Keysight B2980B Series Femto/Picoammeter Electrometer/High Resistance Meter

SCPI Command Reference

KEYSIGHT TECHNOLOGIES

Notices

Copyright Notice

Manual Part Number

B2980-90130

Edition

Edition 1, April 2021

Published by:

Keysight Technologies Japan K.K.
9-1, Takakura-cho, Hachioji-shi, Tokyo
192-8550 Japan

Technology Licenses

The hardware and/or software described in this document are furnished under a license and may be used or copied only in accordance with the terms of such license.

U.S. Government Rights

The Software is "commercial computer software," as defined by Federal Acquisition Regulation ("FAR") 2.101. Pursuant to FAR 12.212 and 27.405-3 and Department of Defense FAR Supplement ("DFARS") 227.7202, the U.S. government acquires commercial computer software under the same terms by which the software is customarily provided to the public. Accordingly, Keysight provides the Software to U.S. government customers under its standard commercial license, which is embodied in its End User License Agreement (EULA), a copy of which can be found at http://www.keysight.com/find/sweula. The license set forth in the EULA represents the exclusive authority by which the U.S. government may use, modify, distribute, or disclose the Software. The EULA and the license set forth therein, does not require or permit, among other things, that Keysight: (1) Furnish technical information related to commercial computer software or commercial computer software documentation that is not customarily provided to the public; or (2) Relinquish to, or otherwise provide, the government rights in excess of these rights customarily provided to the public to use, modify, reproduce, release, perform, display, or disclose commercial computer software or commercial computer software documentation. No additional government requirements beyond those set forth in the EULA shall apply, except to the extent that those terms, rights, or licenses are explicitly required from all providers of commercial computer software pursuant to the FAR and the DFARS and are set forth specifically in writing elsewhere in the EULA. Keysight shall be under no obligation to update, revise or otherwise modify the Software. With respect to any technical data as defined by FAR 2.101, pursuant to FAR 12.211 and 27.404.2 and DFARS 227.7102, the U.S. government acquires no greater than Limited Rights as defined in FAR 27.401 or DFAR 227.7103-5 (c), as applicable in any technical data.

Warranty

THE MATERIAL CONTAINED IN THIS DOCUMENT IS PROVIDED "AS IS," AND IS SUBJECT TO BEING CHANGED, WITHOUT NOTICE, IN FUTURE EDITIONS. FURTHER, TO THE MAXIMUM EXTENT PERMITTED BY APPLICABLE LAW, KEYSIGHT DISCLAIMS ALL WARRANTIES, EITHER EXPRESS OR IMPLIED, WITH REGARD TO THIS MANUAL AND ANY INFORMATION CONTAINED HEREIN, INCLUDING BUT NOT LIMITED TO THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. KEYSIGHT SHALL NOT BE LIABLE FOR ERRORS OR FOR INCIDENTAL OR CONSEQUENTIAL DAMAGES IN CONNECTION WITH THE FURNISHING, USE, OR PERFORMANCE OF THIS DOCUMENT OR OF ANY INFORMATION CONTAINED HEREIN. SHOULD KEYSIGHT AND THE USER HAVE A SEPARATE WRITTEN AGREEMENT WITH WARRANTY TERMS COVERING THE MATERIAL IN THIS DOCUMENT THAT CONFLICT WITH THESE TERMS, THE WARRANTY TERMS IN THE SEPARATE AGREEMENT SHALL CONTROL.

Open Software License

A portion of the software in this product is licensed under terms of the General Public License Version 2 ("GPLv2"). The text of the license and source code can be found at:

www.keysight.com/find/GPLV2

Declaration of Conformity

Declarations of Conformity for this product and for other Keysight products may be downloaded from the Web. Go to http://www.keysight.com/go/conformity. You can then search by product number to find the latest Declaration of Conformity.

Latest Information

To get the latest firmware/software/electronic manuals/specifications/support information, go to www.keysight.com and type in the product number in the Search field at the top of the page.

In This Manual

This manual contains reference information to help you program the Keysight B2980 series over the remote interface using the SCPI programming language. The B2980 supports the SCPI programming language on all of its remote I/O interfaces.

- Chapter 1, "Programming Basics"

  Describes a basic information for programming the B2980, and contains a brief introduction to the SCPI programming language, the data output format, the status system diagram, and the non-volatile settings.

- Chapter 2, "Subsystem Command Summary"

  Lists the B2980 SCPI subsystem commands and summary descriptions.

- Chapter 3, "Common Commands"

  Provides reference information such as description and command syntax of SCPI common commands available for the B2980.

- Chapter 4, "Subsystem Commands"

  Provides reference information such as description and command syntax of device specific SCPI commands available for the B2980.

- Chapter 5, "Error Messages"

  Lists the B2980 error messages, and provides error number, error message and description.

See *User's Guide* for information about the B2980 itself.

Contents

4 Subsystem Commands

1 Programming Basics

This chapter describes a basic information on programming Keysight B2980.

SCPI Commands

SCPI (Standard Commands for Programmable Instruments) is a programming language for controlling test and measurement instruments. SCPI provides instrument control with a standardized command syntax and style, as well as a standardized data interchange format.

SCPI has two types of commands, common and subsystem.

- Common commands

  Common commands are defined by the IEEE 488.2 standard to perform common interface functions such as reset, status, and synchronization. All common commands consist of a three-letter mnemonic preceded by an asterisk: *RST, *IDN?, *SRE 8. Common commands belong to the IEEE-488.2 Common Commands group.

- Subsystem commands

  Subsystem commands perform specific instrument functions. They can be a single command or a group of commands. The groups are comprised of commands that extend one or more levels below the root. Subsystem commands are arranged alphabetically according to the function they perform. The following example shows a portion of a subsystem command tree, from which you access the commands located along the various paths. Some [optional] commands have been included for clarity.

  Example:

```
:INPut
      :ZCORrect
           [:STATe] <Bool>
      [:STATe] <Bool>

[:SENSe]
      :FUNCtion
           [:ON] function[, function[, function]]
      :CURRent[:DC]
           :RANGe <range>
                :AUTO <Bool>
```

Multiple Commands in a Message

Multiple SCPI commands can be combined and sent as a single message with one message terminator. There are two important considerations when sending several commands within a single message.

- Use a semicolon to separate commands within a message.
- There is an implied header path that affects how commands are interpreted by the instrument.

The header path can be thought of as a string that is inserted before each command within a message. For the first command in a message, the header path is a null string. For each subsequent command, the header path is defined as the characters that make up the headers of the previous command in the message up to and including the last colon separator. An example of a message with two commands is:

```
INPut:STATe OFF;ZCORrect:STATe ON
```

which shows the use of the semicolon separating the two commands, and also illustrates the header path concept. Note that with the second command, the leading header “INPut” was omitted because after the “INPut:STATe OFF” command, the header path became defined as “INPut” and thus the instrument interpreted the second command as:

```
INPut:ZCORrect:STATe ON
```

In fact, it would have been syntactically incorrect to include the “INPut” explicitly in the second command, since the result after combining it with the header path would be:

```
INPut:INPut:ZCORrect:STATe ON
```

which is incorrect.

Moving Between Subsystems

In order to combine commands from different subsystems, you must reset the header path to a null string within a message. This is done by beginning the command with a colon (:), which discards any previous header path. For example, you could disable the zero correct function and check the status of the Operation Condition register with a single message by using a root specifier as follows:

```
INPut:ZCORrect:STATe OFF;:STATus:OPERation:CONDition?
```

The following message shows how to combine commands from different subsystems as well as within the same subsystem:

```
SENSe:TOUTput:STATe ON;SIGNal TOUT;:CURRent:RANGe:AUTO ON
```

Note the use of the optional header STATe to maintain the correct path within the subsystems, and the use of the root specifier to move between subsystems.

Including Common Commands

You can combine common commands and subsystem commands into a single message. Treat the common command as a message unit by separating it with a semicolon (the message unit separator). Common commands do not affect the header path; you may insert them anywhere in the message.

```
INPut OFF;*RCL 1;INPut ON
```

Using Queries

Observe the following precautions when using queries.

- Add a blank space between the query indicator (?) and any subsequent parameter such as a channel list.
- Allocate a proper number of variables for the returned data.
- Read back all the results of a query before sending another command to the instrument. Otherwise, a Query Interrupted error will occur and the unreturned data will be lost.

Coupled Commands

When commands are coupled, it means that the value sent by one command is affected by the settings of another command. For example, the following commands are coupled:

`[SENSe:]CURRent:RANGe` and `[SENSe:]CURRent:RANGe:AUTO`

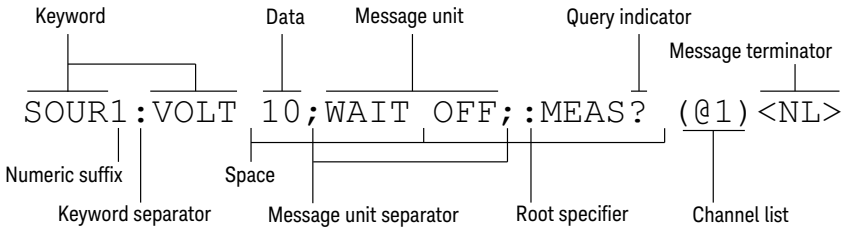
If a range command is sent to place a current measurement range into a specified setting when the present setting is auto-ranging, a specified range setting overrides the present auto-ranging setting.

SCPI Messages

There are two types of SCPI messages, program and response.

- A program message consists of one or more properly formatted SCPI commands sent from the controller to the instrument. The message, which may be sent at any time, requests the instrument to perform some action.
- A response message consists of data in a specific SCPI format sent from the instrument to the controller. The instrument sends the message only when commanded by a program message “query.”

The following figure illustrates the SCPI message structure.



Headers

Headers, also referred to as keywords, are instructions recognized by the instrument. Headers may take a long form or a short form. In the long form, the header is completely spelled out, such as CURRENT, STATUS, and MEDIAN. In the short form, the header includes only the first three or four letters, such as CURR, STAT, and MED.

When the long form notation is used in this document, the capital letters indicate the corresponding short form. For example, when MEASure is the long form, MEAS will be the short form equivalent.

Message Unit

The simplest form of an SCPI command is a single message unit consisting of a command header (or keyword) followed by a message terminator such as a newline. The message unit may include a parameter after the header. The parameter can be a numeric value or a string.

```
*RST<NL>

CURRent:RANGe 0.02<NL>
```

Message Unit Separator

When two or more message units are combined into a compound message, separate the units with semicolons.

```
STATus:OPERation?;QUEStionable?
```

Root Specifier

When it precedes the first header of a message unit, a colon is interpreted as a root specifier. It tells the command parser that this is the root or the top node of the command tree.

Query Indicator

Following a header with a question mark turns it into a query (VOLTage?, VOLTage:TRIGgered?). The ? is the query indicator. If a query contains parameters, place the query indicator at the end of the last header, before the parameters.

```
CURRent:RANGe? MAX
```

Message Terminator

A terminator informs SCPI that it has reached the end of a message. The following message terminators are permitted.

- newline <NL>, expressed in ASCII as decimal 10 or hex 0A
- end or identify <END> (EOI with ATN false)
- both of the above <NL><END>
- also <CR><NL>

In the examples used in this document, there is an assumed message terminator at the end of each message.

Numeric Suffix

All command headers can be accompanied by a numeric suffix for differentiating multiple instances of the same structure, i.e. for multi-channel instruments. The numeric suffix can be appended to both long and short forms. For example, TRIG1 is the short form of TRIGger1. A numeric suffix of 1 is implied on all command headers that do not explicitly define a suffix; thus, TRIG is equivalent to TRIG1.

For B2980 SCPI commands, some commands have a numeric suffix for classifying the channels, the trigger lines, etc. The numeric suffix is an optional character, and can be expressed by [*c*], [*d*], [*h*], [*i*], [*j*], and [*n*], where:

c is the integer 1 or 2, used to specify the channel 1 or 2, respectively. Note that only 1 is available on the present B2980 products.

d is the integer 1 or 2, used to specify the upper half (1) or lower half (2), respectively, of the display area on the front panel display.

h is an integer between 1 and 100, used to specify a variable in a program memory.

i is the integer 1 or 2, used to specify the internal trigger line 1 or 2, respectively.

j is the integer 1 or 2, used to specify the primary (1) or secondary (2), respectively.

n is an integer between 1 and 7, used to specify a Digital I/O pin.

Abbreviating the numeric suffix gives the same result as specifying 1.

Channel List Parameter

The channel list parameter is used for identifying the channel number as well as the numeric suffix.

The notation `(@1,2)` specifies a channel list that includes channels 1 and 2.

The notation `(@1:2)` specifies a channel list that includes channels 1 to 2.

In the B2980 SCPI commands, the channel list parameter is only available on certain commands which requires specification of the channel itself (e.g. some commands of the MMEMory Subsystem).

Note that only `(@1)` is available on the present B2980 products.

SCPI Command Completion

SCPI commands sent to the instrument are processed either sequentially or in parallel. Sequential commands finish execution before the subsequent command is started. Parallel commands allow other commands to begin executing while the parallel command is still executing.

The *WAI, *OPC, and *OPC? common commands provide different ways of indicating when all transmitted commands, including any parallel ones, have completed their operations. Some practical considerations for using these commands are as follows:

*WAI - prevents the instrument from processing subsequent commands until all pending operations are completed.

*OPC? - places a 1 in the Output Queue when all pending operations have completed. Since it requires your program to read the returned value before executing the next program statement, *OPC? can be used to cause the controller to wait for commands to complete before proceeding with its program.

*OPC - sets the OPC status bit when all pending operations have completed. Since your program can read this status bit on an interrupt basis, *OPC allows subsequent commands to be executed.

NOTE: The trigger subsystem must be in the Idle state for the status OPC bit to be true. As far as triggers are concerned, OPC is false whenever the trigger subsystem is in the Initiated state.

Device Clear

You can send a Device Clear at any time to abort an SCPI command that may be hanging up the GPIB interface. Device Clear aborts all transient and acquire actions, clears the input and output buffers of the instrument and prepares the instrument to accept a new command string. The error queue and all configuration states are left unchanged by Device Clear.

SCPI Conventions and Data Formats

The SCPI conventions shown in Table 1-1 are used throughout this document.

Data programmed or queried from the instrument is coded in ASCII. The data may contain numeric values or character strings.

Table 1-1 SCPI Conventions and Data Formats

Convention	Description
Angle brackets < >	Items within angle brackets are parameter abbreviations. For example, <NR1> indicates a specific form of numerical data.
Vertical bar |	Vertical bars separate alternative parameters. For example, <VOLT|CURR> indicates that either VOLT or CURR must be placed there.
Square brackets [ ]	Items within square brackets are optional. The representation [SENSe:]CURRent means that SENSe: may be omitted.
Parentheses ( )	Items within parentheses are used in place of the usual parameter types to specify a channel list. The notation (@1:3) specifies a channel list that includes channels 1, 2, and 3. The notation (@1,3) specifies a channel list that includes only channels 1 and 3.
Braces { }	Braces indicate parameters that may be repeated zero or more times. It is used especially for representing arrays. The notation <A>{,<B>} shows that parameter "A" must be entered, while parameter "B" omitted or may be entered one or more times.
<NR1>	Digits with an implied decimal point assumed at the right of the least-significant digit. Example: 273
<NR2>	Digits with an explicit decimal point. Example: 27.3
<NR3>	Digits with an explicit decimal point and an exponent. Example: 2.73E+02
<NRf>	Extended format that includes <NR1>, <NR2> and <NR3>. Examples: 273, 27.3, 2.73E+02

Convention	Description
<NRf+>	Expanded decimal format that includes <NRf>, MIN, and MAX. Examples: 273, 27.3, 2.73E+02, MAX. MIN and MAX are the minimum and maximum limit values that are implicit in the range specification for the parameter.
<NDN>	Non-decimal numeric value. May also be represented in binary preceded by "#B", octal preceded by "#Q", or hexadecimal preceded by "#H". Examples: 29 (decimal), #B11101 (binary), #Q35 (octal), #H1D (hexadecimal)
<Bool>	Boolean data. Can be numeric (0, 1), or named (OFF, ON).
<SPD>	String program data. Programs string parameters enclosed in single or double quotes.
<CPD>	Character program data. Programs discrete parameters. Accepts both short form and long form.
<SRD>	String response data. Returns string parameters enclosed in single or double quotes.
<CRD>	Character response data. Returns discrete parameters. Only the short form of the parameter is returned.
<AARD>	Arbitrary ASCII response data. Permits the return of un-delimited 7-bit ASCII. This data type has an implied message terminator.
<Block>	Arbitrary block response data. Permits the return of definite length and indefinite length arbitrary response data. This data type has an implied message terminator.
<Expr>	Channel list, group list, or math expression. Channel list: Parenthetical data beginning with "@" Group list: Parenthetical data beginning with "@" Math expression: Parenthetical math expression (see :CALCulate:MATH[:EXPRession][:DEFine] command)

Data Output Format

B2980 supports the following data output formats for sending the result data. The data contains all of the elements specified by the :FORMat:ELEMents:SENSe or :FORMat:ELEMents:CALCulate command. Available elements are voltage measurement data, current measurement data, resistance measurement data, calculation result data, time data, status data, source output setting data, temperature, and humidity data.
A terminator <newline> (0x0a, 1 byte) is attached to the end of each data.

NOTE B2981B/B2983B supports current measurement data, time data, and status data only.

- ASCII data format, set by :FORMat[:DATA] ASCii

  Returns the result data in the comma-separated format. If the data contains three elements, B2980 sends the data as shown in the following example.

  Example: +1.000001E–06,+1.000002E–06,+9.999999E–07<newline>

  +9.910000E+37 indicates “not a number.”

  +9.900000E+37 indicates positive infinity.

  –9.900000E+37 indicates negative infinity.

- IEEE-754 single precision format, set by :FORMat[:DATA] REAL,32

  4-byte definite length block data, #<number of digits for byte length><byte length><byte>...<byte><terminator>. For example, two data elements are sent by a data block which consists of a header (3 bytes, #18), two 4-byte data, and a terminator (1 byte). A 4-byte data is used for each data element. Each element consists of a fraction (bits 0 (LSB) to 22), exponent (bits 23 to 30), and sign (bit 31).

  Order of bytes set by :FORMat:BORDer NORMal (default): byte 1 to 4

  Order of bytes set by :FORMat:BORDer SWAPped: byte 4 to 1

  NaN indicates “not a number.”

  +infinity indicates positive infinity.

–infinity indicates negative infinity.

- IEEE-754 double precision format, set by :FORMat[:DATA] REAL,64

  8-byte definite length block data, #<number of digits for byte length><byte length><byte>...<byte><terminator>. For example, one data element is sent by a data block which consists of a header (3 bytes, #18), one 8-byte data, and a terminator (1 byte). An 8-byte data is used for each data element. Each element consists of a fraction (bits 0 (LSB) to 51), exponent (bits 52 to 62), and sign (bit 63).

  Order of bytes set by :FORMat:BORDer NORMal (default): byte 1 to 8

  Order of bytes set by :FORMat:BORDer SWAPped: byte 8 to 1

  NaN indicates “not a number.”

  +infinity indicates positive infinity.

  –infinity indicates negative infinity.

Status Data

B2980 sends the status data with the result data if it is specified by the :FORMat:ELEMents:SENSe or :FORMat:ELEMents:CALCulate command.

The status data is given by a binary-weighted sum of all bits set in the binary data. For example, if bit 3 (decimal value = 8) and bit 5 (decimal value = 32) are set to 1, the status data returns 40.

Bit definitions of the status data are shown in Table 1-2. This table shows the meaningful bits only. Ignore the other bits.

Table 1-2 Bit Definitions of Status Data

Bit	Description	Decimal value
0	Measurement range overflow (Current/Charge measurement) 0: No or 1: Yes	0 or 1
1	Measurement range overflow (Voltage measurement) 0: No or 1: Yes	0 or 2
2	Measurement out of limit state (Current/Charge measurement) 0: No or 1: Yes	0 or 4
3	Measurement out of limit state (Voltage measurement) 0: No or 1: Yes	0 or 8
4	Zero correct function state 0: Disabled or 1: Enabled	0 or 16
5	Reference function (null offset cancel) state (Current/Charge measurement) 0: Disabled or 1: Enabled	0 or 32
6	Reference function (null offset cancel) state (Voltage measurement) 0: Disabled or 1: Enabled	0 or 64
7	Reference function (null offset cancel) state (Resistance measurement) 0: Disabled or 1: Enabled	0 or 128
8	Voltage force over current 0: No or 1: Yes	0 or 256
9	Median filter state 0: Disabled or 1: Enabled	0 or 512

Bit	Description	Decimal value
10	Moving filter state 0: Disabled or 1: Enabled	0 or 1024
11	Auto resistance measurement state 0: Disabled or 1: Enabled	0 or 2048
12	Voltage select for resistance measurement 0: Voltage measurement value or 1: Voltage force value	0 or 4096
16 to 20	Composite limit test result, 0 to 31	0 or (1 to 31)×2^{16} (0 or 65536 to 2031616)

GPIB Capability

Th functions in the following table provide the means of for an instrument to receive, process, and transmit commands, data, and status over the GPIB bus.

Table 1-3 GPIB Capabilities and Functions of the B2980

Interface Function	Code	Description
Source Handshake	SH1	Complete capability
Acceptor Handshake	AH1	Complete capability
Talker	T6	Basic Talker: YES Serial Poll: YES Talk Only Mode: NO Unaddress if MLA (my listen address): YES
Listener	L4	Basic Listener: YES Unaddress if MTA (my talk address): YES Listen Only Mode: NO
Service Request	SR1	Complete capability
Remote/Local	RL1	Complete capability (with local lockout)
Parallel Poll	PP0	No capability
Device Clear	DC1	Complete capability
Device Trigger	DT1	Complete capability
Controller Function	C0	No capability
Driver Electronics	E1	Open Collector

Status Byte

Status byte bits are turned off or on (0 or 1) to represent the instrument operation status. When you execute a serial poll, an external computer (controller) reads the contents of the status byte, and responds accordingly. When an unmasked status bit is set to "1", the instrument sends an SRQ to the controller, causing the controller to perform an interrupt service routine.

Table 1-4 Status Byte Description

Bit	Decimal Value	Description
0	1	Measurement status summary
1	2	Not used
2	4	Error queue not empty
3	8	Questionable status summary
4	16	Output buffer
5	32	Event status byte summary
6	64	Master status summary (Request for service)
7	128	Operation status summary

The status byte register can be read with either a serial poll or the *STB? query command. Serial poll is a low-level GPIB command.

In general, use serial polling (not *STB?) inside interrupt service routines. Use *STB? in other cases (not in interrupt service routine) when you want to know the value of the Status Byte.

Status System Diagram

Figure 1-1 B2980 Status System Overview

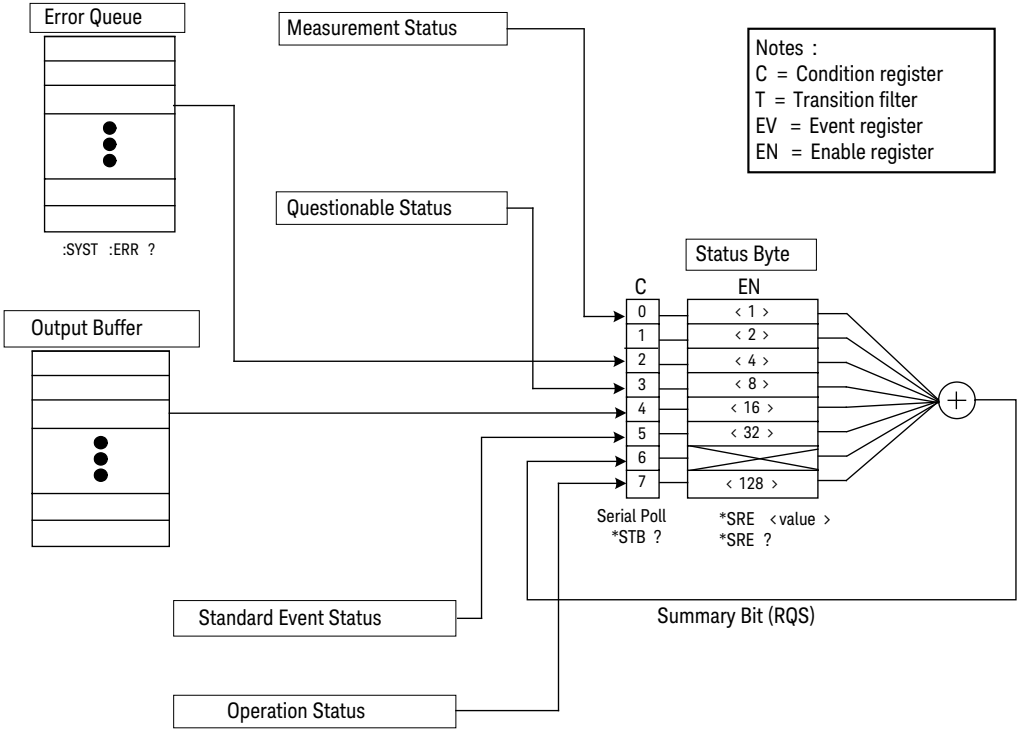


Figure 1-2 Measurement Status Register

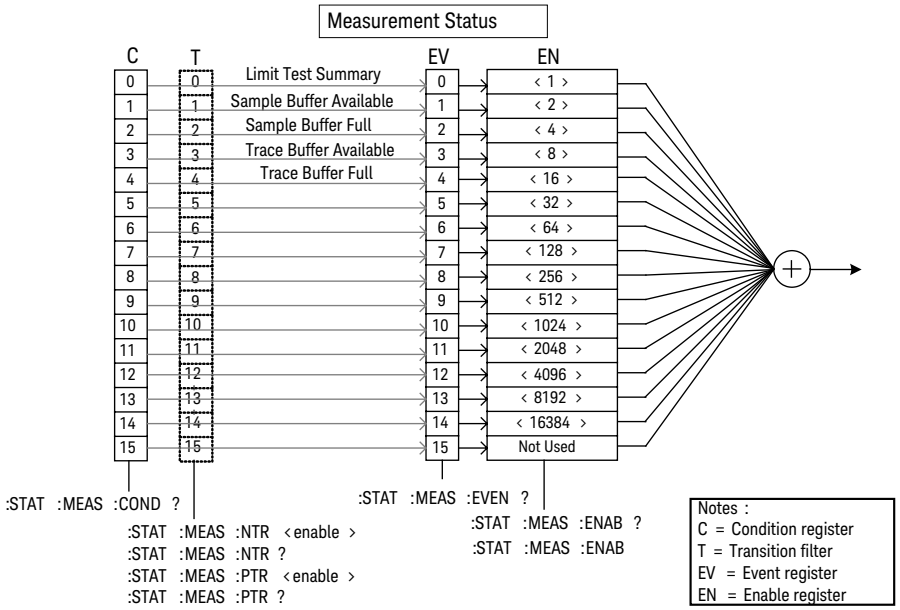


Figure 1-3 Questionable Status Register

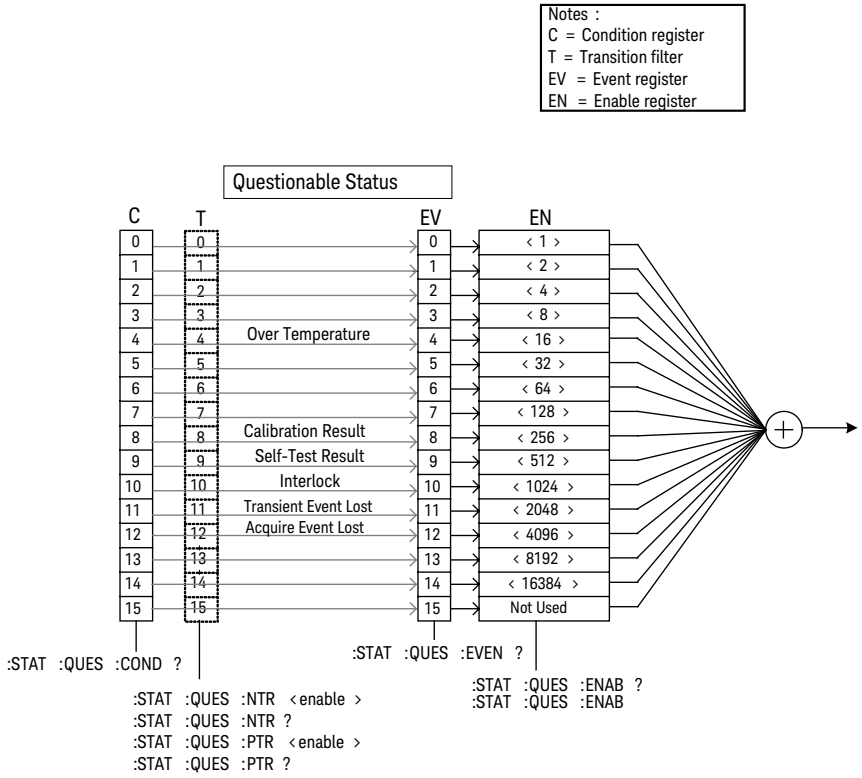


Figure 1-4 Standard Event Status Register

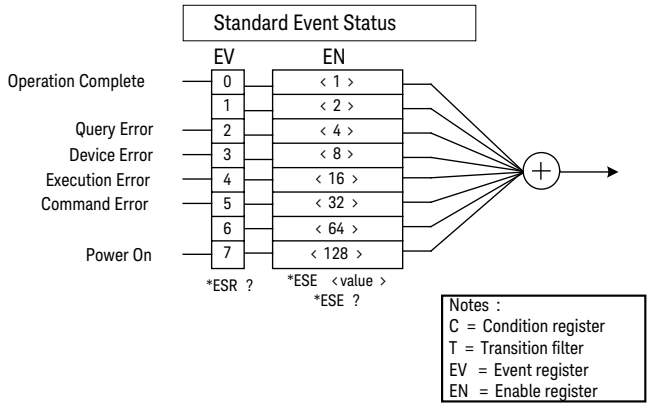
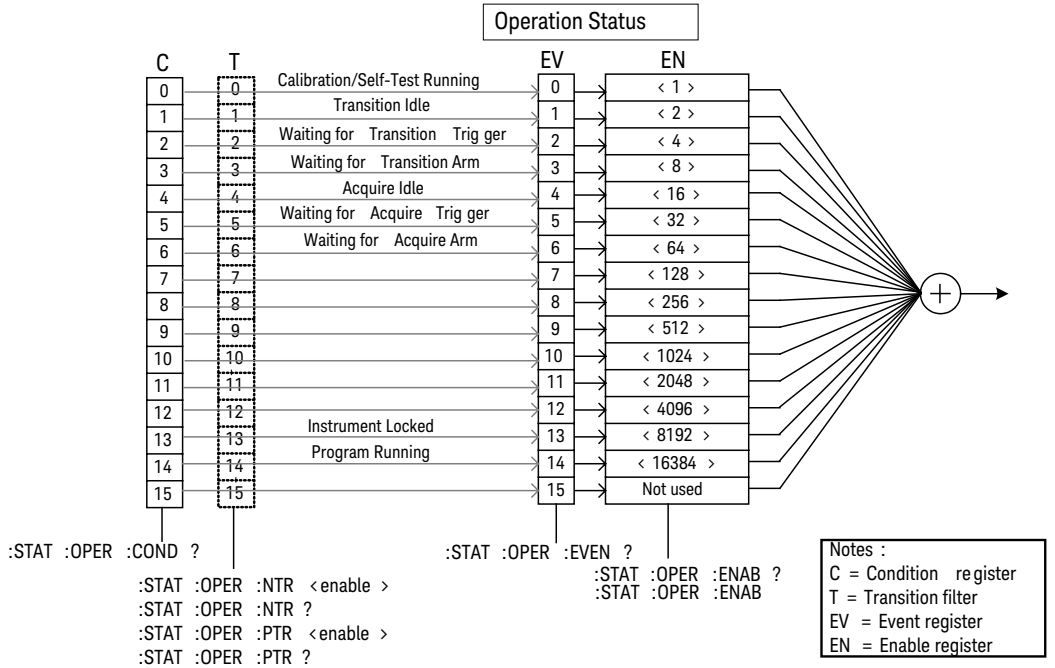


Figure 1-5 Operation Status Register



Non-Volatile Settings

The following tables show the factory-shipped non-volatile settings of the instrument. Information in non-volatile memory is NOT lost when power is turned off. These settings are all customer-configurable.

Table 1-5 Non-Volatile Communication Settings

Setup item	Factory default setting
DHCP	Enabled
IP address	169.254.5.2
Subnet mask	255.255.0.0
Default gateway	0.0.0.0
Obtain DNS server from DHCP	Enabled
DNS server	0.0.0.0
WINS server	0.0.0.0
Hostname	K-B29*xx*B-*nnnnn*
Desired hostname	B29*xx*B: model number
Desired service name	*nnnnn*: suffix of serial number.
mDNS	Enabled
Use DNS naming service	Enabled
Use NetBIOS naming service	Enabled
Domain name	Not set
GPIB address	23
LXI identify	Disabled
GPIB command interface	Enabled
USB command interface	Enabled

Setup item	Factory default setting
VXI-11 command interface	Enabled
SCPI telnet command interface	Enabled
SCPI socket command interface	Enabled
SCPI HiSLIP command interface	Enabled
Web interface	Enabled
Command prompt for a Telnet session	B2900B>
Welcome message for a Telnet session	Welcome to Keysight B2900B Series

Table 1-6 Other Non-volatile Settings

Setup item	Factory default setting
Remote display	Enabled
Display color set	1
Beeper	Enabled
Graphical web interface (web server)	Enabled
Power-on program	Not set
Line frequency	50 Hz
Power-on line frequency detection	Enabled
Dual measure result display	ON
Temperature display	ON
Humidity display	ON
Immediate voltage update by knob	OFF
Temperature sensor	Thermocouple
Temperature unit	°C

Setup item	Factory default setting
Voltmeter inner shield connection	Guard
Measured Value Format (Meter View)	ENG

2 Subsystem Command Summary

This chapter lists all of the SCPI subsystem commands for Keysight B2980 and provides the summary information on the command.

- "Setting Measurement Conditions"

  "INPut Subsystem" for enabling/disabling the current/charge input or zero correct function

  "SENSe Subsystem" for measurement setup

  "Measurement Ranges"
- "Setting Voltage Source"

  "OUTPut Subsystem" for using source output functions

  "SOURce Subsystem" for source setup

  "Voltage Output Ranges"
- "Controlling Source/Measure Trigger"

  "TRIGger Subsystem" for triggering source output and measurement

  "LXI Subsystem" for using LXI trigger event functions
- "Reading Source/Measure Data"

  "FETCh Subsystem" only for reading data

  "FORMat Subsystem" for data output format

  "READ Subsystem" for performing measurements

  "MEASure Subsystem" for a spot measurement

  "CALCulate Subsystem" for using math functions

"TRACe Subsystem" for using trace buffer

- "Using Advanced Functions"

"HCOPy Subsystem" for getting screen dump

"DISPlay Subsystem" for front panel display setup

"MMEMory Subsystem" for managing data memory

"PROGram Subsystem" for using program memory

"SYSTem Subsystem" for using system functions

"STATus Subsystem" for using status system

NOTE In the tables, Reset setting gives the initial setting or the default setting which is set to the instrument when it is turned on or it receives the *RST command.

NOTE The following subsystem commands are classified under the TRIGger subsystem because they are used for trigger operations.

- ABORt
- ARM
- IDLE
- INITiate
- TRIGger

For examples of the source output and measurement operation, see Figure 2-7. Also see Figures 2-6 and 2-8 for using the trigger commands. For these commands, see Table 2-10.

NOTE For details on numeric suffixes [*c*], [*d*], [*h*], [*i*], [*j*], and [*n*], see "Numeric Suffix" on page 27.

NOTE

FETCh:TEMPerature?, MEASure:VOLTage?, READ:CHARge?, and SENSe:RESistance:XXXX commands are not available on B2981B/B2983B, which does not have charge, voltage, resistance, temperature, humidity measurement functions. See the following table.

OUTPut and SOURce subsystem commands, and TRIGger:TRANsient: commands are not available on B2981B/B2983B, which does not have Voltage Source.

The commands related to battery operation are not available for B2981B/B2985B.

Model No.	Battery Op.	Measurement Function						Voltage Source
		Current	Voltage	Charge	Resistance	Temperature	Humidity	
B2981B	n.a.	yes	n.a.	n.a.	n.a.	n.a.	n.a.	n.a.
B2983B	yes							
B2985B	n.a.		yes	yes	yes	yes	yes	yes
B2987B	yes							

Setting Measurement Conditions

NOTE Keysight B2981B/B2983B supports current measurement only.

Table 2-1 INPut Subsystem

Command	Summary	Reset setting
:INP[*c*][:STAT] *mode* :INP[*c*][:STAT]?	Enables or disables the current/charge measure input. *mode* = 1|ON|0|OFF	OFF
:INP[*c*]:ZCOR[:STAT] *mode* :INP[*c*]:ZCOR[:STAT]?	Enables or disables the zero correct function. *mode* = 1|ON|0|OFF	OFF

Table 2-2 SENSe Subsystem

Command	Summary	Reset setting
[:SENS[*c*]]:ARSP *mode* [:SENS[*c*]]:ARSP?	Specifies the automatic ranging operation. *mode* = NORM|FAST	NORM
(B2985B/B2987B) [:SENS[*c*]]:CHAR:ADIS:LEV *level* [:SENS[*c*]]:CHAR:ADIS:LEV?	Specifies the level of the automatic discharge function for charge measurement. *level* = *value*|MIN|MAX|DEF	2.0E–6 C
(B2985B/B2987B) [:SENS[*c*]]:CHAR:ADIS[:STAT] *mode* [:SENS[*c*]]:CHAR:ADIS[:STAT]?	Enables or disables the automatic discharge function. *mode* = 1|ON|0|OFF	OFF
(B2985B/B2987B) [:SENS[*c*]]:CHAR:DISC	Discharges the capacitor used for charge measurement.	

Command	Summary	Reset setting
(B2985B/B2987B) [:SENS[*c*]]:CHAR:RANG:AUTO:GRO *group* [:SENS[*c*]]:CHAR:RANG:AUTO:GRO?	Selects the range group for the automatic measurement ranging operation at charge measurement. *group* = HIGH|LOW	HIGH
[:SENS[*c*]]:CURR[:DC]:APER *time* [:SENS[*c*]]:CURR[:DC]:APER? [DEFault|MINimum|MAXimum]	Sets the integration time for one point measurement. *time* = MINimum|MAXimum|DEFault| +1E–5 to +2 seconds The integration time can be expressed by the following formula by using the NPLC value set by the :SENS:<CHAR|CURR|RES|VOLT>:NPLC command. So the last command setting is effective for both *time* and *nplc*. *time = nplc / power line frequency*	0.1 PLC, = 0.1/*power line frequency*
(B2985B/B2987B) [:SENS[*c*]]:CHAR:APER *time* [:SENS[*c*]]:RES:APER *time* [:SENS[*c*]]:VOLT[:DC]:APER *time* [:SENS[*c*]]:CHAR:APER? [DEFault|MINimum|MAXimum] [:SENS[*c*]]:RES:APER? [DEFault|MINimum|MAXimum] [:SENS[*c*]]:VOLT[:DC]:APER? [DEFault|MINimum|MAXimum]		
[:SENS[*c*]]:CURR[:DC]:APER:AUTO *mode* [:SENS[*c*]]:CURR[:DC]:APER:AUTO?	Enables or disables the automatic aperture function. *mode* = 1|ON|0|OFF The automatic aperture on/off works with the automatic NPLC on/off set by the :SENS:<CHAR|CURR|RES|VOLT>:NPLC:AUTO command. So the last command setting is effective for both functions.	ON
(B2985B/B2987B) [:SENS[*c*]]:CHAR:APER:AUTO *mode* [:SENS[*c*]]:RES:APER:AUTO *mode* [:SENS[*c*]]:VOLT[:DC]:APER:AUTO *mode* [:SENS[*c*]]:CHAR:APER:AUTO? [:SENS[*c*]]:RES:APER:AUTO? [:SENS[*c*]]:VOLT[:DC]:APER:AUTO?		

Command	Summary	Reset setting
[:SENS[*c*]]:CURR[:DC]:APER:AUTO:MODE *mode* [:SENS[*c*]]:CURR[:DC]:APER:AUTO:MODE?	Selects the mode of automatic aperture function. *mode* = SHOR|MED|LONG The automatic aperture mode is enabled or disabled by the :SENS:<CHAR|CURR|RES|VOLT>:APER:AUTO command. The mode of automatic aperture function works with the mode of automatic NPLC function set by the :SENS:<CHAR|CURR|RES|VOLT>:NPLC:AUTO:MODE command.	MED
(B2985B/B2987B) [:SENS[*c*]]:CHAR:APER:AUTO:MODE *mode* [:SENS[*c*]]:RES:APER:AUTO:MODE *mode* [:SENS[*c*]]:VOLT[:DC]:APER:AUTO:MODE *mode* [:SENS[*c*]]:CHAR:APER:AUTO:MODE? [:SENS[*c*]]:RES:APER:AUTO:MODE? [:SENS[*c*]]:VOLT[:DC]:APER:AUTO:MODE?		
[:SENS[*c*]]:CURR[:DC]:AVER:MOV:COUN *mov_count* [:SENS[*c*]]:CURR[:DC]:AVER:MOV:COUN?	Sets the number of measurement samples used for the moving average. *mov_count* = 1 to 100 *mov_count* is common to all measurement functions, which are current, charge, resistance, and voltage measurements. So the last command setting is effective for all measurement functions.	DEF = 1
(B2985B/B2987B) [:SENS[*c*]]:CHAR:AVER:MOV:COUN *mov_count* [:SENS[*c*]]:RES:AVER:MOV:COUN *mov_count* [:SENS[*c*]]:VOLT[:DC]:AVER:MOV:COUN *mov_count* [:SENS[*c*]]:CHAR:AVER:MOV:COUN? [:SENS[*c*]]:RES:AVER:MOV:COUN? [:SENS[*c*]]:VOLT[:DC]:AVER:MOV:COUN?		

Command	Summary	Reset setting
[:SENS[*c*]]:CURR[:DC]:AVER:MOV[:STAT] *mode* [:SENS[*c*]]:CURR[:DC]:AVER:MOV[:STAT]?	Enables or disables the moving average filter. *mode* = 1|ON|0|OFF Enabling or disabling the moving average filter is common to all measurement functions, which are current, charge, resistance, and voltage measurements. So the last command setting is effective for all measurement functions.	OFF
(B2985B/B2987B) [:SENS[*c*]]:CHAR:AVER:MOV[:STAT] *mode* [:SENS[*c*]]:RES:AVER:MOV[:STAT] *mode* [:SENS[*c*]]:VOLT[:DC]:AVER:MOV[:STAT] *mode* [:SENS[*c*]]:CHAR:AVER:MOV[:STAT]? [:SENS[*c*]]:RES:AVER:MOV[:STAT]? [:SENS[*c*]]:VOLT[:DC]:AVER:MOV[:STAT]?		
[:SENS[*c*]]:CURR[:DC]:NPLC *nplc* [:SENS[*c*]]:CURR[:DC]:NPLC? [DEFault|MINimum|MAXimum]	Sets the number of power line cycles (NPLC) value instead of setting the integration time for one point measurement. *nplc* = MINimum|MAXimum|DEFault|+5E–4 to +100 for 50 Hz or +6E–4 to +120 for 60 Hz The NPLC value can be expressed by the following formula by using the integration time set by the :SENS:<CHAR|CURR|RES|VOLT>:APER command. So the last command setting is effective for both *nplc* and *time*. *nplc* = *time* × *power line frequency*	0.1 PLC, = 0.1/*power line frequency*
(B2985B/B2987B) [:SENS[*c*]]:CHAR:NPLC *nplc* [:SENS[*c*]]:RES:NPLC *nplc* [:SENS[*c*]]:VOLT[:DC]:NPLC *nplc* [:SENS[*c*]]:CHAR:NPLC? [DEFault|MINimum|MAXimum] [:SENS[*c*]]:RES:NPLC? [DEFault|MINimum|MAXimum] [:SENS[*c*]]:VOLT[:DC]:NPLC? [DEFault|MINimum|MAXimum]		

Command	Summary	Reset setting
[:SENS[*c*]]:CURR[:DC]:NPLC:AUTO *mode* [:SENS[*c*]]:CURR[:DC]:NPLC:AUTO?	Enables or disables the automatic NPLC function. *mode* = 1|ON|0|OFF The automatic NPLC on/off works with the automatic aperture on/off set by the :SENS:<CHAR|CURR|RES|VOLT>:APER:AUTO command. So the last command setting is effective for both functions. The mode of automatic NPLC function works with the mode of automatic aperture function set by the :SENS:<CHAR|CURR|RES|VOLT>:APER:AUTO:MODE command.	ON
(B2985B/B2987B) [:SENS[*c*]]:CHAR:NPLC:AUTO *mode* [:SENS[*c*]]:RES:NPLC:AUTO *mode* [:SENS[*c*]]:VOLT[:DC]:NPLC:AUTO *mode* [:SENS[*c*]]:CHAR:NPLC:AUTO? [:SENS[*c*]]:RES:NPLC:AUTO? [:SENS[*c*]]:VOLT[:DC]:NPLC:AUTO?		
[:SENS[*c*]]:CURR[:DC]:NPLC:AUTO:MODE *mode* [:SENS[*c*]]:CURR[:DC]:NPLC:AUTO:MODE?	Selects the mode of automatic NPLC function. *mode* = SHOR|MED|LONG The automatic NPLC function is enabled or disabled by the :SENS:<CHAR|CURR|RES|VOLT>:NPLC:AUTO command.	MED
(B2985B/B2987B) [:SENS[*c*]]:CHAR:NPLC:AUTO:MODE *mode* [:SENS[*c*]]:RES:NPLC:AUTO:MODE *mode* [:SENS[*c*]]:VOLT[:DC]:NPLC:AUTO:MODE *mode* [:SENS[*c*]]:CHAR:NPLC:AUTO:MODE? [:SENS[*c*]]:RES:NPLC:AUTO:MODE? [:SENS[*c*]]:VOLT[:DC]:NPLC:AUTO:MODE?		

Command	Summary	Reset setting
[:SENS[*c*]]:CURR[:DC]:RANG:AUTO *mode* [:SENS[*c*]]:CURR[:DC]:RANG:AUTO?	Enables or disables the automatic ranging function. *mode* = 0|OFF|1|ON	ON
(B2985B/B2987B) [:SENS[*c*]]:CHAR:RANG:AUTO *mode* [:SENS[*c*]]:RES:RANG:AUTO *mode* [:SENS[*c*]]:VOLT[:DC]:RANG:AUTO *mode* [:SENS[*c*]]:CHAR:RANG:AUTO? [:SENS[*c*]]:RES:RANG:AUTO? [:SENS[*c*]]:VOLT[:DC]:RANG:AUTO?		
[:SENS[*c*]]:CURR[:DC]:RANG:AUTO:LLIM *range* [:SENS[*c*]]:CURR[:DC]:RANG:AUTO:LLIM?	Specifies the lower limit for the automatic measurement ranging operation and sets the minimum measurement range which provides the best resolution to measure the specified value. If the minimum measurement range is same as the maximum measurement range, the measurement is performed by using this range. *range* = *value*|MIN|MAX|DEF *value* for current measurement, 2 pA to 20 mA. *value* for voltage measurement, 2 V to 20 V. *value* for resistance measurement, 1 MΩ to 100 GΩ.	2 pA, 1 MΩ, or 2 V
(B2985B/B2987B) [:SENS[*c*]]:RES:RANG:AUTO:LLIM *range* [:SENS[*c*]]:VOLT[:DC]:RANG:AUTO:LLIM *range* [:SENS[*c*]]:RES:RANG:AUTO:LLIM? [:SENS[*c*]]:VOLT[:DC]:RANG:AUTO:LLIM?		

Command	Summary	Reset setting
[:SENS[*c*]]:CURR[:DC]:RANG:AUTO:ULIM *range* [:SENS[*c*]]:CURR[:DC]:RANG:AUTO:ULIM? (B2985B/B2987B) [:SENS[*c*]]:RES:RANG:AUTO:ULIM *range* [:SENS[*c*]]:VOLT[:DC]:RANG:AUTO:ULIM *range* [:SENS[*c*]]:RES:RANG:AUTO:ULIM? [:SENS[*c*]]:VOLT[:DC]:RANG:AUTO:ULIM?	Specifies the upper limit for the automatic measurement ranging operation and sets the maximum measurement range which provides the best resolution to measure the specified value. If the minimum measurement range is same as the maximum measurement range, the measurement is performed by using this range. *range* = *value*|MIN|MAX|DEF *value* for current measurement, 2 pA to 20 mA. *value* for voltage measurement, 2 V to 20 V. *value* for resistance measurement, 1 MΩ to 100 GΩ.	20 mA, 100 GΩ, or 20 V
[:SENS[*c*]]:CURR[:DC]:RANG[:UPP] *range* [:SENS[*c*]]:CURR[:DC]:RANG[:UPP]? (B2985B/B2987B) [:SENS[*c*]]:CHAR:RANG[:UPP] *range* [:SENS[*c*]]:RES:RANG[:UPP] *range* [:SENS[*c*]]:VOLT[:DC]:RANG[:UPP] *range* [:SENS[*c*]]:CHAR:RANG[:UPP]? [:SENS[*c*]]:RES:RANG[:UPP]? [:SENS[*c*]]:VOLT[:DC]:RANG[:UPP]?	Specifies the expected measurement value and sets the measurement range which provides the best resolution to measure the specified value. *range* = *value*|UP|DOWN|MIN|MAX|DEF *value* for current measurement, 2 pA to 20 mA. *value* for charge measurement, 2 nC to 2 μC. *value* for voltage measurement, 2 V to 20 V. *value* for resistance measurement, 1 MΩ to 1 PΩ.	2 μA, 20 nC, 10 MΩ, or 20 V

Command	Summary	Reset setting
[:SENS[*c*]]:CURR[:DC]:REF *reference* [:SENS[*c*]]:CURR[:DC]:REF?	Sets the reference value used for each measurement data. *reference* = –9.999999E+20 to +9.999999E+20|MIN|MAX|DEF	0.0
(B2985B/B2987B) [:SENS[*c*]]:CHAR:REF *reference* [:SENS[*c*]]:RES:REF *reference* [:SENS[*c*]]:VOLT[:DC]:REF *reference* [:SENS[*c*]]:CHAR:REF? [:SENS[*c*]]:RES:REF? [:SENS[*c*]]:VOLT[:DC]:REF?		
[:SENS[*c*]]:CURR[:DC]:REF:ACQ	Automatically sets the reference value used for each measurement data.	
(B2985B/B2987B) [:SENS[*c*]]:CHAR:REF:ACQ [:SENS[*c*]]:RES:REF:ACQ [:SENS[*c*]]:VOLT[:DC]:REF:ACQ		
[:SENS[*c*]]:CURR[:DC]:REF:STAT *mode* [:SENS[*c*]]:CURR[:DC]:REF:STAT?	Enables or disables the reference function (null offset cancel) for each measurement data. *mode* = 1|ON|0|OFF	OFF
(B2985B/B2987B) [:SENS[*c*]]:CHAR:REF:STAT *mode* [:SENS[*c*]]:RES:REF:STAT *mode* [:SENS[*c*]]:VOLT[:DC]:REF:STAT *mode* [:SENS[*c*]]:CHAR:REF:STAT? [:SENS[*c*]]:RES:REF:STAT? [:SENS[*c*]]:VOLT[:DC]:REF:STAT?		

Command	Summary	Reset setting
[:SENS[*c*]]:CURR[:DC]:MED:RANK *rank* [:SENS[*c*]]:CURR[:DC]:MED:RANK?	Sets the rank of the median filter function. *rank* = *value*|MIN|MAX|DEF If the median filter is enable, the measurement value is calculated central value from the *N* measurement samples. *N* is calculated from *rank* as follows: $N = 2 \times rank + 1$	0
[:SENS[*c*]]:CURR[:DC]:MED[:STAT] *mode* [:SENS[*c*]]:CURR[:DC]:MED[:STAT]?	Enables or disables the median filter function. *mode* = 0|OFF|1|ON	OFF
[:SENS[*c*]]:DATA? [*offset*[,*size*]]	Returns the array data which contains all data for the element specified by the :FORM:ELEM:SENS command. *offset* = CURRent|STARt|0 to maximum *size* = 1 to maximum	
[:SENS[*c*]]:DATA:ACQ	Executes a spot measurement (one-shot measurement). Measurement conditions must be set by SCPI commands of front panel operation before executing this command.	
[:SENS[*c*]]:DATA:CLE	Clears all measurement data.	
[:SENS[*c*]]:DATA:LAT?	Returns the latest data for the element specified by the :FORM:ELEM:SENS command.	
[:SENS[*c*]]:FUNC[:ON] *fctn*[,*fctn*[,*fctn*]] [:SENS[*c*]]:FUNC[:ON]?	Enables the specified measurement functions. *fctn* = "CHARge"|"CURRent[:DC]"|"VOLTage[:DC]"|"RESistance" B2981B/B2983B can specify "CURRent[:DC]" only.	"CURR" for B2981B/ B2983B "VOLT", "CURR" for B2985B/ B2987B

Command	Summary	Reset setting
(B2985B/B2987B) [:SENS[*c*]]:RES:AVAL[:STAT] *mode* [:SENS[*c*]]:RES:AVAL[:STAT]?	Sets the resistance measurement result to absolute format or not. *mode* = 1|ON|0|OFF	OFF
(B2985B/B2987B) [:SENS[*c*]]:RES:VSC *mode* [:SENS[*c*]]:RES:VSC?	Selects the voltage source control mode for resistance measurement. *mode* = MAN|AUTO	MAN
(B2985B/B2987B) [:SENS[*c*]]:RES:VSEL *select* [:SENS[*c*]]:RES:VSEL?	Selects the voltage source value for calculation of resistance measurement, voltage source setting (VSO) or measured value (VME). *select* = VSO|VME	VSO
[:SENS[*c*]]:TOUT:SIGN *output*{,output} [:SENS[*c*]]:TOUT:SIGN?	Selects the trigger output for the status change between the trigger layer and the acquire device action. *output* = INT1|INT2|LAN|EXT1|EXT2|EXT3|EXT4|EXT5|EXT6|EXT7|TOUT	EXT1
[:SENS[*c*]]:TOUT[:STAT] *mode* [:SENS[*c*]]:TOUT[:STAT]?	Enables or disables the trigger output for the status change between the trigger layer and the acquire device action. *mode* = 1|ON|0|OFF	OFF
(B2985B/B2987B) [:SENS[*c*]]:VOLT:ATT *att* [:SENS[*c*]]:VOLT:ATT?	Sets the attenuation level for the voltage input. When performing voltage measurement with the Keysight N1413A/N1414A adapter, you can set the proper attenuation level for the adapter using this command, in order to return actual voltage value. *att* = 1|100	1

Command	Summary	Reset setting
(B2985B/B2987B) [:SENS[*c*]]:VOLT[:DC]:GUAR *mode* [:SENS[*c*]]:VOLT[:DC]:GUAR?	Selects if the inner-shield of voltage input terminal connects to the Guard or Common. *mode* = 1|ON|0|OFF *mode* = 1 or ON connects to Guard *mode* = 0 or OFF connects to Common	ON
(B2985B/B2987B) [:SENS[*c*]]:WAIT:AUTO *mode* [:SENS[*c*]]:WAIT:AUTO?	Enables or disables the initial wait time used for calculating the measurement wait time for the specified channel. See [:SENS[c]]:WAIT[:STAT]. *mode* = 1|ON|0|OFF	ON
(B2985B/B2987B) [:SENS[*c*]]:WAIT:GAIN *gain* [:SENS[*c*]]:WAIT:GAIN? [DEFault|MINimum|MAXimum]	Sets the gain value used for calculating the measurement wait time for the specified channel. *gain* = MINimum|MAXimum|DEFault|0 to 100	1

Command	Summary	Reset setting
(B2985B/B2987B) [:SENS[*c*]]:WAIT:OFFS *offset* [:SENS[*c*]]:WAIT:OFFS? [DEFault|MINimum|MAXimum]	Sets the offset value used for calculating the measurement wait time for the specified channel. *offset* = MINimum|MAXimum|DEFault| 0 to 1 seconds	0
(B2985B/B2987B) [:SENS[*c*]]:WAIT[:STAT] *mode* [:SENS[*c*]]:WAIT[:STAT]?	Enables or disables the measurement wait time for the specified channel. The wait time is defined as the time the measurement channel cannot start measurement after the start of a DC output. mode = 0 or OFF disables the measurement wait time. The wait time is set to 0. mode = 1 or ON enables the measurement wait time given by the following formula. • [:SENS[c]]:WAIT:AUTO ON|1 condition: wait time = *gain* × initial wait time + *offset* • [:SENS[c]]:WAIT:AUTO OFF|0 condition: wait time = *offset* The initial wait time is automatically set by the instrument and cannot be changed.	ON

Measurement Ranges

Table 2-3 Current Measurement Range

Range value	Current measurement value	Resolution
2 pA	0 ≤ |I| ≤ 2.1 pA	1 aA
20 pA	0 ≤ |I| ≤ 21 pA	10 aA
200 pA	0 ≤ |I| ≤ 210 pA	100 aA
2 nA	0 ≤ |I| ≤ 2.1 nA	1 fA
20 nA	0 ≤ |I| ≤ 21 nA	10 fA
200 nA	0 ≤ |I| ≤ 210 nA	100 fA
2 μA	0 ≤ |I| ≤ 2.1 μA	1 pA
20 μA	0 ≤ |I| ≤ 21 μA	10 pA
200 μA	0 ≤ |I| ≤ 210 μA	100 pA
2 mA	0 ≤ |I| ≤ 2.1 mA	1 nA
20 mA	0 ≤ |I| ≤ 21 mA	10 nA

Table 2-4 Charge Measurement Range (B2985B/B2987B)

Range value	Charge measurement value	Resolution
2 nC	0 ≤ |Q| ≤ 2.1 nC	1 fC
20 nC	0 ≤ |Q| ≤ 21 nC	10 fC
200 nC	0 ≤ |Q| ≤ 210 nC	100 fC
2 μC	0 ≤ |Q| ≤ 2.1 μC	1 pC

Table 2-5 Voltage Measurement Range (B2985B/B2987B)

Range value	Voltage measurement value	Resolution
2 V	$0 \le \|V\| \le 2.1$ V	1 μV
20 V	$0 \le \|V\| \le 21$ V	10 μV

Table 2-6 Resistance Measurement Range, Value, and Resolution (B2985B/B2987B)

Range value	Current range used for measurement	Output value set to voltage source	Measurement value	Display resolution
1 MΩ	200 μA	20 V	100 kΩ ≤ |R|	1 Ω
10 MΩ	20 μA		1 MΩ ≤ |R|	10 Ω
100 MΩ	2 μA		10 MΩ ≤ |R|	100 Ω
1 GΩ	200 nA		100 MΩ ≤ |R|	1 kΩ
10 GΩ	20 nA		1 GΩ ≤ |R|	10 kΩ
100 GΩ	2 nA		10 GΩ ≤ |R|	100 kΩ
1 TΩ	2 nA	200 V	100 GΩ ≤ |R|	1 MΩ
10 TΩ	200 pA		1 TΩ ≤ |R|	10 MΩ
100 TΩ	20 pA		10 TΩ ≤ |R|	100 MΩ
1 PΩ	2 pA		100 TΩ ≤ |R|	1 GΩ

Setting Voltage Source

NOTE This section is not applicable to Keysight B2981B/B2983B which does not have Voltage Source.

Figure 2-1 Constant Output

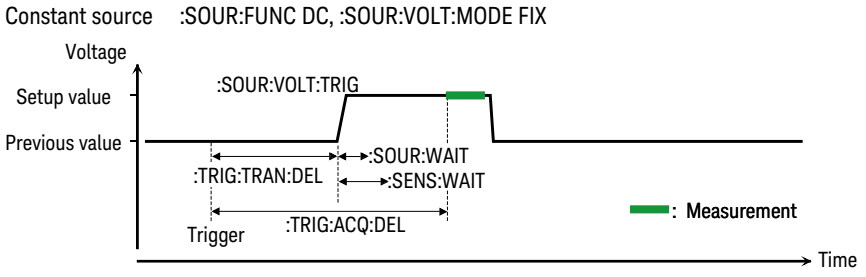


Figure 2-2 Arbitrary Waveforms

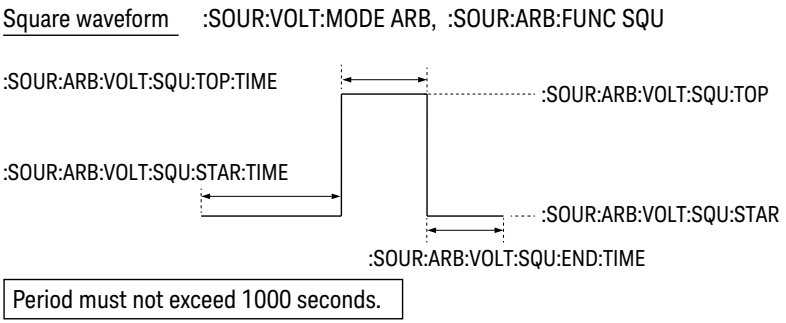


Figure 2-3 Variety of Sweep Outputs

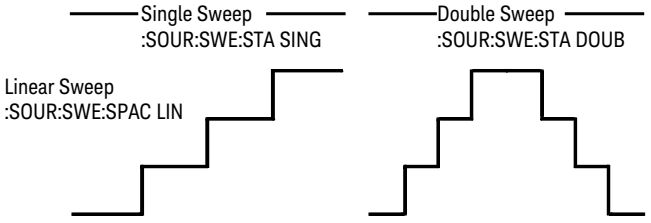


Figure 2-4 Staircase Sweep Output

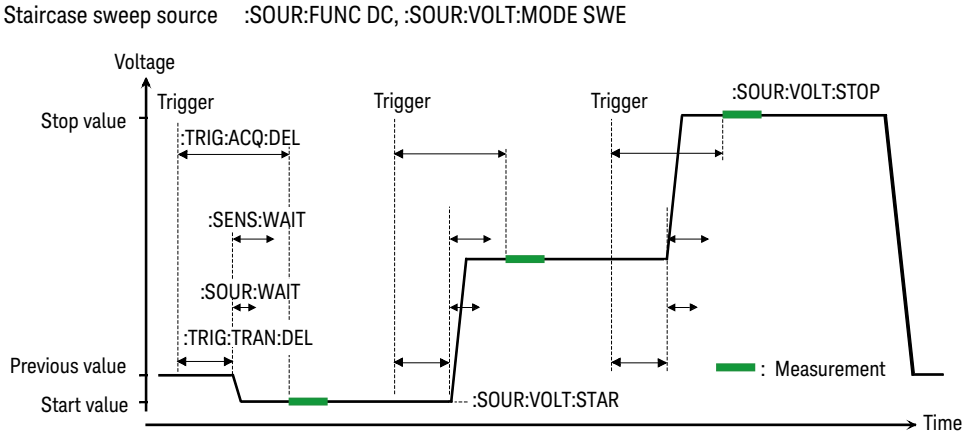


Table 2-7 OUTPut Subsystem

Command	Summary	Reset setting
:OUTP[*c*]:LOW *low_state* :OUTP[*c*]:LOW?	Selects the state of the low terminal. *low_state* = FLOat|COMMon	COMM
:OUTP[*c*]:OFF:MODE *mode* :OUTP[*c*]:OFF:MODE?	Selects the source condition after output off. *mode* = ZERO|HIZ|NORM	NORM
:OUTP[*c*][:STAT] *mode* :OUTP[*c*][:STAT]?	Enables or disables the source output. *mode* = 1|ON|0|OFF	OFF

Table 2-8 SOURce Subsystem

Command	Summary	Reset setting
:SOUR[*c*]:ARB:COUN *count* :SOUR[*c*]:ARB:COUN? [DEFault|MINimum|MAXimum]	Sets the number of arbitrary waveforms for the voltage output. *count* = DEFault|MINimum|MAXimum|INFinite| 1 to 100000	1
:SOUR[*c*]:ARB:FUNC[:SHAP] *shape* :SOUR[*c*]:ARB:FUNC[:SHAP]?	Selects the shape of the arbitrary waveform output. *shape* = SQUare	SQU
:SOUR[*c*]:ARB:VOLT:SQU:END:TIME *time* :SOUR[*c*]:ARB:VOLT:SQU:END:TIME? [DEFault|MINimum|MAXimum]	Set the end time of the square waveform output. Waveform period must not exceed 1000 seconds. *time* = MINimum|MAXimum|DEFault|0 to 1000 seconds.	0
:SOUR[*c*]:ARB:VOLT:SQU:STAR[:LEV] *level* :SOUR[*c*]:ARB:VOLT:SQU:STAR[:LEV]? [DEFault|MINimum|MAXimum]	Set the start level of the square waveform output. *level* = MINimum|MAXimum|DEFault|minimum to maximum source value, in V. See "Voltage Output Ranges" on page 72.	0
:SOUR[*c*]:ARB:VOLT:SQU:STAR:TIME *time* :SOUR[*c*]:ARB:VOLT:SQU:STAR:TIME? [DEFault|MINimum|MAXimum]	Set the start time of the square waveform output. Waveform period must not exceed 1000 seconds. *time* = MINimum|MAXimum|DEFault|1E–4 to 1000 seconds.	1.00E–04 seconds
:SOUR[*c*]:ARB:VOLT:SQU:TOP[:LEV] *level* :SOUR[*c*]:ARB:VOLT:SQU:TOP[:LEV]? [DEFault|MINimum|MAXimum]	Set the top level of the square waveform output. *level* = MINimum|MAXimum|DEFault|minimum to maximum source value, in V. See "Voltage Output Ranges" on page 72.	0

Command	Summary	Reset setting
:SOUR[*c*]:ARB:VOLT:SQU:TOP:TIME *time* :SOUR[*c*]:ARB:VOLT:SQU:TOP:TIME? [DEFault|MINimum|MAXimum]	Set the top time of the square waveform output. Waveform period must not exceed 1000 seconds. *time* = MINimum|MAXimum|DEFault|1E–4 to 1000 seconds.	1.00E–04 seconds
:SOUR:DIG:DATA *data* :SOUR:DIG:DATA?	Sets the output data to the GPIO pins (digital control port) and read data from the GPIO pins. *data* = 0 to 127	
:SOUR:DIG:EXT[*n*][:FUNC] *function* :SOUR:DIG:EXT[*n*][:FUNC]?	Assigns the input/output function to the specified GPIO pin. *function* = DIO|DINPut|TINPut| TOUT	DINP
:SOUR:DIG:EXT[*n*]:POL *polarity* :SOUR:DIG:EXT[*n*]:POL?	Sets the polarity of the input/output function for the specified GPIO pin. *polarity* = NEG|POS	NEG
:SOUR:DIG:EXT[*n*]:TOUT[:EDGE]:POS *position* :SOUR:DIG:EXT[*n*]:TOUT[:EDGE]:POS?	Selects the trigger output timing for the specified GPIO pin. *position* = BEFore|AFTer|BOTH	BOTH
:SOUR:DIG:EXT[*n*]:TOUT[:EDGE]:WIDT *width* :SOUR:DIG:EXT[*n*]:TOUT[:EDGE]:WIDT? [DEFault|MINimum|MAXimum]	Sets the pulse width of the output trigger for the specified GPIO pin. *width* = MINimum|MAXimum|DEFault| 1E–5 to 1E–2 seconds	1.00E–04 seconds
:SOUR:DIG:EXT[*n*]:TOUT:TYPE *type* :SOUR:DIG:EXT[*n*]:TOUT:TYPE?	Selects the output trigger type for the specified GPIO pin. *type* = EDGE|LEVel	EDGE
:SOUR:DIG:INT[*i*]:TOUT[:EDGE]:POS *position* :SOUR:DIG:INT[*i*]:TOUT[:EDGE]:POS?	Selects the trigger output timing for the internal trigger line 1 or 2. *position* = BEFore|AFTer|BOTH	BOTH

Command	Summary	Reset setting
:SOUR[*c*]:FUNC:MODE *mode* :SOUR[*c*]:FUNC:MODE?	Selects the source output mode of the specified channel. *mode* = VOLTage	VOLT
:SOUR[*c*]:FUNC:TRIG:CONT *mode* :SOUR[*c*]:FUNC:TRIG:CONT?	Enables or disables continuous trigger output for the specified channel. *mode* = 0|OFF|1|ON	OFF
:SOUR[*c*]:LIST:VOLT *list* :SOUR[*c*]:LIST:VOLT?	Sets the source output voltage data for the specified channel. *list*: List of the output voltage data. Maximum of 100000 data can be set to *list*. Each data must be separated by a comma.	0
:SOUR[*c*]:LIST:VOLT:APP *append_list*	Adds the source output voltage data to the end of the list set by the :SOUR[*c*]:LIST:VOLT command, to which some data might be appended to by this command. Total number of data in the list must be ≤ 100000. *append_list*: List of the output voltage data. Multiple data can be set to *append_list*. Each data must be separated by a comma.	
:SOUR[*c*]:LIST:VOLT:POIN?	Returns the number of data in the list set by the :SOUR[*c*]:LIST:VOLT command, to which some data might be appended to by the :SOUR[*c*]:LIST:VOLT:APP command.	
:SOUR[*c*]:LIST:VOLT:STAR *start* :SOUR[*c*]:LIST:VOLT:STAR?	Specifies the list sweep start point by using the index of the list. *start*: Index of the list. 1 to 100000. *start* = 1 indicates the first data in the list (top of the list). *start* = 0 or the value greater than 100000 causes an error.	1

Command	Summary	Reset setting
:SOUR[*c*]:SWE:DIR *direction* :SOUR[*c*]:SWE:DIR?	Sets the sweep direction, UP or DOWN, for the specified channel. *direction* = DOWN|UP	UP
:SOUR[*c*]:SWE:POIN *points* :SOUR[*c*]:SWE:POIN? DEFault|MINimum|MAXimum	Sets the number of sweep steps for the specified channel. This command setting is effective for both current sweep and voltage sweep. *points* = DEFault|MINimum|MAXimum| 1 to 100000 The points value can be expressed by the following formula, using the step value set by the :SOUR[*c*]:VOLT:STEP command and the span value set by the :SOUR[*c*]:VOLT:SPAN command. *points* = *span*/*step* + 1 (where *step* is not 0) *points* = 1 sets *step* = 0.	1
:SOUR[*c*]:SWE:RANG *mode* :SOUR[*c*]:SWE:RANG?	Selects the output ranging mode of the sweep output for the specified channel. *mode* = BEST|FIXed	BEST
:SOUR[*c*]:SWE:SPAC *mode* :SOUR[*c*]:SWE:SPAC?	Selects the scale of the sweep output for the specified channel. See Figure 2-3. *mode* = LINear	LIN
:SOUR[*c*]:SWE:STA *mode* :SOUR[*c*]:SWE:STA?	Sets the sweep mode for the specified channel. *mode* = SINGle|DOUBle	SING

Command	Summary	Reset setting
:SOUR[*c*]:TOUT:SIGN *output*{,output} :SOUR[*c*]:TOUT:SIGN *output*{,output} :SOUR[*c*]:TOUT:SIGN *output*{,output} :SOUR[*c*]:TOUT:SIGN? :SOUR[*c*]:TOUT:SIGN?	Selects the trigger output for the status change between the trigger layer and the transient device action. *output* = INT1|INT2|LAN|EXT1|EXT2|EXT3|EXT4|EXT5|EXT6|EXT7|TOUT	EXT1
:SOUR[*c*]:TOUT[:STAT] *mode* :SOUR[*c*]:TOUT[:STAT] *mode* :SOUR[*c*]:TOUT[:STAT] *mode* :SOUR[*c*]:TOUT[:STAT]? :SOUR[*c*]:TOUT[:STAT]?	Enables or disables the trigger output for the status change between the trigger layer and the transient device action. *mode* = 1|ON|0|OFF	OFF
:SOUR[*c*]:VOLT:CENT *data* :SOUR[*c*]:VOLT:CENT? [DEFault|MINimum|MAXimum]	Sets the center or span value of the voltage sweep output. *data* = MINimum|MAXimum|DEFault| minimum to maximum source value, in V. See "Voltage Output Ranges" on page 72. The center and span values can be expressed by the following formula, using the start and stop values set by the :SOUR[*c*]:VOLT:<STAR|STOP> command. So the last command setting is effective for these sweep parameters. *center* = (*start* + *stop*)/2 *span* = *stop* – *start*	0
:SOUR[*c*]:VOLT:SPAN *data* :SOUR[*c*]:VOLT:SPAN? [DEFault|MINimum|MAXimum]		0
:SOUR[*c*]:VOLT[:LEV][:IMM][:AMPL] *level* :SOUR[*c*]:VOLT[:LEV][:IMM][:AMPL]? [DEFault|MINimum|MAXimum]	Changes the output level of the specified source channel immediately. *level* = MINimum|MAXimum|DEFault| minimum to maximum source value, in V. See "Voltage Output Ranges" on page 72.	0

Command	Summary	Reset setting
:SOUR[*c*]:VOLT[:LEV]:TRIG[:AMPL] *level* :SOUR[*c*]:VOLT[:LEV]:TRIG[:AMPL]? [DEFault|MINimum|MAXimum]	Changes the output level of the specified source channel immediately by receiving a trigger from the trigger source set by the :TRIG[c]<:TRAN|[:ALL]>:SOUR command. *level* = MINimum|MAXimum|DEFault| minimum to maximum source value, in V. See "Voltage Output Ranges" on page 72.	0
:SOUR[*c*]:VOLT:MODE *mode* :SOUR[*c*]:VOLT:MODE?	Selects the source mode, arbitrary waveform, fixed, sweep, or list sweep of the specified source channel. *mode* = ARB|FIXed|SWEep|LIST	FIX
:SOUR[*c*]:VOLT:POIN *points* :SOUR[*c*]:VOLT:POIN? [DEFault| MINimum|MAXimum]	Sets the number of sweep steps for the voltage sweep output. *points* = MINimum|MAXimum|DEFault| 1 to 100000 The points value can be expressed by the following formula, using the step value set by the :SOUR[*c*]:VOLT:STEP command and the span value set by the :SOUR[*c*]:VOLT:SPAN command. *points* = *span/step* + 1 (where *step* is not 0) *points* = 1 sets *step* = 0.	1
:SOUR[*c*]:VOLT:RANG *range* :SOUR[*c*]:VOLT:RANG?	Sets the voltage output range of the specified source channel. This command is effective when the automatic ranging function is off. *range* = MINimum|MAXimum|DEFault| minimum to maximum source value, in V. See "Voltage Output Ranges" on page 72.	20 V for voltage range

Command	Summary	Reset setting
:SOUR[*c*]:VOLT:RLIM:STAT *mode* :SOUR[*c*]:VOLT:RLIM:STAT?	Selects if 20 MΩ current-limiting resistor connects to HI of Voltage Output in series or not. *mode* = 1|ON|0|OFF	OFF
:SOUR[*c*]:VOLT:STAR *data* :SOUR[*c*]:VOLT:STAR? [DEFault|MINimum|MAXimum]	Sets the start or stop value for the voltage sweep output. *data* = MINimum|MAXimum|DEFault| minimum to maximum source value, in V. See "Voltage Output Ranges" on page 72. The start and stop values can be expressed by the following formula, using the center and span values set by the :SOUR[*c*]:VOLT:<CENT|SPAN> command. So the last command setting is effective for these sweep parameters. *start* = *center* – *span*/2 *stop* = *center* + *span*/2	0
:SOUR[*c*]:VOLT:STOP *data* :SOUR[*c*]:VOLT:STOP? [DEFault|MINimum|MAXimum]		0
:SOUR[*c*]:VOLT:STEP *step* :SOUR[*c*]:VOLT:STEP? [DEFault|MINimum|MAXimum]	Sets the sweep step value of the voltage sweep output. *step* = MINimum|MAXimum|DEFault| minimum to maximum source value, in V. See "Voltage Output Ranges" on page 72. The step value can be expressed by the following formula, using the points value set by the :SOUR[*c*]:VOLT:POIN command and the span value set by the :SOUR[*c*]:VOLT:SPAN command. *step* = *span*/(*points* – 1) (where *points* is not 1) *points* = 1 sets *step* = 0.	0

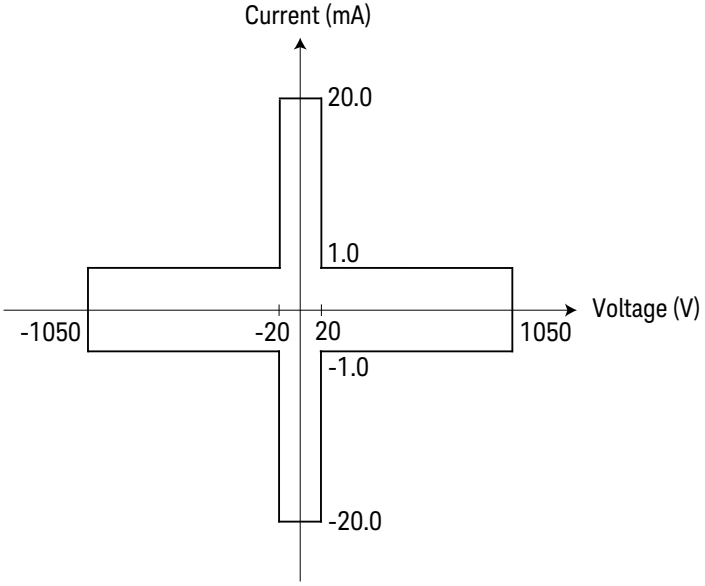
Command	Summary	Reset setting
:SOUR[*c*]]:WAIT:AUTO *mode* :SOUR[*c*]]:WAIT:AUTO?	Enables or disables the initial wait time used for calculating the source wait time for the specified channel. See :SOUR[c]:WAIT[:STAT]. *mode* = 1|ON|0|OFF	ON
:SOUR[*c*]:WAIT:GAIN *gain* :SOUR[*c*]:WAIT:GAIN? [DEFault|MINimum|MAXimum]	Sets the gain value used for calculating the source wait time for the specified channel. *gain* = MINimum|MAXimum|DEFault| 0 to 100	1
:SOUR[*c*]:WAIT:OFFS *offset* :SOUR[*c*]:WAIT:OFFS? [DEFault|MINimum|MAXimum]	Sets the offset value used for calculating the source wait time for the specified channel. *offset* = MINimum|MAXimum|DEFault| 0 to 1 seconds	0
:SOUR[*c*]:WAIT[:STAT] *mode* :SOUR[*c*]:WAIT[:STAT]?	Enables or disables the source wait time for the specified channel. The wait time is defined as the time the source channel cannot change the output after the start of a DC output. mode = 0 or OFF disables the source wait time. The wait time is set to 0. mode = 1 or ON enables the source wait time given by the following formula. • :SOUR[c]:WAIT:AUTO ON|1 condition: wait time = *gain* × initial wait time + *offset* • :SOUR[c]:WAIT:AUTO OFF|0 condition: wait time = *offset* The initial wait time is automatically set by the instrument and cannot be changed.	ON

Voltage Output Ranges

Table 2-9 Voltage Output Range

Range value	Setting resolution	DC output voltage	Maximum current
20 V	700 μV	0 V < |V| ≤ 21 V	±20 mA
1000 V	35 mV	–1 V ≤ V ≤ 1000 V	±1.0 mA
–1000 V	35 mV	–1000 V ≤ V ≤ 1 V	±1.0 mA

Figure 2-5 Maximum Voltage and Current



Controlling Source/Measure Trigger

NOTE Transient action is not applicable to Keysight B2981B/B2983B which does not have Voltage Source.

Figure 2-6 Transient and Acquire Device Actions

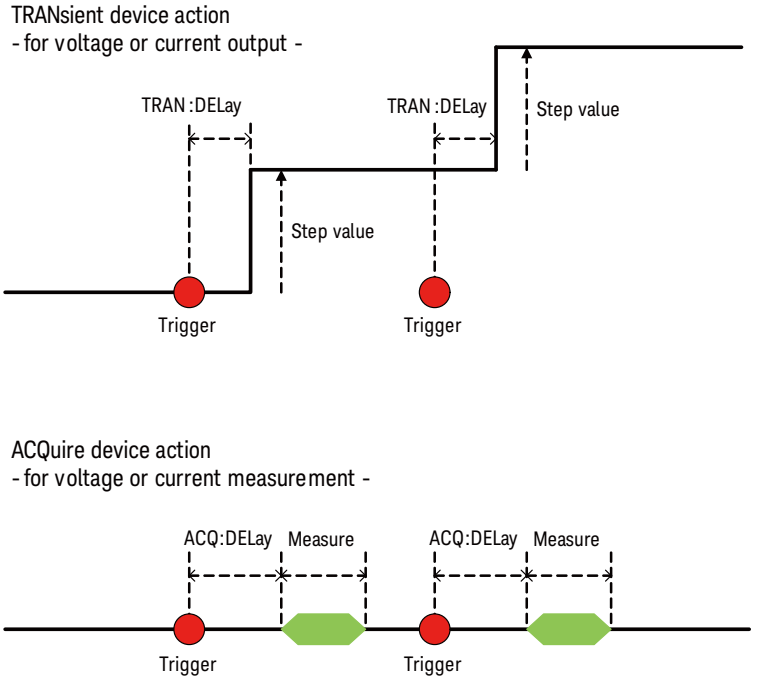


Figure 2-7 Operation Example Using Trigger Delay and AINT Trigger Source

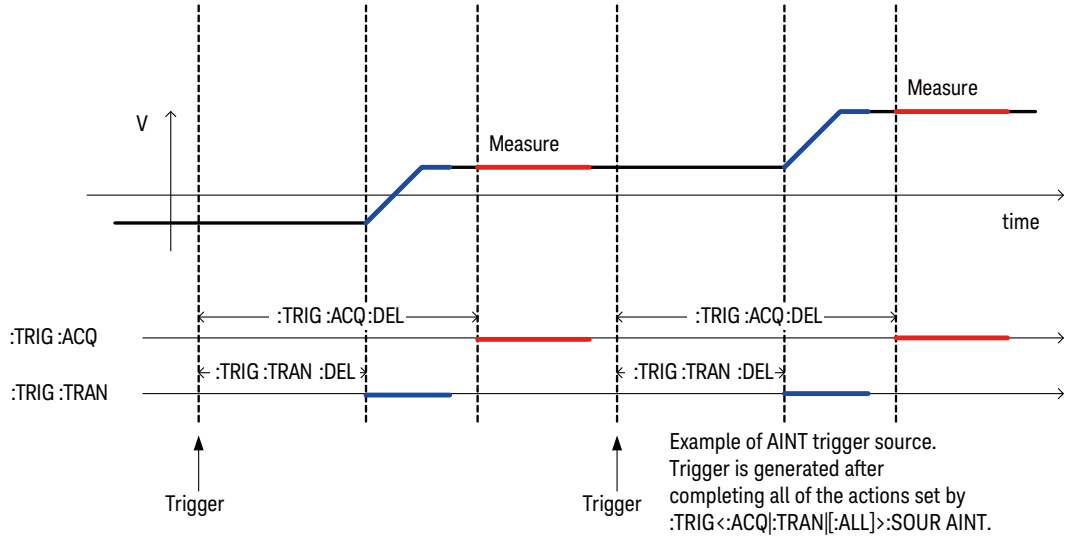


Figure 2-8 B2980 Trigger System

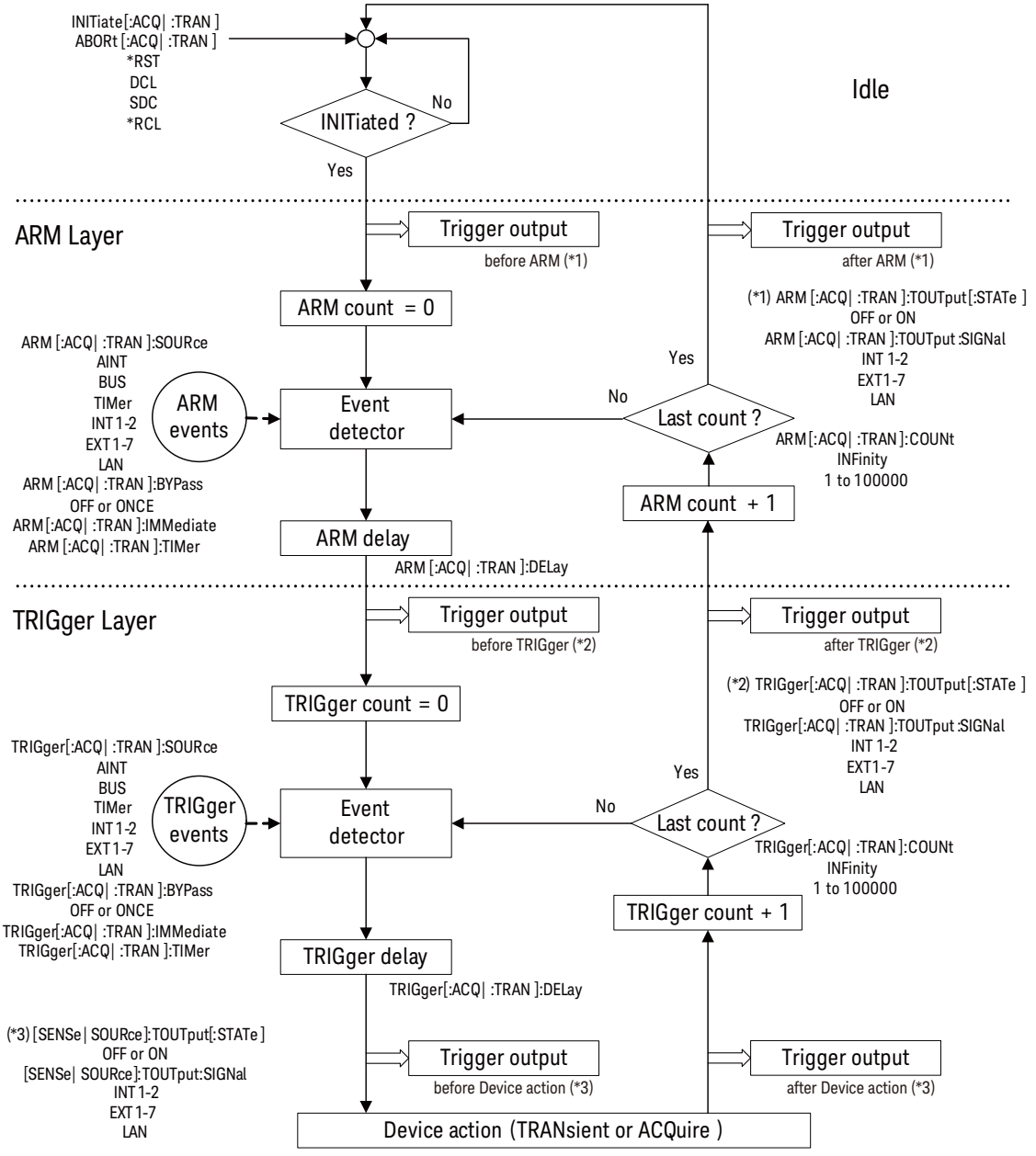


Table 2-10 TRIGger Subsystem

Command	Summary	Reset setting
:ABOR:ACQ [*chanlist*] :ABOR[:ALL] [*chanlist*]	Aborts the specified device action for the specified channel. Trigger status is changed to idle. *chanlist* = (@1)	
(B2985B/B2987B) :ABOR:TRAN [*chanlist*]		
:ARM:ACQ[:IMM] [*chanlist*] :ARM[:ALL][:IMM] [*chanlist*]	Sends an immediate arm trigger for the specified device action to the specified channel. When the status of the specified device action is initiated, the arm trigger causes a layer change from arm to trigger. *chanlist* = (@1)	
(B2985B/B2987B) :ARM:TRAN[:IMM] [*chanlist*]		
:ARM[*c*]:ACQ[:LAY]:BYP *bypass* :ARM[*c*][:ALL][:LAY]:BYP *bypass* :ARM[*c*]:ACQ[:LAY]:BYP?	Enables or disables a bypass for the event detector in the arm layer. *bypass* = ONCE|OFF ONCE enables the bypass only for the first passage.	OFF
(B2985B/B2987B) :ARM[*c*]:TRAN[:LAY]:BYP *bypass* :ARM[*c*]:TRAN[:LAY]:BYP?		
:ARM[*c*]:ACQ[:LAY]:COUN *count* :ARM[*c*][:ALL][:LAY]:COUN *count* :ARM[*c*]:ACQ[:LAY]:COUN? [*count*] :ARM[*c*][:ALL][:LAY]:COUN? *count*	Sets the arm count for the specified device action. *count* = INFinity|MINimum|MAXimum| DEFault| 1 to 100000 or 2147483647 *count* = 2147483647 indicates infinity. Query does not support *count* = INFinity, 1 to 100000 and 2147483647. *Arm count* × *Trigger count* must be less than 100001.	1
(B2985B/B2987B) :ARM[*c*]:TRAN[:LAY]:COUN *count* :ARM[*c*]:TRAN[:LAY]:COUN? [*count*]		

Command	Summary	Reset setting
:ARM[*c*]:ACQ[:LAY]:DEL *delay* :ARM[*c*][:ALL][:LAY]:DEL *delay* :ARM[*c*]:ACQ[:LAY]:DEL? [*delay*] :ARM[*c*][:ALL][:LAY]:DEL? *delay*	Sets the arm delay for the specified device action. *delay* = MINimum|MAXimum|DEFault| 0 to 100000 seconds Query does not support *delay* = 0 to 100000.	0
(B2985B/B2987B) :ARM[*c*]:TRAN[:LAY]:DEL *delay* :ARM[*c*]:TRAN[:LAY]:DEL? [*delay*]		
:ARM[*c*]:ACQ[:LAY]:SOUR:LAN *lan_id*{,*lan_id*} :ARM[*c*][:ALL][:LAY]:SOUR:LAN *lan_id*{,*lan_id*} :ARM[*c*]:ACQ[:LAY]:SOUR:LAN?	Specifies one or more LXI triggers used for the arm source for the specified device action. *lan_id* = LAN0|LAN1|LAN2|LAN3|LAN4| LAN5|LAN6|LAN7	All is selected.
(B2985B/B2987B) :ARM[*c*]:TRAN[:LAY]:SOUR:LAN *lan_id*{,*lan_id*} :ARM[*c*]:TRAN[:LAY]:SOUR:LAN?		
:ARM[*c*]:ACQ[:LAY]:SOUR[:SIGN] *source* :ARM[*c*][:ALL][:LAY]:SOUR[:SIGN] *source* :ARM[*c*]:ACQ[:LAY]:SOUR[:SIGN]?	Selects the arm source for the specified device action. *source* = AINT|BUS|TIMer|INT1|INT2|LAN| EXT1|EXT2|EXT3|EXT4|EXT5| EXT6|EXT7|TIN	AINT
(B2985B/B2987B) :ARM[*c*]:TRAN[:LAY]:SOUR[:SIGN] *source* :ARM[*c*]:TRAN[:LAY]:SOUR[:SIGN]?		

Command	Summary	Reset setting
:ARM[*c*]:ACQ[:LAY]:TIM *interval* :ARM[*c*][:ALL][:LAY]:TIM *interval* :ARM[*c*]:ACQ[:LAY]:TIM? [*interval*] :ARM[*c*][:ALL][:LAY]:TIM? *interval*	Sets the interval of the TIMer arm source for the specified device action. *interval* = MINimum|MAXimum|DEFault|1E–5 to 1E+5 seconds For TRAN and ALL of B2985B/B2987B: *interval* = MINimum|MAXimum|DEFault|1E–4 to 1E+5 seconds Query does not support *interval* = 1E–5 to 1E+5.	1E–4 seconds
(B2985B/B2987B) :ARM[*c*]:TRAN[:LAY]:TIM *interval* :ARM[*c*]:TRAN[:LAY]:TIM? [*interval*]		
:ARM[*c*]:ACQ[:LAY]:TOUT:SIGN *output*{,output} :ARM[*c*][:ALL][:LAY]:TOUT:SIGN *output*{,output} :ARM[*c*]:ACQ[:LAY]:TOUT:SIGN?	Selects the trigger output for the status change between the idle state and the arm layer. *output* = INT1|INT2|LAN|EXT1|EXT2|EXT3|EXT4|EXT5|EXT6|EXT7|TOUT	EXT1
(B2985B/B2987B) :ARM[*c*]:TRAN[:LAY]:TOUT:SIGN *output*{,output} :ARM[*c*]:TRAN[:LAY]:TOUT:SIGN?		
:ARM[*c*]:ACQ[:LAY]:TOUT[:STAT] *mode* :ARM[*c*][:ALL][:LAY]:TOUT[:STAT] *mode* :ARM[*c*]:ACQ[:LAY]:TOUT[:STAT]?	Enables or disables the trigger output for the status change between the idle state and the arm layer. *mode* = 1|ON|0|OFF	OFF
(B2985B/B2987B) :ARM[*c*]:TRAN[:LAY]:TOUT[:STAT] *mode* :ARM[*c*]:TRAN[:LAY]:TOUT[:STAT]?		
:IDLE[*c*]:ACQ? :IDLE[*c*][:ALL]?	Checks the status of the specified device action for the specified channel, and waits until the status is changed to idle.	
(B2985B/B2987B) :IDLE[*c*]:TRAN?		

Command	Summary	Reset setting
:INIT[:IMM]:ACQ [*chanlist*] :INIT[:IMM][:ALL] [*chanlist*]	Initiates the specified device action for the specified channel. Trigger status is changed from idle to initiated. *chanlist* = (@1)	
(B2985B/B2987B) :INIT[:IMM]:TRAN [*chanlist*]		
:TRIG[*c*]:ACQ:BYP *bypass* :TRIG[*c*][:ALL]:BYP *bypass* :TRIG[*c*]:ACQ:BYP?	Enables or disables a bypass for the event detector in the trigger layer. *bypass* = ONCE|OFF ONCE enables the bypass only for the first passage.	OFF
(B2985B/B2987B) :TRIG[*c*]:TRAN:BYP *bypass* :TRIG[*c*]:TRAN:BYP?		
:TRIG[*c*]:ACQ:COUN *count* :TRIG[*c*][:ALL]:COUN *count* :TRIG[*c*]:ACQ:COUN? [*count*] :TRIG[*c*][:ALL]:COUN? *count*	Sets the trigger count for the specified device action. *count* = INFinity|MINimum|MAXimum| DEFault| 1 to 100000 or 2147483647 *count* = 2147483647 indicates infinity. Query does not support *count* = INFinity, 1 to 100000 and 2147483647. *Arm count* × *Trigger count* must be less than 100001.	1
(B2985B/B2987B) :TRIG[*c*]:TRAN:COUN *count* :TRIG[*c*]:TRAN:COUN? [*count*]		
:TRIG[*c*]:ACQ:DEL *delay* :TRIG[*c*][:ALL]:DEL *delay* :TRIG[*c*]:ACQ:DEL? [*delay*] :TRIG[*c*][:ALL]:DEL? *delay*	Sets the trigger delay for the specified device action. *delay* = MINimum|MAXimum|DEFault| 0 to 100000 seconds Query does not support *delay* = 0 to 100000.	0
(B2985B/B2987B) :TRIG[*c*]:TRAN:DEL *delay* :TRIG[*c*]:TRAN:DEL? [*delay*]		

Command	Summary	Reset setting
:TRIG:ACQ[:IMM] [*chanlist*] :TRIG[:ALL][:IMM] [*chanlist*]	Sends an immediate trigger for the specified device action to the specified channel. When the status of the specified device action is initiated, the trigger causes the specified device action. *chanlist* = (@1)	
(B2985B/B2987B) :TRIG:TRAN[:IMM] [*chanlist*]		
:TRIG[*c*]:ACQ[:LAY]:SOUR:LAN *lan_id*{,*lan_id*} :TRIG[*c*][:ALL][:LAY]:SOUR:LAN *lan_id*{,*lan_id*} :TRIG[*c*]:ACQ[:LAY]:SOUR:LAN?	Specifies one or more LXI triggers used for the trigger source for the specified device action. *lan_id* = LAN0|LAN1|LAN2|LAN3|LAN4|LAN5|LAN6|LAN7	All is selected.
(B2985B/B2987B) :TRIG[*c*]:TRAN[:LAY]:SOUR:LAN *lan_id*{,*lan_id*} :TRIG[*c*]:TRAN[:LAY]:SOUR:LAN?		
:TRIG[*c*]:ACQ:SOUR[:SIGN] *source* :TRIG[*c*][:ALL]:SOUR[:SIGN] *source* :TRIG[*c*]:ACQ:SOUR[:SIGN]?	Selects the trigger source for the specified device action. *source* = AINT|BUS|TIMer|INT1|INT2|LAN|EXT1|EXT2|EXT3|EXT4|EXT5| EXT6|EXT7|TIN	AINT
(B2985B/B2987B) :TRIG[*c*]:TRAN:SOUR[:SIGN] *source* :TRIG[*c*]:TRAN:SOUR[:SIGN]?		
:TRIG[*c*]:ACQ:TIM *interval* :TRIG[*c*][:ALL]:TIM *interval* :TRIG[*c*]:ACQ:TIM? [*interval*] :TRIG[*c*][:ALL]:TIM? *interval*	Sets the interval of the TIMer trigger source for the specified device action. *interval* = MINimum|MAXimum|DEFault|1E–5 to 1E+5 seconds For TRAN and ALL of B2985B/B2987B: *interval* = MINimum|MAXimum|DEFault|1E–4 to 1E+5 seconds Query does not support *interval* = 1E–5 to 1E+5.	1E–4 seconds
(B2985B/B2987B) :TRIG[*c*]:TRAN:TIM *interval* :TRIG[*c*]:TRAN:TIM? [*interval*]		

Command	Summary	Reset setting
:TRIG[*c*]:ACQ:TOUT:SIGN *output*{,output} :TRIG[*c*][:ALL]:TOUT:SIGN *output*{,output} :TRIG[*c*]:ACQ:TOUT:SIGN?	Selects the trigger output for the status change between the arm layer and the trigger layer. *output* = INT1|INT2|LAN|EXT1|EXT2|EXT3|EXT4|EXT5|EXT6|EXT7|TOUT	EXT1
(B2985B/B2987B) :TRIG[*c*]:TRAN:TOUT:SIGN *output*{,output} :TRIG[*c*]:TRAN:TOUT:SIGN?		
:TRIG[*c*]:ACQ:TOUT[:STAT] *mode* :TRIG[*c*][:ALL]:TOUT[:STAT] *mode* :TRIG[*c*]:ACQ:TOUT[:STAT]?	Enables or disables the trigger output for the status change between the arm layer and the trigger layer. *mode* = 1|ON|0|OFF	OFF
(B2985B/B2987B) :TRIG[*c*]:TRAN:TOUT[:STAT] *mode* :TRIG[*c*]:TRAN:TOUT[:STAT]?		

LXI Trigger Events

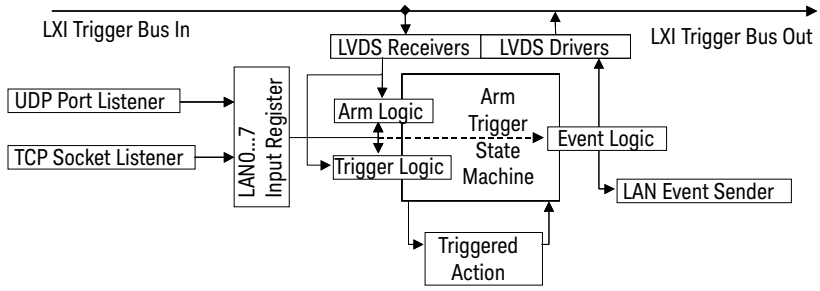
B2980 provides a subset of the LXI Trigger Events (IVI-3.15 IviLxiSync) functionality in the trigger system.

- Device Model

  Figure 2-9 shows the high-level LXI device model defined in IVI-3.15. B2980 does not have the LXI Trigger Bus, but has the UDP Port/TCP Socket Listener and the LAN Event Sender in the system.

  You can configure the trigger systems to send/receive LAN*n* (*n*: 0 to 7) trigger events by the instrument specific trigger event, slope, drive logic, destination, and filter.

Figure 2-9 High-Level LXI Device Model



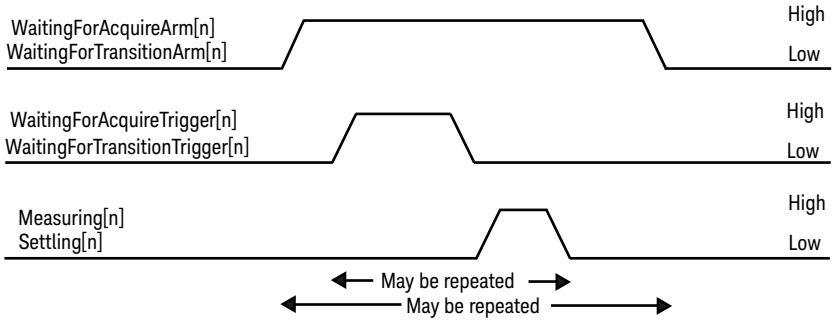
- Instrument Specific Events

  B2980 has the ARM-TRIGger model for each channel and actions (transition and acquire), and provides following events.

  - WaitingForAcquireArm1
  - WaitingForAcquireTrigger1
  - WaitingForTransitionArm1
  - WaitingForTransitionTrigger1
  - Measuring1
  - Settling1

  All events can be configured by the signal level (or edge), destination, and other parameters defined in IVI 3.15.

Figure 2-10 Trigger State Machine Signal Relationships



- Limitations

  LXI trigger event functions provided by B2980 are a subset of the IEEE-1588 required by LXI Class-B. The following limitations exist for B2980.

  - Timestamp in the event are ignored. (immediate trigger only)
  - Delay and other timing parameters cannot be set. (always 0)
  - It is not allowed to add/delete any events.
  - The :ARM:LXI:COUNt command is not effective. (ignored)

Table 2-11 LXI Subsystem

Command	Summary	Reset setting
:ARM:LXI:COUN *intRepetitions* :ARM:LXI:COUN?	This command is ignored.	0
:ARM:LXI:DEL *delay* :ARM:LXI:DEL?	Delay time must be zero. *delay*: Delay time, in seconds	0
:ARM:LXI:LAN[:SET]:DET *event*,*detect* :ARM:LXI:LAN[:SET]:DET? *event*	Specifies the style of arm source detection for the specified event. *event*: LAN event name *detect* = RISE|FALL|HIGH|LOW	RISE

Command	Summary	Reset setting
:ARM:LXI:LAN[:SET]:ENAB *event*,*status* :ARM:LXI:LAN[:SET]:ENAB? *event*	Enables or disables the arm source of the specified event. *event*: LAN event name *status* = 0|OFF|1|ON	OFF
:ARM:LXI:LAN[:SET]:FILT *event*,*filter* :ARM:LXI:LAN[:SET]:FILT? *event*	Specifies a filter for restricting arm sources of the specified event. *event*: LAN event name *filter*: Filter. It will be an existing LAN arm sources, e.g. one of the items returned by :LXI:EVEN:INP:LAN:LIST? or :LXI:EVEN[:OUTP]:LAN:LIST?.	"ALL:5044"
:ARM:LXI:LAN[:SET]:IDEN *event*,*id* :ARM:LXI:LAN[:SET]:IDEN? *event*	Specifies the LAN event identifier that is associated with this arm source. *event*: LAN event name *id*: Identifier	*event*
:LXI:EVEN:DOM *domain* :LXI:EVEN:DOM?	Specifies the LXI LAN domain. *domain* = 0 to 255	0
:LXI:EVEN:INP:LAN:ADD *event*	This command is ignored.	
:LXI:EVEN:INP:LAN:COUN?	Returns an integer as the total number of defined input LAN events (includes both enabled and disabled events).	8
:LXI:EVEN:INP:LAN:DIS:ALL	Disables all input events.	
:LXI:EVEN:INP:LAN:LIST?	Returns a quoted string with the list of defined input event names.	
:LXI:EVEN:INP:LAN:REM:ALL	This command is ignored.	
:LXI:EVEN:INP:LAN:REM *event*	This command is ignored.	

Command	Summary	Reset setting
:LXI:EVEN:INP:LAN[:SET]:CONF *event*,*status*, *detect*[,*delay*],*filter*,*id*	Configures the most common attributes of LXI LAN input events. *event*: LAN event name *status*: Status *detect*: Detection type *delay*: Delay time *filter*: Filter *id*: Identifier	
:LXI:EVEN:INP:LAN[:SET]:DEL *event*,*delay* :LXI:EVEN:INP:LAN[:SET]:DEL? *event*	Delay time must be zero. *event*: LAN event name *delay*: Delay time, in seconds	0
:LXI:EVEN:INP:LAN[:SET]:DET *event*,*detect* :LXI:EVEN:INP:LAN[:SET]:DET? *event*	Specifies the trigger detection method and polarity for the input event. *event*: LAN event name *detect* = RISE|FALL|HIGH|LOW	RISE
:LXI:EVEN:INP:LAN[:SET]:ENAB *event*,*status* :LXI:EVEN:INP:LAN[:SET]:ENAB? *event*	Enables or disables the specified input event. *event*: LAN event name *status* = 0|OFF|1|ON	OFF
:LXI:EVEN:INP:LAN[:SET]:FILT *event*,*filter* :LXI:EVEN:INP:LAN[:SET]:FILT? *event*	Creates a filter for incoming input events. *event*: LAN event name *filter*: Filter. It will be an existing LAN event, e.g. one of the items returned by :LXI:EVEN:INP:LAN:LIST?.	"ALL:5044"

Command	Summary	Reset setting
:LXI:EVEN:INP:LAN[:SET]:IDEN *event*,*id* :LXI:EVEN:INP:LAN[:SET]:IDEN? *event*	Specifies the string that is expected to arrive over the LAN for a given input event to occur. *event*: LAN event name *id*: Identifier	*event*
:LXI:EVEN:LOG:ALL?	Returns the contents of the event log.	
:LXI:EVEN:LOG:CIRC[:ENAB] *status* :LXI:EVEN:LOG:CIRC[:ENAB]?	Selects how new entries are handled when the LXI event log is full. *status* = 0|OFF|1|ON	ON
:LXI:EVEN:LOG:CIRC:FBE	Selects the most recently added event log entry to be used as the reference for :LXI:EVEN:LOG:ENTR?.	
:LXI:EVEN:LOG:CLE	Removes all existing entries from the event log.	
:LXI:EVEN:LOG:COUN?	Returns an integer as the total number of entries in the LXI event log.	
:LXI:EVEN:LOG:ENAB *status* :LXI:EVEN:LOG:ENAB?	Enables or disables LXI event logging. *status* = 0|OFF|1|ON	ON
:LXI:EVEN:LOG:ENTR? *index*	Retrieves the event log entry referenced by *index*. *index* = 0 to 2147483647	
:LXI:EVEN:LOG:SIZE *size* :LXI:EVEN:LOG:SIZE?	Sets the maximum number of entries the LXI event log can hold. *size* = 1to 200	100
:LXI:EVEN[:OUTP]:LAN:ADD *event*	This command is ignored.	
:LXI:EVEN[:OUTP]:LAN:COUN?	Returns an integer as the number of configured LXI output LAN events.	
:LXI:EVEN[:OUTP]:LAN:DIS:ALL	Disables all configured LXI output LAN events.	

Command	Summary	Reset setting
:LXI:EVEN[:OUTP]:LAN:LIST?	Returns a quoted string containing a list of all configured LAN output event names.	
:LXI:EVEN[:OUTP]:LAN:REM:ALL	This command is ignored.	
:LXI:EVEN[:OUTP]:LAN:REM *event*	This command is ignored.	
:LXI:EVEN[:OUTP]:LAN:SEND *event*,*type*	Forces the instrument to send the specified output event. *event*: LAN event name *type* = RISE|FALL	
:LXI:EVEN[:OUTP]:LAN[:SET]:CONF *event*, *status*,*source*,*slope*,*drive*,*destination*	Configures the most common attributes of LXI LAN output events. *event*: LAN event name *status*: Status *source*: Event name *slope*: Slope attribute *drive*: Drive behavior *destination*: Destination	
:LXI:EVEN[:OUTP]:LAN[:SET]:DEST *event*, *destination* :LXI:EVEN[:OUTP]:LAN[:SET]:DEST? *event*	Sets the destination for the specified outgoing LAN event to the hosts specified by *destination*. *event*: LAN event name *destination*: Destination	"ALL:5044"
:LXI:EVEN[:OUTP]:LAN[:SET]:DRIV *event*,*drive* :LXI:EVEN[:OUTP]:LAN[:SET]:DRIV? *event*	Specifies the trigger drive behavior for the specified LAN output event. *event*: LAN event name *drive* = OFF|NORMal|WOR	OFF

Command	Summary	Reset setting
:LXI:EVEN[:OUTP]:LAN[:SET]:ENAB *event*,*status* :LXI:EVEN[:OUTP]:LAN[:SET]:ENAB? *event*	Enables or disables the specified LXI LAN output event. *event*: LAN event name *status* = 0|OFF|1|ON	OFF
:LXI:EVEN[:OUTP]:LAN[:SET]:IDEN *event*,*id* :LXI:EVEN[:OUTP]:LAN[:SET]:IDEN? *event*	Specifies the custom string that will be transmitted as part of the output event. *event*: LAN event name *id*: Identifier	*event*
:LXI:EVEN[:OUTP]:LAN[:SET]:SLOP *event*,*slope* :LXI:EVEN[:OUTP]:LAN[:SET]:SLOP? *event*	Sets the slope of the event transition. *event*: LAN event name *slope* = POSitive|NEGative	POS
:LXI:EVEN[:OUTP]:LAN[:SET]:SOUR *event*, *anyEvent* :LXI:EVEN[:OUTP]:LAN[:SET]:SOUR? *event*	Designates the instrument that the specified LAN output event is tied to. *event*: LAN event name *anyEvent*: One of the following event names. WaitingForAcquireArm1, WaitingForAcquireTrigger1, WaitingForTransitionArm1, WaitingForTransitionTrigger1, Measuring1, or Settling1	""
:LXI:EVEN[:OUTP]:LAN[:SET]:TSD *event*,*delay* :LXI:EVEN[:OUTP]:LAN[:SET]:TSD? *event*	Sets the delay that occurs between the generation of the specified event and the remote instruments action on it. The delay time must be zero. *event*: LAN event name *delay*: Delay time, in seconds	0
:LXI:IDEN[:STAT] *mode* :LXI:IDEN[:STAT]?	Changes the LXI status indicator state. *mode* = 0|OFF|1|ON	

Command	Summary	Reset setting
:LXI:MDNS:ENAB *mode* :LXI:MDNS:ENAB?	Enables or disables mDNS (multicast DNS) function. *mode* = 0|OFF|1|ON	ON
:LXI:MDNS:HNAM[:RES]?	Returns the resolved mDNS hostname.	
:LXI:MDNS:SNAM:DES *name* :LXI:MDNS:SNAM:DES?	Sets the desired mDNS service name. *name*: Desired mDNS service name	
:LXI:MDNS:SNAM[:RES]?	Returns the resolved mDNS service name.	
:TRIG:LXI:LAN[:SET]:DEL *delay* :TRIG:LXI:LAN[:SET]:DEL?	Delay time must be zero. *delay*: Delay time, in seconds	0
:TRIG:LXI:LAN[:SET]:DET *event*,*detect* :TRIG:LXI:LAN[:SET]:DET? *event*	Specifies the behavior of the trigger signal. *event*: LAN event name *detect* = RISE|FALL|HIGH|LOW	RISE
:TRIG:LXI:LAN[:SET]:ENAB *event*,*status* :TRIG:LXI:LAN[:SET]:ENAB? *event*	Enables or disables the specified LAN trigger. *event*: LAN event name *status* = 0|OFF|1|ON	OFF
:TRIG:LXI:LAN[:SET]:FILT *event*,*filter* :TRIG:LXI:LAN[:SET]:FILT? *event*	Allows user to create a filter expression for the specified LAN trigger event. *event*: LAN event name *filter*: Filter	"ALL:5044"
:TRIG:LXI:LAN[:SET]:IDEN *event*,*id* :TRIG:LXI:LAN[:SET]:IDEN? *event*	Sets the string that is expected to arrive over the LAN for a given trigger LAN event to occur. *event*: LAN event name *id*: Identifier	*event*

Reading Source/Measure Data

Figure 2-11 Measurement Data Flow (B2981B/B2983B supports CURR, TIME, and STAT only)

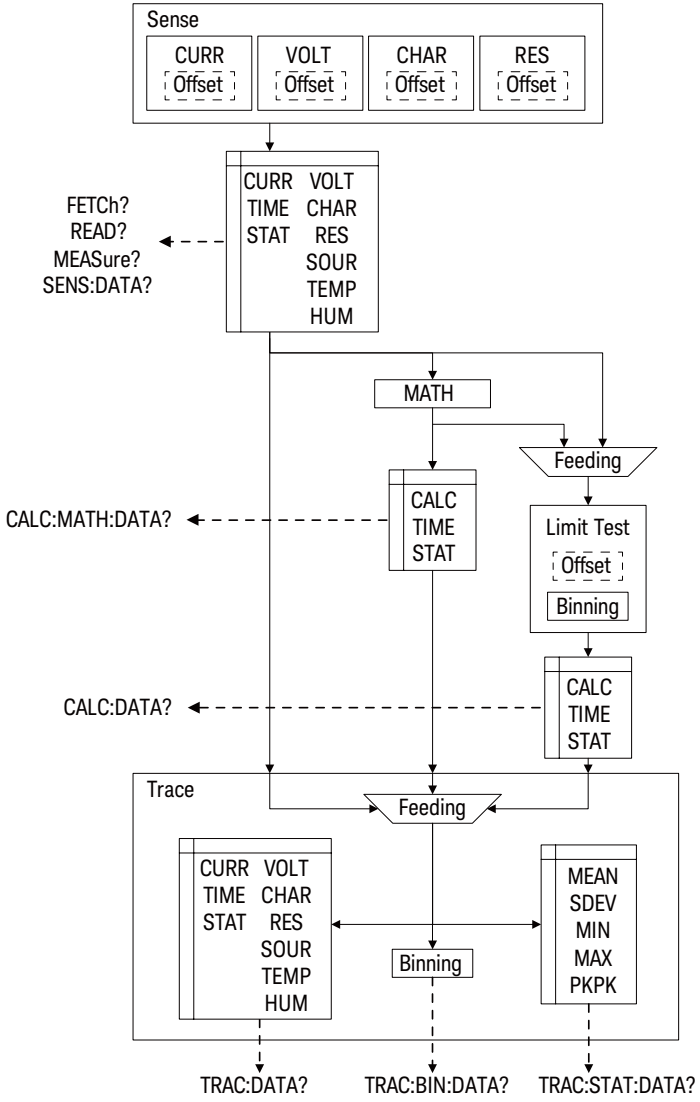


Table 2-12 FETCh Subsystem

Command	Summary	Reset setting
:FETC:ARR? [*chanlist*]	Returns the array data which contains all of the voltage measurement data, current measurement data, resistance measurement data, time data, status data, or source output setting data specified by the :FORM:ELEM:SENS command. *chanlist* = (@1) B2981B/B2983B returns current measurement data, time data, and status data only.	
:FETC:ARR:CURR? [*chanlist*] :FETC:ARR:STAT? [*chanlist*] :FETC:ARR:TIME? [*chanlist*] **(B2985B/B2987B)** :FETC:ARR:CHAR? [*chanlist*] :FETC:ARR:HUM? [*chanlist*] :FETC:ARR:RES? [*chanlist*] :FETC:ARR:SOUR? [*chanlist*] :FETC:ARR:TEMP? [*chanlist*] :FETC:ARR:VOLT? [*chanlist*]	Returns the array data which contains all of the current measurement data, resistance measurement data, source output setting data, status data, time data, or voltage measurement data specified by CHAR, CURR, HUM, RES, SOUR, STAT, TEMP, TIME, or VOLT. *chanlist* = (@1)	
:FETC[:SCAL]? [*chanlist*]	Returns the latest voltage measurement data, current measurement data, resistance measurement data, time data, status data, or source output setting data specified by the :FORM:ELEM:SENS command. *chanlist* = (@1) B2981B/B2983B returns current measurement data, time data, and status data only.	

Command	Summary	Reset setting
:FETC[:SCAL]:CURR? [*chanlist*] :FETC[:SCAL]:STAT? [*chanlist*] :FETC[:SCAL]:TIME? [*chanlist*]	Returns the latest current measurement data, resistance measurement data, source output setting data, status data, time data, or voltage measurement data specified by CHAR, CURR, HUM, RES, SOUR, STAT, TEMP, TIME, or VOLT. *chanlist* = (@1)	
(B2985B/B2987B) :FETC[:SCAL]:CHAR? [*chanlist*] :FETC[:SCAL]:HUM? [*chanlist*] :FETC[:SCAL]:RES? [*chanlist*] :FETC[:SCAL]:SOUR? [*chanlist*] :FETC[:SCAL]:TEMP? [*chanlist*] :FETC[:SCAL]:VOLT? [*chanlist*]		

Table 2-13 FORMat Subsystem

Command	Summary	Reset setting
:FORM:BORD *byte_order* :FORM:BORD?	Sets the byte order of binary output data. *byte_order* = NORMal|SWAPped	NORM
:FORM[:DATA] *format* :FORM[:DATA]?	Sets the data output format. *format* = ASCii|REAL,32|REAL,64	ASC
:FORM:DIG *format* :FORM:DIG?	Sets the response format of the bit pattern. *format* = ASCii|BINary|OCTal| HEXadecimal	ASC
:FORM:ELEM:CALC *type*{,*type*} :FORM:ELEM:CALC?	Specifies the elements included in the calculation result data. *type* = CALC|TIME|STATus Order of returned data: *calc*, *time*, *status*	CALC

Command	Summary	Reset setting
:FORM:ELEM:SENS *type*{,*type*} :FORM:ELEM:SENS?	Specifies the elements included in the sense or measurement result data. *type* = VOLTage|CURRent|CHARge| RESistance|TIME|STATus|SOURce| TEMPerature|HUMidity Order of returned data: *voltage*, *current(charge)*, *resistance*, *time*, *status*, *source*, *temp*, *hum*	CURR, TIME, STAT for B2981B/ B2983B VOLT, CURR, RES, TIME, STAT, SOUR, TEMP, HUM for B2985B/ B2987B
:FORM:SREG *format* :FORM:SREG?	Sets the response format of the status byte register. *format* = ASCii|BINary|OCTal| HEXadecimal	ASC

Table 2-14 READ Subsystem

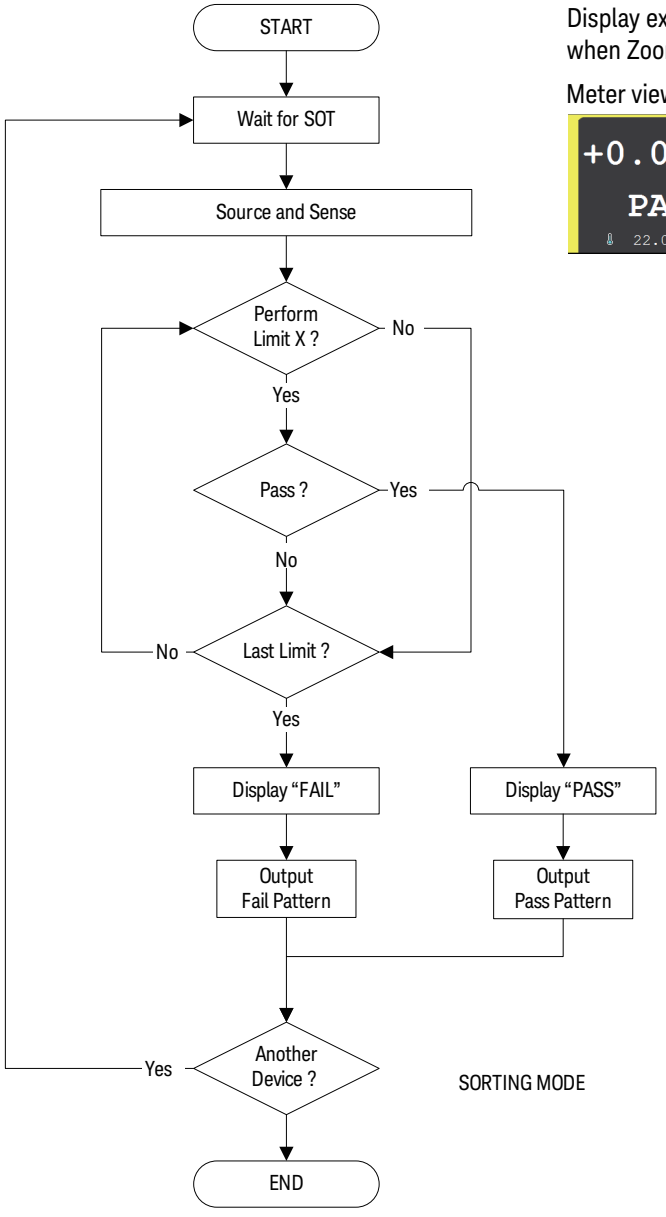
Command	Summary	Reset setting
:READ:ARR? [*chanlist*]	Executes the :INIT command and the :FETC:ARR? command in series, and returns the array data which contains all data for the element specified by the :FORM:ELEM:SENS command. *chanlist* = (@1) B2981B/B2983B returns current measurement data, time data, and status data only.	
:READ:ARR:CURR? [*chanlist*] :READ:ARR:STAT? [*chanlist*] :READ:ARR:TIME? [*chanlist*] **(B2985B/B2987B)** :READ:ARR:CHAR? [*chanlist*] :READ:ARR:HUM? [*chanlist*] :READ:ARR:RES? [*chanlist*] :READ:ARR:SOUR? [*chanlist*] :READ:ARR:TEMP? [*chanlist*] :READ:ARR:VOLT? [*chanlist*]	Executes the :INIT command and the :FETC:ARR:<CHAR|CURR|HUM|RES|SOUR|STAT|TEMP|TIME|VOLT>? command in series, and returns the array data which contains all data for the element specified by CHAR, CURR, HUM, RES, SOUR, STAT, TEMP, TIME, or VOLT. *chanlist* = (@1)	
:READ[:SCAL]? [*chanlist*]	Executes the :INIT command and the :FETC? command in series, and returns the latest data for the element specified by the :FORM:ELEM:SENS command. *chanlist* = (@1) B2981B/B2983B returns current measurement data, time data, and status data only.	

Command	Summary	Reset setting
:READ[:SCAL]:CURR? [*chanlist*] :READ[:SCAL]:STAT? [*chanlist*] :READ[:SCAL]:TIME? [*chanlist*]	Executes the :INIT command and the :FETC:<CHAR|CURR|HUM|RES|SOUR|STAT|TEMP|TIME|VOLT>? command in series, and returns the latest data for the element specified by CHAR, CURR, HUM, RES, SOUR, STAT, TEMP, TIME, or VOLT. *chanlist* = (@1)	
(B2985B/B2987B) :READ[:SCAL]:CHAR? [*chanlist*] :READ[:SCAL]:HUM? [*chanlist*] :READ[:SCAL]:RES? [*chanlist*] :READ[:SCAL]:SOUR? [*chanlist*] :READ[:SCAL]:TEMP? [*chanlist*] :READ[:SCAL]:VOLT? [*chanlist*]		

Table 2-15 MEASure Subsystem

Command	Summary	Reset setting
:MEAS? [*chanlist*]	Executes a spot (one-shot) measurement for the parameters specified by the :SENS:FUNC command., and returns the measurement result data specified by the :FORM:ELEM:SENS command. *chanlist* = (@1)	
:MEAS:CURR[:DC]? [*chanlist*]	Executes a spot (one-shot) measurement and returns the measurement result data. Measurement items can be specified by CHAR, CURR, RES, or VOLT. *chanlist* = (@1)	
(B2985B/B2987B) :MEAS:CHAR? [*chanlist*] :MEAS:RES? [*chanlist*] :MEAS:VOLT[:DC]? [*chanlist*]		

Figure 2-12 Composite Limit Test Flowchart Example for Sorting Mode



Display example of PASS status when Zoom is ON:

Meter view

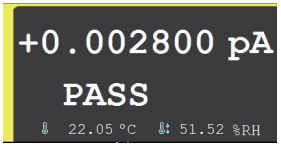


Figure 2-13 Composite Limit Test Flowchart Example for Grading Mode

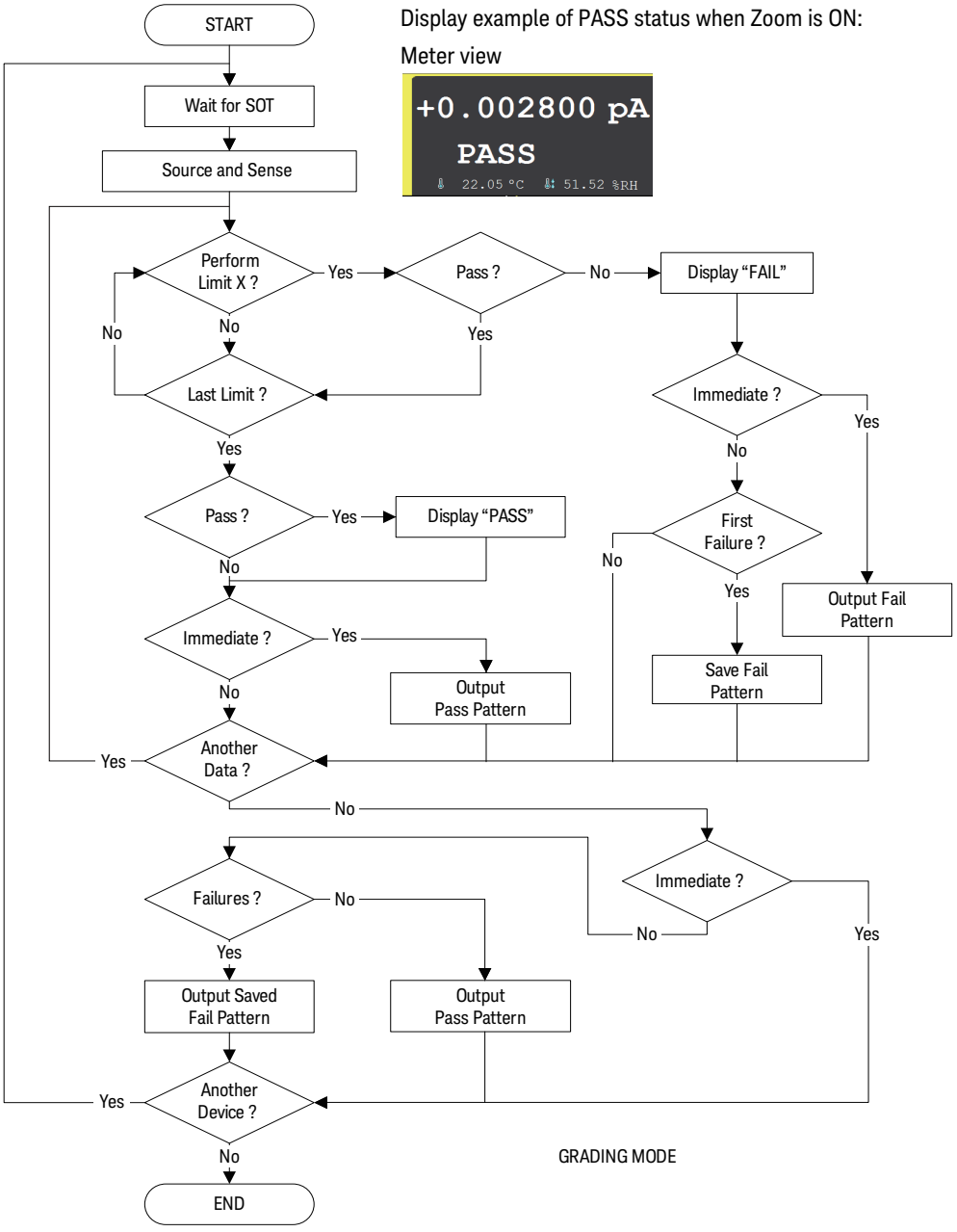


Table 2-16 CALCulate Subsystem

Command	Summary	Reset setting
:CALC[*c*]:CLIM:CLE:AUTO *mode* :CALC[*c*]:CLIM:CLE:AUTO?	Enables or disables the automatic clear function of the composite limit test. *mode* = 0|OFF|1|ON	ON
:CALC[*c*]:CLIM:CLE:AUTO:DEL *time* :CALC[*c*]:CLIM:CLE:AUTO:DEL?	Sets the delay time for the automatic clear of the composite limit test. *time* = MINimum|MAXimum|DEFault| 1E–5 to 60 seconds Query does not support *time* = 1E–5 to 60.	1E–4
:CALC[*c*]:CLIM:CLE[:IMM]	Clears the composite limit test results and the GPIO lines immediately.	
:CALC[*c*]:CLIM:FAIL:DIG[:DATA] *pattern* :CALC[*c*]:CLIM:FAIL:DIG[:DATA]?	Defines a fail pattern that appears near the end of the flowcharts shown in Figures 2-12 and 2-13. This is a bit pattern used to indicate the composite limit test *fail*. *pattern* = 0 to 127	0
:CALC[*c*]:CLIM:MODE *mode* :CALC[*c*]:CLIM:MODE?	Sets the operation mode of the composite limit test. *mode* = GRADing|SORTing.	GRAD
:CALC[*c*]:CLIM:PASS:DIG[:DATA] *pattern* :CALC[*c*]:CLIM:PASS:DIG[:DATA]?	Defines a pass pattern that appears near the end of the flowcharts shown in Figures 2-12 and 2-13. This is a bit pattern used to indicate the composite limit test *pass*. *pattern* = 0 to 127	0
:CALC[*c*]:CLIM:STAT *mode* :CALC[*c*]:CLIM:STAT?	Enables or disables the composite limit test. *mode* = 0|OFF|1|ON	ON
:CALC[*c*]:CLIM:STAT:ANY?	Checks if the present composite limit test contains a limit test, which is a pass/fail judgment. It is performed at the “Pass?” step in Figures 2-12 and 2-13.	

Command	Summary	Reset setting
:CALC[*c*]:CLIM:UPD *result* :CALC[*c*]:CLIM:UPD?	Only for the GRAD composite limit test. Enables or disables the immediate result output or update. See "Immediate?" shown in Figure 2-13. *result* = END|IMMediate	IMM
:CALC[*c*]:DATA? [*offset*[,*size*]]	:CALC[*c*]:DATA? returns limit test data. *offset* = CURRent|STARt|0 to maximum *size* = 1 to maximum :CALC[*c*]:DATA:LAT? returns the last limit test data. Elements of the returned data are specified by the :FORM:ELEM:CALC command. The limit test data can be expressed by the following formula. *limit test data = input data – null offset*	
:CALC[*c*]:DATA:LAT?		
:CALC[*c*]:DIG:BIT *pin* :CALC[*c*]:DIG:BIT?	Assigns the GPIO pins used for the result output. The output is the pass/fail bit pattern. *pin* = EXT*n*|NONE. *n* = 1 to 7.	NONE
:CALC[*c*]:DIG:BUSY *pin* :CALC[*c*]:DIG:BUSY?	Assigns the GPIO pin for the BUSY (busy) line for the composite limit test. *pin* = EXT*n*|NONE. *n* = 1 to 7.	NONE
:CALC[*c*]:DIG:EOT *pin* :CALC[*c*]:DIG:EOT?	Assigns the GPIO pin for the EOT (end of test) line for the composite limit test. *pin* = EXT*n*|NONE. *n* = 1 to 7.	NONE
:CALC[*c*]:DIG:SOT *pin* :CALC[*c*]:DIG:SOT?	Assigns the GPIO pin for the SOT (start of test) line for the composite limit test. *pin* = EXT*n*|NONE. *n* = 1 to 7.	NONE

Command	Summary	Reset setting
:CALC[*c*]:FEED *type* :CALC[*c*]:FEED?	Specifies the *input data* value used for calculating the limit test data. The limit test is a pass/fail judgement performed during a composite limit test. The limit test is performed at the "Pass?" step in Figures 2-12 and 2-13. *type* = MATH|CHARge|CURRent|RESistance|VOLTage	CURR
:CALC[*c*]:LIM[*m*]:COMP:DIG[:DATA] *pattern* :CALC[*c*]:LIM[*m*]:COMP:DIG[:DATA]?	Defines the bit pattern used to indicate a *failure* of the measurement range overflow status check specified by *m*. *pattern* = 0 to 127	0
:CALC[*c*]:LIM[*m*]:COMP:FAIL *criteria* :CALC[*c*]:LIM[*m*]:COMP:FAIL?	Sets the judgement criteria for the measurement range overflow status check specified by *m*. *criteria* = IN determines that the limit test has failed if the channel goes into the measurement range overflow state. *criteria* = OUT determines that the limit test has failed if the channel comes out of the measurement range overflow state.	IN
:CALC[*c*]:LIM[*m*]:FAIL?	Returns the result of the limit test specified by *m*. 0: Passed, 1: Failed.	
:CALC[*c*]:LIM[*m*]:FUNC *type* :CALC[*c*]:LIM[*m*]:FUNC?	Sets the type of the limit test specified by *m*. *type* = COMP sets the measurement range overflow status check which checks if the channel is in the measurement range overflow status. *type* = LIM sets the limit test which checks if the measurement value is between the upper limit and the lower limit.	LIM

Command	Summary	Reset setting
:CALC[*c*]:LIM[*m*]:LOW *limit* :CALC[*c*]:LIM[*m*]:LOW? [MINimum|MAXimum|DEFault]	Sets a lower limit used for the limit test specified by *m*. *limit* = MINimum|MAXimum|DEFault| –9.999999E+20 to +9.999999E+20	–1
:CALC[*c*]:LIM[*m*]:LOW:DIG[:DATA] *pattern* :CALC[*c*]:LIM[*m*]:LOW:DIG[:DATA]?	Defines the bit pattern used to indicate *failed-by-exceeding-lower-limit* of the limit test specified by *m*. The bit pattern is used for the GRAD composite limit test. *pattern* = 0 to 127	0
:CALC[*c*]:LIM[*m*]:PASS:DIG[:DATA] *pattern* :CALC[*c*]:LIM[*m*]:PASS:DIG[:DATA]?	Defines the bit pattern used to indicate a *pass* of the limit test specified by *m*. The bit pattern is used for the SORT composite limit test. *pattern* = 0 to 127	0
:CALC[*c*]:LIM[*m*]:STAT *mode* :CALC[*c*]:LIM[*m*]:STAT?	Enables or disables the limit test specified by *m*. *mode* = 1|ON|0|OFF	OFF
:CALC[*c*]:LIM[*m*]:UPP *limit* :CALC[*c*]:LIM[*m*]:UPP? [MINimum|MAXimum|DEFault]	Sets a upper limit used for the limit test specified by *m*. *limit* = MINimum|MAXimum|DEFault| –9.999999E+20 to +9.999999E+20	1
:CALC[*c*]:LIM[*m*]:UPP:DIG[:DATA] *pattern* :CALC[*c*]:LIM[*m*]:UPP:DIG[:DATA]?	Defines a bit pattern used to indicate *failed-by-exceeding-upper-limit* of the limit test specified by *m*. The bit pattern is used for the GRAD composite limit test. *pattern* = 0 to 127	0

Command	Summary	Reset setting
:CALC[*c*]:MATH:DATA? [*offset*[,*size*]]	:CALC[*c*]:MATH:DATA? returns the calculation result data. *offset* = CURRent|STARt|0 to maximum *size* = 1 to maximum	
:CALC[*c*]:MATH:DATA:LAT?	:CALC[*c*]:MATH:DATA:LAT? returns the latest calculation result data. Elements of the returned data are specified by the :FORM:ELEM:CALC command. Math expression for the calculation is specified by the :CALC:MATH[:EXPR]:NAME and :CALC:MATH[:EXPR][:DEF] commands.	
:CALC[*c*]:MATH[:EXPR]:CAT?	Returns the list of all the predefined and user-defined math expression names.	
:CALC[*c*]:MATH[:EXPR][:DEF] *definition*	Defines a math expression which will be a user-defined math expression. Maximum of 32 math expressions can be defined including the predefined math expressions. *definition*: Up to 256 ASCII characters.	(M * CURR + B)
:CALC[*c*]:MATH[:EXPR]:DEL:ALL	Deletes all user-defined math expressions.	
:CALC[*c*]:MATH[:EXPR]:DEL[:SEL] *name*	Deletes an user-defined math expression. *name*: Up to 32 ASCII characters.	
:CALC[*c*]:MATH[:EXPR]:NAME *name* :CALC[*c*]:MATH[:EXPR]:NAME?	Selects a math expression used for calculation. *name*: Up to 32 ASCII characters without any control characters, space characters, single and double quotes, and comma.	"MXPLUSB"
:CALC[*c*]:MATH:STAT *mode* :CALC[*c*]:MATH:STAT?	Enables or disables the math expression. *mode* = 1|ON|0|OFF	OFF

Command	Summary	Reset setting
:CALC[*c*]:MATH:UNIT *name* :CALC[*c*]:MATH:UNIT?	Defines the unit name for the math expression. *name*: Up to 32 ASCII characters.	" " (Space)
:CALC[*c*]:MATH:VAR:CAT?	Returns the list of all the predefined and user-defined math variable names.	
:CALC[*c*]:MATH:VAR[:DEF] *definition*	Defines a math variable which will be a user-defined math variable.	
:CALC[*c*]:MATH:VAR:DEL:ALL	Deletes all user-defined math variables.	
:CALC[*c*]:MATH:VAR:DEL[:SEL] *name*	Deletes an user-defined math variable. *name*: Up to 16 ASCII characters.	
:CALC[*c*]:MATH:VAR:NAME *name* :CALC[*c*]:MATH:VAR:NAME?	Selects a math variable used for calculation.	"M"
:CALC[*c*]:OFFS *offset* :CALC[*c*]:OFFS? [MINimum|MAXimum|DEFault]	Sets the *null offset* value used for calculating the limit test data. *offset* = MINimum|MAXimum|DEFault| –9.999999E+20 to +9.999999E+20	0
:CALC[*c*]:OFFS:ACQ	Automatically sets the *null offset* value used for calculating the limit test data.	
:CALC[*c*]:OFFS:STAT *mode* :CALC[*c*]:OFFS:STAT?	Enables or disables the null offset function used for calculating the limit test data. *mode* = 1|ON|0|OFF	OFF

Table 2-17 TRACe Subsystem

Command	Summary	Reset setting
:TRAC[*c*]:BIN:CENT *center* :TRAC[*c*]:BIN:CENT?	Sets the center value of histogram on Histogram view.	0.0
:TRAC[*c*]:BIN:COUN *count* :TRAC[*c*]:BIN:COUN?	Sets the number of bins for histogram on Histogram view.	10
:TRAC[*c*]:BIN:COUN:ACT?	Gets the actual number of bins for histogram on Histogram view.	
:TRAC[*c*]:BIN:DATA:CURR? :TRAC[*c*]:BIN:DATA:LIM? :TRAC[*c*]:BIN:DATA:MATH?	Returns the number of data in each bin on Histogram view.	
(B2985B/B2987B) :TRAC[*c*]:BIN:DATA:RES? :TRAC[*c*]:BIN:DATA:VOLT?		
:TRAC[*c*]:BIN:DATA:CURR:OOB? :TRAC[*c*]:BIN:DATA:LIM:OOB? :TRAC[*c*]:BIN:DATA:MATH:OOB?	Returns the number of data that is out of bins range on Histogram view.	
(B2985B/B2987B) :TRAC[*c*]:BIN:DATA:RES:OOB? :TRAC[*c*]:BIN:DATA:VOLT:OOB?		
:TRAC[*c*]:BIN:WIDT *width* :TRAC[*c*]:BIN:WIDT?	Sets the bin width on Histogram view.	0.004
:TRAC[*c*]:CLE	Clears the trace buffer of the specified channel. This command is effective when the trace buffer control mode is set to NEV by the :TRAC[*c*]:FEED:CONT command.	

Command	Summary	Reset setting
:TRAC[*c*]:DATA? [*offset*[,*size*]]	Returns the data in the trace buffer. *offset* = CURRent|STARt|0 to maximum *size* = 1 to maximum	
:TRAC[*c*]:DATA:CURR? [*offset*[,*size*]] :TRAC[*c*]:DATA:LIM? [*offset*[,*size*]] :TRAC[*c*]:DATA:MATH? [*offset*[,*size*]] (B2985B/B2987B) :TRAC[*c*]:DATA:RES? [*offset*[,*size*]] :TRAC[*c*]:DATA:VOLT? [*offset*[,*size*]]	Returns the specified measurement data in the trace buffer. The data stored in the buffer is specified by the :TRAC:FEED command. *offset* = CURRent|STARt|0 to maximum *size* = 1 to maximum	
:TRAC[*c*]:FEED *type* :TRAC[*c*]:FEED?	Specifies the data placed in the trace buffer. This command is effective when the trace buffer control mode is set to NEV by the :TRAC[*c*]:FEED:CONT command. *type* = LIM|MATH|SENSe	SENS
:TRAC[*c*]:FEED:CONT *mode* :TRAC[*c*]:FEED:CONT?	Selects the trace buffer control. *mode* = NEXT|NEVer	NEV
:TRAC[*c*]:FREE?	Returns the available size (*available*) and the total size (*total*) of the trace buffer. Response is *available*,*total*.	
:TRAC[*c*]:POIN *points* :TRAC[*c*]:POIN? [MINimum|MAXimum|DEFault]	Sets the size of the trace buffer. This command is effective when the trace buffer control mode is set to NEV by the :TRAC[*c*]:FEED:CONT command. *points* = MINimum|MAXimum|DEFault| 1 to 100000	100000
:TRAC[*c*]:POIN:ACT?	Returns the number of data in the trace buffer.	

Command	Summary	Reset setting
:TRAC[*c*]:STAT:DATA?	Returns the result of the statistical operation for the data stored in the trace buffer.	
:TRAC[*c*]:STAT:FORM *operation* :TRAC[*c*]:STAT:FORM?	Selects the statistical operation performed by the :TRAC[*c*]:STAT:DATA? command. *operation* = MINimum|MAXimum|MEAN|SDEViation|PKPK	MEAN
:TRAC[*c*]:TST:FORM *rule* :TRAC[*c*]:TST:FORM?	Selects the rule for reading the timestamp data in the trace buffer. *rule* = DELTa|ABSolute	ABS

Using Advanced Functions

Table 2-18 HCOPy Subsystem

Command	Summary	Reset setting
:HCOP:SDUM:DATA?	Returns the data of the front panel screen image. The response is a definite length arbitrary binary block.	
:HCOP:SDUM:FORM *format* :HCOP:SDUM:FORM?	Sets the format of the image data. *format* = JPG|BMP|PNG|WMF	JPG

Table 2-19 DISPlay Subsystem

Command	Summary	Reset setting
:DISP:CSET *color* :DISP:CSET?	Selects the color set of the front panel display. *color* = 1 (default color set)|2 (alternative color set)	
:DISP:ENAB *mode* :DISP:ENAB?	Enables or disables the front panel display under remote operation. *mode* = 1|ON|0|OFF	
:DISP:VIEW *mode* :DISP:VIEW?	Sets the display mode, single 1, graph, roll, or histogram. *mode* = SINGle1|GRAPh|ROLL|HIST	SING1
:DISP:VIEW:GRAP:CURS:STAT *mode* :DISP:VIEW:ROLL:CURS:STAT *mode* :DISP:VIEW:GRAP:CURS:STAT? :DISP:VIEW:ROLL:CURS:STAT?	Select if the cursor is displayed or not. *mode* = 1|ON|0|OFF	OFF

Command	Summary	Reset setting
:DISP:VIEW:GRAP:RLIN:STAT *mode* :DISP:VIEW:ROLL:RLIN:STAT *mode* :DISP:VIEW:GRAP:RLIN:STAT? :DISP:VIEW:ROLL:RLIN:STAT?	Select if the reference line function is enabled or disabled. *mode* = 1|ON|0|OFF	OFF
:DISP:VIEW:GRAP:RLIN:STOR	Saves the displayed line data as the reference line data.	
:DISP:VIEW:GRAPH:SCAL:AUTO *mode* :DISP:VIEW:HIST:SCAL:AUTO *mode* :DISP:VIEW:ROLL:SCAL:AUTO *mode*	Changes the graph scale to fit the trace automatically. *mode* = ONCE	
:DISP:VIEW:GRAP:X[:ELEM] *mode* :DISP:VIEW:GRAP:Y[:ELEM] *mode* :DISP:VIEW:GRAP:X[:ELEM]? :DISP:VIEW:GRAP:Y[:ELEM]?	Sets the data type assignment for X-axis or Y-axis on the Graph view. *mode* = CHAR|CURR|RES|MATH|SOUR|TIME|VOLT B2981B/B2983B can specify CURR, MATH, or TIME only. SOUR/TIME is only for X-axis.	
:DISP:VIEW:GRAP:X:MAX *value* :DISP:VIEW:GRAP:Y:MAX *value* :DISP:VIEW:GRAP:X:MAX? :DISP:VIEW:GRAP:Y:MAX?	Sets the maximum value of X-axis or Y-axis on the Graph view.	
:DISP:VIEW:GRAP:X:MIN *value* :DISP:VIEW:GRAP:Y:MIN *value* :DISP:VIEW:GRAP:X:MIN? :DISP:VIEW:GRAP:Y:MIN?	Sets the minimum value of X-axis or Y-axis on the Graph view.	

Command	Summary	Reset setting
:DISP:VIEW:GRAP:X:SPAC *mode* :DISP:VIEW:GRAP:Y:SPAC *mode* :DISP:VIEW:GRAP:X:SPAC? :DISP:VIEW:GRAP:Y:SPAC?	Selects linear scale of logarithmic scale for X-axis or Y-axis on the Graph view. *mode* = LIN|LOG	LIN
:DISP:VIEW:HIST:Y:ELEM *mode* :DISP:VIEW:ROLL:Y:ELEM *mode* :DISP:VIEW:HIST:Y:ELEM? :DISP:VIEW:ROLL:Y:ELEM?	Sets the data type assignment for Y-axis or Y-axis on the Histogram or Roll view. *mode* = CHAR|CURR|RES|VOLT B2981B/B2983B can specify CURR, only.	
:DISP:VIEW:HIST:Y:MAX *value* :DISP:VIEW:HIST:Y:MAX?	Sets the maximum value of Y-axis on the Histogram view.	
:DISP:VIEW:ROLL:X:OFFS *value* :DISP:VIEW:ROLL:X:PDIV *value* :DISP:VIEW:ROLL:X:OFFS? :DISP:VIEW:ROLL:X:PDIV?	Sets the value of scale division or offset value of X-axis on Roll view.	
:DISP:VIEW:ROLL:Y:OFFS:CURR *value* :DISP:VIEW:ROLL:Y:OFFS:CURR?	Sets the offset value of Y-axis on Roll view.	
(B2985B/B2987B) :DISP:VIEW:ROLL:Y:OFFS:CHAR *value* :DISP:VIEW:ROLL:Y:OFFS:RES *value* :DISP:VIEW:ROLL:Y:OFFS:VOLT *value* :DISP:VIEW:ROLL:Y:OFFS:CHAR? :DISP:VIEW:ROLL:Y:OFFS:RES? :DISP:VIEW:ROLL:Y:OFFS:VOLT?		

Command	Summary	Reset setting
:DISP:VIEW:ROLL:Y:PDIV:CURR *value* :DISP:VIEW:ROLL:Y:PDIV:CURR?	Sets the scale division of Y-axis on Roll view.	
(B2985B/B2987B) :DISP:VIEW:ROLL:Y:PDIV:CHAR *value* :DISP:VIEW:ROLL:Y:PDIV:RES *value* :DISP:VIEW:ROLL:Y:PDIV:VOLT *value* :DISP:VIEW:ROLL:Y:PDIV:CHAR? :DISP:VIEW:ROLL:Y:PDIV:RES? :DISP:VIEW:ROLL:Y:PDIV:VOLT?		
:DISP:VIEW:SING:FORM *mode* :DISP:VIEW:SING:FORM?	Selects the display format of measured value to be displayed on Meter view. *mode* = ENG|EXP	ENG
:DISP:VIEW:SING:SPAN :DISP:VIEW:SING:SPAN?	Selects the item to be displayed on sub-panel of Meter view.	
:DISP[:WIND[*d*]]:DATA?	Returns the data displayed on the front panel display.	
:DISP[:WIND[*d*]]:TEXT:DATA *text* :DISP[:WIND[*d*]]:TEXT:DATA?	Sets the text message displayed on the center of the upper or lower display area of the front panel display. *text*: Up to 32 ASCII characters.	“”
:DISP[:WIND[*d*]]:TEXT:STAT :DISP[:WIND[*d*]]:TEXT:STAT?	Shows or hides the text message set by the :DISP[:WIND[*d*]]:TEXT:DATA command. *mode* = 1|ON|0|OFF	OFF
:DISP:ZOOM *mode* :DISP:ZOOM?	Enables or disables the zoom function of the front panel display. *mode* = 1|ON|0|OFF	

Table 2-20 MMEMory Subsystem

Command	Summary	Reset setting
:MMEM:CAT? [*directory*]	Returns the memory usage and availability. Also returns the list of files and folders in the current specified directory. *directory* = <path>|USB:\<path>	
:MMEM:CDIR *directory* :MMEM:CDIR?	Changes the current directory to the specified directory. *directory* = <path>|USB:\<path>	
:MMEM:COPY *source*,*destination*	Makes a copy of an existing file in the current directory. *source*: Source file name. *destination*: Copy file name. Or directory name, <path>|USB:\<path>.	
:MMEM:DEL *file_name*	Deletes a file in the current directory. *file_name*: Name of the file to delete.	
:MMEM:LOAD:LIST:VOLT *file*[,*chlist*]	Loads a list sweep data from the specified file in the current directory. *file*: Name of the file to load the specified data. *chlist* = (@1)	
:MMEM:LOAD:MACR macro,*file_name*	Loads a macro from the specified file in the current directory. *macro*: Name of macro. *file_name*: Name of the file which contains the macro.	

Command	Summary	Reset setting
:MMEM:LOAD:STAT *file_name*	Loads an instrument setup from the specified file in the current directory. *file_name*: Name of the file which contains the instrument setup.	
:MMEM:MDIR *directory*	Creates a new directory. *directory* = <path>|USB:\<path>	
:MMEM:MOVE *source*,*destination*	Moves or renames an existing file in the current directory. *source*: Source file name. *destination*: New file name. Or directory name, <path>|USB:\<path>.	
:MMEM:RDIR *directory*	Removes the specified empty directory. *directory* = <path>|USB:\<path>	
:MMEM:STOR:DATA[:ALL] *file*[,*chlist*] :MMEM:STOR:DATA:LIM *file*[,*chlist*] :MMEM:STOR:DATA:MATH *file*[,*chlist*] :MMEM:STOR:DATA:SENS *file*[,*chlist*]	Saves the limit test data, the math expression result data, sense data, or all of these data for the specified channel to the specified file in the current directory. *file*: Name of the file to save the specified data. *chlist* = (@1)	
:MMEM:STOR:LIST:VOLT *file*[,*chlist*]	Saves the list sweep data to the specified file in the current directory. *file*: Name of the file to save the specified data. *chlist* = (@1)	

Command	Summary	Reset setting
:MMEM:STOR:MACR macro,*file_name*	Saves the macro to the specified file in the current directory. *macro*: Name of macro. *file_name*: Name of the file to save the macro.	
:MMEM:STOR:STAT *file_name*	Saves the instrument setup to the specified file in the current directory. *file_name*: Name of the file to save the instrument setup.	
:MMEM:STOR:TRAC *file_name*[,*chlist*]	Saves all data in the trace buffer for the specified channel to the specified file in the current directory. *file_name*: Name of the file to save the specified data. *chlist* = (@1)	

Table 2-21 PROGram Subsystem

Command	Summary	Reset setting
:PROG:CAT?	Returns the names of all programs defined in the program memory.	
:PROG:PON:COPY *name*	Specifies the power-on program. *name*: Name of the program used for the power-on program.	
:PROG:PON:DEL	Clears the power-on program.	
:PROG:PON:RUN *mode*	Enables or disables the power-on program. *mode* = 1|ON|0|OFF	

Command	Summary	Reset setting
:PROG[:SEL]:APP *program_code*	Adds a program code to the end of a program stored in the program memory. *program_code*: Program code. Up to 256 byte per execution. Sum of all program size in the program memory must be up to 100 KB.	
:PROG[:SEL]:DEF *program_code* :PROG[:SEL]:DEF?	Defines a program in the program memory by entering the initial program code. *program_code*: Program code. Up to 256 byte per execution. Sum of all program size in the program memory must be up to 100 KB. Maximum of 100 programs can be memorized.	
:PROG[:SEL]:DEL:ALL	Deletes all programs stored in the program memory.	
:PROG[:SEL]:DEL[:SEL]	Deletes a program stored in the program memory.	
:PROG[:SEL]:EXEC	Executes a program stored in the program memory.	
:PROG[:SEL]:NAME *name* :PROG[:SEL]:NAME?	Selects the program for performing the action by the following commands. If *name* does not specify the program stored in the program memory, this command creates a new program with the specified name and selects the program. If *name* specifies an existing program, this command selects the program.	
:PROG[:SEL]:STAT *operation* :PROG[:SEL]:STAT?	Changes the execution status of a program stored in the program memory. *operation* = RUN|PAUSe|STEP|STOP|CONTinue	

Command	Summary	Reset setting
:PROG[:SEL]:WAIT? *timeout*	Blocks other commands until the program execution status changes to Paused or Stopped. *timeout*: Timeout value, in seconds.	
:PROG:VAR[*h*] *value* :PROG:VAR[*h*]?	Sets a value to the variable specified by *h*. The variable is used in the program memory. A variable can be used in a program as %*h*% (*h*: integer. 1 to 100). *value*: Value of the variable specified by *h*. Up to 32 ASCII characters.	

Table 2-22 SYSTem Subsystem

Command	Summary	Reset setting
:SYST:AOUT *function* :SYST:AOUT?	Selects the measurement mode of which results are applied to the analog output function. *function* = IM|QM|VM On B2981B/B2983B, only IM can be specified.	IM
(B2983B/B2987B) :SYST:BATT?	Returns a percentage of the remaining battery capacity.	
(B2983B/B2987B) :SYST:BATT:CYCL?	Returns the battery cycle count.	
(B2983B/B2987B) :SYST:BATT:TEST?	Performs self-test on the battery and return the result. 0: test passed 1: test failed	

Command	Summary	Reset setting
:SYST:BEEP[:IMM] *frequency*,*time*	Generates a beep sound of the specified frequency and duration. *frequency* = 55 to 6640 Hz *time* = 0.05 to 12.75 seconds	
:SYST:BEEP:STAT *mode* :SYST:BEEP:STAT?	Enables or disables the beeper. *mode* = 0|OFF|1|ON	
:SYST:COMM:ENAB *mode*,*interface* :SYST:COMM:ENAB? *interface*	Enables or disables the remote interface GPIB, USB, or LAN, the remote service Sockets, Telnet, VXI-11, HiSLIP, or the built-in Web Interface. The setting is effective after rebooting the instrument. *mode* = 0|OFF|1|ON *interface* = GPIB|USB|LAN|SOCKets|TELNet|VXI11|HISLip|WEB	
:SYST:COMM:GPIB[:SELF]:ADDR *address* :SYST:COMM:GPIB[:SELF]:ADDR?	Sets the GPIB address of the instrument. *address* = 0 to 30	
:SYST:COMM:LAN:ADDR *address* :SYST:COMM:LAN:ADDR? [CURR|STAT]	Sets the static LAN (IP) address of the instrument. The setting is enabled by the :SYST:COMM:LAN:UPD command. *address* = *A*.*B*.*C*.*D*, 15 characters maximum. *A*, *B*, *C*, and *D* must be a number from 0 to 225. CURR: Present setup value STAT: Reserved value for the next startup	
:SYST:COMM:LAN:BST?	Returns the LAN boot status of the instrument. Response is LAN_AUTO_IP, LAN_DHCP, LAN_FAULT, or LAN_STATIC.	

Command	Summary	Reset setting
:SYST:COMM:LAN:CONT? :SYST:COMM:TCP:CONT?	Returns the control connection port number of the specified port.	
:SYST:COMM:LAN:DHCP *mode* :SYST:COMM:LAN:DHCP?	Enables or disables the use of the Dynamic Host Configuration Protocol (DHCP). The setting is enabled by the :SYST:COMM:LAN:UPD command. *mode* = 0|OFF|1|ON	
:SYST:COMM:LAN:DNS[*j*] *address* :SYST:COMM:LAN:DNS[*j*]? [CURR|STAT]	Sets the IP address of the DNS server. *address* = *A*.*B*.*C*.*D*, 15 characters maximum. *A*, *B*, *C*, and *D* must be a number from 0 to 255. CURR: Present setup value STAT: Reserved value for the next startup	
:SYST:COMM:LAN:DOM?	Returns the domain name of the network to which the instrument is connected.	
:SYST:COMM:LAN:GAT *address* :SYST:COMM:LAN:GATE *address* :SYST:COMM:LAN:GAT? [CURR|STAT] :SYST:COMM:LAN:GATE? [CURR|STAT]	Sets the IP address of the default gateway. The setting is enabled by the :SYST:COMM:LAN:UPD command. *address* = *A*.*B*.*C*.*D*, 15 characters maximum. *A*, *B*, *C*, and *D* must be a number from 0 to 225. CURR: Present setup value STAT: Reserved value for the next startup	
:SYST:COMM:LAN:HNAM *hostname* :SYST:COMM:LAN:HOST *hostname* :SYST:COMM:LAN:HNAM? [CURR|STAT] :SYST:COMM:LAN:HOST? [CURR|STAT]	Sets the host name of the instrument. The setting is enabled by the :SYST:COMM:LAN:UPD command. *hostname*: Host name. Up to 15 characters. CURR: Present setup value STAT: Reserved value for the next startup	

Command	Summary	Reset setting
:SYST:COMM:LAN:MAC?	Returns the MAC address of the instrument.	
:SYST:COMM:LAN:SMAS *subnet_mask* :SYST:COMM:LAN:SMAS? [CURR|STAT]	Sets the static subnet mask. The setting is enabled by the :SYST:COMM:LAN:UPD command. *subnet_mask* = *A*.*B*.*C*.*D*, 15 characters maximum. *A*, *B*, *C*, and *D* must be a number from 0 to 255. CURR: Present setup value STAT: Reserved value for the next startup	
:SYST:COMM:LAN:TELN:PROM *prompt* :SYST:COMM:LAN:TELN:PROM?	Sets the command prompt displayed during a Telnet session for establishing communication with the instrument. *prompt*: Command prompt. Up to 15 characters.	
:SYST:COMM:LAN:TELN:WMES *message* :SYST:COMM:LAN:TELN:WMES?	Sets the welcome message displayed during a Telnet session when starting the communication with the instrument. *message*: Welcome message. Up to 63 characters.	
:SYST:COMM:LAN:UPD	Disconnects all active LAN and Web Interface connections, updates the LAN setup, and restarts the LAN interface with the new setup.	
:SYST:COMM:LAN:WINS[*j*] *address* :SYST:COMM:LAN:WINS[*j*]? [CURR|STAT]	Sets the IP address of the WINS server. *address* = *A*.*B*.*C*.*D*, 15 characters maximum. *A*, *B*, *C*, and *D* must be a number from 0 to 255. CURR: Present setup value STAT: Reserved value for the next startup	

Command	Summary	Reset setting
:SYST:DATA:QUAN? [*chanlist*]	Returns the number of data for the specified channel in the data buffer. *chanlist* = (@1)	
:SYST:DATE *year*,*month*,*day* :SYST:DATE?	Sets the date of the internal clock. *year*: Year. 4-digit integer. *month*: Month. Integer from 1 to 12. *day*: Day. Integer from 1 to 31.	
:SYST:ERR:ALL?	Reads and returns all items in the error/event queue, and clears the queue.	
:SYST:ERR:CODE:ALL?	Reads all items in the error/event queue, returns all codes, and clears the queue.	
:SYST:ERR:CODE[:NEXT]?	Reads and removes the top item in the error/event queue, and returns the top code.	
:SYST:ERR:COUN?	Returns the number of items in the error/event queue.	
:SYST:ERR[:NEXT]?	Reads and removes the top item in the error/event queue, and returns the top code and message.	
(B2985B/B2987B) :SYST:HUM?	Returns the humidity measurement data from the humidity sensor.	
(B2985B/B2987B) :SYST:INT:TRIP?	Returns if the interlock circuit is close or open. Response is 0 or 1 that indicates the interlock circuit is close or open, respectively	
:SYST:LFR *frequency* :SYST:LFR?	Selects the line frequency. *frequency* = 50 (for 50 Hz)|60 (for 60 Hz)	

Command	Summary	Reset setting
:SYST:LFR:DET:AUTO *mode* :SYST:LFR:DET:AUTO?	Selects if auto-detection of power line frequency at power on is enabled or disabled. *mode* = 0|OFF|1|ON	
:SYST:LFR:DET?	Detects the power line frequency and returns the result.	
:SYST:LOCK:NAME?	Returns the current I/O interface (the I/O interface in use by the querying computer).	
:SYST:LOCK:OWN?	Returns the I/O interface that currently has a lock.	
:SYST:LOCK:REL	Decrements the lock count by one, and may release the I/O interface from which the command is executed.	
:SYST:LOCK:REQ?	Requests a lock of the current I/O interface.	
:SYST:PON *memory*	Specifies the power-on state. *memory* = RST|RCL0|RCL1|RCL2|RCL3|RCL4	
:SYST:PRES	Presets the instrument settings and the front panel display.	
:SYST:SET *data* :SYST:SET?	Sends or loads the instrument setup data. *data*: Instrument setup data. Parameter data type is a definite length arbitrary binary block.	
(B2985B/B2987B) :SYST:TEMP?	Returns the temperature measurement data from the currently-selected temperature sensor.	
(B2985B/B2987B) :SYST:TEMP:SEL *sensor* :SYST:TEMP:SEL?	Selects the temperature sensor for temperature measurement. *sensor* = TC|HSEN	

Command	Summary	Reset setting
(B2985B/B2987B) :SYST:TEMP:UNIT *unit* :SYST:TEMP:UNIT?	Selects the unit for temperature measurement. *sensor* = C|CEL|F|FAR|K	C
:SYST:TIME *hour*,*minute*,*second* :SYST:TIME?	Sets the time of the internal clock. *hour*: Hour. Integer from 0 to 23. *minute*: Minute. Integer from 0 to 59. *second*: Second. Integer from 0 to 59.	
:SYST:TIME:TIM:COUN?	Returns the present count of the timer.	
:SYST:TIME:TIM:COUN:RES:AUTO *mode* :SYST:TIME:TIM:COUN:RES:AUTO?	Enables or disables the automatic reset function of the timer. If this function is enabled, the timer count is reset when the initiate action occurs. *mode* = 0|OFF|1|ON	ON
:SYST:TIME:TIM:COUN:RES[:IMM]	Resets the timer count immediately.	
:SYST:TIN:POL *polarity* :SYST:TOUT:POL *polarity* :SYST:TIN:POL? :SYST:TOUT:POL?	Sets the polarity of the trigger input/output function for the specified BNC connector. *polarity* = NEG|POS	NEG
:SYST:TOUT[:EDGE]:POS *position* :SYST:TOUT[:EDGE]:POS?	Selects the trigger output timing for the specified GPIO pin. *polarity* = BEF|ARG|BOTH	BOTH
:SYST:TOUT[:EDGE]:WIDT *width* :SYST:TOUT[:EDGE]:WIDT?	Sets the pulse width of the output from the specified GPIO pin. *polarity* = (1E–5 to 1E–2)|MIN|MAX|DEF	0.1 ms

Command	Summary	Reset setting
:SYST:TOUT:TYPE *type* :SYST:TOUT:TYPE?	Selects the output trigger type for the specified GPIO pin. *polarity* = EDGE|LEV	EDGE
:SYST:VERS?	Returns the version of the SCPI standard.	

Table 2-23 STATus Subsystem

Command	Summary	Reset setting
:STAT:MEAS:COND? :STAT:OPER:COND? :STAT:QUES:COND?	Returns the value of the measurement, operation, or questionable status condition register.	
:STAT:MEAS:ENAB *mask* :STAT:OPER:ENAB *mask* :STAT:QUES:ENAB *mask* :STAT:MEAS:ENAB? :STAT:OPER:ENAB? :STAT:QUES:ENAB?	Sets the measurement, operation, or questionable status enable register. *mask* = 0 to 65535 (decimal)	0
:STAT:MEAS[:EVEN]? :STAT:OPER[:EVEN]? :STAT:QUES[:EVEN]?	Returns the value of the measurement, operation, or questionable status event register.	
:STAT:MEAS:NTR *filter* :STAT:OPER:NTR *filter* :STAT:QUES:NTR *filter* :STAT:MEAS:NTR? :STAT:OPER:NTR? :STAT:QUES:NTR?	Sets the negative transition filter in the measurement, operation, or questionable status register. If you set a bit of the filter, a 1-to-0 transition of its register bit sets the corresponding bit of the event register. *filter* = 0 to 65535 (decimal)	0

Command	Summary	Reset setting
:STAT:MEAS:PTR *filter* :STAT:OPER:PTR *filter* :STAT:QUES:PTR *filter* :STAT:MEAS:PTR? :STAT:OPER:PTR? :STAT:QUES:PTR?	Sets the positive transition filter in the measurement, operation, or questionable status register. If you set a bit of the filter, a 0-to-1 transition of its register bit sets the corresponding bit of the event register. *filter* = 0 to 65535 (decimal)	32767
:STAT:PRES	Sets all defined bits in the status system's PTR registers and clears the all bits in the NTR and Enable registers. The registers are returned to the default condition.	

3 Common Commands

This chapter describes common commands and queries of *IEEE 488.2*. The commands available for Keysight B2980 are listed in Table 3-1.

Table 3-1 Common Commands Available for B2980

Mnemonic	Name
*CAL?	Calibration query
*CLS	Clear status
*ESE	Standard event status enable command (query)
*ESR?	Standard event status register query
*IDN?	Identification query
*OPC	Operation complete command (query)
*RCL	Recall command
*RST	Reset command
*SAV	Save command
*SRE	Service request enable command (query)
*STB?	Read status byte query
*TRG	Trigger command
*TST?	Self-test query
*WAI	Wait-to-continue command

*CAL?

This query command performs the self-calibration, and returns the execution result.

Execution Conditions Open the measurement terminals before starting the self-calibration.

Syntax *CAL?

Query response *result* <newline><^END>

result is 0 or 1 that indicates the calibration result. Response data type is NR1.

0: Passed

1: Failed

*CLS

This command clears the Status Byte register, the Standard Event Status register, and the Error Queue. This command does not clear the enable registers. For the SCPI status system, see "Status System Diagram" on page 37.

Also, this command stops the monitoring of pending operations by the *OPC command.

This command does not have query form.

Syntax *CLS

*ESE

This command sets the bits of the Standard Event Status Enable register. This command programs the Standard Event Status Enable register bits. The programming determines which events of the Standard Event Status Enable register are allowed to set the ESB (Event Summary Bit) of the Status Byte register. A 1 in the bit position enables the corresponding event. For the SCPI status system, see "Status System Diagram" on page 37.

Syntax *ESE *enable_number*

*ESE?

Parameter

enable_number	Decimal value that is the sum of the binary-weighted values for the desired bits, hexadecimal, octal, or binary value. Parameter data type is NR1 or NDN.

Query response *enable_number* <newline><^END>

enable_number is the sum of the binary-weighted values of the Enable register bits. The return format can be selected by the :FORMat:SREGister command. Response data type is NR1 or NDN.

Remarks Bit definitions of the Standard Event register are shown in Table 3-2. All of the enabled events of the Standard Event Status Enable register are logically ORed to cause the Event Summary Bit (ESB) of the Status Byte register to be set.

The *CLS (clear status) command will not clear the enable register but it does clear all bits in the event register.

The :STATus:PRESet command does not clear the bits in the Status Byte register.

See Also :FORMat:SREGister and :STATus:PRESet

*ESR?

This query returns the present contents of the Standard Event Status register. The event register is a read-only register, which stores (latches) all standard events. Reading the Standard Event Status Enable register clears it. For the SCPI status system, see "Status System Diagram" on page 37.

Syntax *ESR?

Query response *register* <newline><^END>

register is the binary-weighted sum of all bits set in the register. For example, if bit 3 (decimal value = 8) and bit 7 (decimal value = 128) are enabled, the query command will return 136. Response data type is NR1.

Remarks Bit definitions of the Standard Event register are shown in Table 3-2.

To be reported to the Standard Event register, the corresponding bits in the event register must be enabled using the *ESE command.

Once a bit is set, it remains set until cleared by reading the event register or the *CLS (clear status) command.

See Also *ESE

Table 3-2 Standard Event Register Bit Definitions

Bit	Decimal value	Description	Definition
0	1	OPC (operation complete)	All commands prior to and including *OPC have been executed.
1	2	Not used	0 is returned.
2	4	QYE (query error)	The instrument tried to read the output buffer but it was empty. Or, a new command line was received before a previous query has been read. Or, both the input and output buffers are full.
3	8	DDE (device-dependent error)	A self-test or calibration error occurred (an error in the -300 range or any positive error has been generated). For a complete listing of the error messages, see Chapter 5, "Error Messages."
4	16	EXE (execution error)	An execution error occurred (an error in the -200 range has been generated).
5	32	CME (command error)	A command syntax error occurred (an error in the -100 range has been generated).
6	64	Not used	0 is returned.
7	128	PON (power on)	Power has been turned off and on since the last time the event register was read or cleared.

*IDN?

This query command returns the instrument's (mainframe) identification string which contains four comma-separated fields.

Syntax *IDN?

Query response Keysight Technologies,*model*,*serial*,*revision* <newline><^END>

model: mainframe model number

serial: mainframe serial number

revision: firmware revision number

Response data type is AARD.

*OPC

This command starts to monitor pending operations, and sets/clears the operation complete (OPC) bit in the Standard Event Status register as follows.

- If there is no pending operation, the OPC bit is set to 1.
- If there are any pending operations, the OPC bit is set to 0. The bit will be set to 1 again when all pending operations are completed.

The *OPC command is required to enable the OPC bit. To stop monitoring pending operations (disable OPC bit), execute the *CLS command.

Other commands cannot be executed until this command completes.

Syntax *OPC

*OPC?

Query response 1 <newline><^END>

The query returns 1 if the instrument has completed all pending operations sent before this command. Response data type is NR1.

See Also *WAI

*RCL

This command restores the instrument to a state that was previously stored in one of the memory locations 0 through 9 with the *SAV command.

Syntax *RCL *memory*

Parameter

memory	One of the memory locations 0 to 9. Parameter data type is NR1.

Remarks The device state stored in the location 0 is automatically recalled at power turn-on when the Output Power-On state is set to *RCL 0.

You cannot recall the instrument state from a storage location that is empty or was deleted. You can only recall a state from a location that contains a previously stored state.

The *RST command does not affect the configurations stored in memory. Once a state is stored, it remains until it is overwritten or specifically deleted.

*RST

This command performs an instrument reset. This command resets the volatile memory of the instrument to the initial setting.

Syntax *RST

Remarks This command cancels any measurement or output trigger actions presently in process, and resets the Waiting for arm and trigger bits in the Status Operation Condition register.

*SAV

This command stores the present state of the instrument to the specified location in non-volatile memory. Up to 10 states can be stored in the memory locations 0 through 9. Any state previously stored in the same location will be overwritten. Use the *RCL command to retrieve instrument states.

Syntax *SAV *memory*

Parameter

memory	One of the memory locations 0 to 9. Parameter data type is NR1. The locations 0 to 4 are in the non-volatile memory, and the locations 5 to 9 are in the volatile memory.

Remarks If a particular state is desired at power-on, it should be stored in the location 0. It will then be automatically recalled at power turn-on if the Output Power-On state is set to *RCL 0.

Data described in "Non-Volatile Settings" on page 41 is not affected by the *SAV command.

The *RST command does not affect the configurations stored in memory. Once a state is stored, it remains until it is overwritten or specifically deleted.

CAUTION This command causes a write cycle to non-volatile memory. Non-volatile memory has a finite maximum number of write cycles. Programs that repeatedly cause write cycles to non-volatile memory can eventually exceed the maximum number of write cycles and cause the memory to fail.

*SRE

This command sets the value of the Service Request Enable register. This register determines which bits from the Status Byte register are summed to set the Master Status Summary (MSS) bit and the Request for Service (RQS) summary bit. A 1 in the bit position enables the corresponding event. For the SCPI status system, see "Status System Diagram" on page 37.

The query reads the enable register and returns a decimal value which corresponds to the binary-weighted sum of all bits set in the register.

Syntax *SRE *value*

*SRE?

Parameter

value	Decimal value which corresponds to the binary-weighted sum of the bits in the register (see Table 3-3). Parameter data type is NRf.

For example, to enable bit 0 (decimal value = 1), bit 3 (decimal value = 8), and bit 6 (decimal value = 64), the corresponding decimal value would be 73 (1 + 8 + 64).

Query response *value* <newline><^END>

value is the binary-weighted sum of all bits set in the register. For example, if bit 3 (decimal value = 8) and bit 7 (decimal value = 128) are enabled, the query command will return 136. Response data type is NR1.

Remarks Bit definitions of the Status Byte register are shown in Table 3-3.

All of the enabled events of the Standard Event Status Enable register are logically ORed to cause the Event Summary Bit (ESB) of the Status Byte register to be set. All such enabled bits are then logically ORed to cause the MSS bit (bit 6) of the Status Byte register to be set.

When the controller conducts a serial poll in response to SRQ, the RQS bit is cleared, but the MSS bit is not. When *SRE is cleared (by programming it with 0), the power system cannot generate an SRQ to the controller.

The *CLS (clear status) command will not clear the enable register but it does clear all bits in the event register.

A :STATus:PRESet command does not clear the bits in the Status Byte register.

Table 3-3 Status Byte Register Bit Definitions

Bit	Decimal value	Description	Definition
0	1	Measurement status summary	One or more bits are set in the Measurement Status register (bits must be enabled, see :STATus:<MEASurement|OPERation|QUEStionable>:ENABle command).
1	2	Not used	0 is returned.
2	4	Error queue not empty	One or more errors have been stored in the Error Queue (see :SYSTem:ERRor[:NEXT]? command).
3	8	Questionable status summary	One or more bits are set in the Questionable Status register (bits must be enabled, see :STATus:<MEASurement|OPERation|QUEStionable>:ENABle command).
4	16	Output buffer	Data is available in the instrument's output buffer.
5	32	Event status byte summary	One or more bits are set in the Standard Event register (bits must be enabled, see *ESE command).
6	64	Master status summary (Request for service)	One or more bits are set in the Status Byte register (bits must be enabled, see *SRE command). Also used to indicate a request for service.
7	128	Operation status summary	One or more bits are set in the Operation Status register (bits must be enabled, see :STATus:<MEASurement|OPERation|QUEStionable>:ENABle command).

*STB?

This query reads the Status Byte register, which contains the status summary bits and the Output Queue MAV bit. The Status Byte register is a read-only register and the bits are not cleared when it is read. For the SCPI status system, see "Status System Diagram" on page 37.

Syntax *STB?

Query response *register* <newline><^END>

register is the binary-weighted sum of all bits set in the register. For example, if bit 1 (decimal value = 2) and bit 4 (decimal value = 16) are set (and the corresponding bits are enabled), this command will return 18. Response data type is NR1.

Remarks Bit definitions of the Status Byte register are shown in Table 3-3.

The input summary bits are cleared when the appropriate event registers are read. The MAV bit is cleared at power-on, by *CLS, or when there is no more response data available.

A serial poll also returns the value of the Status Byte register, except that bit 6 returns Request for Service (RQS) instead of Master Status Summary (MSS). A serial poll clears RQS, but not MSS. When MSS is set, it indicates that the instrument has one or more reasons for requesting service.

*TRG

This common command generates a trigger when the trigger subsystem has BUS selected as its source. The command has the same affect as the Group Execute Trigger (GET) command.

Syntax *TRG

*TST?

This query causes the instrument to do a self-test and report any errors. A 0 indicates the instrument passed self-test. If all tests pass, you can have a high confidence that the instrument is operational.

Syntax *TST?

Query response *result* <newline><^END>

result is 0 or 1 that indicates the self-test result. Response data type is NR1.

0: all tests passed

1: one or more tests failed

Remarks If one or more tests fail, a 1 is returned and an error is stored in the error queue. For a complete listing of the error messages related to self-test failures, see Chapter 5, "Error Messages."

If one or more tests fail, see the Service Guide for instructions on returning the instrument to Keysight for service.

*TST? also forces an *RST command.

*WAI

This command instructs the instrument not to process any further commands until all pending operations are completed. Pending operations are as defined under the *OPC command.

Syntax *WAI

Remarks *WAI can be aborted only by sending the instrument a Device Clear command.

See Also *OPC

4 Subsystem Commands

This chapter describes subsystem commands available for Keysight B2980 in alphabetical order.

CALCulate Subsystem

For the numeric suffix [*c*], see "Numeric Suffix" on page 27.

:CALCulate:CLIMits:CLEar:AUTO

Enables or disables the automatic clear function of the composite limit test.

Syntax :CALCulate[*c*]:CLIMits:CLEar:AUTO *mode*

:CALCulate[*c*]:CLIMits:CLEar:AUTO?

Parameter ***mode*** 0|OFF|1|ON (default). Parameter data type is boolean.

mode = 1 or ON enables the automatic clear function which clears the composite limit test results and ports (GPIO lines) automatically with each :INITiate command. See ":INITiate[:IMMediate]<:ACQuire|:TRANsient|[:ALL]>" on page 344.

mode = 0 or OFF disables the automatic clear function. The composite limit test results and ports (GPIO lines) must be cleared manually before the next composite limit test is started. Execute the :CALCulate:CLIMits:CLEar[:IMMediate] command to clear them immediately.

Query response *mode* <newline>

mode is 0 or 1, and indicates that the automatic clear function is off or on, respectively. Response data type is NR1.

Example :CALC:CLIM:CLE:AUTO 1

:CALC:CLIM:CLE:AUTO?

:CALCulate:CLIMits:CLEar:AUTO:DELay

Sets the delay time for the automatic clear of the composite limit test. See ":CALCulate:CLIMits:CLEar:AUTO" on page 144. The delay time is defined as the time before the automatic clear is performed after the measurement is completed.

Syntax :CALCulate[*c*]:CLIMits:CLEar:AUTO:DELay *time*

:CALCulate[*c*]:CLIMits:CLEar:AUTO:DELay? [*time*]

Parameter	*time*	*value* (+1E–5 to 60 seconds)|MINimum|MAXimum|DEFault (default is +1E–4). Parameter data type is NRf+. Query does not support *time* = *value*. If you specify the value less than MIN or greater than MAX, *time* is automatically set to MIN or MAX.

Query response *time* <newline>

time returns the present setting of delay time for the automatic clear. If a parameter is specified, *time* returns the value assigned to DEF, MIN, or MAX. Response data type is NR3.

Example :CALC:CLIM:CLE:AUTO:DEL 1E–3

:CALC:CLIM:CLE:AUTO:DEL?

:CALCulate:CLIMits:CLEar[:IMMediate]

Clears the composite limit test results and ports (GPIO lines) immediately.

Syntax :CALCulate[*c*]:CLIMits:CLEar[:IMMediate]

Example :CALC:CLIM:CLE:IMM

:CALC:CLIM:CLE

:CALCulate:CLIMits:<FAIL|PASS>:DIGital[:DATA]

Defines a fail/pass pattern that appears near the end of the flowcharts shown in Figures 2-12 and 2-13. This is a bit pattern used to indicate the composite limit test result (fail or pass). It must be entered in the format set by the :FORMat:DIGital command.

Syntax :CALCulate[*c*]:CLIMits:<FAIL|PASS>:DIGital[:DATA] *bit_pattern*

:CALCulate[*c*]:CLIMits:<FAIL|PASS>:DIGital[:DATA]?

For <FAIL|PASS>, specify PASS for a pass pattern or FAIL for a fail pattern.

Parameter	*bit_pattern*	0 (default setting in decimal expression) to 127. Parameter data type is NR1 or NDN.

Query response *bit_pattern* <newline>

bit_pattern returns the fail/pass bit pattern in the format specified by the :FORMat:DIGital command. Response data type is NR1 or NDN.

Example :CALC:CLIM:FAIL:DIG:DATA 64

:CALC:CLIM:PASS:DIG?

:CALCulate:CLIMits:MODE

Sets the operation mode of the composite limit test to GRADing or SORTing.

Syntax :CALCulate[*c*]:CLIMits:MODE *mode*

:CALCulate[*c*]:CLIMits:MODE?

Parameter ***mode*** SORT (sorting)|GRAD (grading, default). Parameter data type is CPD.

mode = GRAD performs limit tests for up to 12 test limits until a failure is detected. See Figure 2-13 for an example of a flowchart under the grading mode.

mode = SORT performs limit tests for up to 12 test limits until a pass is detected. See Figure 2-12 for an example of a flowchart under the sorting mode.

A limit test is a pass/fail judgement performed during a composite limit test. It is performed at the “Pass?” step in Figures 2-12 and 2-13.

Query response *mode* <newline>

mode returns GRAD or SORT. Response data type is CPD.

Example :CALC:CLIM:MODE SORT

:CALC:CLIM:MODE?

:CALCulate:CLIMits:STATe

Enables or disables the composite limit test.

Syntax :CALCulate[*c*]:CLIMits:STATe *mode*

:CALCulate[*c*]:CLIMits:STATe?

Parameter ***mode*** 1|ON (default)|0|OFF. Parameter data type is boolean.

mode = 1 or ON enables the composite limit test.

mode = 0 or OFF disables the composite limit test.

Query response *mode* <newline>

mode is 0 or 1, and indicates that the composite limit test is off or on, respectively. Response data type is NR1 or NDN.

Example :CALC:CLIM:STAT 1

:CALC:CLIM:STAT?

:CALCulate:CLIMits:STATe:ANY?

Checks if the present composite limit test contains a limit test, which is a pass/fail judgement. It is performed at the "Pass?" step in Figures 2-12 and 2-13.

Syntax :CALCulate[*c*]:CLIMits:STATe:ANY?

Query response *status* <newline>

status returns 0 or 1. Response data type is NR1.

0: No limit test exists.

1: At least one limit test exists.

Example :CALC:CLIM:STAT:ANY?

:CALCulate:CLIMits:UPDate

Only for the GRAD composite limit test. Enables or disables the immediate result output or update. See "Immediate?" shown in Figure 2-13.

When enabled, the result output/update is executed immediately when the first failure or all pass is detected. The result is the pass/fail bit pattern defined by the :CALCulate:CLIMits:<FAIL|PASS>:DIGital[:DATA] command. If all pass is detected, the pattern will be the pass pattern.

Syntax :CALCulate[*c*]:CLIMits:UPDate *result*

:CALCulate[*c*]:CLIMits:UPDate?

Parameter ***result*** END|IMMediate (default). Parameter data type is CPD.

result = IMM enables immediate result output.

result = END disables immediate result output.

Query response *result* <newline>

result returns IMM or END. Response data type is CRD.

Example :CALC:CLIM:UPD END

:CALC:CLIM:UPD?

:CALCulate:DATA?

Returns limit test data. Elements of the returned data are specified by the :FORMat:ELEMents:CALCulate command. The limit test data can be expressed by the following formula.

limit test data = input data – null offset

input data: Data specified by :CALCulate:FEED

null offset: Data set by :CALCulate:OFFSet or :CALCulate:OFFSet:ACQuire

If the null offset function is disabled by the :CALCulate:OFFSet:STATe command, *null offset* = 0.

Syntax :CALCulate[*c*]:DATA? [*offset*[, *size*]]

Parameter

offset	Indicates the beginning of the data received. *n*|CURRent|STARt (default). Parameter data type is NR1 or CPD. *offset* = *n* specifies the *n+1*th data. *n* is an integer, 0 to maximum (depends on the buffer state). *offset* = CURR specifies the present data position. *offset* = STAR specifies the top of the data buffer. Same as *offset* = 0.
size	Number of data to be received. 1 to maximum (depends on the buffer state). Parameter data type is NR1. If this parameter is not specified, all data from *offset* is returned.

Query response *data* <newline>

Response data type is NR3. See "Data Output Format" on page 31.

Example :CALC:DATA? 0,10

:CALCulate:DATA:LATest?

Returns the latest limit test data. Elements of the returned data are specified by the :FORMat:ELEMents:CALCulate command. The limit test data can be expressed by the following formula.

limit test data = *input data* – *null offset*

input data: Data specified by :CALCulate:FEED

null offset: Data set by :CALCulate:OFFSet or :CALCulate:OFFSet:ACQuire

If the null offset function is disabled by the :CALCulate:OFFSet:STATe command, *null offset* = 0.

Syntax :CALCulate[*c*]:DATA:LATest?

Query response *data* <newline>

Response data type is NR3. See "Data Output Format" on page 31.

Example :CALC:DATA:LAT?

:CALCulate:DIGital:BIT

Assigns the GPIO pins used for the result output. The result is the pass/fail bit pattern defined by the :CALCulate:CLIMits:<FAIL|PASS>:DIGital[:DATA] command.

Syntax :CALCulate[*c*]:DIGital:BIT *pin*

:CALCulate[*c*]:DIGital:BIT?

Parameter

pin	EXT*n*|NONE (default). Parameter data type is CPD. EXT*n* specifies a GPIO pin, which is an output port of the Digital I/O D-sub connector on the rear panel. *n* = 1 to 7.

pin = NONE does not assign the GPIO pins.

To assign the GPIO pins, *pin* must be a comma separated EXT string like `EXTn,EXTn+1,`.... , and `EXTn` must be LSB. The specified pins must be continuous. For example, *pin* = `EXT2,EXT3,EXT4` is effective for this command, and EXT2, EXT3 and EXT4 are assigned to BIT0 (LSB), BIT1, and BIT2, respectively. On the contrary, non-continuous pin assignment such as *pin* = `EXT2,EXT3,EXT7` is not effective.

Query response *pin* <newline>

pin returns NONE or a comma separated EXT string. Response data type is CRD.

Example :CALC:DIG:BIT EXT2,EXT3,EXT4

:CALC:DIG:BIT?

:CALCulate:DIGital:<BUSY|EOT|SOT>

Assigns the GPIO pin for the BUSY (busy), EOT (end of test), or SOT (start of test) signal line for the composite limit test.

Syntax :CALCulate[*c*]:DIGital:<BUSY|EOT|SOT> *pin*

:CALCulate[*c*]:DIGital:<BUSY|EOT|SOT>?

For <BUSY|EOT|SOT>, specify BUSY for assigning the busy line, EOT for assigning the end of test line, or SOT for assigning the start of test line.

Parameter

pin	EXT*n*|NONE (default). Parameter data type is CPD. EXT*n* specifies a GPIO pin, which is an output port of the Digital I/O D-sub connector on the rear panel. *n* = 1 to 7.

To assign the GPIO pin, *pin* must be a EXT string like EXT*n*. For example, *pin* = EXT7.

pin = NONE does not assign the GPIO pin.

Query response *pin* <newline>

pin returns NONE or a EXT string. Response data type is CRD.

Example :CALC:DIG:EOT EXT7

:CALC:DIG:SOT?

:CALCulate:FEED

Specifies the *input data* value used for calculating the limit test data. The limit test is a pass/fail judgement performed during a composite limit test. The limit test is performed at the “Pass?” step in Figures 2-12 and 2-13. The limit test data is returned by the :CALCulate:DATA? or :CALCulate:DATA:LATest? command.

Syntax :CALCulate[*c*]:FEED *type*

:CALCulate[*c*]:FEED?

Parameter	*type*	Data type. MATH|RESistance|CURRent(default)|CHARge|VOLTage. Parameter data type is CPD.

NOTE On B2981B/B2983B, only MATH and CURR can be specified.

type = VOLT specifies the voltage measurement data.

type = CURR specifies the current measurement data.

type = CHAR specifies the charge measurement data.

type = RES specifies the resistance calculation data given by the following formula.

Resistance = Vsour/Imeas or Vmeas/Imeas

Where, Vsour is the voltage force data, Vmeas is the voltage measurement data, and Imeas is the current measurement data.

type = MATH specifies the data given by a math expression. The math expression must be specified before the :CALC:FEED MATH command is executed.

An existing math expression can be specified by the :CALCulate:MATH[:EXPRession]:NAME command.

A new math expression can be defined by the :CALCulate:MATH[:EXPRession]:NAME and :CALCulate:MATH[:EXPRession][:DEFine] commands.

Query response *type* <newline>

type returns the present setting of data type, MATH, RES, CURR, CHAR or VOLT. Response data type is CRD.

Example

```
:CALC:FEED MATH
:CALC:FEED?
```

:CALCulate:LIMit:COMPliance:DIGital[:DATA]

Defines the bit pattern used to indicate a *failure* of the measurement range overflow status check specified by *m*. It must be entered in the format set by the :FORMat:DIGital command.

Syntax :CALCulate[*c*]:LIMit[*m*]:COMPliance:DIGital[:DATA] *bit_pattern*

:CALCulate[*c*]:LIMit[*m*]:COMPliance:DIGital[:DATA]?

Parameter ***bit_pattern*** 0 (default setting in decimal expression) to 127. Parameter data type is NR1 or NDN.

Query response *bit_pattern* <newline>

bit_pattern returns the fail bit pattern in the format specified by the :FORMat:DIGital command. Response data type is NR1 or NDN.

Example :CALC:LIM:COMP:DIG:DATA 64

:CALC:LIM12:COMP:DIG?

:CALCulate:LIMit:COMPliance:FAIL

Sets the judgement criteria for the measurement range overflow status check specified by *m*.

Syntax :CALCulate[*c*]:LIMit[*m*]:COMPliance:FAIL *criteria*

:CALCulate[*c*]:LIMit[*m*]:COMPliance:FAIL?

Parameter ***criteria*** OUT|IN (default). Parameter data type is CPD.

criteria = IN determines that the limit test has failed if the channel goes into the measurement range overflow state.

criteria = OUT determines that the limit test has failed if the channel comes out of the measurement range overflow state.

Query response *criteria* <newline>

criteria returns IN or OUT which indicates the present setting of the judgement criteria. Response data type is CRD.

Example :CALC:LIM:COMP:FAIL OUT

:CALC:LIM12:COMP:FAIL?

:CALCulate:LIMit:FAIL?

Returns the result of the limit test specified by *m*.

Syntax :CALCulate[*c*]:LIMit[*m*]:FAIL?

Query response *result* <newline>

result returns 0 or 1. Response data type is NR1.

0: Passed

1: Failed

Example :CALC:LIM12:FAIL?

:CALCulate:LIMit:FUNCtion

Sets the type of the limit test specified by *m*.

Syntax :CALCulate[*c*]:LIMit[*m*]:FUNCtion *type*

:CALCulate[*c*]:LIMit[*m*]:FUNCtion?

Parameter *type* COMPliance|LIMit (default). Parameter data type is CPD.

type = COMP sets the measurement range overflow status check which checks if the channel is in the measurement range overflow status.

type = LIM sets the limit test which checks if the measurement value is between the upper limit and the lower limit.

Query response *type* <newline>

type returns the present setting of the type, COMP or LIM. Response data type is CRD.

Example :CALC:LIM:FUNC COMP

:CALC:LIM12:FUNC?

:CALCulate:LIMit:<LOWer|UPPer>

Sets a lower/upper limit used for the limit test specified by *m*.

Syntax :CALCulate[*c*]:LIMit[*m*]:<LOWer|UPPer> *limit*

:CALCulate[*c*]:LIMit[*m*]:<LOWer|UPPer>? [*limit*]

For <LOWer|UPPer>, specify LOWer for lower limit, or UPPer for upper limit.

Parameter ***limit*** *value* (–9.999999E+20 to +9.999999E+20)|MINimum|MAXimum|DEFault (default is –1 for the lower limit and +1 for the upper limit). Parameter data type is NRf+. Query does not support *limit* = *value*.

Query response *limit* <newline>

limit returns the present setting of the lower/upper limit used for the limit test specified by *m*. If a parameter is specified, *limit* returns the value assigned to DEF, MIN, or MAX. Response data type is NR3.

Example :CALC:LIM:LOW –2.5

:CALC:LIM12:UPP?

:CALCulate:LIMit:<LOWer|UPPer>:DIGital[:DATA]

Defines the bit pattern used to indicate *failed-by-exceeding-lower-limit* or *failed-by-exceeding-upper-limit* of the limit test specified by *m*. It must be entered in the format set by the :FORMat:DIGital command. The bit pattern defined by this command is used for the GRAD composite limit test.

Syntax :CALCulate[*c*]:LIMit[*m*]:<LOWer|UPPer>:DIGital[:DATA] *bit_pattern*

:CALCulate[*c*]:LIMit[*m*]:<LOWer|UPPer>:DIGital[:DATA]?

For <LOWer|UPPer>, specify LOWer for *failed-by-exceeding-lower-limit*, or UPPer for *failed-by-exceeding-upper-limit*.

Parameter ***bit_pattern*** 0 (default setting in decimal expression) to 127. Parameter data type is NR1 or NDN.

Query response *bit_pattern* <newline>

bit_pattern returns the fail bit pattern in the format specified by the :FORMat:DIGital command. Response data type is NR1 or NDN.

Example :CALC:LIM:LOW:DIG:DATA 64

:CALC:LIM12:UPP:DIG?

:CALCulate:LIMit:PASS:DIGital[:DATA]

Defines the bit pattern used to indicate a *pass* of the limit test specified by *m*. It must be entered in the format set by the :FORMat:DIGital command. The bit pattern defined by this command is used for the SORT composite limit test.

Syntax :CALCulate[*c*]:LIMit[*m*]:PASS:DIGital[:DATA] *bit_pattern*

:CALCulate[*c*]:LIMit[*m*]:PASS:DIGital[:DATA]?

Parameter ***bit_pattern*** 0 (default setting in decimal expression) to 127. Parameter data type is NR1 or NDN.

Query response *bit_pattern* <newline>

bit_pattern returns the pass bit pattern in the format specified by the :FORMat:DIGital command. Response data type is NR1 or NDN.

Example :CALC:LIM:PASS:DIG:DATA 64

:CALC:LIM12:PASS:DIG?

:CALCulate:LIMit:STATe

Enables or disables the limit test specified by *m*.

Syntax :CALCulate[*c*]:LIMit[*m*]:STATe *mode*

:CALCulate[*c*]:LIMit[*m*]:STATe?

Parameter ***mode*** 1|ON|0|OFF (default). Parameter data type is boolean.

mode = 1 or ON enables the limit test specified by *m*.

mode = 0 or OFF disables the limit test specified by *m*.

Query response *mode* <newline>

mode is 0 or 1, and indicates that the limit test is off or on, respectively. Response data type is NR1.

Example :CALC:LIM:STAT 1

:CALC:LIM12:STAT?

:CALCulate:MATH:DATA?

Returns the calculation result data. Elements of the returned data are specified by the :FORMat:ELEMents:CALCulate command. Math expression for the calculation is defined by the :CALCulate:MATH[:EXPRession]:NAME and :CALCulate:MATH[:EXPRession][:DEFine]commands.

Syntax :CALCulate[*c*]:MATH:DATA? [*offset*[, *size*]]

Parameter

offset	Indicates the beginning of the data received. *n*|CURRent|STARt (default). Parameter data type is NR1 or CPD. *offset* = *n* specifies the *n+1*th data. *n* is an integer, 0 to maximum (depends on the buffer state). *offset* = CURR specifies the present data position. *offset* = STAR specifies the top of the data buffer. Same as *offset* = 0.
size	Number of data to be received. 1 to maximum (depends on the buffer state). Parameter data type is NR1. If this parameter is not specified, all data from *offset* is returned.

Query response *data* <newline>

Response data type is NR3.

See "Data Output Format" on page 31.

Example :CALC:MATH:DATA? 0,10

Remarks If the math expression contains some measurement results, measurement may be performed many times to obtain the result. For example, measurement must be performed twice to get the result of the following math expression.

math expression = (CURR[1]–CURR[0])

:CALCulate:MATH:DATA:LATest?

Returns the latest calculation result data. Elements of the returned data are specified by the :FORMat:ELEMents:CALCulate command. Math expression for the calculation is defined by the :CALCulate:MATH[:EXPRession]:NAME and :CALCulate:MATH[:EXPRession][:DEFine]commands.

Syntax :CALCulate[*c*]:MATH:DATA:LATest?

Query response *data* <newline>

Response data type is NR3. See "Data Output Format" on page 31.

Example :CALC:MATH:DATA:LAT?

:CALCulate:MATH[:EXPRession]:CATalog?

Returns the list of all the predefined and user-defined math expression names.

Syntax :CALCulate[*c*]:MATH[:EXPRession]:CATalog?

Query response *catalog* <newline>

catalog returns all of the predefined and user-defined math expression names. Response data type is AARD.

Example :CALC:MATH:EXPR:CAT?

:CALCulate:MATH[:EXPRession][:DEFine]

Defines a math expression which will be a user-defined math expression. For the resources effective for the expression, see "Resources used in the expressions" on page 158. Also see "Predefined math expressions" on page 160 for the definition of predefined expressions.

Before executing this command, the math expression must be selected by the:CALCulate:MATH[:EXPRession]:NAME command.

Syntax :CALCulate[*c*]:MATH[:EXPRession][:DEFine] *definition*

:CALCulate[*c*]:MATH[:EXPRession][:DEFine]?

Parameter

definition	Definition of a math expression. Up to 256 ASCII characters. Parameter data type is Expr. The expression must be enclosed by parentheses. Maximum of 32 math expressions can be defined including the predefined math expressions.

Query response *definition* <newline>

definition returns the definition of the math expression currently selected. Response data type is Expr.

Example

```
:CALC:MATH:EXPR:NAME “Expression”
:CALC:MATH:EXPR:DEF ((CURR[1]–CURR[0])*(CURR[1]))

:CALC:MATH?
```

Resources used in the expressions

The following resources can be used in user-defined math expressions.

- Reserved variables

  The variables listed in Table 4-1 are reserved for reading the channel output or measurement data.

  Scalar variable is used for spot measurement data.

  Vector (array) variable is used for sweep measurement data.

- Reserved user variables

  The variables listed in Table 4-2 are reserved for reading the channel output or measurement data.

  Scalar variable is used for spot measurement data.

  Vector (array) variable is used for sweep measurement data.

- Math operators

  The following operators are available.

  - Arithmetic operators: +, –, *, /, ^, see Table 4-3
  - Elementary functions: ln, log, sin, cos, tan, exp

  The functions log and ln perform the operation after calculating the absolute value. So if a negative value is specified, they do not result in an error, but calculate as if a positive value was specified. For example, log(–10) results in log(10) = 1.

- Numeric value

  Decimal (0 to 4294967294, 4294967295 indicates –1), binary (32 bit, 0 to 0b11111111111111111111111111111111), or hexadecimal (0 to 0xFFFFFFFF).

Table 4-1 Reserved Variables

Reserved variable [a]		Description
Scalar	Vector	
CURR[*c*]	CURR[*c*][]	Current measurement data
CHAR[*c*]	CHAR[*c*][]	Charge measurement data
VOLT[*c*]	VOLT[*c*][]	Voltage measurement data
RES[*c*]	RES[*c*][]	Resistance measurement data
SOUR[*c*]	SOUR[*c*][]	Source output setting data
TIME[*c*]	TIME[*c*][]	Time (timestamp) data
TEMP[*c*]	TEMP[*c*][]	Temperature data
HUM[*c*]	HUM[*c*][]	Humidity data

a. The numeric suffix [*c*] is effective for specifying the channel. See "Numeric Suffix" on page 27.
On B2981B/B2983B, only CURR can be used.

Table 4-2 Reserved User Variables

Reserved variable	Description
M, B	Used in MXPLUSB, MRECPXPLUSB
TARGET	Used in RATIO, PERCENT, DEVIATION, PERDIV
TARGET	Voltage measurement data
A0, A1, A2	Used in POLYNOMINAL
EPER	Effective perimeter for surface resistivity measurement
GLEN	Gap length for surface resistivity measurement
EAR	Effective area for volume resistivity measurement
STH	Sample thickness for volume resistivity measurement

Table 4-3 Arithmetic and Unary Operators

Priority of task	Operator	Description
High	( )	Parentheses
⋮	+ and –	Unary plus operator and unary minus operator
⋮	^	Exponentiation operator
⋮	* and /	Multiplication operator and division operator
Low	+ and –	Additive operator and subtraction operator

Predefined math expressions

The following math expressions are already defined in the instrument. The predefined math expressions are not cleared by the power off/on operations.

- MX+B (MXPLUSB)
- M/X+B (MRECPXPLUSB)
- Ratio (RATIO)
- Percentage (PERCENT)
- Deviation (DEVIATION)
- Percent Deviation (PERDEV)
- Logarithm (LOG10)
- Polynomial (POLYNOMIAL)
- Surface Resistivity (SRESISTIVITY)
- Volume Resistivity (VRESISTIVITY)
- Power (POWER)
- Offset Compensated Ohms (OFFCOMPOHM)
- Voltage Coefficient (VOLTCOEF)
- Varistor Alpha (VARALPHA)

MXPLUSB Calculates MX+B from user variables M and B using the following formula.

M * CURR[*c*] + B

MRECPXPLUSB Calculates M/X+B from user variables M and B using the following formula.

M / CURR[*c*] + B

RATIO Calculates ratio from user variable TARGET using the following formula.

CURR[*c*] / TARGET

PERCENT Calculates percentage from user variable TARGET using the following formula.

CURR[*c*] / TARGET * 100

DEVIATION Calculates deviation from user variable TARGET using the following formula.

(CURR[*c*] – TARGET) / TARGET

PERDIV Calculates percent deviation from user variable TARGET using the following formula.

(CURR[*c*] – TARGET) / TARGET * 100

LOG10 Calculates logarithm from user variable TARGET using the following formula.

Log(CURR[*c*])

POLYNOMIAL Calculates polynomial from user variables A0, A1, and A2 using the following formula.

A2 * (CURR[*c*])^2 + A1 * CURR[*c*] + A0

SRESISTIVITY Calculates surface resistivity from user variables EPER and GLEN using the following formula.

NOTE B2985B/B2987B can use this math expression.

EPER / GLEN * RES[*c*]

VRESISTIVITY Calculates volume resistivity from user variables EAR and STH using the following formula.

NOTE B2985B/B2987B can use this math expression.

EAR / STH * RES[*c*] / 10

POWER Calculates power using the following formula.

NOTE B2985B/B2987B can use this math expression.

POWER = VOLT[*c*] * CURR[*c*]

OFFCOMPOHM Calculates offset compensated ohms (resistance) using the following formula.

NOTE B2985B/B2987B can use this math expression.

OFFCOMPOHM = (VOLT[*c*][1]–VOLT[*c*][0]) / (CURR[*c*][1]–CURR[*c*][0])

where, VOLT[*c*][0] and CURR[*c*][0] are data measured in a condition, and VOLT[*c*][1] and CURR[*c*][1] are data measured in another condition.

VARALPHA Calculates varistor alpha using the following formula.

NOTE B2985B/B2987B can use this math expression.

VARALPHA = log(CURR[*c*][1] / CURR[*c*][0]) / log(VOLT[*c*][1] / VOLT[*c*][0])

where, CURR[*c*][0] and VOLT[*c*][0] are measurement data at a point on a varistor's non-linear I-V characteristics curve, and CURR[*c*][1] and VOLT[*c*][1] are data at another point.

VOLTCOEF Calculates voltage coefficient using the following formula.

NOTE B2985B/B2987B can use this math expression.

VOLTCOEF = (RES[*c*][1]–RES[*c*][0]) / (RES[*c*][1] * (VOLT[*c*][1]–VOLT[*c*][0])) * 100%

where, RES[*c*][0] and RES[*c*][1] are resistance measurement data at the first and second measurement points, respectively, and VOLT[*c*][0] and VOLT[*c*][1] are voltage measurement data at the first and second measurement points, respectively.

The voltage coefficient is known as the ratio of the fractional change for a resistor whose resistance varies with voltage.

:CALCulate:MATH[:EXPRession]:DELete:ALL

Deletes all user-defined math expressions. This command cannot delete predefined math expressions.

Syntax :CALCulate[*c*]:MATH[:EXPRession]:DELete:ALL

Example :CALC:MATH:EXPR:DEL:ALL

:CALCulate:MATH[:EXPRession]:DELete[:SELected]

Deletes an user-defined math expression. This command cannot delete a predefined math expression.

Syntax :CALCulate[*c*]:MATH[:EXPRession]:DELete[:SELected] *name*

Parameter

name	Name of the math expression to delete. Up to 32 ASCII characters. Parameter data type is SPD.

Example :CALC:MATH:EXPR:DEL:SEL “TempExpression1”

:CALCulate:MATH[:EXPRession]:NAME

Selects a math expression used for calculation. A predefined math expression or an user-defined math expression can be specified by the *name* parameter.

See “Predefined math expressions” on page 160 for the definition of predefined math expressions.

A new user-defined math expression can be added by executing this command with a new name, and executing the :CALCulate:MATH[:EXPRession][:DEFine] command with a new definition.

Existing user-defined math expression can be changed by executing this command with its name, and executing the :CALCulate:MATH[:EXPRession][:DEFine] command with a new definition.

Syntax :CALCulate[*c*]:MATH[:EXPRession]:NAME *name*

:CALCulate[*c*]:MATH[:EXPRession]:NAME?

Parameter

name	Name of a math expression. Up to 32 ASCII characters without any control characters, space characters, single and double quotes, and comma. Parameter data type is SPD.

Query response *name* <newline>

name returns the name of the math expression currently selected. For example, *name* returns “Expression1”. Response data type is SRD.

Example :CALC:MATH:EXPR:NAME “Expression1”

:CALC:MATH:NAME?

:CALCulate:MATH:STATe

Enables or disables the math expression.

Syntax :CALCulate[*c*]:MATH:STATe *mode*

:CALCulate[*c*]:MATH:STATe?

Parameter ***mode*** 1|ON|0|OFF (default). Parameter data type is boolean.

mode = 1 or ON enables the math expression.

mode = 0 or OFF disables the math expression.

Query response *mode* <newline>

mode is 0 or 1, and indicates that the math expression is off or on, respectively. Response data type is NR1.

Example :CALC:MATH:STAT 1

:CALC:MATH:STAT?

:CALCulate:MATH:UNITs

Defines the unit name for the math expression.

Syntax :CALCulate[*c*]:MATH:UNITs *unit*

:CALCulate[*c*]:MATH:UNITs?

Parameter ***unit*** Unit name. Up to 32 ASCII characters. Parameter data type is SPD.

Query response *unit* <newline>

unit returns the unit name of the math expression. Response data type is SRD.

Example :CALC:MATH:UNIT “amps”

:CALC:MATH:UNIT?

:CALCulate:MATH:VARiable:CATalog?

Returns the list of all the predefined and user-defined math variable names.

Syntax :CALCulate[*c*]:MATH:VARiable:CATalog?

Query response *catalog* <newline>

catalog returns all of the predefined and user-defined math variable names. Response data type is AARD. For example, if the instrument stores the math variables M, B, A0, A1, A2, TARGET, EPER, GLEN and STH, *catalog* returns M,B,A0,A1,A2,TARGET,EPER,GLEN,STH.

Example :CALC:MATH:VAR:CAT?

:CALCulate:MATH:VARiable[:DEFine]

Defines a math variable which will be a user-defined math variable. For the resources effective for the variable, see “Resources used in the expressions” on page 158. Also see “Predefined math expressions” on page 160 for the definition of predefined variables.

Before executing this command, the math variable must be selected by the:CALCulate:MATH:VARiable:NAME command.

Syntax :CALCulate[*c*]:MATH:VARiable[:DEFine] *definition*

:CALCulate[*c*]:MATH:VARiable[:DEFine]?

Parameter ***definition*** Definition of a math variable. Parameter data type is NRf.

A range of math variable value must be from –9.999999E+20 to +9.999999E+20. Default value of defined math variable is 0.0.

Query response *definition* <newline>

definition returns the value of the math variable currently selected. Response data type is NR3.

Example :CALC:MATH:VAR:NAME “TempVariable1”
:CALC:MATH:VAR:DEF 10.0

:CALCulate:MATH:VARiable:DELete:ALL

Deletes all user-defined math variables. This command does not delete predefined math variables.

Syntax :CALCulate[*c*]:MATH:VARiable:DELete:ALL

Example :CALC:MATH:VAR:DEL:ALL

:CALCulate:MATH:VARiable:DELete[:SELected]

Deletes an user-defined math variable. This command cannot delete a predefined math variable.

Syntax :CALCulate[*c*]:MATH:VARiable:DELete[:SELected] *name*

Parameter

name	Name of the math variable to delete. Up to 16 ASCII characters. Parameter data type is SPD.

Example :CALC:MATH:VAR:DEL:SEL “TempVariable1”

:CALCulate:MATH:VARiable:NAME

Selects a math variable used for calculation. A predefined math variable or an user-defined math variable can be specified by the *name* parameter.

See “Predefined math expressions” on page 160 for the definition of predefined math variables.

A new user-defined math variable can be added by executing this command with a new name, and executing the :CALCulate:MATH:VARiable[:DEFine] command with a new definition.

Existing user-defined math variable can be changed by executing this command with its name, and executing the :CALCulate:MATH:VARiable[:DEFine] command with a new definition.

Syntax :CALCulate[*c*]:MATH:VARiable:NAME *name*

:CALCulate[*c*]:MATH:VARiable:NAME?

Parameter

name	Name of a math variable. Up to 16 ASCII characters without any control characters, space characters, single and double quotes, and comma. Parameter data type is SPD.

Query response *name* <newline>

name returns the name of the math variable currently selected. For example, *name* returns “TempVariable1”. Response data type is SRD.

Example :CALC:MATH:VAR:NAME “TempVariable1”

:CALC:MATH:NAME?

:CALCulate:OFFSet

Sets the *null offset* value used for calculating the limit test data.

The null offset function is enabled by the :CALCulate:OFFSet:STATe command.

Syntax :CALCulate[*c*]:OFFSet *offset*

:CALCulate[*c*]:OFFSet? [*offset*]

Parameter

offset	*value* (–9.999999E+20 to +9.999999E+20)|MINimum|MAXimum|DEFault (default is 0.0). Parameter data type is NRf+. Query does not support *offset* = *value*.

Query response *offset* <newline>

offset returns the present setting of the null offset value. If a parameter is specified, *offset* returns the value assigned to DEF, MIN, or MAX. Response data type is NR3.

Example :CALC:OFFS 0.5

:CALC:OFFS?

:CALCulate:OFFSet:ACQuire

Automatically sets the *null offset* value used for calculating the limit test data. The *null offset* value will be the currently available value read by the :CALCulate:DATA:LATest? or [:SENSe]:DATA:ACQuire command. Or, it will be 0.0 if a currently available value does not exist.

Syntax :CALCulate[*c*]:OFFSet:ACQuire

Example :CALC:OFFS:ACQ

:CALC:OFFS:ACQ

:CALCulate:OFFSet:STATe

Enables or disables the null offset function used for calculating the limit test data.

The null offset value is set by the :CALCulate:OFFSet or :CALCulate:OFFSet:ACQuire command.

Syntax :CALCulate[*c*]:OFFSet:STATe *mode*

:CALCulate[*c*]:OFFSet:STATe?

Parameter ***mode*** 1|ON|0|OFF (default). Parameter data type is boolean.

mode = 1 or ON enables the null offset.

mode = 0 or OFF disables the null offset.

Query response *mode* <newline>

mode is 0 or 1, and indicates that the null offset is off or on, respectively. Response data type is NR1.

Example :CALC:OFFS:STAT 1

:CALC:OFFS:STAT?

DISPlay Subsystem

For the numeric suffix [*d*], see "Numeric Suffix" on page 27.

:DISPlay:CSET

Selects the color set of the front panel display. This command setting is not changed by power off or the *RST command.

Syntax :DISPlay:CSET *color*

:DISPlay:CSET?

Parameter ***color*** Color set of the front panel display. 1|2. Parameter data type is NR1.

color = 1 selects the default color set.

color = 2 selects the alternative color set.

Query response *response* <newline>

response is 1 or 2, and indicates the color set of the front panel display. Response data type is NR1.

Example :DISP:CSET 1

:DISP:CSET?

:DISPlay:ENABle

Enables or disables the front panel display under remote operation. This command setting is not changed by power off or the *RST command.

Regardless of this command setting, the front panel keys and the rotary knob are disabled during remote operation. However, only the *Local* key is effective for returning the instrument to local operation.

Syntax :DISP:ENAB *mode*

:DISP:ENAB?

Parameter *mode* 0|OFF|1|ON. Parameter data type is boolean.

mode = 1 or ON enables the front panel display.

mode = 0 or OFF disables the front panel display.

Query response *mode* <newline>

mode is 0 or 1, and indicates that front panel display is off or on, respectively. Response data type is NR1.

Example :DISP:ENAB OFF

:DISP:ENAB?

:DISPlay:VIEW

Sets the display mode to single 1, graph, roll, or histogram.

Syntax :DISPlay:VIEW *mode*

:DISPlay:VIEW?

Parameter *mode* SINGle1|GRAPh|ROLL|HISTogram. Parameter data type is CPD.

mode = SINGle1 sets the display mode to display channel 1 data on the Meter view. Default setting for 1-channel models.

mode = GRAPh sets the display mode to display the sweep measurement results on the Graph view.

mode = ROLL sets the display mode to display the time domain measurement results on the Roll view.

mode = HISTogram sets display mode to perform the simple statistical analysis on the Histogram view.

Query response *mode* <newline>

mode returns SING1, GRAP, ROLL, or HIST. Response data type is CRD.

Example :DISP:VIEW GRAP

:DISP:VIEW?

:DISPlay:VIEW:GRAPh:<X|Y>[:ELEMent]

Sets the data type assignment for X-axis or Y-axis on the Graph view.

Syntax :DISPlay:VIEW:GRAPh:<X|Y>[:ELEMent] *mode*

:DISPlay:VIEW:GRAPh:<X|Y>[:ELEMent]?

For <X|Y>, specify X for the X-axis or Y for the Y-axis.

Parameter

mode	CHARge|CURRent (default for Y-axis)|RESistance| MATH|SOURce (default for X-axis of B2985B/B2987B)| TIME (default for X-axis of B2981B/B2983B)|VOLTage. Parameter data type is CPD.

NOTE B2981B/B2983B can specify CURRent, MATH, or TIME only.

mode = CHARge selects the charge measurement data.

mode = CURRent selects the current measurement data.

mode = RESistance selects the resistance measurement data.

mode = MATH selects the math results.

mode = SOURce selects the source setting values.(Only for X-axis.)

mode = TIME selects the time data.(Only for X-axis.)

mode = VOLTage selects the voltage measurement data.

Query response *mode* <newline>

mode returns CHAR, CURR, RES, MATH, SOUR, TIME or VOLT. Response data type is CRD.

Example

```
:DISP:VIEW:GRAP:X CURR
:DISP:VIEW:GRAP:Y:ELEM?
```

:DISPlay:VIEW:GRAPh:<X|Y>:MAXimum

Sets the maximum value of X-axis or Y-axis on the Graph view.

Syntax :DISPlay:VIEW:GRAPh:<X|Y>:MAXimum *value*

:DISPlay:VIEW:GRAPh:<X|Y>:MAXimum?

For <X|Y>, specify X for the X-axis or Y for the Y-axis.

Parameter ***value*** Maximum value. Parameter data type is NRf.

Query response *value* <newline>

value returns the present setting of the maximum value of the X-axis or Y-axis. Response data type is NR3.

Example :DISP:VIEW:GRAP:X:MAX 1E+2

:DISP:VIEW:GRAP:Y:MAX?

:DISPlay:VIEW:GRAPh:<X|Y>:MINimum

Sets the minimum value of X-axis or Y-axis on the Graph view.

Syntax :DISPlay:VIEW:GRAPh:<X|Y>:MINimum *value*

:DISPlay:VIEW:GRAPh:<X|Y>:MINimum?

For <X|Y>, specify X for the X-axis or Y for the Y-axis.

Parameter ***value*** Minimum value. Parameter data type is NRf.

Query response *value* <newline>

value returns the present setting of the minimum value of the X-axis or Y-axis. Response data type is NR3.

Example :DISP:VIEW:GRAP:X:MIN 1E–12

:DISP:VIEW:GRAP:Y:MIN?

:DISPlay:VIEW:GRAPh:<X|Y>:SPACing

Selects linear scale or logarithmic scale for X-axis or Y-axis on the Graph view.

Syntax :DISPlay:VIEW:GRAPh:<X|Y>:SPACing *mode*

:DISPlay:VIEW:GRAPh:<X|Y>:SPACing?

For <X|Y>, specify X for the X-axis or Y for the Y-axis.

Parameter *mode* LINear(default)|LOGarithmic. Parameter data type is CPD.

mode = LINear selects the linear scale.

mode = LOGarithmic selects the logarithmic scale.

Query response *mode* <newline>

mode returns LIN or LOG. Response data type is CRD.

Example :DISP:VIEW:GRAP:X:SPAC LIN

:DISP:VIEW:GRAP:Y:SPAC?

:DISPlay:VIEW:<GRAPh|HISTogram|ROLL>:SCALe:AUTO

Changes the graph scale to fit the trace automatically on the specified display mode.

Syntax :DISPlay:VIEW:<GRAPh|HISTogram|ROLL>:SCALe:AUTO [*mode*]

For <GRAPh|HISTogram|ROLL>, specify GRAPh for auto scaling on the Graph view, HISTogram for auto scaling on the Histogram view, or ROLL for auto scaling on the Roll view.

Parameter *mode* ONCE. Parameter data type is CPD.

mode = ONCE performs an auto scaling function once.

If this parameter is not specified, *mode* = ONCE is set.

Example :DISP:VIEW:GRAP:SCAL:AUTO

:DISPlay:VIEW:<GRAPh|ROLL>:CURSor:STATe

Selects if the cursor on the specified display mode is displayed or not.

Syntax :DISPlay:VIEW:<GRAPh|ROLL>:CURSor:STATe *mode*

:DISPlay:VIEW:<GRAPh|ROLL>:CURSor:STATe?

For <GRAPh|ROLL>, specify GRAPh for cursor display on Graph view, or ROLL for cursor display on the Roll view.

Parameter *mode* 1|ON|0|OFF (default). Parameter data type is boolean.

mode = 1 or ON enables the cursor display.

mode = 0 or OFF disables the cursor display.

Query response *mode* <newline>

mode is 0 or 1, and indicates that the cursor display on the specified display mode is disabled or enabled, respectively. Response data type is NR1.

Example :DISP:VIEW:GRAP:CURS:STAT ON

:DISP:VIEW:ROLL:CURS:STAT?

:DISPlay:VIEW:<GRAPh|ROLL>:RLINe:STATe

Selects if the reference line function on the specified display mode is enabled or disabled.

Syntax :DISPlay:VIEW:<GRAPh|ROLL>:RLINe:STATe *mode*

:DISPlay:VIEW:<GRAPh|ROLL>:RLINe:STATe?

For <GRAPh|ROLL>, specify GRAPh to display reference line on the Graph view, or ROLL to display reference line on the Roll view.

Parameter ***mode*** 1|ON|0|OFF (default). Parameter data type is boolean.

mode = 1 or ON enables the reference line function.

mode = 0 or OFF disables the reference line function.

Query response *mode* <newline>

mode is 0 or 1, and indicates that the reference line function is disabled or enabled, respectively. Response data type is NR1.

Example :DISP:VIEW:GRAP:RLIN:STAT ON

:DISP:VIEW:ROLL:RLIN:STAT?

:DISPlay:VIEW:<GRAPh|ROLL>:RLINe:STORe

Saves the displayed line data as the reference line data on the specified display mode.

Syntax :DISPlay:VIEW:<GRAPh|ROLL>:RLINe:STORe

For <GRAPh|ROLL>, specify GRAPh to save line data on the Graph view, or ROLL to save line data on the Roll view.

Example :DISP:VIEW:GRAP:RLIN:STOR

:DISPlay:VIEW:HISTogram:Y:MAXimum

Sets the maximum value of Y-axis on the Histogram view.

Syntax :DISPlay:VIEW:HISTogram:Y:MAXimum *value*

:DISPlay:VIEW:HISTogram:Y:MAXimum?

Parameter ***value*** Maximum value. Parameter data type is NRf.

Query response *value* <newline>

value returns the present maximum value setting of Y-axis. Response data type is NR3.

Example :DISP:VIEW:HIST:Y:MAX 2.1E–2

:DISP:VIEW:HIST:Y:MAX?

:DISPlay:VIEW:<HISTogram|ROLL>:Y:ELEMent

Sets the data type assignment for Y-axis on the Histogram or Roll view.

Syntax :DISPlay:VIEW:<HISTogram|ROLL>:Y:ELEMent *mode*

:DISPlay:VIEW:<HISTogram|ROLL>:Y:ELEMent?

For <HISTogram|ROLL>, specify HIStogram for the data type assignment on Histogram view or ROLL for the data type assignment on Roll view.

Parameter ***mode*** CHARge|CURRent(default)|RESistance|VOLTage. Parameter data type is CPD.

NOTE B2981B/B2983B can specify CURRent only.

mode = CHARge selects the charge measurement data.

mode = CURRent selects the current measurement data.

mode = RESistance selects the resistance measurement data.

mode = VOLTage selects the voltage measurement data.

Query response *mode* <newline>

mode returns CHAR, CURR, RES or VOLT. Response data type is CRD.

Example :DISP:VIEW:HIST:Y:ELEM CURR

:DISP:VIEW:ROLL:Y:ELEM?

:DISPlay:VIEW:ROLL:X:<OFFSet|PDIVision>

Sets the value of scale division or offset value of X-axis on the Roll view.

Syntax :DISPlay:VIEW:ROLL:X:<OFFSet|PDIVision> *value*

:DISPlay:VIEW:ROLL:X:<OFFSet|PDIVision>?

Parameter ***value*** Offset value or value of scale division. Parameter data type is NRf.

Query response *value* <newline>

value returns the present setting of the offset or scale division of X-axis. Response data type is NR3.

Example :DISP:VIEW:ROLL:X:OFFS 1.0E–5

:DISP:VIEW:ROLL:X:PDIV?

:DISPlay:VIEW:ROLL:Y:OFFSet:<CHARge|CURRent|RESistance|VOLTage>

Sets the offset value of Y-axis on Roll view.

Syntax :DISPlay:VIEW:ROLL:Y:OFFSet:<CHARge|CURRent|RESistance|VOLTage> *value*

:DISPlay:VIEW:ROLL:Y:OFFSet:<CHARge|CURRent|RESistance|VOLTage>*?*

For <CHARge|CURRent|RESistance|VOLTage>, specify CHARge for charge offset value of Y-axis, CURRent for current offset value of Y-axis, RESistance for resistance offset value of Y-axis, or VOLTage for voltage offset value of Y-axis.

Parameter ***value*** Offset value. Parameter data type is NRf.

Query response *value* <newline>

value returns the present offset value setting of Y-axis. Response data type is NR3.

Example :DISP:VIEW:ROLL:Y:OFFS:CURR 1.0E–6

:DISP:VIEW:ROLL:Y:OFFS:CURR?

:DISPlay:VIEW:ROLL:Y:PDIVision:<CHARge|CURRent|RESistance|VOLTage>

Sets the value of scale division of Y-axis on the Roll view.

Syntax :DISPlay:VIEW:ROLL:Y:PDIVision:<CHARge|CURRent|RESistance|VOLTage> *value*

:DISPlay:VIEW:ROLL:Y:PDIVision:<CHARge|CURRent|RESistance|VOLTage>?

For <CHARge|CURRent|RESistance|VOLTage>, specify CHARge to set the scale division value of Y-axis for charge measurement, CURRent to set the scale division value of Y-axis for current measurement, RESistance to set the scale division value of Y-axis for resistance measurement, or VOLTage to set the scale division value of Y-axis for voltage measurement.

Parameter ***value*** Scale division value. Parameter data type is NRf.

Query response *value* <newline>

value returns the present setting of scale division value for Y-axis. Response data type is NR3.

Example :DISP:VIEW:ROLL:Y:PDIV:CURR 1.0E–6

:DISP:VIEW:ROLL:Y:PDIV:CURR?

:DISPlay:VIEW:SINGle:FORMat

Selects the display format of measured value to be displayed on Meter view.

Syntax :DISPlay:VIEW:SINGle:FORMat *mode*

:DISPlay:VIEW:SINGle:FORMat?

Parameter ***mode*** ENGineering(default)|EXPonential. Parameter data type is CPD.

mode = ENGineering displays the measured value in engineering format.

mode = EXPonential displays the measured value in exponential format.

Query response *mode* <newline>

mode returns ENG or EXP. Response data type is CRD.

Example :DISP:VIEW:SING:FORM EXP

:DISP:VIEW:SING:FORM?

:DISPlay:VIEW:SINGle:SPANel

Selects the item to be displayed on sub-panel of Meter view.

Syntax :DISPlay:VIEW:SINGle:SPANel *mode*

:DISPlay:VIEW:SINGle:SPANel?

Parameter ***mode*** RANGe|TRIGger|FUNCtion|ROLL|HISTogram. Parameter data type is CPD.

mode = RANGe displays the Range setup.

mode = TRIGger displays the Trigger setup.

mode = FUNCtion displays the voltage source range setup.

mode = ROLL displays the condensed roll graph.

mode = HISTogram displays the condensed histogram.

Query response *mode* <newline>

mode returns RANG, TRIG, FUNC, ROLL or HIST. Response data type is CRD.

Example :DISP:VIEW:SING:SPAN ROLL

:DISP:VIEW:SING:SPAN?

:DISPlay[:WINDow]:DATA?

Returns the data displayed on the front panel display.

Syntax :DISPlay[:WINDow[*d*]]:DATA?

Query response *response* <newline>

response returns the measured values and source output value displayed on the front panel display, as shown below. Each data is separated by a comma. *response* returns ---------- (hyphen) for the empty data. Response data type is SRD.

Characters μ and Ω are converted to u and ohm, respectively.

For the SING1 display mode, the :DISP:DATA? command returns the data displayed on the upper display area.

For the GRAPh, ROLL or HISTogram display mode, the :DISP:DATA? command returns "----------,----------" (if B2981B/B2983B) or "----------,----------,----------,----------,----------" (if B2985B/B2987B).

Example :DISP:DATA?

:DISPlay[:WINDow]:TEXT:DATA

Sets the text message displayed on the center of the upper or lower display area of the front panel display.

Syntax :DISPlay[:WINDow[*d*]]:TEXT:DATA *text*

:DISPlay[:WINDow[*d*]]:TEXT:DATA?

Parameter ***text*** Text. Up to 32 ASCII characters. Parameter data type is SPD.

Query response *text* <newline>

text returns the text message. Response data type is SRD.

Example :DISP:TEXT:DATA "Sweep measurement"

:DISPlay[:WINDow]:TEXT:STATe

Shows or hides the text message set by the :DISPlay[:WINDow]:TEXT:DATA command.

Syntax :DISPlay[:WINDow[*d*]]:TEXT:STATe *mode*

:DISPlay[:WINDow[*d*]]:TEXT:STATe?

Parameter ***mode*** 1|ON|0|OFF (default). Parameter data type is boolean.

mode = 1 or ON shows the text message.

mode = 0 or OFF hides the text message.

Query response *mode* <newline>

mode is 0 or 1, and indicates that the text message is off or on, respectively. Response data type is NR1.

Example :DISP:TEXT:STAT 1

:DISPlay:ZOOM

Enables or disables the zoom function of the front panel display. This function is effective for the SING1 view.

If this function is enabled, the setup information is not displayed and the measurement result is zoomed.

Syntax :DISPlay:ZOOM *mode*

:DISPlay:ZOOM?

Parameter ***mode*** 1|ON|0|OFF (default). Parameter data type is boolean.

mode = 1 or ON enables the zoom function of the front panel display.

mode = 0 or OFF disables the zoom function of the front panel display.

Query response *mode* <newline>

mode is 0 or 1, and indicates that the front panel zoom function is off or on, respectively. Response data type is NR1.

Example :DISP:ZOOM ON

:DISP:ZOOM?

FETCh Subsystem

:FETCh:ARRay?

Returns the array data which contains all of the voltage measurement data, current measurement data, resistance measurement data, time data, status data, source output setting data, temperature data, or humidity data specified by the :FORMat:ELEMents:SENSe command. The data is not cleared until the :INITiate, :MEASure, :READ, or :SENSe:DATA:CLEar command is executed.

NOTE B2981B/B2983B returns current measurement data, time data, and status data only.

Syntax :FETCh:ARRay? [*chanlist*]

Parameter

chanlist Channel to return the data. Parameter data type is channel list. (@1). See "Channel List Parameter" on page 27.

(@1) selects channel 1.

If this parameter is not specified, *chanlist* = (@1) is set.

Query response *response* <newline>

response returns the array data specified by the :FORMat:ELEMents:SENSe command. Response data type is NR3. See "Data Output Format" on page 31.

response returns the channel 1 data. See the following example. With the ASCII data output format, each data is separated by a comma.

ch1curr1,*ch1time1*,*ch1curr2*,*ch1time2*, ...
ch1curr10,*ch1time10*

This example shows the data containing the current data (*ch1currN*) and time data (*ch1timeN*) of the 10 measurements by channel 1.

If the measurement function is not enabled or no data exists, *response* returns +9.910000E+37 (ASCII) or NaN (IEEE-754) which indicates "not a number."

Example
```
:FORM:ELEM:SENS CURR,TIME
:FETC:ARR? (@1)
```

:FETCh:ARRay:<CHARge|CURRent|HUMidity|RESistance|SOURce|STATus|TEMPerature|TIME|VOLTage>?

Returns the array data which contains all of the charge measurement data, current measurement data, resistance measurement data, source output setting data, status data, time data, or voltage measurement data specified by CHARge, CURRent, HUMidity, RESistance, SOURce, STATus, TEMPerature, TIME, or VOLTage. The data is not cleared until the :INITiate, :MEASure, :READ, or :SENSe:DATA:CLEar command is executed.

Syntax :FETCh:ARRay:<CHARge|CURRent|HUMidity|RESistance|SOURce|STATus|TEMPerature|TIME|VOLTage>? [*chanlist*]

For <CHARge|CURRent|HUMidity|RESistance|SOURce|STATus|TEMPerature|TIME|VOLTage>, specify CHARge for charge measurement data, CURRent for current measurement data, HUMidity for humidity data, RESistance for resistance measurement data, SOURce for source output setting data, STATus for status data, TEMPerature for temperature data, TIME for time data, or VOLTage for voltage measurement data.

NOTE On B2981B/B2983B, only CURRent, STATus, and TIME are available.

NOTE The unit of temperature data is specified by :SYSTem:TEMPerature:UNIT command.

Parameter

chanlist	Channel to return the data. Parameter data type is channel list. (@1). See "Channel List Parameter" on page 27.

(@1) selects channel 1.

If this parameter is not specified, *chanlist* = (@1) is set.

Query response *response* <newline>

response returns the array data specified by CHARge, CURRent, RESistance, SOURce, STATus, TIME, or VOLTage. Response data type is NR3. See "Data Output Format" on page 31.

response returns the channel 1 data. See the following example. With the ASCII data output format, each data is separated by a comma.

ch1curr1,*ch1curr2*, ... ,*ch1curr10*

This example shows the data containing the current data (*ch1currN*) of the 10 measurements by channel 1.

If the measurement function is not enabled or no data exists, *response* returns +9.910000E+37 (ASCII) or NaN (IEEE-754) which indicates "not a number."

Example :FETC:ARR:CURR? (@1)

:FETCh[:SCALar]?

Returns the latest voltage measurement data, current measurement data, charge measurement data, resistance measurement data, time data, status data, source output setting data, temperature data, or humidity data specified by the :FORMat:ELEMents:SENSe command. The data is not cleared until the :INITiate, :MEASure, :READ, or :SENSe:DATA:CLEar command is executed.

NOTE B2981B/B2983B can return current measurement data, time data, and status data only.

NOTE The unit of temperature data is specified by :SYSTem:TEMPerature:UNIT command.

Syntax :FETCh[:SCALar]? [*chanlist*]

Parameter

chanlist	Channel to return the data. Parameter data type is channel list. (@1). See "Channel List Parameter" on page 27.

(@1) selects channel 1.

If this parameter is not specified, *chanlist* = (@1) is set.

Query response *response* <newline>

response returns the latest data specified by the :FORMat:ELEMents:SENSe command. Response data type is NR3. See "Data Output Format" on page 31.

response returns the channel 1 data. See the following example. With the ASCII data output format, each data is separated by a comma.

ch1curr10,*ch1time10*

This example shows the data containing the latest current data (*ch1curr10*) and time data (*ch1time10*) of the 10 measurements by channel 1.

If the measurement function is not enabled or no data exists, *response* returns +9.910000E+37 (ASCII) or NaN (IEEE-754) which indicates "not a number."

Example

```
:FORM:ELEM:SENS CURR,TIME
:FETC? (@1)
```

:FETCh[:SCALar]:<CHARge|CURRent|HUMidity|RESistance|SOURce|STATus|TEMPerature|TIME|VOLTage>?

Returns the latest voltage measurement data, current measurement data, charge measurement data, resistance measurement data, time data, status data, source output setting data, temperature data, or humidity data specified by CHARge, CURRent, HUMidity, RESistance, SOURce, STATus, TEMPerature, TIME, or VOLTage. The data is not cleared until the :INITiate, :MEASure, :READ, or :SENSe:DATA:CLEar command is executed.

Syntax

:FETCh[:SCALar]:<CHARge|CURRent|HUMidity|RESistance|SOURce|STATus|TEMPerature|TIME|VOLTage>? [*chanlist*]

For <CHARge|CURRent|HUMidity|RESistance|SOURce|STATus|TEMPerature|TIME|VOLTage>, specify CHARge for charge measurement data, CURRent for current measurement data, HUMidity for humidity data, RESistance for resistance measurement data, SOURce for source output setting data, STATus for status data, TEMPerature for temperature data, TIME for time data, or VOLTage for voltage measurement data.

NOTE For B2981B or B2983B, only CURRent, STATus, or TIME is available.

NOTE The unit of temperature data is specified by :SYSTem:TEMPerature:UNIT command.

Parameter *chanlist* Channel to return the data. Parameter data type is channel list. (@1). See "Channel List Parameter" on page 27.

(@1) selects channel 1.

If this parameter is not specified, *chanlist* = (@1) is set.

Query response *response* <newline>

response returns the latest data specified by CHARge, CURRent, RESistance, SOURce, STATus, TIME, or VOLTage. Response data type is NR3. See "Data Output Format" on page 31.

response returns the channel 1 data. See the following example. With the ASCII data output format, each data is separated by a comma.

ch1curr10

This example shows the data containing the latest current data (*ch1curr10*) of the 10 measurements by channel 1.

If the measurement function is not enabled or no data exists, *response* returns +9.910000E+37 (ASCII) or NaN (IEEE-754) which indicates "not a number."

Example :FETC:CURR? (@1)

FORMat Subsystem

:FORMat:BORDer

This command is effective when the data output format is set to the IEEE-754 binary format by using the :FORMat[:DATA] command. This command sets the byte order of binary output data.

Syntax :FORMat:BORDer *byte_order*

:FORMat:BORDer?

Parameter ***byte_order*** NORMal (default)|SWAPped. Parameter data type is CPD.

byte_order = NORMal sets the normal byte order. For the IEEE-754 single precision format, byte 1 to byte 4 are sent in this order. For the IEEE-754 double precision format, byte 1 to byte 8 are sent in this order.

byte_order = SWAPped sets the reverse byte order. For the IEEE-754 single precision format, byte 4 to byte 1 are sent in this order. For the IEEE-754 double precision format, byte 8 to byte 1 are sent in this order.

Query response *byte_order* <newline>

byte_order returns NORM or SWAP. Response data type is CRD.

Example :FORM:BORD SWAP

:FORM:BORD?

:FORMat[:DATA]

Sets the data output format. See "Data Output Format" on page 31.

Syntax :FORMat [:DATA] *format*

:FORMat [:DATA]?

Parameter ***format*** Data output format. Parameter data type is CPD.

ASCii|REAL,32|REAL,64. RERL,64 is only for the default language mode set by the :SYST:LANG "DEF" command.

format = ASCii specifies the ASCII format (default).

format = REAL,32 specifies the IEEE-754 single precision format. 4-byte data.

format = REAL,64 specifies the IEEE-754 double precision format. 8-byte data.

Query response *format* <newline>

format returns ASC, REAL,32, or REAL,64. Response data type is CRD.

Example :FORM REAL32

:FORM?

:FORMat:DIGital

Sets the response format of the bit pattern defined by the following commands:

- :CALCulate:CLIMits:<FAIL|PASS>DIGital[:DATA]
- :CALCulate:LIMit:COMPliance:DIGital[:DATA]
- :CALCulate:LIMit:<LOWer|UPPer>:DIGital[:DATA]
- :CALCulate:LIMit:PASS:DIGital[:DATA]

Syntax :FORMat:DIGital *format*

:FORMat:DIGital?

Parameter ***format*** Response format. ASCii (decimal, default)| BINary|OCTal|HEXadecimal. Parameter data type is CPD.

Query response *format* <newline>

format returns ASC, BIN, OCT, or HEX. Response data type is CRD.

Example :FORM:DIG BIN

:FORM:DIG?

:FORMat:ELEMents:CALCulate

Specifies the elements included in the calculation result data returned by the :CALCulate:DATA?, :CALCulate:DATA:LATest?, :CALCulate:MATH:DATA?, :CALCulate:MATH:DATA:LATest?, or :TRACe:DATA? command.

For the data stored in the trace buffer, this command is effective for the calculation result data that is specified by the :TRACe:FEED MATH command.

If all elements are specified by this command, the result data contains the all elements shown below. Then the order of elements is exclusive. For example, if TIME is not specified, the data contains the calc and status data in this order. If this command is not entered, the data contains the *calc* data only.

Elements and their order: *calc*, *time*, *status*

Syntax :FORMat:ELEMents:CALCulate *type*{,*type*}

:FORMat:ELEMents:CALCulate?

Parameter *type* Data element included in the data. CALC (calculation data, default)|TIME|STATus. Parameter data type is CPD.

CALC selects the calculation data *calc*.

TIME selects the time (timestamp) data *time*.

STAT selects the status data *status*.

Query response *type*{,*type*} <newline>

type returns CALC, TIME, or STAT. Response data type is CRD. Multiple responses are separated by a comma.

Example :FORM:ELEM:CALC CALC,TIME,STAT

:FORM:ELEM:CALC?

:FORMat:ELEMents:SENSe

Specifies the elements included in the sense or measurement result data returned by the :FETCh?, :READ?, :MEASure?, or :TRACe:DATA? command.

For the data stored in the trace buffer, this command is effective for the measurement result data that is specified by the :TRACe:FEED SENS command.

If this command is not entered or if all elements are specified by this command, the sense or measurement result data contains the all elements shown below. Then the order of elements is exclusive. For example, if VOLTage and RESistance are not specified, the data contains the current, time, status, source, temperature and humidity data in this order. It will not contain the voltage and resistance data.

Model	Elements and their order
B2981B, B2983B	*current*, *time*, *status*
B2985B, B2987B	*voltage, current(charge), resistance, time*, *status, source, temperature, humidity*

Syntax :FORMat:ELEMents:SENSe *type*{,*type*}

:FORMat:ELEMents:SENSe?

Parameter *type* Data element included in the data. VOLTage|CURRent|CHARge|RESistance|TIME|STATus|SOURce|TEMPerature|HUMidity. Parameter data type is CPD.

NOTE For B2981B/B2983B, CURRent, STATus and TIME can be specified.

VOLT selects the voltage measurement data *voltage*.

CURR selects the current measurement data *current*.

CHAR selects the charge measurement data *charge*.

NOTE Can't specify both CURR and CHAR.

RES selects the resistance measurement data *resistance*.

TIME selects the time data *time* (timestamp of the measurement start trigger).

STAT selects the status data *status*.

SOUR selects the source output setting data *source*.

TEMP selects the temperature data *temperature*.

NOTE The unit of temperature data is specified by :SYSTem:TEMPerature:UNIT command.

HUM selects the humidity data *humidity*.

Query response *type*{,*type*} <newline>

type returns VOLT, CURR, CHAR, RES, TIME, STAT, SOUR, TEMP, or HUM. Response data type is CRD. Multiple responses are separated by a comma.

Example :FORM:ELEM:SENS CURR,TIME,STAT

:FORM:ELEM:SENS?

:FORMat:SREGister

Sets the response format of the status byte register.

Syntax :FORMat:SREGister *format*

:FORMat:SREGister?

Parameter

format	Response format. ASCii (decimal, default)| BINary|OCTal|HEXadecimal. Parameter data type is CPD.

Query response *format* <newline>

format returns ASC, BIN, OCT, or HEX. Response data type is CRD.

Example :FORM:SREG BIN

:FORM:SREG?

HCOPy Subsystem

:HCOPy:SDUMp:DATA?

Returns the data of the front panel screen image. The format of the image data is set by the :HCOPy:SDUMp:FORMat command.

Syntax :HCOPy:SDUMp:DATA?

Query response The response is a definite length arbitrary binary block.

Example :HCOP:SDUM:DATA?

:HCOPy:SDUMp:FORMat

Sets the format of the image data. The front panel screen image will be created in the format set by this command. The image data will be returned by the :HCOPy:SDUMp:DATA? command.

Syntax :HCOPy:SDUMp:FORMat *format*

HCOPy:SDUMp:FORMat?

Parameter

format	Format of image data. JPG (default)|BMP|PNG|WMF. Parameter data type is CPD.

Query response *format* <newline>

format returns JPG, BMP, PNG, or WMF. Response data type is CRD.

Example :HCOP:SDUM:FORM BMP

:HCOP:SDUM:FORM?

INPut Subsystem

For the numeric suffix [*c*], see "Numeric Suffix" on page 27.

:INPut[:STATe]

Enables or disables the current or charge input.

NOTE B2981B/B2983B supports current measurement only.

Syntax :INPut[*c*][:STATe] *mode*

:INPut[*c*][:STATe]?

Parameter *mode* 1|ON|0|OFF (default). Parameter data type is boolean.

mode = 1 or ON enables the current or charge input.

mode = 0 or OFF disables the current or charge input.

Query response *mode* <newline>

mode is 0 or 1, and indicates that the current or charge input is off or on, respectively. Response data type is NR1.

Example :INP 1

:INP:STAT?

:INPut:ZCORrect[:STATe]

Enables or disables zero correct function for the current or charge measurement.

NOTE B2981B/B2983B supports current measurement only.

Syntax :INPut[*c*]:ZCORrect[:STATe] *mode*

:INPut[*c*]:ZCORrect[:STATe]?

Parameter *mode* 1|ON|0|OFF (default). Parameter data type is boolean.

mode = 1 or ON enables zero correct function for the current or charge measurement.

mode = 0 or OFF disables zero correct function for the current or charge measurement.

Query response *mode* <newline>

mode is 0 or 1, and indicates that zero correct function for the current/charge input is off or on, respectively. Response data type is NR1.

Example :INP:ZCOR 1

:INP:ZCOR:STAT?

NOTE **ZC indicator on the display**

This indicator shows the zero correct function state as follows.

(not lit): The function is disabled.

ZC (gray): The function is enabled. Zero correction was not performed.

ZC (white): The function is enabled. Zero correction was performed.

LXI Subsystem

:ARM:LXI:COUNt

Specifies an integer as the number of times the arm has to occur to complete the arm loop; that is, the number of arms that are accepted before the measurement must be initiated again. intRepetitions must be greater than zero (0) and is limited by either your application or the maximum 32 bit signed integer value.This is not enforced by LxiMiddleware; your SCPI action for this command will need to enforce the limits.

NOTE The LXI Middleware only stores this value. In other words, you essentially have to implement the count (e.g., you need to keep track of how many ARM events have occurred, and only arm the instrument when the total COUNt has occurred).

Syntax :ARM:LXI:COUNt *intRepetitions*

:ARM:LXI:COUNt?

Parameter ***intRepetitions*** Number of arms required to complete the arm loop. Parameter data type is NRf.

Query response *intRepetitions* <newline>

intRepetitions returns the number of arms required to complete the arm loop. Response data type is NR1.

Example :ARM:LXI:COUN 10

:ARM:LXI:COUN?

:ARM:LXI:DELay

Specifies the delay, in seconds, from when the arm logic satisfied until the waiting for trigger state is entered.

NOTE Any <delay> value other than 0.0 is changed to 0 (zero) and a warning is inserted into the SCPI error queue.

This implementation assumes that there is no Precision Time Protocol (PTP) implementation on the instrument. Delay requires PTP, so without it the delay must be zero.

Syntax :ARM:LXI:DELay *delay*

:ARM:LXI:DELay?

Parameter

delay	Delay, in seconds. Parameter data type is NRf.

Query response *delay* <newline>

delay returns the delay, in seconds. Response data type is NR3.

Example :ARM:LXI:DEL 10.0

:ARM:LXI:DEL?

:ARM:LXI:LAN[:SET]:DETection

Specifies the style of arm source detection for the specified event. If the source is a LAN event and the source detection is set to rise, this Arm repeated capability will be satisfied when the designated LAN packet arrives with a True indication. If the source detection is set to fall, this Arm repeated capability will be satisfied when a LAN packet arrives with a False indication. If the detection is set to high, the source will be satisfied when the designated LAN packet arrives with a True indication and remain satisfied until the designated LAN packet arrives with a False indication. If the detection is to low, the source will be satisfied when the designated LAN packet arrives with a False indication and remain satisfied until the designated LAN packet arrives with a True indication.

Syntax :ARM:LXI:LAN[:SET]:DETection *strLanEvent*, *detection*

:ARM:LXI:LAN[:SET]:DETection? strLanEvent

Parameter

strLanEvent	LAN event name. String enclosed in quotes. Parameter data type is SPD.
detection	Detection type. RISE (default)|FALL|HIGH|LOW. Parameter data type is CPD.

RISE sets the instrument to trigger on the receipt of a signal LOW LAN event followed by a signal HIGH LAN event (rising edge).

FALL sets the instrument to trigger on the receipt of a signal HIGH LAN event followed by a signal LOW LAN event (falling edge).

HIGH sets the instrument to trigger on every signal HIGH LAN event.

LOW sets the instrument to trigger on every signal LOW LAN event.

Query response *detection* <newline>

detection returns a string with the DETection setting for the specified event. Response data type is CRD.

Example :ARM:LXI:LAN:DET “LAN0”,RISE

:ARM:LXI:LAN:DET? “LAN0”

:ARM:LXI:LAN[:SET]:ENABle

Enables or disables the arm source of the specified event.

Syntax :ARM:LXI:LAN[:SET]:ENABle *strLanEvent*, *status*

:ARM:LXI:LAN[:SET]:ENABle? strLanEvent

Parameter

strLanEvent	LAN event name. String enclosed in quotes. Parameter data type is SPD.
status	Status. 0|OFF (default)|1|ON. Parameter data type is boolean.

Query response *status* <newline>

status returns an integer (0 or 1) denoting the enabled status of the specified event. Response data type is boolean.

Example :ARM:LXI:LAN:ENAB “LAN0”,1

:ARM:LXI:LAN:ENAB? “LAN0”

:ARM:LXI:LAN[:SET]:FILTer

Specifies a filter for restricting arm sources of the specified event.

Syntax :ARM:LXI:LAN[:SET]:FILTer *strLanEvent*, *strFilterExpression*

:ARM:LXI:LAN[:SET]:FILTer? *strLanEvent*

Parameter ***strLanEvent*** LAN event name. String enclosed in quotes. Parameter data type is SPD.

strFilterExpression Filter. It will be an existing LAN arm sources, e.g. one of the items returned by :LXI:EVENt:INPut:LAN:LIST? or :LXI:EVENt[:OUTPut]:LAN:LIST?. String enclosed in quotes. Parameter data type is SPD.

Query response *strFilterExpression* <newline>

strFilterExpression returns a string with the filter expression for the specified event. Response data type is SRD.

Example :ARM:LXI:LAN:FILT “LAN0”,”ALL:5024”

:ARM:LXI:LAN:FILT? “LAN0”

:ARM:LXI:LAN[:SET]:IDENtifier

Specifies the LAN event identifier that is associated with this arm source. LAN Events with this identifier are accepted from the source described in the filter.

Syntax :ARM:LXI:LAN[:SET]:IDENtifier *strLanEvent*, *strCustomId*

:ARM:LXI:LAN[:SET]:IDENtifier? *strLanEvent*

Parameter ***strLanEvent*** LAN event name. String enclosed in quotes. Parameter data type is SPD.

strCustomId Identifier. String of up to 16 characters enclosed in quotes. You can use letters (A-Z, a-z), numbers (0-9), and printable characters like “@”, “%”, “*”, etc. Parameter data type is SPD.

Query response *strCustomId* <newline>

strCustomId returns the identifier string for the specified event. Response data type is SRD.

Example :ARM:LXI:LAN:IDEN “LAN0”,”MyEvent”

:ARM:LXI:LAN:IDEN? “LAN0”

:LXI:EVENt:DOMain

Specifies the LXI LAN domain. <bytDomain> is an integer, 0 to 255. A domain facilitates multiple systems on a single LAN. LXI modules ignore all LXI messages except those in its own domain, as defined by the domain byte.

Syntax :LXI:EVENt:DOMain *bytDomain*

:LXI:EVENt:DOMain?

Parameter ***bytDomain*** LXI LAN domain. 0 (default) to 255. Parameter data type is NRf.

Query response *bytDomain* <newline>

bytDomain returns the LXI LAN domain. Response data type is NR1.

Example :LXI:EVEN:DOM 1

:LXI:EVEN:DOM?

:LXI:EVENt:INPut:LAN:ADD

Creates a new input event to receive.

Syntax :LXI:EVENt:INPut:LAN:ADD *strLanEvent*

Parameter ***strLanEvent*** LAN event name. String of up to 16 characters enclosed in quotes. You can use letters (A-Z, a-z), numbers (0-9), and printable characters like “@”, “%”, “*”, etc. Parameter data type is SPD.

Example :LXI:EVEN:INP:LAN:ADD “MyEvent”

:LXI:EVENt:INPut:LAN:COUNt?

Returns an integer as the total number of defined input LAN events (includes both enabled and disabled events).

Syntax :LXI:EVENt:INPut:LAN:COUNt?

Query response *count* <newline>

count returns an integer as the total number of defined input LAN events. Response data type is NR1.

Example :LXI:EVEN:INP:LAN:COUN?

:LXI:EVENt:INPut:LAN:DISable:ALL

Disables all input events.

Syntax :LXI:EVENt:INPut:LAN:DISable:ALL

Example :LXI:EVEN:INP:LAN:DIS:ALL

:LXI:EVENt:INPut:LAN:LIST?

Returns a quoted string with the list of defined input event names.

Syntax :LXI:EVENt:INPut:LAN:LIST?

Query response *list* <newline>

list returns a quoted string with the list of defined input event names. Response data type is SRD.

Example :LXI:EVEN:INP:LAN:LIST?

:LXI:EVENt:INPut:LAN:REMove

Removes the specified input event from the list of named events. Predefined LAN event names (LAN0 - LAN7) cannot be removed.

The event is added by :LXI:EVENt:INPut:LAN:ADD.

Syntax :LXI:EVENt:INPut:LAN:REMove *strLanEvent*

Parameter ***strLanEvent*** LAN event name. String enclosed in quotes. Parameter data type is SPD.

Example :LXI:EVEN:INP:LAN:REM “MyEvent”

:LXI:EVENt:INPut:LAN:REMove:ALL

Removes all of the input events that were added using :LXI:EVENt:INPut:LAN:ADD. Predefined LAN names (LAN0 - LAN7) cannot be removed.

Syntax :LXI:EVENt:INPut:LAN:REMove:ALL

Example :LXI:EVEN:INP:LAN:REM:ALL

:LXI:EVENt:INPut:LAN[:SET]:CONFigure

Configures the most common attributes of LXI LAN input events. The parameters are also available from their corresponding commands. For more details on the configurable parameters, refer to the individual SCPI commands below.

Syntax :LXI:EVENt:INPut:LAN[:SET]:CONFigure *strLanEvent*, *enable*, *detection* [, *delay*], *filter*, *identifier*

Parameter

strLanEvent	LAN event name. String enclosed in quotes. Parameter data type is SPD.
enable	Status. 0|OFF|1|ON. Parameter data type is boolean.
detection	Detection type. RISE|FALL|HIGH|LOW. Parameter data type is CPD.
delay	Delay, in seconds. Parameter data type is NRf.
filter	Filter. String enclosed in quotes. You can use letters (A-Z, a-z), numbers (0-9), and printable characters like “@”, “%”, “*”, etc. Parameter data type is SPD.
identifier	Identifier. String of up to 16 characters enclosed in quotes. You can use letters (A-Z, a-z), numbers (0-9), and printable characters like “@”, “%”, “*”, etc. Parameter data type is SPD.

RISE sets the instrument to trigger on the receipt of a signal LOW LAN event followed by a signal HIGH LAN event (rising edge).

FALL sets the instrument to trigger on the receipt of a signal HIGH LAN event followed by a signal LOW LAN event (falling edge).

HIGH sets the instrument to trigger on every signal HIGH LAN event.

LOW sets the instrument to trigger on every signal LOW LAN event.

Example :LXI:EVEN:INP:LAN:CONF “LAN0”,ON,”HIGH”,0,”ALL:5044”,”LAN0”

:LXI:EVENt:INPut:LAN[:SET]:DELay

The delay is optional. <delay> is a double representing the delay, in seconds. Any <delay> value other than 0.0 is changed to 0 (zero) and a warning is inserted into the SCPI error queue.

NOTE This implementation assumes that there is no Precision Time Protocol (PTP) implementation on the instrument. Delay requires PTP, so without it the delay must be zero.

Syntax :LXI:EVENt:INPut:LAN[:SET]:DELay *strLanEvent*, *delay*

:LXI:EVENt:INPut:LAN[:SET]:DELay? *strLanEvent*

Parameter

strLanEvent	LAN event name. String enclosed in quotes. Parameter data type is SPD.
delay	Delay, in seconds. Parameter data type is NRf.

Query response *delay* <newline>

delay returns the delay, in seconds, for the specified LAN trigger event. Response data type is NR3.

Example :LXI:EVEN:INP:LAN:DEL “LAN0”,10.0

:LXI:EVEN:INP:LAN:DEL? “LAN0”

:LXI:EVENt:INPut:LAN[:SET]:DETection

Specifies the trigger detection method and polarity for the input event.

Syntax :LXI:EVENt:INPut:LAN[:SET]:DETection *strLanEvent*, *detection*

:LXI:EVENt:INPut:LAN[:SET]:DETection? *strLanEvent*

Parameter

strLanEvent	LAN event name. String enclosed in quotes. Parameter data type is SPD.
detection	Detection type. RISE (default)|FALL|HIGH|LOW. Parameter data type is CPD.

RISE sets the instrument to trigger on the receipt of a signal LOW LAN event followed by a signal HIGH LAN event (rising edge).

FALL sets the instrument to trigger on the receipt of a signal HIGH LAN event followed by a signal LOW LAN event (falling edge).

HIGH sets the instrument to trigger on every signal HIGH LAN event.

LOW sets the instrument to trigger on every signal LOW LAN event.

Query response *detection* <newline>

detection returns a character string with the detection setting. Response data type is CRD.

Example :LXI:EVEN:INP:LAN:DET “LAN0”,RISE

:LXI:EVEN:INP:LAN:DET? “LAN0”

:LXI:EVENt:INPut:LAN[:SET]:ENABle

Enables or disables the specified input event. The event is ignored if disabled.

Syntax :LXI:EVENt:INPut:LAN[:SET]:ENABle *strLanEvent*, *status*

:LXI:EVENt:INPut:LAN[:SET]:ENABle? *strLanEvent*

Parameter

strLanEvent LAN event name. String enclosed in quotes. Parameter data type is SPD.

status Status. 0|OFF (default)|1|ON. Parameter data type is boolean.

Query response *status* <newline>

status returns an integer (0 or 1) denoting the enabled status of the specified event. Response data type is boolean.

Example :LXI:EVEN:INP:LAN:ENAB “LAN0”,1

:LXI:EVEN:INP:LAN:ENAB? “LAN0”

:LXI:EVENt:INPut:LAN[:SET]:FILTer

Creates a filter for incoming input events. Only input events coming from hosts matching the filter string are processed.

Syntax :LXI:EVENt:INPut:LAN[:SET]:FILTer *strLanEvent*, *strFilterExpression*

:LXI:EVENt:INPut:LAN[:SET]:FILTer? *strLanEvent*

Parameter

strLanEvent LAN event name. String enclosed in quotes. Parameter data type is SPD.

strFilterExpression Filter which specifies an existing LAN event, e.g. one of the items returned by :LXI:EVENt:INPut:LAN:LIST?. String enclosed in quotes. Parameter data type is SPD.

Query response *strFilterExpression* <newline>

strFilterExpression returns a string with the filter expression for the specified event. Response data type is SRD.

Example :LXI:EVEN:INP:LAN:FILT “LAN0”,1

:LXI:EVEN:INP:LAN:FILT? “LAN0”

:LXI:EVENt:INPut:LAN[:SET]:IDENtifier

Specifies the string that is expected to arrive over the LAN for a given input event to occur. For example, "Relay Closed" as an identifier for "LAN0" sent from a switch box.

Syntax :LXI:EVENt:INPut:LAN[:SET]:IDENtifier *strLanEvent*, *strCustomId*

:LXI:EVENt:INPut:LAN[:SET]:IDENtifier? *strLanEvent*

Parameter

strLanEvent	LAN event name. String enclosed in quotes. Parameter data type is SPD.
strCustomId	Identifier. String of up to 16 characters enclosed in quotes. You can use letters (A-Z, a-z), numbers (0-9), and printable characters like “@”, “%”, “*”, etc. Parameter data type is SPD.

Query response *strCustomId* <newline>

strCustomId returns a string with the identifier assigned to the specified input event. Response data type is SRD.

Example :LXI:EVEN:INP:LAN:IDEN “LAN0”,”MyEvent”

:LXI:EVEN:INP:LAN:IDEN? “LAN0”

:LXI:EVENt:LOG:ALL?

Returns the contents of the event log. A complete log is a set of entries delimited by a semicolon. The contents of the event log are returned as a single quoted string containing all the event log entries. This is a non-destructive read.

Syntax :LXI:EVENt:LOG:ALL?

Query response *log* <newline>

log returns the contents of the event log. Response data type is SRD.

Example :LXI:EVEN:LOG:ALL?

:LXI:EVENt:LOG:CIRCular[:ENABle]

Selects how new entries are handled when the LXI event log is full.

Syntax :LXI:EVENt:LOG:CIRCular[:ENABle] *status*

:LXI:EVENt:LOG:CIRCular[:ENABle]?

Parameter | ***status*** | Circular logging status. 0|OFF|1|ON (default). Parameter data type is boolean.

Query response *status* <newline>

status returns an integer (0 or 1) indicating the behavior of the LXI event log. Response data type is boolean.

Example :LXI:EVEN:LOG:CIRC ON

:LXI:EVEN:LOG:CIRC?

:LXI:EVENt:LOG:CIRCular:FBEntry

Selects the most recently added event log entry to be used as the reference for :LXI:EVENt:LOG:ENTRy?. The log must be in circular mode for the command to function.

Syntax :LXI:EVENt:LOG:CIRCular:FBEntry

Example :LXI:EVEN:LOG:CIRC:FBE

:LXI:EVENt:LOG:CLEar

Removes all existing entries from the event log.

Syntax :LXI:EVENt:LOG:CLEar

Example :LXI:EVEN:LOG:CLE

:LXI:EVENt:LOG:COUNt?

Returns an integer as the total number of entries in the LXI event log.

Syntax :LXI:EVENt:LOG:COUNt?

Query response *count* <newline>

count returns an integer as the total number of entries in the LXI event log. Response data type is NR1.

Example :LXI:EVEN:LOG:COUN?

:LXI:EVENt:LOG:ENABle

Enables or disables LXI event logging.

Syntax :LXI:EVENt:LOG:ENABle *status*

:LXI:EVENt:LOG:ENABle?

Parameter ***status*** LXI event logging status. 0|OFF|1|ON (default). Parameter data type is boolean.

Query response *status* <newline>

status returns an integer (0 or 1) indicating the behavior of the LXI event log. Response data type is boolean.

Example :LXI:EVEN:LOG:ENAB ON

:LXI:EVEN:LOG:ENAB?

:LXI:EVENt:LOG:ENTRy?

Retrieves the event log entry referenced by <intIndex>. When the log is in circular mode, this index value is relative to the entry selected by :LXI:EVENt:LOG:CIRCular:FBEntry.

Syntax :LXI:EVENt:LOG:ENTRy? *intIndex*

Parameter ***intIndex*** Reference point of the event log. 0 to 2147483647. Parameter data type is NRf.

Query response *log* <newline>

log returns an event log entry, nine comma separated character strings, as follows.

"<chrDate>,<chrTime>,<chrEventType>,<chrEventName>,<chrEventEdge>, <chrSourceEvent>, <chrEventIdentifier>,<chrSrcAddress>,<chrDstAddress>"

Example :LXI:EVEN:LOG:ENTR? 100

This example returns the 100th entry in the event log.

Remarks The command returns an empty string if the specified index is out of range or the entry no longer exists.

The event log records internal status events as well as all LXI event activity. As LXI LAN events are sent or received, the activity is noted in the event log with an IEEE 1588 timestamp.

The fields recorded in the event log are:

- The date the event occurred (GMT).
- The time the event occurred (GMT).
- The type of event: LAN Input, LAN Output, Status, Alarm, Trigger Alarm, Trigger LAN.
- The name of the event.
- The edge associated with the event.
- The source event is only valid for LAN Output, Trigger LAN, and Trigger Alarm event types.
- The event's identifier appears as an ASCII character on the LAN.
- The source address is only valid for LAN Input event types. It is the address from which the message originated.
- The destination address is only valid for LAN Output event types. It is the address (or addresses) that the message will be sent to. For UDP messages, this field will read "ALL".

See also ":LXI:EVENt:LOG:CLEar" on page 204.

:LXI:EVENt:LOG:SIZE

Sets the maximum number of entries the LXI event log can hold.

Syntax :LXI:EVENt:LOG:SIZE *maxLogEntries*

:LXI:EVENt:LOG:SIZE?

Parameter ***maxLogEntries*** Size of the LXI event log. 1 to 200. Default is 100. Parameter data type is NRf.

Query response *maxLogEntries* <newline>

maxLogEntries returns an integer as the current size (maximum number of entries) of the LXI event log. Response data type is NR1.

Example :LXI:EVEN:LOG:SIZE 200

:LXI:EVEN:LOG:SIZE?

:LXI:EVENt[:OUTPut]:LAN:ADD

Creates a new output event as specified by <strLanEvent>.

Syntax :LXI:EVENt[:OUTPut]:LAN:ADD *strLanEvent*

Parameter ***strLanEvent*** LAN event name. String of up to 16 characters enclosed in quotes. You can use letters (A-Z, a-z), numbers (0-9), and printable characters like “@”, “%”, “*”, etc. Parameter data type is SPD.

Example :LXI:EVEN:LAN:ADD “MyEvent”

:LXI:EVENt[:OUTPut]:LAN:COUNt?

Returns an integer as the number of configured LXI output LAN events.

Syntax :LXI:EVENt[:OUTPut]:LAN:COUNt?

Query response *count* <newline>

count returns an integer as the total number of defined output LAN events. Response data type is NR1.

Example :LXI:EVEN:LAN:COUN?

:LXI:EVENt[:OUTPut]:LAN:DISable:ALL

Disables all configured LXI output LAN events.

Syntax :LXI:EVENt[:OUTPut]:LAN:DISable:ALL

Example :LXI:EVEN:LAN:DIS:ALL

:LXI:EVENt[:OUTPut]:LAN:LIST?

Returns a quoted string containing a list of all configured LAN output event names.

Syntax :LXI:EVENt[:OUTPut]:LAN:LIST?

Query response *list* <newline>

list returns a quoted string with the list of defined output event names. Response data type is SRD.

Example :LXI:EVEN:LAN:LIST?

:LXI:EVENt[:OUTPut]:LAN:REMove

Disables and removes the specified custom LAN output event. Predefined LAN event names (LAN0 - LAN7) cannot be removed.

The event is added by :LXI:EVENt[:OUTPut]:LAN:ADD.

Syntax :LXI:EVENt[:OUTPut]:LAN:REMove *strLanEvent*

Parameter ***strLanEvent*** LAN event name. String enclosed in quotes. Parameter data type is SPD.

Example :LXI:EVEN:LAN:REM “MyEvent”

:LXI:EVENt[:OUTPut]:LAN:REMove:ALL

Disables and removes all custom LAN events added using :LXI:EVENt[:OUTPut]:LAN:ADD. Predefined LAN names (LAN0 - LAN7) cannot be removed.

Syntax :LXI:EVENt[:OUTPut]:LAN:REMove:ALL

Example :LXI:EVEN:LAN:REM:ALL

:LXI:EVENt[:OUTPut]:LAN:SEND

Forces the instrument to send the specified output event. The output event must be enabled, otherwise this command is ignored; in other words, you must send :LXI:EVENt[:OUTPut]:LAN[:SET]:ENABle <strLanEvent>, ON to enable the lanEvent prior to sending SEND, or SEND will not send anything.

Syntax :LXI:EVENt[:OUTPut]:LAN:SEND *strLanEvent*,*type*

Parameter

strLanEvent	LAN event name. String enclosed in quotes. Parameter data type is SPD.
type	Event type. RISE|FALL. Parameter data type is CPD.

Example :LXI:EVEN:LAN:SEND “LAN0”,FALL

:LXI:EVENt[:OUTPut]:LAN[:SET]:CONFigure

Configures the most common attributes of LXI LAN output events. The parameters are also available from their corresponding commands. For more details on the configurable parameters, refer to the individual SCPI commands below.

Syntax :LXI:EVENt[:OUTPut]:LAN[:SET]:CONFigure *strLanEvent*, *enable*, *source*, *slope*, *drive*, *destination*

Parameter

strLanEvent	LAN event name. String enclosed in quotes. Parameter data type is SPD.
enable	Status. 0|OFF|1|ON. Parameter data type is boolean.
source	Event name, should be one of following. Parameter data type is SPD. • WaitingForAcquireArm1 • WaitingForAcquireTrigger1 • WaitingForTransitionArm1 • WaitingForTransitionTrigger1 • Measuring1 • Settling1

	slope	Slope attribute. POSitive|NEGative. Parameter data type is CPD.
	drive	Drive behavior. OFF|NORMal|WOR. Parameter data type is CPD. See ":LXI:EVENt[:OUTPut]:LAN[:SET]:DRIVe" on page 210.
	destination	Destination. String enclosed in quotes. You can use letters (A-Z, a-z), numbers (0-9), and printable characters like ""@", "%", "*", etc. Parameter data type is SPD.

Example :LXI:EVEN:LAN:CONF "LAN0",ON,"WaitingForAcquireTrigger1",POS,NORM, "ALL:5044"

:LXI:EVENt[:OUTPut]:LAN[:SET]:DESTination

Sets the destination for the specified outgoing LAN event to the hosts specified by <strDestExpression>. The expression takes the form of "host1:port1, host2:port2,". The port numbers are optional and will default to the IANA assigned TCP port (5044).

Syntax :LXI:EVENt[:OUTPut]:LAN[:SET]:DESTination *strLanEvent*, *strDestExpression*

:LXI:EVENt[:OUTPut]:LAN[:SET]:DESTination? *strLanEvent*

Parameter	***strLanEvent***	LAN event name. String enclosed in quotes. Parameter data type is SPD.
	strDestExpression	Destination. String enclosed in quotes. You can use letters (A-Z, a-z), numbers (0-9), and printable characters like "@", "%", "*", etc. Parameter data type is SPD.

Query response *strDestExpression* <newline>

strDestExpression returns the Destination Expression string. Response data type is SRD.

Example :LXI:EVEN:LAN:DEST "LAN0","ALL:5044"

:LXI:EVEN:LAN:DEST? "LAN0"

:LXI:EVENt[:OUTPut]:LAN[:SET]:DRIVe

Specifies the trigger drive behavior for the specified LAN output event.

Syntax :LXI:EVENt[:OUTPut]:LAN[:SET]:DRIVe *strLanEvent*, *drive*

:LXI:EVENt[:OUTPut]:LAN[:SET]:DRIVe? *strLanEvent*

Parameter

strLanEvent	LAN event name. String enclosed in quotes. Parameter data type is SPD.
drive	Drive behavior. OFF (default)|NORMal|WOR. Parameter data type is CPD.

OFF disables the LAN event.

NORMal designates typical operation where both edges of the instrument event are transmitted.

WOR (Wired.OR) causes only one edge to be transmitted.

Query response *drive* <newline>

drive returns a string representing the drive behavior. Response data type is CRD.

Example :LXI:EVEN:LAN:DRIV “LAN0”,WOR

:LXI:EVEN:LAN:DRIV? “LAN0”

:LXI:EVENt[:OUTPut]:LAN[:SET]:ENABle

Enables or disables the specified LXI LAN output event.

Syntax :LXI:EVENt[:OUTPut]:LAN[:SET]:ENABle *strLanEvent*, *status*

:LXI:EVENt[:OUTPut]:LAN[:SET]:ENABle? *strLanEvent*

Parameter

strLanEvent	LAN event name. String enclosed in quotes. Parameter data type is SPD.
status	LXI LAN output event status. 0|OFF (default)|1|ON. Parameter data type is boolean.

Query response *status* <newline>

status returns an integer (0 or 1) as the state of the specified output event. Response data type is boolean.

Example :LXI:EVEN:LAN:ENAB “LAN0”,ON

:LXI:EVEN:LAN:ENAB? “LAN0”

:LXI:EVENt[:OUTPut]:LAN[:SET]:IDENtifier

Specifies the custom string that will be transmitted as part of the output event.

Syntax :LXI:EVENt[:OUTPut]:LAN[:SET]:IDENtifier *strLanEvent*, *strCustomId*

:LXI:EVENt[:OUTPut]:LAN[:SET]:IDENtifier? *strLanEvent*

Parameter

strLanEvent	LAN event name. String enclosed in quotes. Parameter data type is SPD.
strCustomId	Identifier. String of up to 16 characters enclosed in quotes. You can use letters (A–Z, a–z), numbers (0–9), and printable characters like “@”, “%”, “*”, etc. Parameter data type is SPD.

Query response *strCustomId* <newline>

strCustomId returns a string with the identifier assigned to the specified LXI LAN output event. Response data type is SRD.

Example :LXI:EVEN:LAN:IDEN “LAN0”,”MyEvent”

:LXI:EVEN:LAN:IDEN? “LAN0”

:LXI:EVENt[:OUTPut]:LAN[:SET]:SLOPe

Sets the slope of the event transition. It determines which instrument event transition will result in a LAN packet being sent and whether or not that edge is inverted.

Syntax :LXI:EVENt[:OUTPut]:LAN[:SET]:SLOPe *strLanEvent*, *slope*

:LXI:EVENt[:OUTPut]:LAN[:SET]:SLOPe? *strLanEvent*

Parameter

strLanEvent	LAN event name. String enclosed in quotes. Parameter data type is SPD.
slope	Slope attribute. POSitive (default)|NEGative. Parameter data type is CPD.

Query response *slope* <newline>

slope returns a character string representing the slope attribute of the LXI LAN packet. Response data type is CRD.

Example :LXI:EVEN:LAN:SLOP “LAN0”,POS

:LXI:EVEN:LAN:SLOP? “LAN0”

:LXI:EVENt[:OUTPut]:LAN[:SET]:SOURce

Designates the instrument that the specified LAN output event is tied to. <strAnyEvent> is a discrete value.

Syntax :LXI:EVENt[:OUTPut]:LAN[:SET]:SOURce *strLanEvent*, *strAnyEvent*

:LXI:EVENt[:OUTPut]:LAN[:SET]:SOURce? *strLanEvent*

Parameter

strLanEvent LAN event name. String enclosed in quotes. Parameter data type is SPD.

strAnyEvent Event name, should be one of following. Parameter data type is SPD.

- WaitingForAcquireArm1
- WaitingForAcquireTrigger1
- WaitingForTransitionArm1
- WaitingForTransitionTrigger1
- Measuring1
- Settling1

Query response *strAnyEvent* <newline>

strAnyEvent returns a character string indicating the event type for the specified LAN event name. Response data type is SRD.

Example :LXI:EVEN:LAN:SOUR “LAN0”,“Measuring1”

:LXI:EVEN:LAN:SOUR? “LAN0”

:LXI:EVENt[:OUTPut]:LAN[:SET]:TSDelta

Sets the delay that occurs between the generation of the specified event and the remote instruments action on it. The parameter <dblSeconds> represents a time in seconds to add to the timestamp of the output LAN event. Since precision timed interrupts (also known as alarms) are not supported at this time, <dblSeconds> must be 0.0.

Syntax :LXI:EVENt[:OUTPut]:LAN[:SET]:TSDelta *strLanEvent*, *dblSeconds*

:LXI:EVENt[:OUTPut]:LAN[:SET]:TSDelta? *strLanEvent*

Parameter

strLanEvent	LAN event name. String enclosed in quotes. Parameter data type is SPD.
dblSeconds	Delay, in seconds. Parameter data type is NRf.

Query response *dblSeconds* <newline>

dblSeconds returns the delay, in seconds. Response data type is NR3.

Example :LXI:EVEN:LAN:TSD “LAN0”,1.0

:LXI:EVEN:LAN:TSD? “LAN0”

:LXI:IDENtify[:STATe]

Changes the LXI status indicator state.

Syntax :LXI:IDENtify[:STATe] *mode*

:LXI:IDENtify[:STATe]?

Parameter

mode	0|OFF (default)|1|ON. Parameter data type is boolean.

mode = 1 or ON changes the LXI status indicator to the Identify state.

mode = 0 or OFF changes the LXI status indicator to the No Fault state.

Query response *mode* <newline>

mode returns 0 or 1, and indicates that the LXI status indicator is No Fault or Identify, respectively. Response data type is NR1.

Example :LXI:IDEN 0

:LXI:IDEN:STAT?

:LXI:MDNS:ENABle

Enables or disables mDNS (multicast DNS) function.

Syntax :LXI:MDNS:ENABle *mode*

:LXI:MDNS:ENABle?

Parameter *mode* 0|OFF|1|ON (default). Parameter data type is boolean.

mode = 1 or ON enables the mDNS function.

mode = 0 or OFF disables the mDNS function.

Query response *mode* <newline>

mode returns 0 or 1, and indicates that the mDNS function is disable or enable, respectively. Response data type is NR1.

Example :LXI:MDNS:ENAB 0

:LXI:MDNS:ENAB?

:LXI:MDNS:HNAMe[:RESolved]?

Returns the resolved mDNS hostname.

Syntax :LXI:MDNS:HNAMe[:RESolved]?

Query response *desired mDNS hostname-N* <newline>

N is an integer appended as necessary to make the name unique. Response data type is SRD.

Example :LXI:MDNS:HNAM?

:LXI:MDNS:SNAMe:DESired

Sets the desired mDNS service name.

Syntax :LXI:MDNS:SNAMe:DESired *name*

:LXI:MDNS:SNAMe:DESired?

Parameter *name* Desired mDNS service name. Up to 15 ASCII characters. Parameter data type is SPD.

Query response *name* <newline>

name returns the desired mDNS service name. Response data type is SRD.

Example :LXI:MDNS:SNAM:DES “B2980”

:LXI:MDNS:SNAM:DES?

:LXI:MDNS:SNAMe[:RESolved]?

Returns the resolved mDNS service name.

Syntax :LXI:MDNS:SNAMe[:RESolved]?

Query response *desired mDNS service name-N* <newline>

N is an integer appended as necessary to make the name unique. Response data type is SRD.

Example :LXI:MDNS:SNAM?

:TRIGger:LXI:LAN[:SET]:DELay

Optional command. <delay> is a double representing the delay, in seconds. Any <delay> value other than 0.0 is changed to 0 (zero) and a warning is inserted into the SCPI error queue.

NOTE This implementation assumes that there is no Precision Time Protocol (PTP) implementation on the instrument. Delay requires PTP, so without it the delay must be zero.

Syntax :TRIGger:LXI:LAN[:SET]:DELay *delay*

:TRIGger:LXI:LAN[:SET]:DELay?

Parameter ***delay*** Delay, in seconds. Parameter data type is NRf.

Query response *delay* <newline>

delay returns the delay, in seconds. Response data type is NR3.

Example :TRIG:LXI:LAN:DEL 10.0

:TRIG:LXI:LAN:DEL?

:TRIGger:LXI:LAN[:SET]:DETection

Specifies the behavior of the trigger signal.

Syntax :TRIGger:LXI:LAN[:SET]:DETection *strLanEvent*, *detection*

:TRIGger:LXI:LAN[:SET]:DETection? *strLanEvent*

Parameter

strLanEvent	LAN event name. String enclosed in quotes. Parameter data type is SPD.
detection	Detection type. RISE (default)|FALL|HIGH|LOW. Parameter data type is CPD.

RISE sets the instrument to trigger on the receipt of a signal LOW LAN event followed by a signal HIGH LAN event (rising edge).

FALL sets the instrument to trigger on the receipt of a signal HIGH LAN event followed by a signal LOW LAN event (falling edge).

HIGH sets the instrument to trigger on every signal HIGH LAN event.

LOW sets the instrument to trigger on every signal LOW LAN event.

Query response *detection* <newline>

detection returns a string with the DETection setting for the specified event. Response data type is CRD.

Example :TRIG:LXI:LAN:DET “LAN0”,RISE

:TRIG:LXI:LAN:DET? “LAN0”

:TRIGger:LXI:LAN[:SET]:ENABle

Enables or disables the specified LAN trigger. When enabled the instrument will trigger upon receiving any event from the LXI trigger LAN event list.

Syntax :TRIGger:LXI:LAN[:SET]:ENABle *strLanEvent*, *status*

:TRIGger:LXI:LAN[:SET]:ENABle? *strLanEvent*

Parameter

strLanEvent	LAN event name. String enclosed in quotes. Parameter data type is SPD.
status	Status. 0|OFF (default)|1|ON. Parameter data type is boolean.

Query response *status* <newline>

status returns an integer (0 or 1) denoting the enabled status of the specified event. Response data type is boolean.

Example :TRIG:LXI:LAN:ENAB “LAN0”,1

:TRIG:LXI:LAN:ENAB? “LAN0”

:TRIGger:LXI:LAN[:SET]:FILTer

Allows user to create a filter expression for the specified LAN trigger event. Only LXI trigger LAN events coming from hosts matching the filter string are processed.

Syntax :TRIGger:LXI:LAN[:SET]:FILTer *strLanEvent*, *strFilterExpression*

:TRIGger:LXI:LAN[:SET]:FILTer? *strLanEvent*

Parameter

strLanEvent LAN event name. String enclosed in quotes. Parameter data type is SPD.

strFilterExpression Filter. String enclosed in quotes. You can use letters (A-Z, a-z), numbers (0-9), and printable characters like “@”, “%”, “*”, etc. Parameter data type is SPD.

Query response *strFilterExpression* <newline>

strFilterExpression returns a string with the filter expression for the specified event. Response data type is SRD.

Example :TRIG:LXI:LAN:FILT “LAN0”,”ALL:5024”

:TRIG:LXI:LAN:FILT? “LAN0”

:TRIGger:LXI:LAN[:SET]:IDENtifier

Sets the string that is expected to arrive over the LAN for a given trigger LAN event to occur.

Syntax :TRIGger:LXI:LAN[:SET]:IDENtifier *strLanEvent*, *strCustomId*

:TRIGger:LXI:LAN[:SET]:IDENtifier? *strLanEvent*

Parameter

strLanEvent LAN event name. String enclosed in quotes. Parameter data type is SPD.

strCustomId Identifier. String of up to 16 characters enclosed in quotes. You can use letters (A-Z, a-z), numbers (0-9), and printable characters like “@”, “%”, “*”, etc. Parameter data type is SPD.

Query response *strCustomId* <newline>

strCustomId returns the identifier string for the specified event. Response data type is SRD.

Example :TRIG:LXI:LAN:IDEN “LAN0”,”MyEvent”

:TRIG:LXI:LAN:IDEN? “LAN0”

MEASure Subsystem

:MEASure?

Executes a spot (one-shot) measurement for the parameters specified by the [:SENSe]:FUNCtion[:ON] command, and returns the measurement result data specified by the :FORMat:ELEMents:SENSe command. Measurement conditions must be set by SCPI commands or front panel operation before executing this command.

Syntax :MEASure? [*chanlist*]

Parameter

chanlist	Channels to perform measurement. Parameter data type is channel list. (@1). See "Channel List Parameter" on page 27.

(@1) selects channel 1.

If this parameter is not specified, *chanlist* = (@1) is set.

Query response *response* <newline>

response returns the measurement result data. Response data type is NR3. See "Data Output Format" on page 31.

See the following example. With the ASCII data output format, each data is separated by a comma.

ch1curr,*ch1time*

This example shows the data containing the current data (*ch1curr*) and time data (*ch1time*) of channel 1.

If the measurement function is not enabled or no data exists, *response* returns +9.910000E+37 (ASCII) or NaN (IEEE-754) which indicates "not a number."

Example

```
:SENS:FUNC CURR
:FORM:ELEM:SENS CURR,TIME
:MEAS? (@1)
```

:MEASure:<CHARge|CURRent[:DC]|RESistance|VOLTage[:DC]>?

Executes a spot (one-shot) measurement and returns the measurement result data. Measurement conditions must be set by SCPI commands or front panel operation before executing this command. Measurement item can be set to CHARge, CURRent, RESistance, or VOLTage.

Syntax :MEASure:<CHARge|CURRent[:DC]|RESistance|VOLTage[:DC]>? [*chanlist*]

For <CHARge|CURRent[:DC]|RESistance|VOLTage[:DC]>, select CHARge for charge measurement, CURRent[:DC] for current measurement, RESistance for resistance measurement, or VOLTage[:DC] for voltage measurement.

NOTE B2981B/B2983B can use CURRent[:DC] only.

Parameter

chanlist	Channels to perform measurement. Parameter data type is channel list. (@1). See "Channel List Parameter" on page 27.

(@1) selects channel 1.

If this parameter is not specified, *chanlist* = (@1) is set.

Query response *response* <newline>

response returns the measurement result data. Response data type is NR3. See "Data Output Format" on page 31.

See the following example.

ch1curr

This example shows the data containing the current data (*ch1curr*) of channel 1.

If the measurement function is not enabled or no data exists, *response* returns +9.910000E+37 (ASCII) or NaN (IEEE-754) which indicates "not a number."

Example

```
:SENS:FUNC CURR
:MEAS:CURR? (@1)
```

MMEMory Subsystem

:MMEMory:CATalog?

Returns the memory usage and availability. Also returns the list of files and folders in the current specified directory.

Syntax :MMEMory:CATalog? [*directory*]

Parameter

directory	Directory name, <path>|USB:\<path>. Either / (slash) or \ (backslash) can be used as the path separator. Up to 255 ASCII characters. Parameter data type is SPD.

If *directory* is not set, this function is applied to the current directory.

If *directory* = <path>, this function is applied to <current directory>\<path>.

If *directory* = USB:\<path>, this function is applied to USB:\<path>. Where, USB:\ is the root directory of the USB flash drive connected to the front panel.

Error occurs if the specified directory does not exist or is set to hidden or system.

Query response *used*,*free*{,item}<newline>

used returns the size of the used space, in bytes. Response data type is NR1.

free returns the size of the free space, in bytes. Response data type is NR1.

item returns the file or directory information. Response data type is SRD.

For files, *item* returns a string *name*,*type*,*size* which indicates the file name, file type, and file size. Where, *type* returns “ASC”, “BIN”, “STAT” or “MACR” for the file extensions “csv”, “dat”, “sta”, and “mac”, respectively.

For a directory, *item* returns a string *name*,*type*,*size*. Where, *name* indicates the directory name, and *type*,*size* always returns “FOLD,0”.

Example :MMEM:CAT? “USB:\b2980\device1\result”

:MMEM:CAT? “b2980\device1\result”

:MMEMory:CDIRectory

Changes the current directory to the specified directory.

Syntax :MMEMory:CDIRectory *directory*

:MMEMory:CDIRectory?

Parameter

directory	Directory name, <path>|USB:\<path>. Either / (slash) or \ (backslash) can be used as the path separator. Up to 255 ASCII characters. Parameter data type is SPD.

If *directory* = <path>, the next current directory will be <current directory>\<path>.

If *directory* = USB:\<path>, the next current directory will be USB:\<path>. Where, USB:\ is the root directory of the USB flash drive connected to the front panel.

Error occurs if the specified directory does not exist or is set to hidden or system.

Query response *directory* <newline>

directory returns the full path of the current directory. Response data type is SRD.

Example :MMEM:CDIR “USB:\b2980\device1\result”

:MMEM:CDIR?

:MMEMory:COPY

Makes a copy of an existing file in the current directory.

Syntax :MMEMory:COPY *source*,*destination*

Parameter

source	Source file name. Name of the original file.
destination	Copy file name. Or directory name, <path>|USB:\<path>. Either / (slash) or \ (backslash) can be used as the path separator.

Length of parameters is up to 255 ASCII characters. Parameter data type is SPD.

If *destination* is a file name, the copy file is created in the current directory.

If *destination* = <path>, the source file is duplicated in <current directory>\<path>.

If *destination* = USB:\<path>, the source file is duplicated in USB:\<path>. Where, USB:\ is the root directory of the USB flash drive connected to the front panel.

Error occurs if the source file does not exist or the destination file already exists.

Example :MMEM:COPY “original.dat”,“original_copy.dat”

:MMEM:COPY “original.dat”,“USB:\b2980\device1\result”

:MMEMory:DELete

Deletes a file in the current directory.

Syntax :MMEMory:DELete *file_name*

Parameter

file_name	Name of the file to delete. Up to 255 ASCII characters. Parameter data type is SPD.

Error occurs if the specified file does not exist.

Example :MMEM:DEL “original_copy.dat”

:MMEMory:LOAD:LIST:VOLTage

Loads a list sweep data from the specified file in the current directory.

Syntax :MMEMory:LOAD:LIST:VOLTage *file_name*[,*chanlist*]

Parameter

file_name	Name of the file used to save the specified data. Up to 255 ASCII characters. Parameter data type is SPD. File extension must be *csv, txt, or prn*, which is meaningful for the :MMEMory:CATalog? result.
chanlist	Channels to collect the data for. Parameter data type is channel list. (@1). See “Channel List Parameter” on page 27.

(@1) selects channel 1.

If *chanlist* is not specified, *chanlist* = (@1) is set.

Example :MMEM:LOAD:LIST:VOLT “VoltageList.csv”

:MMEMory:LOAD:MACRo

Loads a macro from the specified file in the current directory.

Syntax :MMEMory:LOAD:MACRo *macro*,*file_name*

Parameter *macro* Name of macro.

file_name Name of the file which contains the macro. File extension must be *mac*.

Length of parameters is up to 255 ASCII characters. Parameter data type is SPD.

Example :MMEM:LOAD:MACR “abc”,“MacroData1.mac”

:MMEMory:LOAD:STATe

Loads an instrument setup from the specified file in the current directory.

Syntax :MMEMory:LOAD:STATe *file_name*

Parameter *file_name* Name of the file which contains the instrument setup. File extension must be *sta*. Up to 255 ASCII characters. Parameter data type is SPD.

Example :MMEM:LOAD:STAT “SetupData1.sta”

:MMEMory:MDIRectory

Creates a new directory.

Syntax :MMEMory:MDIRectory *directory*

Parameter *directory* Directory name, <path>|USB:\<path>. Either / (slash) or \ (backslash) can be used as the path separator. Up to 255 ASCII characters. Parameter data type is SPD.

If *directory* = <path>, this command creates a <current directory>\<path> directory.

If *directory* = USB:\<path>, this command creates a USB:\<path> directory. Where, USB:\ is the root directory of the USB flash drive connected to the front panel.

Example :MMEM:MDIR “USB:\b2980\device1\setup”

:MMEMory:MOVE

Moves or renames an existing file in the current directory.

Syntax :MMEMory:MOVE *source*,*destination*

Parameter

source	Source file name. Name of the original file.
destination	New file name. Or directory name, <path>|USB:\<path>. Either / (slash) or \ (backslash) can be used as the path separator.

Length of parameters is up to 255 ASCII characters. Parameter data type is SPD.

If *destination* is a file name, the source file is renamed to the new file name in the current directory.

If *destination* = <path>, the source file is moved to <current directory>\<path>.

If *destination* = USB:\<path>, the source file is moved to USB:\<path>. Where, USB:\ is the root directory of the USB flash drive connected to the front panel.

Error occurs if the source file does not exist or the destination file already exists.

Example :MMEM:MOVE “original.dat”,“new.dat”

:MMEM:MOVE “original.dat”,“USB:\b2980\device1\result”

:MMEMory:RDIRectory

Removes the specified empty directory.

Syntax :MMEMory:RDIRectory *directory*

Parameter

directory	Directory name, <path>|USB:\<path>. Either / (slash) or \ (backslash) can be used as the path separator. Up to 255 ASCII characters. Parameter data type is SPD.

If *directory* = <path>, this command removes the <current directory>\<path> directory.

If *directory* = USB:\<path>, this command removes the USB:\<path> directory. Where, USB:\ is the root directory of the USB flash drive connected to the front panel.

Error occurs if the specified directory is not empty.

Example :MMEM:RDIR “USB:\b2980\device1\setup”

:MMEMory:STORe:DATA<:LIMit|:MATH|:SENSe|[:ALL]>

Saves the limit test data, math expression result data, sense data, or all of these data for the specified channel to the specified file in the current directory.

Syntax :MMEMory:STORe:DATA<:LIMit|:MATH|:SENSe|[:ALL]> *file_name*[,*chanlist*]

For <:LIMit|:MATH|:SENSe|[:ALL]>, specify :LIMit for limit test data, :MATH for math expression result data, :SENSe for sense data, or [:ALL] for all of these data.

Parameter

file_name	Name of the file used to save the specified data. Up to 255 ASCII characters. Parameter data type is SPD. File extension must be *dat*, which is meaningful for the :MMEMory:CATalog? result.
chanlist	Channel to collect the data for. Parameter data type is channel list. (@1). See "Channel List Parameter" on page 27.

(@1) selects channel 1.

If *chanlist* is not specified, *chanlist* = (@1) is set.

Example :MMEM:STOR:DATA "AllData1.dat"

:MMEMory:STORe:LIST:VOLTage

Saves the list sweep data to the specified file in the current directory.

Syntax :MMEMory:STORe:LIST:|VOLTage *file_name*[,*channel*]

Parameter

file_name	Name of the file used to save the specified data. Up to 255 ASCII characters. Parameter data type is SPD. File extension must be *csv, txt, or prn*, which is meaningful for the :MMEMory:CATalog? result.
channel	Channel to collect the data for. Parameter data type is channel list. (@1). See "Channel List Parameter" on page 27.

If *channel* is not specified, *channel* = (@1) is set.

Example :MMEM:STOR:LIST:VOLT "VoltageData1.csv"

:MMEMory:STORe:MACRo

Saves the macro to the specified file in the current directory.

Syntax :MMEMory:STORe:MACRo *macro*,*file_name*

Parameter

macro	Name of macro.
file_name	Name of the file used to save the macro. File extension must be *mac*, which is meaningful for the :MMEMory:CATalog? result.

Length of parameters is up to 255 ASCII characters. Parameter data type is SPD.

Example :MMEM:STOR:MACR “abc”,“MacroData1.mac”

:MMEMory:STORe:STATe

Saves the instrument setup to the specified file in the current directory.

Syntax :MMEMory:STORe:STATe *file_name*

Parameter

file_name	Name of the file used to save the instrument setup. Up to 255 ASCII characters. Parameter data type is SPD. File extension must be *sta*, which is meaningful for the :MMEMory:CATalog? result.

Example :MMEM:STOR:STAT “SetupData1.sta”

:MMEMory:STORe:TRACe

Saves all data in the trace buffer for the specified channel to the specified file in the current directory.

Syntax :MMEMory:STORe:TRACe *file_name*[,*chanlist*]

Parameter

file_name	Name of the file used to save the specified data. Up to 255 ASCII characters. Parameter data type is SPD. File extension must be *tra*, which is meaningful for the :MMEMory:CATalog? result.
chanlist	Channel to get the data. Parameter data type is channel list. (@1). See “Channel List Parameter” on page 1-8.

(@1) selects channel 1.

If *chanlist* is not specified, *chanlist* = (@1) is set.

Example :MMEM:STOR:TRAC “AllTraceData1.dat”

OUTPut Subsystem

NOTE OUTPut subsystem commands are available on B2985B/B2987B.

For the numeric suffix [*c*], see "Numeric Suffix" on page 27.

:OUTPut:LOW

NOTE This command is available on B2985B/B2987B.

Selects if the low terminal is connected to common or is floating.

Before executing this command, the source output must be disabled by the :OUTPut[:STATe] command. Or else, an error occurs.

Syntax :OUTPut[*c*]:LOW *low_state*

:OUTPut[*c*]:LOW?

Parameter ***low_state*** COMMon|FLOat(default). Parameter data type is CPD.

low_state = COMMon connects the low terminal to COMMON terminal on the rear panel.

low_state = FLOat makes the low terminal to floating state.

Query response *low_state* <newline>

If *low_state* is FLO, the low terminal is floating.
If *low_state* is COMM, the low terminal is connected to COMMON terminal.

Response data type is CRD.

Example :OUTP:LOW FLO

:OUTP:LOW?

:OUTPut:OFF:MODE

NOTE This command is available on B2985B/B2987B.

Selects the source condition after output off.

Syntax :OUTPut[*c*]:OFF:MODE *mode*

:OUTPut[*c*]:OFF:MODE?

Parameter ***mode*** ZERO|HIZ|NORMal (default). Parameter data type is CPD.

mode = NORMal selects the following source setup.

- Output voltage: 0 V
- Output relay: off (open or break)

mode = HIZ selects the following source setup.

- Voltage source setup is not changed if the source applies 21 V or less.
- Output relay: off (open or break)

mode = ZERO selects the following setup.

- Output voltage: 0 V
- Output relay: on

NOTE The source condition by this command is not applied to the output-off process triggered by the emergency condition such as the interlock open, and over temperature protection. Then the output voltage is immediately set to 0 V and the output switch is set to off.

Query response *mode* <newline>

mode is NORM, HIZ, or ZERO, and indicates the source condition after output off. Response data type is CRD.

Example :OUTP:OFF:MODE HIZ

:OUTP:OFF:MODE?

:OUTPut[:STATe]

NOTE This command is available on B2985B/B2987B.

Enables or disables the source output.

Syntax :OUTPut[*c*][:STATe] *mode*

:OUTPut[*c*][:STATe]?

Parameter ***mode*** 1|ON|0|OFF (default). Parameter data type is boolean.

mode = 1 or ON enables the source output.

mode = 0 or OFF disables the source output.

Query response *mode* <newline>

mode is 0 or 1, and indicates that the source output is off or on, respectively. Response data type is NR1.

Example :OUTP 1

:OUTP:STAT?

PROGram Subsystem

For the numeric suffix [*h*], see "Numeric Suffix" on page 27.

:PROGram:CATalog?

Returns the names of all programs defined in the program memory.

Even if a name is selected by the :PROGram[:SELected]:NAME command, this command does not return the name if the program is empty.

Syntax :PROGram:CATalog?

Query response *program_names* <newline>

program_names returns the names of all programs defined in the program memory. Response data type is AARD.

Example :PROG:CAT?

:PROGram:PON:COPY

Specifies the power-on program.

Syntax :PROGram:PON:COPY *name*

Parameter

name	Name of the program used for the power-on program. Parameter data type is SPD.

Example :PROG:PON:COPY "program1"

:PROGram:PON:DELete

Clears the power-on program.

Syntax :PROGram:PON:DELete

Example :PROG:PON:DEL

:PROGram:PON:RUN

Enables or disables the power-on program. The specified program automatically runs with each power-on. The program is specified by the :PROGram:PON:COPY command.

Syntax :PROGram:PON:RUN *mode*

mode 1|ON|0|OFF (default). Parameter data type is boolean.

mode = 1 or ON enables power-on program.

mode = 0 or OFF disables power-on program.

Query response *mode* <newline>

mode is 0 or 1, and indicates that the power-on program is disable or enable, respectively. Response data type is NR1.

Example :PROG:PON:RUN 1

:PROG:PON:RUN?

:PROGram[:SELected]:APPend

Adds a program code to the end of a program stored in the program memory.

Before executing this command, the program must be selected by the :PROGram[:SELected]:NAME command. Or else, an error occurs.

Syntax :PROGram[:SELected]:APPend *program_code*

Parameter ***program_code*** Program code. Up to 256 byte per execution. Sum of all program size in the program memory must be up to 100 KB. Parameter data type is block. Both definite length block and indefinite length block are available. Program code cannot contain control characters except for the trailing linefeed.

See the :PROGram[:SELected]:DEFine command for details.

Example
```
:PROG:NAME “program1”
:PROG:APP #212:INP:STAT ON
```

:PROGram[:SELected]:DEFine

Defines a program in the program memory by entering the initial program code.

Before executing this command, the program must be selected by the :PROGram[:SELected]:NAME command with a new program name. Or else, an error occurs.

Attempting to overwrite an existing program causes an error. Delete the program first by using the :PROGram[:SELected]:DELete[:SELected] command.

Syntax :PROGram[:SELected]:DEFine *program_code*

:PROGram[:SELected]:DEFine?

Parameter

program_code	Program code. Up to 256 byte per execution. Sum of all program size in the program memory must be up to 100 KB. Maximum of 100 programs can be memorized. Parameter data type is block. Both definite length block and indefinite length block are available. Program code cannot contain control characters except for the trailing linefeed.

For the definite length block, *program_code* must be #*nms* which consists of the header #*nm* and the command string *s*. For example, #213:OUTP:STAT ON.

- *n*: Number of digits for *m*. (ex: 2)
- *m*: Number of characters (8-bit data bytes) for the command string. (ex: 13)
- *s*: Command string. (ex: :OUTP:STAT ON (total 13 characters))

For the indefinite length block, *program_code* must be #0*s* which consists of the header #0 and the command string *s*. For example, #0:OUTP:STAT ON.

In the command string, the following characters have special meaning.

- \n: Command delimiter
- %%: Percent (%) character
- #: Comment header

program_code does not support the following.

- Query commands
- SCPI common commands except for *CLS, *ESE, and *SRE commands

- Program subsystem commands except for :PROG:STAT command

program_code supports variables in the format %*h*% (*h*: integer. 1 to 100). It is replaced with the value set by the :PROGram:VARiable command before executing the program.

Example :PROG:NAME “program1”
:PROG:DEF #212:INP:STAT ON

:PROGram[:SELected]:DELete:ALL

Deletes all programs stored in the program memory.

If any of the programs are in the RUN state, this command causes an error and does not delete any program.

Syntax :PROGram[:SELected]:DELete:ALL

Example :PROG:DEL:ALL

:PROGram[:SELected]:DELete[:SELected]

Deletes a program stored in the program memory.

Before executing this command, the program must be selected by the :PROGram[:SELected]:NAME command. Or else, an error occurs.

If any of the programs are in the RUN state, this command causes an error and does not delete the selected program.

Syntax :PROGram[:SELected]:DELete[:SELected]

Example :PROG:NAME “program1”
:PROG:DEL

:PROGram[:SELected]:EXECute

Executes a program stored in the program memory.

Before executing this command, the program must be selected by the :PROGram[:SELected]:NAME command. Or else, an error occurs.

If any of the programs are in the RUN state, this command causes an error and does not execute the selected program.

Syntax :PROGram[:SELected]:EXECute

Example
```
:PROG:NAME “program1”
:PROG:EXEC
```

:PROGram[:SELected]:NAME

Selects the program for performing the action by the following commands.

If *name* does not specify the program stored in the program memory, this command creates a new program with the specified name and selects the program.

If *name* specifies an existing program, this command selects the program.

Syntax :PROGram[:SELected]:NAME *name*

:PROGram[:SELected]:NAME?

Parameter

name	Program name. Up to 32 ASCII characters without any control characters, space characters, single and double quotes, and comma. Parameter data type is SPD.

Query response *name* <newline>

name returns the name of the program currently selected. Response data type is SRD.

Example
```
:PROG:NAME “program1”
:PROG:SEL:NAME?
```

:PROGram[:SELected]:STATe

Changes the execution status of a program stored in the program memory.

Before executing this command, the program must be selected by the :PROGram[:SELected]:NAME command. Or else, an error occurs.

Syntax :PROGram[:SELected]:STATe *operation*

:PROGram[:SELected]:STATe?

Parameter

operation	RUN|PAUSe|STEP|STOP|CONTinue. Parameter data type is CPD. See Table 4-4 for the status changed by this command.

Query response *status* <newline>

status returns the present execution status, Running, Paused, or Stopped. Response data type is CRD.

Example :PROG:STAT PAUS

:PROG:SEL:STAT?

Table 4-4 Execution Status Changed by :PROG:STAT Command

operation	**Present execution status**		
	Running	**Paused**	**Stopped**
RUN	Error	to Running	to Running
PAUSe	to Paused	Paused	Stopped
STEP	Error	to Running to Paused	to Running to Paused
STOP	to Stopped	to Stopped	Stopped
CONTinue	Error	to Running	Error

:PROGram[:SELected]:WAIT?

Blocks other commands until the program execution status changes to Paused or Stopped.

Syntax :PROGram[:SELected]:WAIT? *timeout*

Parameter ***timeout*** Timeout value, in seconds. Parameter data type is NRf+.

Query response *status* <newline>

status returns 1 if the execution status changes to Paused or Stopped within the specified timeout, or 0 if a timeout occurs and the status is still in Running. Response data type is NR1.

Example :PROG:WAIT? 1

:PROGram:VARiable

Sets a value to the variable specified by *h*.

Variables can be used in the memory program. They must be expressed as %*h*% (*h*: integer. 1 to 100) in the memory program.

Syntax :PROGram:VARiable[*h*] *value*

:PROGram:VARiable[*h*]?

Parameter

value	Value of the variable specified by *h*. Up to 32 ASCII characters. Parameter data type is SPD.

Example :PROG:VAR “1”

:PROG:VAR100?

READ Subsystem

:READ:ARRay?

Executes the :INITiate:ACQuire command and the :FETCh:ARRay? command in series, and returns the array data which contains all of the voltage measurement data, current measurement data, charge measurement data, resistance measurement data, time data, status data, source output setting data, temperature data, or humidity data specified by the :FORMat:ELEMents:SENSe command. The data is not cleared until the :INITiate, :MEASure, :READ, or [:SENSe]:DATA:CLEar command is executed.

NOTE B2981B/B2983B support only current measurement data, time data, and status data.

NOTE The unit of temperature data is specified by :SYSTem:TEMPerature:UNIT command.

Syntax :READ:ARRay? [*chanlist*]

Parameter

chanlist Channels to return the data. Parameter data type is channel list. (@1). See "Channel List Parameter" on page 27.

(@1) selects channel 1.

If this parameter is not specified, *chanlist* = (@1) is set.

Query response *response* <newline>

response returns the array data specified by the :FORMat:ELEMents:SENSe command. Response data type is NR3. See "Data Output Format" on page 31.

response returns the channel 1 data. See the following example. With the ASCII data output format, each data is separated by a comma.

ch1curr1,*ch1sour1*,*ch1curr2*,*ch1sour2*, ...
ch1curr5,*ch1sour5*,*ch1curr6*,*ch1sour6*,+9.910000E+37,+9.910000E+37, ...
ch1curr10,*ch1sour10*,+9.910000E+37,+9.910000E+37

This example shows the data containing the current data (*ch1currN*) and source data (*ch1sourN*) of the 10-step sweep measurement by channel 1.

If the measurement function is not enabled or no data exists, *response* returns +9.910000E+37 (ASCII) or NaN (IEEE-754) which indicates "not a number."

Example

```
:FORM:ELEM:SENS CURR,TIME
:READ:ARR? (@1)
```

:READ:ARRay:<CHARge|CURRent|HUMidity|RESistance|SOURce|STATus|TEMPerature|TIME|VOLTage>?

Executes the :INITiate:ACQuire command and the :FETCh:ARRay:<CHARge|CURRent|HUMidity|RESistance|SOURce|STATus|TEMPerature|TIME|VOLTage>? command in series, and returns the array data which contains all of the charge measurement data, current measurement data, resistance measurement data, source output setting data, status data, time data, or voltage measurement data specified by CURRent, HUMidity, RESistance, SOURce, STATus, TEMPerature, TIME, or VOLTage. The data is not cleared until the :INITiate, :MEASure, :READ, or [:SENSe]:DATA:CLEar command is executed.

NOTE The unit of temperature data is specified by :SYSTem:TEMPerature:UNIT command.

Syntax

```
:READ:ARRay:<CHARge|CURRent|HUMidity|RESistance|SOURce|STATus|
TEMPerature|TIME|VOLTage>? [chanlist]
```

For <CHARge|CURRent|RESistance|SOURce|STATus|TIME|VOLTage>, specify CURRent for current measurement data, HUMidity for humidity data, RESistance for resistance measurement data, SOURce for source output setting data, STATus for status data, TEMPerature for temperature data, TIME for time data, or VOLTage for voltage measurement data.

NOTE On B2981B/B2983B, only CURRent, STATus, and TIME can be specified.

Parameter

chanlist	Channels to return the data. Parameter data type is channel list. (@1). See "Channel List Parameter" on page 27.

(@1) selects channel 1.

If this parameter is not specified, *chanlist* = (@1) is set.

Query response *response* <newline>

response returns the array data specified by CURRent, RESistance, SOURce, STATus, TIME, or VOLTage. Response data type is NR3. See "Data Output Format" on page 31.

response returns the channel 1 data. See the following example. With the ASCII data output format, each data is separated by a comma.

ch1curr1,*ch1curr2*, ...
ch1curr5,*ch1curr6*,+9.910000E+37, ... ,*ch1curr10*,+9.910000E+37

This example shows the data containing the current data (*ch1currN*) of the 10-step sweep measurement by channel 1.

If the measurement function is not enabled or no data exists, *response* returns +9.910000E+37 (ASCII) or NaN (IEEE-754) which indicates "not a number."

Example :READ:ARR:CURR? (@1)

:READ[:SCALar]?

Executes the :INITiate:ACQuire command and the :FETCh[:SCALar]? command in series, and returns the latest voltage measurement data, current measurement data, charge measurement data, resistance measurement data, time data, status data, source output setting data, temperature data, or humidity data specified by the :FORMat:ELEMents:SENSe command. The data is not cleared until the :INITiate, :MEASure, :READ, or [:SENSe]:DATA:CLEar command is executed.

NOTE B2981B/B2983B supports current measurement data, time data, and status data only.

NOTE The unit of temperature data is specified by :SYSTem:TEMPerature:UNIT command.

Syntax :READ[:SCALar]? [*chanlist*]

Parameter

chanlist	Channels to return the data. Parameter data type is channel list. (@1). See "Channel List Parameter" on page 27.

(@1) selects channel 1.

If this parameter is not specified, *chanlist* = (@1) is set.

Query response *response* <newline>

response returns the latest data specified by the :FORMat:ELEMents:SENSe command. Response data type is NR3. See "Data Output Format" on page 31.

response returns the channel 1 data. See the following example. With the ASCII data output format, each data is separated by a comma.

ch1curr10,*ch1sour10*

This example shows the data containing the latest current data (*ch1curr10*) and source data (*ch1sour10*) of the 10-step sweep measurement by channel 1.

If the measurement function is not enabled or no data exists, *response* returns +9.910000E+37 (ASCII) or NaN (IEEE-754) which indicates "not a number."

Example

```
:FORM:ELEM:SENS CURR,TIME
:READ? (@1)
```

:READ[:SCALar]:<CHARge|CURRent|HUMidity|RESistance|SOURce|STATus|TEMPerature|TIME|VOLTage>?

Executes the :INITiate:ACQuire command and the :FETCh[:SCALar]:<CHARge|CURRent|HUMidity|RESistance|SOURce|STATus|TEMPerature|TIME|VOLTage>? command in series, and returns the latest voltage measurement data, current measurement data, charge measurement data, resistance measurement data, time data, status data, source output setting data, temperature data, or humidity data specified by CHARge, CURRent, RESistance, SOURce, STATus, TIME, or VOLTage. The data is not cleared until the :INITiate, :MEASure, :READ, or [:SENSe]:DATA:CLEar command is executed.

NOTE The unit of temperature data is specified by :SYSTem:TEMPerature:UNIT command.

Syntax :READ[:SCALar]:<CHARge|CURRent|HUMidity|RESistance|SOURce|STATus|TEMPerature|TIME|VOLTage>? [*chanlist*]

For <CHARge|CURRent|HUMidity|RESistance|SOURce|STATus|TEMPerature|TIME|VOLTage>, specify CHARge for charge measurement data, CURRent for current measurement data, RESistance for resistance measurement data, SOURce for source output setting data, STATus for status data, TIME for time data, or VOLTage for voltage measurement data.

NOTE On B2981B/B2983B, only CURRent, STATus, and TIME can be specified.

Parameter

chanlist	Channels to return the data. Parameter data type is channel list. (@1). See "Channel List Parameter" on page 27.

(@1) selects channel 1.

If this parameter is not specified, *chanlist* = (@1) is set.

Query response *response* <newline>

response returns the latest data specified by CHARge, CURRent, RESistance, SOURce, STATus, TIME, or VOLTage. Response data type is NR3. See "Data Output Format" on page 31.

response returns the channel 1 data. See the following example.

ch1curr10

This example shows the data containing the latest current data (*ch1curr10*) of the 10-step sweep measurement by channel 1.

If the measurement function is not enabled or no data exists, *response* returns +9.910000E+37 (ASCII) or NaN (IEEE-754) which indicates "not a number."

Example :READ:CURR? (@1)

SENSe Subsystem

For the numeric suffix [*c*], see "Numeric Suffix" on page 27.

[:SENSe]:ARSP

This command specifies the automatic ranging operation.

Syntax [:SENSe[*c*]]:ARSP *mode*

[:SENSe[*c*]]:ARSP?

Parameter

mode		Automatic ranging operation. NORMal(default)|FAST. Parameter data type is CPD.

mode = NORMal sets the normal mode. Performs the basic auto-ranging operation. The channel automatically sets the range which provides the best resolution in performing the measurement.

mode = FAST sets the fast mode. For performing high throughput measurement, reduces observation time to determine measurement range. This setup can be used when the measurement signal noise (superposed from AC power, etc.) is low.

NOTE NORMal|FAST parameter of this command corresponds to Automatic ranging speed Normal|Fast in front panel operation.

Query response *mode* <newline>

mode returns the present setting, NORM or FAST. Response data type is CRD.

Example :ARSP NORM

:SENS:ARSP?

[:SENSe]:CHARge:ADIScharge:LEVel

NOTE This command is available on B2985B/B2987B.

Specifies the level of the automatic discharge function for charge measurement.

Syntax [:SENSe[*c*]]:CHARge:ADIScharge:LEVel *level*

[:SENSe[*c*]]:CHARge:ADIScharge:LEVel?

Parameter ***level*** *value* (–2.1E–6 to +2.1E–6)|MINimum|MAXimum|DEFault (default is 2.0E–6 coulomb). Parameter data type is NRf+. Query does not support *offset* = *value*.

A positive side of discharge level is set to |*value*|, which is an absolute value of *value*, and a negative side of discharge level is set to –|*value*|.

If a charge measurement value is larger than or equal to |*value*|, or smaller than or equal to –|*value*|, discharging is executed automatically.

Query response *level* <newline>

level returns the present level of the automatic discharge function. If a parameter is specified, *level* returns the value assigned to DEF, MIN, or MAX. Response data type is NR3.

Example :CHAR:ADIS:LEV 1.0E–6

:SENS:CHAR:ADIS:LEV?

[:SENSe]:CHARge:ADIScharge[:STATe]

NOTE This command is available on B2985B/B2987B.

Enables or disables the automatic discharge function.

The automatic discharge level is set by the [:SENSe]:CHARge:ADIScharge:LEVel command.

Syntax [:SENSe[*c*]]:CHARge:ADIScharge[:STATe] *mode*

[:SENSe[*c*]]:CHARge:ADIScharge[:STATe]?

Parameter ***mode*** 1|ON|0|OFF (default). Parameter data type is boolean.

mode = 1 or ON enables automatic discharge function.

mode = 0 or OFF disables automatic discharge function.

Query response *mode* <newline>

mode is 0 or 1, and indicates that automatic discharge function is off or on, respectively. Response data type is NR1.

Example :CHAR:ADIS 1

:SENS:CHAR:ADIS:STAT?

[:SENSe]:CHARge:DISCharge

NOTE This command is available on B2985B/B2987B.

Discharges the capacitor used for charge measurement.

Syntax [:SENSe[*c*]]:CHARge:DISCharge

Example :SENS:CHAR:DISC

[:SENSe]:CHARge:RANGe:AUTO:GROup

NOTE This command is available on B2985B/B2987B.

Selects the range group for the automatic measurement ranging operation at charge measurement.

Syntax [:SENSe[*c*]]:CHARge:RANGe:AUTO:GROup group

[:SENSe[*c*]]:CHARge:RANGe:AUTO:GROup?

Parameter ***group*** Range group. HIGH(default)|LOW. Parameter data type is CPD.

group = HIGH selects the group of 200 nC range, 2 μC range.

group = LOW selects the group of 2 nC range, 20 nC range.

Query response *response* <newline>

response returns the present setting, HIGH or LOW. Response data type is CRD.

Example :CHAR:RANG:AUTO:GRO LOW

:SENS:CHAR:RANG:AUOTO:GRO?

[:SENSe]:<CHARge|CURRent[:DC]|RESistance|VOLTage[:DC]>:APERture

Sets the integration time for one point measurement.

Syntax :SENSe[*c*]:<CHARge|CURRent[:DC]|RESistance|VOLTage[:DC]>:APERture *time*

:SENSe[*c*]:<CHARge|CURRent[:DC]|RESistance|VOLTage[:DC]>:APERture? [*time*]

For <CHARge|CURRent[:DC]|RESistance|VOLTage[:DC]>, specify CHARge for charge measurement, CURRent[:DC] for current measurement, RESistance for resistance measurement, or VOLTage[:DC] for voltage measurement. Specifying the measurement item is not important because the *time* value is common for all items.

NOTE On B2981B/B2983B, only CURRent[:DC] can be specified.

Parameter

time	*value* (+1E–5 to +2 seconds)|MINimum|MAXimum|DEFault (default is 0.1 PLC, = 0.1/*power line frequency*). Parameter data type is NRf+. Query does not support *time* = *value*. If you specify the value less than MIN or greater than MAX, *time* is automatically set to MIN or MAX.

The integration time can be expressed by the following formula, using the NPLC value set by the [:SENSe]:<CHARge|CURRent[:DC]|RESistance|VOLTage[:DC]>:NPLCycles command. So the last command setting is effective for both *time* and *nplc*.

time = nplc / power line frequency

Query response *time* <newline>

time returns the present setting of the integration time. If a parameter is specified, *time* returns the value assigned to DEF, MIN, or MAX. Response data type is NR3.

Example :SENS:CUR:APER 2E–3

:SENS:CURR:DC:APER?

[:SENSe]:<CHARge|CURRent[:DC]|RESistance|VOLTage[:DC]>:APERture:AUTO

Enables or disables the automatic aperture function.

Syntax :SENSe[*c*]:<CHARge|CURRent[:DC]|RESistance|VOLTage[:DC]>:APERture:AUTO *mode*

:SENSe[*c*]:<CHARge|CURRent[:DC]|RESistance|VOLTage[:DC]>:APERture:AUTO?

For <CHARge|CURRent[:DC]|RESistance|VOLTage[:DC]>, specify CHARge for charge measurement, CURRent[:DC] for current measurement, RESistance for resistance measurement, or VOLTage[:DC] for voltage measurement. Specifying the measurement item is not important because the *mode* value is common for all items.

NOTE On B2981B/B2983B, only CURRent[:DC] can be specified.

Parameter *mode* 0|OFF|1|ON (default). Parameter data type is boolean.

mode = 0 or OFF disables the automatic aperture function.

mode = 1 or ON enables the automatic aperture function. If this function is enabled, the instrument automatically sets the integration time (NPLC value) suitable for the measurement range.

The automatic aperture on/off works with the automatic NPLC on/off set by the [:SENSe]:<CHARge|CURRent[:DC]|RESistance|VOLTage[:DC]>:NPLCycles:AUTO command. So the last command setting is effective for both functions.

Query response *mode* <newline>

mode is 0 or 1, and indicates that the automatic aperture function is off or on, respectively. Response data type is NR1.

Example :SENS:CUR:APER:AUTO 0

:SENS:CURR:DC:APER:AUTO?

[:SENSe]:<CHARge|CURRent[:DC]|RESistance|VOLTage[:DC]>:APERture:AUTO:MODE

Selects the automatic aperture mode.

Syntax :SENSe[*c*]:<CHARge|CURRent[:DC]|RESistance|VOLTage[:DC]>:APERture:AUTO:MODE *mode*

:SENSe[*c*]:<CHARge|CURRent[:DC]|RESistance|VOLTage[:DC]>:APERture:AUTO:MODE?

For <CHARge|CURRent[:DC]|RESistance|VOLTage[:DC]>, specify CHARge for charge measurement, CURRent[:DC] for current measurement, RESistance for resistance measurement, or VOLTage[:DC] for voltage measurement. Specifying the measurement item is not important because the *mode* value is common for all items.

NOTE On B2981B/B2983B, only CURRent[:DC] can be specified.

Parameter

mode	Automatic aperture mode. SHORt|MEDium(default)|LONG. Parameter data type is CPD.

NOTE SHORt|MEDium|LONG parameter of this command corresponds to Automatic aperture mode Quick|Normal|Stable in front panel operation.

Query response *mode* <newline>

mode returns the present setting, SHOR, MED, or LONG. Response data type is CRD.

Example :SENS:CURR:APER:AUTO:MODE MED

:SENS:CURR:APER:AUTO:MODE?

[:SENSe]:<CHARge|CURRent[:DC]|RESistance|VOLTage[:DC]>:AVERage:MOVing:COUNt

Sets the number of measurement samples used for the moving average.

Syntax :SENSe[*c*]:<CHARge|CURRent[:DC]|RESistance|VOLTage[:DC]>:AVERage:MOVing:COUNt *mov_count*

:SENSe[*c*]:<CHARge|CURRent[:DC]|RESistance|VOLTage[:DC]>:AVERage:MOVing:COUNt? [*mov_count*]

For <CHARge|CURRent[:DC]|RESistance|VOLTage[:DC]>, specify CHARge for charge measurement, CURRent[:DC] for current measurement, RESistance for resistance measurement, or VOLTage[:DC] for voltage measurement. Specifying the measurement item is not important because the *mov_count* value is common for all items.

NOTE On B2981B/B2983B, only CURRent[:DC] can be specified.

Parameter *mov_count* number of measurement samples used for moving average. *value* (1 to 100)| MINimum|MAXimum|DEFault (default is 1). Parameter data type is NRf+.

Query does not support *mov_count* = *value*.

Query response *mov_count* <newline>

mov_count returns the present number of samples used for moving average. If a parameter is specified, *mov_count* returns the value assigned to DEF, MIN, or MAX. Response data type is NR1.

Example :SENS:CURR:AVER:MOV:COUN 10

:SENS:CURR:DC:AVER:MOV:COUN?

[:SENSe]:<CHARge|CURRent[:DC]|RESistance|VOLTage[:DC]>:AVERage:MOVing[:STATe]

Enables or disables the moving average filter.

Syntax :SENSe[*c*]:<CHARge|CURRent[:DC]|RESistance|VOLTage[:DC]>:AVERage:MOVing[:STATe] *mode*

:SENSe[*c*]:<CHARge|CURRent[:DC]|RESistance|VOLTage[:DC]>:AVERage:MOVing[:STATe]?

For <CHARge|CURRent[:DC]|RESistance|VOLTage[:DC]>, specify CHARge for charge measurement, CURRent[:DC] for current measurement, RESistance for resistance measurement, or VOLTage[:DC] for voltage measurement. Specifying the measurement item is not important because the *mode* value is common for all items.

NOTE On B2981B/B2983B, only CURRent[:DC] can be specified.

Parameter *mode* 0|OFF(default)|1|ON. Parameter data type is boolean.

mode = 0 or OFF disables the moving average filter.

mode = 1 or ON enables the moving average filter.

Query response *mode* <newline>

mode is 0 or 1, and indicates that the moving average filter is off or on, respectively. Response data type is NR1.

Example :SENS:CURR:AVER:MOV 0

:SENS:CURR:DC:AVER:MOV:STAT?

[:SENSe]:<CHARge|CURRent[:DC]|RESistance|VOLTage[:DC]>:NPLCycles

Sets the number of power line cycles (NPLC) value instead of setting the integration time for one point measurement.

Syntax :SENSe[*c*]:<CHARge|CURRent[:DC]|RESistance|VOLTage[:DC]>:NPLCycles *nplc*

:SENSe[*c*]:<CHARge|CURRent[:DC]|RESistance|VOLTage[:DC]>:NPLCycles? [*nplc*]

For <CHARge|CURRent[:DC]|RESistance|VOLTage[:DC]>, specify CHARge for charge measurement, CURRent[:DC] for current measurement, RESistance for resistance measurement, or VOLTage[:DC] for voltage measurement. Specifying the measurement item is not important because the *nplc* value is common for all items.

NOTE On B2981B/B2983B, only CURRent[:DC] can be specified.

Parameter *nplc* value (+5E–4 to +100 for 50 Hz or +6E–4 to +120 for 60 Hz)|MINimum|MAXimum|DEFault (default is 0.1 PLC, power line cycles). Parameter data type is NRf+. Query does not support *nplc* = *value*. If you specify the value less than MIN or greater than MAX, *nplc* is automatically set to MIN or MAX.

The NPLC value can be expressed by the following formula, using the integration time set by the [:SENSe]:<CHARge|CURRent[:DC]|RESistance|VOLTage[:DC]>:APERture command. So the last command setting is effective for both *nplc* and *time*.

nplc = *time* × *power line frequency*

Query response *nplc* <newline>

nplc returns the present setting of the number of power line cycles. If a parameter is specified, *nplc* returns the value assigned to DEF, MIN, or MAX. Response data type is NR3.

Example :SENS:CUR:NPLC 0.2

:SENS:CURR:DC:NPLC?

[:SENSe]:<CHARge|CURRent[:DC]|RESistance|VOLTage[:DC]>:NPLCycles:AUTO

Enables or disables the automatic NPLC function.

Syntax :SENSe[*c*]:<CHARge|CURRent[:DC]|RESistance|VOLTage[:DC]>:NPLCycles:AUTO *mode*

:SENSe[*c*]:<CHARge|CURRent[:DC]|RESistance|VOLTage[:DC]>:NPLCycles:AUTO?

For <CHARge|CURRent[:DC]|RESistance|VOLTage[:DC]>, specify CHARge for charge measurement, CURRent[:DC] for current measurement, RESistance for resistance measurement, or VOLTage[:DC] for voltage measurement. Specifying the measurement item is not important because the *mode* value is common for all items.

NOTE On B2981B/B2983B, only CURRent[:DC] can be specified.

Parameter *mode* 1|ON (default)|0|OFF. Parameter data type is boolean.

mode = 0 or OFF disables the automatic NPLC function.

mode = 1 or ON enables the automatic NPLC function. If this function is enabled, the instrument automatically sets the NPLC value (integration time) suitable for the measurement range.

The automatic NPLC on/off works with the automatic aperture on/off set by the [:SENSe]:<CHARge|CURRent[:DC]|RESistance|VOLTage[:DC]>:APERture:AUTO command. So the last command setting is effective for both functions.

Query response *mode* <newline>

mode is 0 or 1, and indicates that the automatic NPLC function is off or on, respectively. Response data type is NR1.

Example :SENS:CUR:NPLC:AUTO 0

:SENS:CURR:DC:NPLC:AUTO?

[:SENSe]:<CHARge|CURRent[:DC]|RESistance|VOLTage[:DC]>:NPLCycles:AUTO:MODE

Selects the automatic NPLC mode.

Syntax :SENSe[*c*]:<CHARge|CURRent[:DC]|RESistance|VOLTage[:DC]>:NPLCycles:AUTO:MODE *mode*

:SENSe[*c*]:<CHARge|CURRent[:DC]|RESistance|VOLTage[:DC]>:NPLCycles:AUTO:MODE?

For <CHARge|CURRent[:DC]|RESistance|VOLTage[:DC]>, specify CHARge for charge measurement, CURRent[:DC] for current measurement, RESistance for resistance measurement, or VOLTage[:DC] for voltage measurement. Specifying the measurement item is not important because the *mode* value is common for all items.

NOTE On B2981B/B2983B, only CURRent[:DC] can be specified.

Parameter

mode	Automatic NPLC mode. SHORt|MEDium(default)|LONG. Parameter data type is CPD.

NOTE SHORt|MEDium|LONG parameter of this command corresponds to Automatic aperture mode Quick|Normal|Stable in front panel operation.

Query response *mode* <newline>

mode returns the present setting, SHOR, MED, or LONG. Response data type is CRD.

Example :SENS:CURR:NPLC:AUTO:MODE MED

:SENS:CURR:NPLC:AUTO:MODE?

[:SENSe]:<CHARge|CURRent[:DC]|RESistance|VOLTage[:DC]>:RANGe:AUTO

Enables or disables the automatic ranging function of the specified measurement function.

Syntax :SENSe[*c*]:<CHARge|CURRent[:DC]|RESistance|VOLTage[:DC]>:RANGe:AUTO *mode*

:SENSe[*c*]:<CHARge|CURRent[:DC]|RESistance|VOLTage[:DC]>:RANGe:AUTO?

For <CHARge|CURRent[:DC]|RESistance|VOLTage[:DC]>, specify CHARge for the charge measurement, CURRent[:DC] for the current measurement, RESistance for the resistance measurement, or VOLTage[:DC] for the voltage measurement.

NOTE On B2981B/B2983B, only CURRent[:DC] can be specified.

Parameter ***mode*** 0|OFF|1|ON (default setting). Parameter data type is boolean.

mode = 0 or OFF disables the automatic measurement ranging. If this function is disabled, the measurement is performed by using the range set by the [:SENSe]:<CHARge|CURRent[:DC]|RESistance|VOLTage[:DC]>:RANGe[:UPPer] command.

mode = 1 or ON enables the automatic measurement ranging. If this function is enabled, the channel automatically sets the range which provides the best resolution to perform the measurement.

If a range is manually selected, the automatic ranging is disabled.

Query response *mode* <newline>

mode is 0 or 1 that indicates the automatic measurement ranging off or on respectively. Response data type is NR1.

Example :SENS:CURR:RANG:AUTO 0

:SENS:CURR:DC:RANG:AUTO?

[:SENSe]:<CHARge|CURRent[:DC]|RESistance|VOLTage[:DC]>:RANGe[:UPPer]

Specifies the expected measurement value and sets the measurement range which provides the best resolution to measure the specified value.

Syntax :SENSe[*c*]:<CHARge|CURRent[:DC]|RESistance|VOLTage[:DC]>:RANGe:UPPer *range*

:SENSe[*c*]:<CHARge|CURRent[:DC]|RESistance|VOLTage[:DC]>:RANGe:UPPer? [*range*]

For <CHARge|CURRent[:DC]|RESistance|VOLTage[:DC]>, specify CHARge for the charge measurement, CURRent[:DC] for the current measurement, RESistance for the resistance measurement, or VOLTage[:DC] for the voltage measurement.

NOTE On B2981B/B2983B, only CURRent[:DC] can be specified.

Parameter

range *value*|UP|DOWN|MINimum|MAXimum|DEFault. Parameter data type is NRf+. Query does not support *range* = *value*, UP, and DOWN.

value for current measurement: 2 pA to 20 mA. See Table 2-3.

value for charge measurement: 2 nC to 2 μC. See Table 2-4.

value for voltage measurement: 2 V to 20 V. See Table 2-5.

value for resistance measurement: 1 MΩ to 1 PΩ. See Table 2-6. This is available for the resistance measurements set to the AUTO mode which is selected by the :SENSe:RESistance:MODE command.

range = UP sets the next higher measurement range.

range = DOWN sets the next lower measurement range.

Query response *range* <newline>

range returns the present setting of the measurement range. If a parameter is specified, *range* returns the value assigned to DEF, MIN, or MAX. Response data type is NR3.

Example :SENS:CURR:RANG:UPP 1

:SENS:CURR:DC:RANG:UPP?

[:SENSe]:<CHARge|CURRent[:DC]|RESistance|VOLTage[:DC]>:REFerence

Sets the *reference* value (offset value) used for each measurement data.

The reference function (null offset cancel) is enabled by the [:SENSe]:<CHARge|CURRent[:DC]|RESistance|VOLTage[:DC]>:REFerence:STATe command.

Syntax :SENSe[*c*]:<CHARge|CURRent[:DC]|RESistance|VOLTage[:DC]>:REFerence *reference*

:SENSe[*c*]:<CHARge|CURRent[:DC]|RESistance|VOLTage[:DC]>:REFerence? [*reference*]

For <CHARge|CURRent[:DC]|RESistance|VOLTage[:DC]>, specify CHARge for the charge measurement, CURRent[:DC] for the current measurement, RESistance for the resistance measurement, or VOLTage[:DC] for the voltage measurement.

NOTE On B2981B/B2983B, only CURRent[:DC] can be specified.

Parameter

reference	*value* (–9.999999E+20 to +9.999999E+20)|MINimum|MAXimum|DEFault (default is 0.0). Parameter data type is NRf+. Query does not support *reference* = *value*.

Query response *reference* <newline>

reference returns the present setting of the reference value (offset value). If a parameter is specified, *reference* returns the value assigned to DEF, MIN, or MAX. Response data type is NR3.

Example :SENS:CURR:REF 1E–6

:SENS:CURR:REF?

[:SENSe]:<CHARge|CURRent[:DC]|RESistance|VOLTage[:DC]>:REFerence:ACQuire

Automatically sets the *reference* value (offset value) used for each measurement data.

Syntax :SENSe[*c*]:<CHARge|CURRent[:DC]|RESistance|VOLTage[:DC]>:REFerence:ACQuire

For <CHARge|CURRent[:DC]|RESistance|VOLTage[:DC]>, specify CHARge for the charge measurement, CURRent[:DC] for the current measurement, RESistance for the resistance measurement, or VOLTage[:DC] for the voltage measurement.

NOTE On B2981B/B2983B, only CURRent[:DC] can be specified.

Example :SENS:CURR:REF:ACQ

[:SENSe]:<CHARge|CURRent[:DC]|RESistance|VOLTage[:DC]>:REFerence:STATe

Enables or disables the reference function (null offset cancel) used for each measurement data.

The reference value (offset value) is set by the [:SENSe]:<CHARge|CURRent[:DC]|RESistance|VOLTage[:DC]>:REFerence:ACQuire command.

Syntax :SENSe[*c*]:<CHARge|CURRent[:DC]|RESistance|VOLTage[:DC]>:REFerence:STATe *mode*

:SENSe[*c*]:<CHARge|CURRent[:DC]|RESistance|VOLTage[:DC]>:REFerence:STATe?

For <CHARge|CURRent[:DC]|RESistance|VOLTage[:DC]>, specify CHARge for the charge measurement, CURRent[:DC] for the current measurement, RESistance for the resistance measurement, or VOLTage[:DC] for the voltage measurement.

NOTE On B2981B/B2983B, only CURRent[:DC] can be specified.

Parameter | ***mode*** | 1|ON|0|OFF (default). Parameter data type is boolean.

mode = 1 or ON enables the reference function (null offset cancel).

mode = 0 or OFF disables the reference function (null offset cancel).

Query response *mode* <newline>

mode is 0 or 1, and indicates that the null offset is off or on, respectively. Response data type is NR1.

Example :CURR:REF:STAT 1

:SENS:CURR:REF:STAT?

NOTE **NULL indicator on the display**

This indicator shows the reference function (null offset cancel) state as follows.

(not lit): The function is disabled.

NULL (gray): The function is enabled. Null offset cancel was not applied.

NULL (white): The function is enabled. Null offset cancel was applied for the data. The data is the actual measurement data minus the reference value (offset value).

[:SENSe]:CURRent[:DC]:MEDian:RANK

Sets the rank of the median filter function.

Syntax :SENSe[*c*]:CURRent[:DC]:MEDian:RANK *rank*

:SENSe[*c*]:CURRent[:DC]:MEDian:RANK? [*rank*]

Parameter

rank	Median filter rank. *value* (0 to 15)| MINimum|MAXimum|DEFault (default is 0). Parameter data type is NRf+.
	Query does not support *rank* = *value*.

If the median filter is enable, the measurement value is calculated central value from the *N* measurement samples. *N* is calculated from *rank* as follows:

$N = 2 \times rank + 1$

Query response *rank* <newline>

rank returns the present rank of the median filter function. If a parameter is specified, *rank* returns the value assigned to DEF, MIN, or MAX. Response data type is NR1.

Example :SENS:CURR:MED:RANK 3

:SENS:CURR:DC:MED:RANK?

[:SENSe]:CURRent[:DC]:MEDian[:STATe]

Enables or disables the median filter function.

Syntax :SENSe[*c*]:CURRent[:DC]:MEDian[:STATe] *mode*

:SENSe[*c*]:CURRent[:DC]:MEDian[:STATe]?

Parameter *mode* 0|OFF(default)|1|ON. Parameter data type is boolean.

mode = 0 or OFF disables the median filter function.

mode = 1 or ON enables the median filter function.

NOTE When using the median filter function, the aperture time must be larger than or equal to 160 μs.

NOTE The median filter function is not effective for a spot measurement, which is executed by :MEASure? or :MEASure:<CHARge|CURRent>? command even if the median filter function is enabled by this command.

Query response *mode* <newline>

mode is 0 or 1, and indicates that the median filter is off or on, respectively. Response data type is NR1.

Example :CURR:MED 0

:SENS:CURR:DC:MED:STAT?

[:SENSe]:<CURRent[:DC]|RESistance|VOLTage[:DC]>:RANGe:AUTO:LLIMit

Specifies the lower limit for the automatic measurement ranging operation and sets the minimum measurement range which provides the best resolution to measure the specified value.

If the minimum measurement range is same as the maximum measurement range, the measurement is performed by using this range.

Syntax :SENSe[*c*]:<CURRent[:DC]|RESistance|VOLTage[:DC]>:RANGe:AUTO:LLIMit *range*

:SENSe[*c*]:<CURRent[:DC]|RESistance|VOLTage[:DC]>:RANGe:AUTO:LLIMit? [*range*]

For <CURRent[:DC]|RESistance|VOLTage[:DC]>, specify CURRent[:DC] for the current measurement, RESistance for the resistance measurement, or VOLTage[:DC] for the voltage measurement.

NOTE On B2981B/B2983B, only CURRent[:DC] can be specified.

Parameter

range	*value*|MINimum|MAXimum|DEFault. Parameter data type is NRf+. Query does not support *range* = *value*.

value for current measurement: 2 pA to 20 mA. See Table 2-3.

value for voltage measurement: 2 V to 20 V. See Table 2-5.

value for resistance measurement: 1 MΩ to 100 GΩ. See Table 2-6. This is available for the resistance measurements set to the AUTO mode which is selected by the :SENSe:RESistance:MODE command.

Query response *range* <newline>

range returns the present setting of the minimum measurement range for the auto range. If a parameter is specified, *range* returns the value assigned to DEF, MIN, or MAX. Response data type is NR3.

Example :SENS:CURR:RANG:AUTO:LLIM 2E–6

:SENS:CURR:DC:RANG:AUTO:LLIM?

[:SENSe]:<CURRent[:DC]|RESistance|VOLTage[:DC]>:RANGe:AUTO:ULIMit

Specifies the upper limit for the automatic measurement ranging operation and sets the maximum measurement range which provides the best resolution to measure the specified value.

If the minimum measurement range is same as the maximum measurement range, the measurement is performed by using this range.

Syntax :SENSe[*c*]:<CURRent[:DC]|RESistance|VOLTage[:DC]>:RANGe:AUTO:ULIMit *range*

:SENSe[*c*]:<CURRent[:DC]|RESistance|VOLTage[:DC]>:RANGe:AUTO:ULIMit? [*range*]

For <CURRent[:DC]|RESistance|VOLTage[:DC]>, specify CURRent[:DC] for the current measurement, RESistance for the resistance measurement, or VOLTage[:DC] for the voltage measurement.

NOTE On B2981B/B2983B, only CURRent[:DC] can be specified.

Parameter

range	*value*|MINimum|MAXimum| DEFault. Parameter data type is NRf+. Query does not support *range* = *value*.

value for current measurement: 2 pA to 20 mA (default is 20 mA). See Table 2-3.

value for voltage measurement: 2 V to 20 V (default is 20 V). See Table 2-5.

value for resistance measurement: 1 MΩ to 100 GΩ (default is 100 GΩ). See Table 2-6. This is available for the resistance measurements set to the AUTO mode which is selected by the :SENSe:RESistance:MODE command.

Query response *range* <newline>

range returns the present setting of the maximum measurement range for the auto range. If a parameter is specified, *range* returns the value assigned to DEF, MIN, or MAX. Response data type is NR3.

Example :SENS:CURR:RANG:AUTO:ULIM 2E–6

:SENS:CURR:DC:RANG:AUTO:ULIM?

[:SENSe]:DATA?

Returns the array data which contains all of the current measurement data, voltage measurement data, resistance measurement data, source output setting data, status data, or time data specified by the :FORMat:ELEMents:SENSe command. The data is not cleared until the :INITiate, :MEASure, :READ, or [:SENSe]:DATA:CLEar command is executed.

Syntax [:SENSe[*c*]]:DATA? [*offset*[, *size*]]

Parameter

offset	Indicates the beginning of the data received. *n*|CURRent|STARt (default). Parameter data type is NR1 or CPD. *offset* = *n* specifies the *n+1*th data. *n* is an integer, 0 to maximum (depends on the buffer state). *offset* = CURR specifies the present data position. *offset* = STAR specifies the top of the data buffer. Same as *offset* = 0.
size	Number of data to be received. 1 to maximum (depends on the buffer state). Parameter data type is NR1. If this parameter is not specified, all data from *offset* is returned.

Query response *response* <newline>

response returns the array data specified by the :FORMat:ELEMents:SENSe command. Response data type is NR3. See "Data Output Format" on page 31.

As shown in the following example, *response* may contain multiple data and elements. This example contains the current data (*ch1currN*) and time data (*ch1timeN*) of the 10 measurements by channel 1. With the ASCII data output format, each data is separated by a comma.

ch1curr1,*ch1time1*,*ch1curr2*,*ch1time2*, ... *ch1curr10*,*ch1time10*

If the measurement function is not enabled or no data exists, *response* returns +9.910000E+37 (ASCII) or NaN (IEEE-754) which indicates "not a number."

Example

```
:FORM:ELEM:SENS CURR,TIME
:SENS:DATA?
```

[:SENSe]:DATA:ACQuire

Executes a spot measurement (one-shot measurement). Measurement conditions must be set by SCPI commands or front panel operation before executing this command. Measurement item can be set to CHARge, CURRent, RESistance, or VOLTage.

Syntax [:SENSe[c]]:DATA:ACQuire

Example :SENS:DATA:ACQ

[:SENSe]:DATA:CLEar

Clears all measurement data.

Syntax [:SENSe[c]]:DATA:CLEar

Example :SENS:DATA:CLE

[:SENSe]:DATA:LATest?

Returns the latest current measurement data, voltage measurement data, resistance measurement data, source output setting data, status data, or time data specified by the :FORMat:ELEMents:SENSe command. The data is not cleared until the :INITiate, :MEASure, :READ, or [:SENSe]:DATA:CLEar command is executed.

Syntax [:SENSe]:DATA:LATest?

Query response *response* <newline>

response returns the latest data specified by the :FORMat:ELEMents:SENSe command. Response data type is NR3. See "Data Output Format" on page 31.

As shown in the following example, *response* may contain multiple data elements. This example contains the latest current data (*ch1curr10*) and time data (*ch1time10*) of the 10 measurements by channel 1. With the ASCII data output format, each data is separated by a comma.

ch1curr10,*ch1time10*

If the measurement function is not enabled or no data exists, *response* returns +9.910000E+37 (ASCII) or NaN (IEEE-754) which indicates "not a number."

Example :FORM:ELEM:SENS CURR,TIME
:SENS:DATA:LAT?

[:SENSe]:FUNCtion[:ON]

Enables the specified measurement functions.

Syntax :SENSe[*c*]:FUNCtion[:ON] *function*[, *function*[, *function*]]

:SENSe[*c*]:FUNCtion[:ON]?

Parameter *function* "CHARge"|"CURRent[:DC]"|"VOLTage[:DC]"|"RESistance" Default is "CURR" for B2981B/B2983B; "CURR", "VOLT" for B2985B/B2987B. Case insensitive. Parameter data type is SPD.

NOTE On B2981B/B2983B, only CURRent[:DC] can be specified.

function = "CHARge" selects the charge measurement function.

function = "CURRent[:DC]" selects the current measurement function.

NOTE Can't specify both CURRent and CHARge.

function = "VOLTage[:DC]" selects the voltage measurement function.

function = "RESistance" selects the resistance measurement function.

NOTE If "RESistance" is specified, "CURRent" is also specified implicitly by system.

Query response *function*[, *function*[, *function*]]<newline>

function returns "CHAR", "CURR", "VOLT", or "RES", and indicates that the currently enabled measurement function. If a function is not selected, query returns "" (null string). Response data type is SRD.

Example :SENS:FUNC "CURR"

:SENS:FUNC:ON?

[:SENSe]:RESistance:AVALue[:STATe]

NOTE This command is available on B2985B/B2987B.

Sets the resistance measurement result to absolute format or not.

Syntax [:SENSe[*c*]]:RESistance:AVALue[:STATe] mode

[:SENSe[*c*]]:RESistance:AVALue[:STATe]?

Parameter ***mode*** Absolute value, ON or OFF. 1|ON|0|OFF (default). Parameter data type is boolean.

mode = 1 or ON sets the resistance measurement result to absolute format.

mode = 0 or OFF cancels the absolute format.

Query response *response* <newline>

response returns 1 or 0, and indicates that the absolute format is on or off. Response data type is NR1.

Example :RES:AVAL 1

:SENS:RES:AVAL?

[:SENSe]:RESistance:VSControl

NOTE This command is available on B2985B/B2987B.

Selects the voltage source control mode for the resistance measurement.

Syntax [:SENSe[*c*]]:RESistance:VSControl mode

[:SENSe[*c*]]:RESistance:VSControl?

Parameter ***mode*** Voltage source control. MANual(default)|AUTO. Parameter data type is CPD.

mode = MANual sets the user-specified value as the voltage source output.

mode = AUTO sets the voltage source output corresponding to the resistance range. See Table 2-6.

Query response *response* <newline>

response returns the present setting, MAN or AUTO. Response data type is CRD.

Example :RES:VSC AUTO

:SENS:RES:VSC?

[:SENSe]:RESistance:VSELect

NOTE This command is available on B2985B/B2987B.

Selects the voltage source value for calculation of resistance measurement, Voltage Source setting value or measured value.

Syntax [:SENSe[*c*]]:RESistance:VSELect select

[:SENSe[*c*]]:RESistance:VSELect?

Parameter ***select*** Voltage select VSOurce(default)|VMEasure. Parameter data type is CPD.

select = VSOurce uses Voltage Source setting value.

select = VMEasure uses Measured source value.

Query response *response* <newline>

response returns the present setting, VSO or VME. Response data type is CRD.

Example :RES:VSEL VME

:SENS:RES:VSEL?

[:SENSe]:TOUTput:SIGNal

Selects the trigger output for the status change between the trigger layer and the transient device action. Multiple trigger output ports can be set.

Syntax [:SENSe[*c*]]:TOUTput:SIGNal *output*{,*output*}

[:SENSe[*c*]]:TOUTput:SIGNal?

Parameter ***output*** Trigger output port. EXT1 (default)|EXT2|EXT3|EXT4|EXT5|EXT6|EXT7| LAN|INT1|INT2|TOUT. Parameter data type is CPD.

output = INT1 or INT2 selects the internal bus 1 or 2, respectively.

output = LAN selects a LAN port.

output = EXT*n* selects the GPIO pin *n*, which is an output port of the Digital I/O D-sub connector on the rear panel. *n* = 1 to 7.

output = TOUT selects the BNC Trigger Out.

Query response *response* <newline>

response returns the present setting, INT1, INT2, LAN, or EXT1 through EXT7. Response data type is CRD. Multiple responses are separated by a comma.

Example :TOUT:SIGN EXT3

:SENS:TOUT:SIGN?

[:SENSe]:TOUTput[:STATe]

Enables or disables the trigger output for the status change between the trigger layer and the transient device action.

Syntax [:SENSe[*c*]]:TOUTput[:STATe] *mode*

[:SENSe[*c*]]:TOUTput[:STATe]?

Parameter ***mode*** Trigger output ON or OFF. 1|ON|0|OFF (default). Parameter data type is boolean.

mode = 1 or ON enables the trigger output.

mode = 0 or OFF disables the trigger output.

Query response *response* <newline>

response returns 1 or 0, and indicates that the trigger output is on or off, respectively. Response data type is NR1.

Example :TOUT 1

:SENS:TOUT:STAT?

[:SENSe]:VOLTage:ATTenuation

NOTE This command is available on B2985B/B2987B.

Sets the attenuation level for the voltage input.

When performing voltage measurement with the Keysight N1413A/N1414A adapter, you can set the proper attenuation level for the adapter using this command, in order to return actual voltage value.

Syntax [:SENSe[*c*]]:VOLTage:ATTenuation *att*

[:SENSe[*c*]]:VOLTage:ATTenuation?

Parameter

att	Attenuation level (coefficient), 1(default)|100. Parameter data type is NR1.

att = 1 sets the attenuation level for the voltage input to 1 (×1).

att = 100 sets the attenuation level for the voltage input to 100 (×100).

Query response *att* <newline>

att is 1 or 100, and indicates the attenuation level set for the voltage input. Response data type is NR1.

Example :VOLT:ATT 100

:SENS:VOLT:ATT?

[:SENSe]:VOLTage[:DC]:GUARd

NOTE This command is available on B2985B/B2987B.

Selects if the inner-shield of voltage input terminal connects to the Guard or Common.

Syntax [:SENSe[*c*]]:VOLTage[:DC]:GUARd *mode*

[:SENSe[*c*]]:VOLTage[:DC]:GUARd?

Parameter

mode	1|ON(default)|0|OFF. Parameter data type is boolean.

mode = 1 or ON connects to Guard.

mode = 0 or OFF connects to Common.

Query response *mode* <newline>

mode is 0 or 1, and indicates where the inner-shield of Voltage input terminal connects. Response data type is NR1.

Example :VOLT:GUAR 1

:SENS:VOLT:DC:GUAR?

[:SENSe]:WAIT:AUTO

NOTE This command is available on B2985B/B2987B.

Enables or disables the initial wait time used for calculating the measurement wait time for the specified channel. The initial wait time is automatically set by the instrument and cannot be changed. See [:SENSe]:WAIT[:STATe].

Syntax [:SENSe[*c*]]:WAIT:AUTO *mode*

[:SENSe[*c*]]:WAIT:AUTO?

Parameter ***mode*** 0|OFF|1|ON(default). Parameter data type is boolean.

mode = 1 or ON enables the initial wait time.

mode = 0 or OFF disables the initial wait time. The initial wait time is set to 0.

Query response *mode* <newline>

mode is 0 or 1, and indicates that the initial wait time is disabled or enabled, respectively. Response data type is NR1.

Example :WAIT:AUTO 0

:SENS:WAIT:AUTO?

[:SENSe]:WAIT:GAIN

NOTE This command is available on B2985B/B2987B.

Sets the gain value used for calculating the measurement wait time for the specified channel. See [:SENSe]:WAIT[:STATe].

Syntax [:SENSe[*c*]]:WAIT:GAIN *gain*

[:SENSe[*c*]]:WAIT:GAIN? [*gain*]

Parameter

gain	value (0 to 100)|MINimum|MAXimum|DEFault (default is 1). Parameter data type is NRf. Query does not support *gain* = *value*.

Query response *gain* <newline>

gain returns the present setting of the gain value. If a parameter is specified, *gain* returns the value assigned to DEF, MIN, or MAX. Response data type is NR3.

Example :WAIT:GAIN 0.5

:SENS:WAIT:GAIN?

[:SENSe]:WAIT:OFFSet

NOTE This command is available on B2985B/B2987B.

Sets the offset value used for calculating the measurement wait time for the specified channel. See [:SENSe]:WAIT[:STATe].

Syntax [:SENSe[*c*]]:WAIT:OFFSet *offset*

[:SENSe[*c*]]:WAIT:OFFSet? [*offset*]

Parameter

offset	value (0 to 1 seconds)|MINimum|MAXimum|DEFault (default is 0). Parameter data type is NRf. Query does not support *offset* = *value*.

Query response *offset* <newline>

offset returns the present setting of the offset value. If a parameter is specified, *offset* returns the value assigned to DEF, MIN, or MAX. Response data type is NR3.

Example :WAIT:OFFS 0.5

:SENS:WAIT:OFFS?

[:SENSe]:WAIT[:STATe]

NOTE This command is available on B2985B/B2987B.

Enables or disables the measurement wait time for the specified channel. The wait time is defined as the time the measurement channel cannot start measurement after the start of a DC output.

Syntax [:SENSe[*c*]]:WAIT[:STATe] *mode*

[:SENSe[*c*]]:WAIT[:STATe]?

Parameter *mode* 0|OFF|1|ON (default). Parameter data type is boolean.

mode = 0 or OFF disables the measurement wait time. The wait time is set to 0.

mode = 1 or ON enables the measurement wait time given by the following formula.

- [:SENSe]:WAIT:AUTO ON|1 condition:

  wait time = *gain* × initial wait time + *offset*

- [:SENSe]:WAIT:AUTO OFF|0 condition:

  wait time = *offset*

The initial wait time is automatically set by the instrument and cannot be changed.

gain and *offset* are set by the [:SENSe]:WAIT:GAIN and [:SENSe]:WAIT:OFFSet commands, respectively.

Query response *mode* <newline>

mode is 0 or 1, and indicates that the measurement wait time is disabled or enabled, respectively. Response data type is NR1.

Example

```
:WAIT 0
:SENS:WAIT:STAT?
```

SOURce Subsystem

NOTE SOURce subsystem commands are available on B2985B/B2987B.

For the numeric suffixes [*c*], [*i*], and [*n*], see "Numeric Suffix" on page 27.

:SOURce:ARB:COUNt

NOTE This command is available on B2985B/B2987B.

Sets the number of arbitrary waveforms for the voltage output.

Syntax :SOURce[*c*]:ARB:COUNt *count*

:SOURce[*c*]:ARB:COUNt? [*count*]

Parameter

count	Waveform count of sweep steps. *value* (1 to 100000 or 2147483647)|INFinity|MINimum| MAXimum|DEFault (default is 1). Parameter data type is NRf+. value = 2147483647 indicates infinity. Query does not support *count* = *value*.

count must be less than 100001.

Query response *count* <newline>

count returns the present setting. If a parameter is specified, *count* returns the value assigned to DEF, MIN, or MAX. Response data type is NR1.

Example :SOUR:ARB:COUN 100

:SOUR:ARB:COUN?

:SOURce:ARB:FUNCtion[:SHAPe]

NOTE This command is available on B2985B/B2987B.

Selects the shape of the arbitrary waveform output for the specified source channel.

Syntax :SOURce[*c*]:ARB:FUNCtion[:SHAPe] *shape*

:SOURce[*c*]:ARB:FUNCtion[:SHAPe]?

Parameter *mode* Shape of waveform. SQUare (default). Parameter data type is CPD.

shape = SQUare selects the square voltage waveform.

Query response *shape* <newline>

shape returns SQU. Response data type is CRD.

Example :SOUR:ARB:FUNC:SHAP SQU

:SOUR:ARB:FUNC?

:SOURce:ARB:VOLTage:SQUare:END:TIME

NOTE This command is available on B2985B/B2987B.

Sets the end time of the voltage square waveform output. Waveform period must not exceed 1000 seconds.

Syntax :SOURce[*c*]:ARB:VOLTage:SQUare:END:TIME *time*

:SOURce[*c*]:ARB:VOLTage:SQUare:END:TIME? [*time*]

Parameter *time* Square waveform end time, in seconds. *value* (0 to 1000)|MINimum|MAXimum|DEFault (default is 0). Parameter data type is NRf+. Query does not support *time* = *value*.

Query response *response* <newline>

response returns the present setting. If a parameter is specified, *response* returns the value assigned to DEF, MIN, or MAX. Response data type is NR3.

Example :SOUR:ARB:VOLT:SQU:END:TIME 0.1

:SOUR:ARB:VOLT:SQU:END:TIME?

:SOURce:ARB:VOLTage:SQUare:STARt[:LEVel]

NOTE This command is available on B2985B/B2987B.

Sets the start level of the voltage square waveform output.

Syntax :SOURce[*c*]:ARB:VOLTage:SQUare:STARt[:LEVel] *data*

:SOURce[*c*]:ARB:VOLTage:SQUare:STARt[:LEVel]? [*data*]

Parameter

data	Square waveform start level. *value* (see "Voltage Output Ranges" on page 72) |MINimum|MAXimum|DEFault (default is 0.0). Parameter data type is NRf+. Query does not support *data* = *value*.

Query response *response* <newline>

response returns the present setting. If a parameter is specified, *response* returns the value assigned to DEF, MIN, or MAX. Response data type is NR3.

Example :SOUR:ARB:VOLT:SQU:STAR 1E–3

:SOUR:ARB:VOLT:SQU:STAR?

:SOURce:ARB:VOLTage:SQUare:STARt:TIME

NOTE This command is available on B2985B/B2987B.

Sets the start time of the voltage square waveform output. Waveform period must not exceed 1000 seconds.

Syntax :SOURce[*c*]:ARB:VOLTage:SQUare:STARt:TIME *time*

:SOURce[*c*]:ARB:VOLTage:SQUare:STARt:TIME? [*time*]

Parameter

time	Square waveform start time, in seconds. *value* (1.0E–4 to 1000)| MINimum|MAXimum|DEFault (default is 1.0E–4). Parameter data type is NRf+. Query does not support *time* = *value*.

Query response *response* <newline>

response returns the present setting. If a parameter is specified, *response* returns the value assigned to DEF, MIN, or MAX. Response data type is NR3.

Example :SOUR:ARB:VOLT:SQU:STAR:TIME 0.1

:SOUR:ARB:VOLT:SQU:STAR:TIME?

:SOURce:ARB:VOLTage:SQUare:TOP[:LEVel]

NOTE This command is available on B2985B/B2987B.

Sets the top level of the voltage square waveform output.

Syntax :SOURce[*c*]:ARB:VOLTage:SQUare:TOP[:LEVel] *data*

:SOURce[*c*]:ARB:VOLTage:SQUare:TOP[:LEVel]? [*data*]

Parameter

data	Square waveform top level. *value* (see "Voltage Output Ranges" on page 72) |MINimum|MAXimum|DEFault (default is 0.0). Parameter data type is NRf+. Query does not support *data* = *value*.

Query response *response* <newline>

response returns the present setting. If a parameter is specified, *response* returns the value assigned to DEF, MIN, or MAX. Response data type is NR3.

Example :SOUR:ARB:VOLT:SQU:TOP 1E–3

:SOUR:ARB:VOLT:SQU:TOP?

:SOURce:ARB:VOLTage:SQUare:TOP:TIME

NOTE This command is available on B2985B/B2987B.

Sets the top time of the voltage square waveform output. Waveform period must not exceed 1000 seconds.

Syntax :SOURce[*c*]:ARB:VOLTage:SQUare:TOP:TIME *time*

:SOURce[*c*]:ARB:VOLTage:SQUare:TOP:TIME? [*time*]

Parameter *time* Square waveform top time, in seconds. *value* (1.0E–4 to 1000)|MINimum|MAXimum|DEFault (default is 1.0E–4). Parameter data type is NRf+. Query does not support *time* = *value*.

Query response *response* <newline>

response returns the present setting. If a parameter is specified, *response* returns the value assigned to DEF, MIN, or MAX. Response data type is NR3.

Example :SOUR:ARB:VOLT:SQU:TOP:TIME 0.1

:SOUR:ARB:VOLT:SQU:TOP:TIME?

:SOURce:DIGital:DATA

NOTE This command is available on B2985B/B2987B.

Sets the output data to the GPIO pins (digital control port) and read data from the GPIO pins.

Syntax :SOURce:DIGital:DATA *data*

:SOURce:DIGital:DATA?

Parameter *data* Output data. *value* (0 to 127)(default is 0). Parameter data type is NR1.

Query response *data* <newline>

data returns the data read from the GPIO pins. Response data type is NR1 or NDN selected by the :FORMat:DIGital command.

Example :SOUR:DIG:DATA 100

:SOUR:DIG:DATA?

:SOURce:DIGital:EXTernal:FUNCtion

NOTE This command is available on B2985B/B2987B.

Assigns the input/output function to the specified GPIO pin.

Syntax :SOURce:DIGital:EXTernal[*n*][:FUNCtion] *function*

:SOURce:DIGital:EXTernal[*n*][:FUNCtion]?

Parameter ***function*** Function. DINPut (default)|DIO|TINPut|TOUT. Parameter data type is CPD.

function = DINP assigns the digital input.

function = DIO assigns the digital I/O.

function = TINP assigns the trigger input.

function = TOUT assigns the trigger output.

Query response *function* <newline>

function returns DIO, DINP, TOUT, or TINP. Response data type is CRD.

Example :SOUR:DIG:EXT TOUT

:SOUR:DIG:EXT7:FUNC?

:SOURce:DIGital:EXTernal:POLarity

NOTE This command is available on B2985B/B2987B.

Sets the polarity of the input/output function for the specified GPIO pin. The input/output function is set by the :SOURce:DIGital:EXTernal:FUNCtion command

Syntax :SOURce:DIGital:EXTernal[*n*]:POLarity *polarity*

:SOURce:DIGital:EXTernal[*n*]:POLarity?

Parameter ***polarity*** Polarity of the input/output function. NEG (default)|POS . Parameter data type is CPD.

polarity = POS sets positive polarity.

polarity = NEG sets negative polarity.

Query response *polarity* <newline>

polarity returns POS or NEG. Response data type is CRD.

Example :SOUR:DIG:EXT:POL NEG

:SOUR:DIG:EXT7:POL?

:SOURce:DIGital:EXTernal:TOUTput[:EDGE]:POSition

NOTE This command is available on B2985B/B2987B.

Selects the trigger output timing for the specified GPIO pin.

Syntax :SOURce:DIGital:EXTernal[*n*]:TOUTput[:EDGE]:POSition *position*

:SOURce:DIGital:EXTernal[*n*]:TOUTput[:EDGE]:POSition?

Parameter ***position*** Output trigger timing. BEFore|AFTer|BOTH (default). Parameter data type is CPD.

type = BEFore enables trigger output at the beginning of arm, trigger, and device actions (transient or acquire).

type = AFTer enables trigger output at the end of arm, trigger, and device actions (transient or acquire).

type = BOTH enables trigger output at both beginning and end of arm, trigger, and device actions (transient or acquire).

Query response *response* <newline>

response returns the present setting of output trigger timing, BEF, AFT or BOTH. Response data type is CRD.

Example :SOUR:DIG:EXT:TOUT:POS BEF

:SOUR:DIG:EXT2:TOUT:POS?

:SOURce:DIGital:EXTernal:TOUTput[:EDGE]:WIDTh

NOTE This command is available on B2985B/B2987B.

Sets the pulse width of the output trigger for the specified GPIO pin.

Syntax :SOURce:DIGital:EXTernal[*n*]:TOUTput[:EDGE]:WIDTh *width*

:SOURce:DIGital:EXTernal[*n*]:TOUTput[:EDGE]:WIDTh? [*width*]

Parameter *width* Pulse width. *value* (1E–5 to 1E–2, in seconds)| MINimum|MAXimum|DEFault (default is 0.1 ms). Parameter data type is NRf+. Query does not support *width* = *value*.

Query response *width* <newline>

width returns the present setting. If a parameter is specified, *width* returns the value assigned to DEF, MIN, or MAX. Response data type is NR3.

Example :SOUR:DIG:EXT:TOUT:WIDT 1E–5

:SOUR:DIG:EXT7:TOUT:WIDT?

:SOURce:DIGital:EXTernal:TOUTput:TYPE

NOTE This command is available on B2985B/B2987B.

Selects the output trigger type for the specified GPIO pin.

Syntax :SOURce:DIGital:EXTernal[*n*]:TOUTput:TYPE *type*

:SOURce:DIGital:EXTernal[*n*]:TOUTput:TYPE?

Parameter *type* Trigger type. EDGE (default)|LEVel. Parameter data type is CPD.

type = EDGE selects the Edge trigger.

type = LEVel selects the Level trigger.

Query response *response* <newline>

response returns the present setting of trigger type, EDGE or LEV. Response data type is CRD.

Example :SOUR:DIG:EXT:TOUT:TYPE LEV

:SOUR:DIG:EXT7:TOUT:TYPE?

:SOURce:DIGital:INTernal:TOUTput[:EDGE]:POSition

NOTE This command is available on B2985B/B2987B.

Selects the trigger output timing for the internal trigger line 1 or 2.

Syntax :SOURce:DIGital:INTernal[*i*]:TOUTput[:EDGE]:POSition *position*

:SOURce:DIGital:INTernal[*i*]:TOUTput[:EDGE]:POSition?

Parameter ***position*** Output trigger timing. BEFore|AFTer|BOTH (default). Parameter data type is CPD.

type = BEFore enables trigger output at the beginning of arm, trigger, and device actions (transient or acquire).

type = AFTer enables trigger output at the end of arm, trigger, and device actions (transient or acquire).

type = BOTH enables trigger output at both beginning and end of arm, trigger, and device actions (transient or acquire).

Query response *response* <newline>

response returns the present setting of output trigger timing, BEF, AFT or BOTH. Response data type is CRD.

Example :SOUR:DIG:INT2:TOUT:POS BEF

:SOUR:DIG:INT2:TOUT:POS?

:SOURce:FUNCtion:MODE

NOTE This command is available on B2985B/B2987B.

Selects the source output mode of the specified channel.

Syntax :SOURce[*c*]:FUNCtion:MODE *mode*

:SOURce[*c*]:FUNCtion:MODE?

Parameter ***mode*** Source output mode. VOLTage(default). Parameter data type is CPD.

mode = VOLT sets the specified channel to the voltage source.

Query response *mode* <newline>

mode returns VOLT. Response data type is CRD.

Example :SOUR:FUNC:MODE VOLT

:SOUR:FUNC:MODE?

:SOURce:FUNCtion:TRIGgered:CONTinuous

NOTE This command is available on B2985B/B2987B.

Enables or disables continuous trigger output for the specified channel.

Syntax :SOURce[*c*]:FUNCtion:TRIGgered:CONTinuous *mode*

:SOURce[*c*]:FUNCtion:TRIGgered:CONTinuous?

Parameter ***mode*** 0|OFF (default)|1|ON. Parameter data type is boolean.

mode = 1 or ON enables continuous trigger output. The instrument keeps the output level and range setting even after it changes status from busy to idle. The last output setting is saved as the immediate output setting.

mode = 0 or OFF disables continuous trigger output. The instrument changes the output level and range setting to the previous setting immediately when it changes status from busy to idle. The previous setting must be set by the :SOURce:VOLTage[:LEVel][:IMMediate][:AMPLitude] command and the range setup command.

Query response *mode* <newline>

mode returns 0 or 1, and indicates that continuous trigger is off or on, respectively. Response data type is NR1.

Example :SOUR:FUNC:TRIG:CONT 0

:SOUR:FUNC:TRIG:CONT?

:SOURce:LIST:VOLTage

NOTE This command is available on B2985B/B2987B.

Sets the source output voltage data.

Syntax :SOURce[*c*]:LIST:VOLTage *list*

:SOURce[*c*]:LIST:VOLTage?

Parameter ***list*** List of the output voltage data. Default is 0. Parameter data type is NRf.

Maximum of 100000 data can be set to *list*. Each data must be separated by a comma, for example: *list = 0.1,0.2,0.3*. For effective values of the output voltage data, see "Voltage Output Ranges" on page 72.

Query response *list* <newline>

list returns the present setting of the list. Multiple data is separated by a comma. Response data type is NR3.

Example :SOUR:LIST:VOLT 0.1,0.2,0.3

:SOUR:LIST:VOLT?

:SOURce:LIST:VOLTage:APPend

NOTE This command is available on B2985B/B2987B.

Adds the source output voltage data to the end of the list set by the :SOURce:LIST:VOLTage command, to which some data might be appended to by this command. Total number of data in the list must be ≤ 100000.

Syntax :SOURce[*c*]:LIST:VOLTage:APPend *append_list*

Parameter ***append_list*** List of the output voltage data. Parameter data type is NRf+.

Multiple data can be set to *append_list*. Each data must be separated by a comma, for example: *append_list = 1.1,1.2,1.3*. For effective values of the output voltage data, see "Voltage Output Ranges" on page 72.

Example :SOUR:LIST:VOLT:APP 1.1,1.2,1.3

:SOURce:LIST:VOLTage:POINts?

NOTE This command is available on B2985B/B2987B.

Returns the number of data in the list set by the :SOURce:LIST:VOLTage command, to which some data might be appended to by the :SOURce:LIST:VOLTage:APPend command.

Syntax :SOURce[*c*]:LIST:VOLTage:POINts?

Query response *number_of_data* <newline>

number_of_data returns the number of data in the list. Response data type is NR1.

Example :SOUR:LIST:VOLT:POIN?

:SOURce:LIST:VOLTage:STARt

NOTE This command is available on B2985B/B2987B.

Specifies the list sweep start point by using the index of the list.

Syntax :SOURce[*c*]:LIST:VOLTage:STARt *start*

:SOURce[*c*]:LIST:VOLTage:STARt?

Parameter

start	Index of the list. 1 to 100000. Default is 1. Parameter data type is NR1. *start* = 1 indicates the first data in the list (top of the list). *start* = 0 or the value greater than 100000 causes an error.

Query response *start* <newline>

start returns the present setting of the list sweep start point. Response data type is NR1.

Example :SOUR:LIST:VOLT:STAR 10

:SOUR:LIST:VOLT:STAR?

:SOURce:SWEep:DIRection

NOTE This command is available on B2985B/B2987B.

Sets the sweep direction, UP or DOWN, for the specified channel.

Syntax :SOURce[*c*]:SWEep:DIRection *direction*

:SOURce[*c*]:SWEep:DIRection?

Parameter ***direction*** Sweep direction. DOWN|UP (default). Parameter data type is CPD.

direction = UP sets the sweep direction from start value to stop value. The sweep measurement is performed from the *start* value to the *stop* value given by the following formula, even if the specified stop value does not satisfy it.

stop = *start* + *step* × (*points* –1)

direction = DOWN sets the sweep direction from stop value to start value. The sweep measurement is performed from the *stop* value to the *start* value given by the following formula, even if the specified start value does not satisfy it.

start = *stop* – *step* × (*points* –1)

Query response *direction* <newline>

direction returns the present setting of the sweep direction, UP or DOWN. Response data type is CRD.

Example :SOUR:SWE:DIR DOWN

:SOUR:SWE:DIR?

:SOURce:SWEep:POINts

NOTE This command is available on B2985B/B2987B.

Sets the number of sweep steps for the specified channel. This command setting is effective for both current sweep and voltage sweep.

Syntax :SOURce[*c*]:SWEep:POINts *points*

:SOURce[*c*]:SWEep:POINts? MINimum| MAXimum|DEFault

Parameter ***points*** Number of sweep steps. *value* (1 to 100000)|MINimum| MAXimum|DEFault (default is 1). Parameter data type is NRf+.

The points value can be expressed by the following formula, using the step value set by the :SOURce:VOLTage:STEP command and the span value set by the :SOURce:VOLTage:<CENTer|SPAN> command.

points = *span/step* + 1 (where *step* is not 0)

points = 1 sets *step* = 0.

If *points* is changed, *span* works as a constant and *step* is changed. If *step* is changed, *span* works as a constant and *points* is changed. If *span* is changed, *points* works as a constant and *step* is changed.

The calculated points value is rounded down to an integer.

The sweep measurement is performed from the *start* value to the *stop* value given by the following formula, even if the specified stop value does not satisfy it.

stop = *start* + *step* × (*points* –1)

Query response *points* <newline>

points returns the value assigned to DEF, MIN, or MAX. Response data type is NR1.

Example :SOUR:SWE:POIN 51

:SOUR:SWE:POIN? MAX

:SOURce:SWEep:RANGing

NOTE This command is available on B2985B/B2987B.

Selects the output ranging mode of the sweep output for the specified channel.

Syntax :SOURce[*c*]:SWEep:RANGing *mode*

:SOURce[*c*]:SWEep:RANGing?

Parameter

mode	Ranging mode. BEST (default)|FIXed. Parameter data type is CPD.

If *mode* = BEST is set, the channel automatically sets the range which covers the whole sweep output level for the linear sweep.

If *mode* = FIX is set, the channel uses only the range effective when starting the sweep. Range change is not performed while the sweep output is applied.

Query response *mode* <newline>

mode returns the present setting of the output ranging mode, BEST or FIX. Response data type is CRD.

Example :SOUR:SWE:RANG BEST

:SOUR:SWE:RANG?

:SOURce:SWEep:SPACing

NOTE This command is available on B2985B/B2987B.

Selects the scale of the sweep output for the specified channel.

Syntax :SOURce[*c*]:SWEep:SPACing *mode*

:SOURce[*c*]:SWEep:SPACing?

Parameter ***mode*** Sweep scale. LINear (default). Parameter data type is CPD.

mode = LIN selects the linear scale sweep output.

Query response *mode* <newline>

mode returns the present setting of the scale, LIN. Response data type is CRD.

Example :SOUR:SWE:SPAC LIN

:SOUR:SWE:SPAC?

:SOURce:SWEep:STAir

NOTE This command is available on B2985B/B2987B.

Sets the sweep mode for the specified channel.

Syntax :SOURce[*c*]:SWEep:STAir *mode*

:SOURce[*c*]:SWEep:STAir?

Parameter ***mode*** Sweep mode. SINGle (default)|DOUBle. Parameter data type is CPD.

mode = SINGle sets the sweep mode to single sweep.

mode = DOUBle sets the sweep mode to double sweep. Double sweep performs the sweep from start to stop to start.

Query response *mode* <newline>

mode returns SING or DOUB, and indicates that the sweep mode is single or double, respectively. Response data type is CRD.

Example :SOUR:SWE:STA DOUB

:SOUR:SWE:STA?

:SOURce:TOUTput:SIGNal

NOTE This command is available on B2985B/B2987B.

Selects the trigger output for the status change between the trigger layer and the transient device action. Multiple trigger output ports can be set.

Syntax :SOURce[*c*]:TOUTput:SIGNal *output*{,*output*}

:SOURce[*c*]:TOUTput:SIGNal?

Parameter ***output*** Trigger output port. EXT1 (default)|EXT2|EXT3|EXT4|EXT5|EXT6|EXT7| LAN|INT1|INT2|TOUT. Parameter data type is CPD.

output = INT1 or INT2 selects the internal bus 1 or 2, respectively.

output = LAN selects a LAN port.

output = EXT*n* selects the GPIO pin *n*, which is an output port of the Digital I/O D-sub connector on the rear panel. *n*=1 to 7.

output = TOUT selects the BNC Trigger Out.

Query response *response* <newline>

response returns the present setting, INT1, INT2, LAN, or EXT1 through EXT7. Response data type is CRD. Multiple responses are separated by a comma.

Example :SOUR:TOUT:SIGN EXT3

:SOUR:TOUT:SIGN?

:SOURce:TOUTput[:STATe]

NOTE This command is available on B2985B/B2987B.

Enables or disables the trigger output for the status change between the trigger layer and the transient device action.

Syntax :SOURce[*c*]:TOUTput[:STATe] *mode*

:SOURce[*c*]:TOUTput[:STATe]?

Parameter *mode* Trigger output ON or OFF. 1|ON|0|OFF (default). Parameter data type is boolean.

mode = 1 or ON enables the trigger output.

mode = 0 or OFF disables the trigger output.

Query response *response* <newline>

response returns 1 or 0, and indicates that the trigger output is on or off, respectively. Response data type is NR1.

Example :SOUR:TOUT 1

:SOUR:TOUT:STAT?

:SOURce:VOLTage:<CENTer|SPAN>

NOTE This command is available on B2985B/B2987B.

Sets the center or span value of the voltage sweep output.

Syntax :SOURce[*c*]:VOLTage:<CENTer|SPAN> *data*

:SOURce[*c*]:VOLTage:<CENTer|SPAN>? [*data*]

For <CENTer|SPAN>, specify CENTer for the sweep center value, or SPAN for the sweep span value.

Parameter | *data* | Sweep center or span value. *value* (see "Voltage Output Ranges" on page 72) |MINimum|MAXimum|DEFault (default is 0.0). Parameter data type is NRf+. Query does not support *data* = *value*.

The center and span values can be expressed by the following formula, using the start and stop values set by the :SOURce:VOLTage:<STARt|STOP> command. So the last command setting is effective for these sweep parameters.

center = (*start* + *stop*)/2

span = *stop* – *start*

Query response *data* <newline>

data returns the present setting. If a parameter is specified, *data* returns the value assigned to DEF, MIN, or MAX. Response data type is NR3.

Example

```
:SOUR:VOLT:CENT 1E–3
:SOUR:VOLT:SPAN?
```

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

NOTE This command is available on B2985B/B2987B.

Changes the voltage output level immediately.

Syntax

```
:SOURce[c]:VOLTage[:LEVel][:IMMediate][:AMPLitude] level
:SOURce[c]:VOLTage[:LEVel][:IMMediate][:AMPLitude]? [level]
```

Parameter | ***level*** | Voltage output level. *value* (see "Voltage Output Ranges" on page 72) |MINimum|MAXimum|DEFault (default is 0). Parameter data type is NRf+. Query does not support *level* = *value*.

Query response *level* <newline>

level returns the present setting. If a parameter is specified, *level* returns the value assigned to DEF, MIN, or MAX. Response data type is NR3.

Example

```
:SOUR:VOLT 3
```

:SOUR:VOLT:LEV:IMM:AMPL?

:SOURce:VOLTage[:LEVel]:TRIGgered[:AMPLitude]

NOTE This command is available on B2985B/B2987B.

Changes the voltage output level immediately by receiving a trigger from the trigger source set by the :TRIGger<:ACQuire|:TRANsient|[:ALL]>:SOURce[:SIGNal] command.

For the output level after the trigger status is returned to idle, see ":SOURce:FUNCtion:TRIGgered:CONTinuous" on page 281

Syntax :SOURce[*c*]:VOLTage[:LEVel]:TRIGgered[:AMPLitude] *level*

:SOURce[*c*]:VOLTage[:LEVel]:TRIGgered[:AMPLitude]? [*level*]

Parameter

level	Voltage output level. *value* (see "Voltage Output Ranges" on page 72) |MINimum|MAXimum|DEFault (default is 0). Parameter data type is NRf+. Query does not support *level* = *value*.

Query response *level* <newline>

level returns the present setting. If a parameter is specified, *level* returns the value assigned to DEF, MIN, or MAX. Response data type is NR3.

Example :SOUR:VOLT:TRIG 3

:SOUR:VOLT:LEV:TRIG:AMPL?

:SOURce:VOLTage:MODE

NOTE This command is available on B2985B/B2987B.

Selects the source mode, arbitrary waveform, fixed, sweep, or list sweep, of the specified source channel.

Syntax :SOURce[*c*]:VOLTage:MODE *mode*

:SOURce[*c*]:VOLTage:MODE?

Parameter *mode* Source mode. ARB|FIXed (default)|SWEep|LIST. Parameter data type is CPD.

mode = ARB sets the voltage arbitrary waveform output.

mode = FIX sets the constant voltage output.

mode = SWEep sets the voltage sweep output.

mode = LIST sets the user-specified voltage list sweep output.

Query response *mode* <newline>

mode returns ARB, FIX, SWE, or LIST. Response data type is CRD.

Example :SOUR:VOLT:MODE ARB

:SOUR:VOLT:MODE?

:SOURce:VOLTage:POINts

NOTE This command is available on B2985B/B2987B.

Sets the number of sweep steps for the voltage sweep output.

Syntax :SOURce[*c*]:VOLTage:POINts *points*

:SOURce[*c*]:VOLTage:POINts? [*points*]

Parameter *points* Number of sweep steps. *value* (1 to 100000)|MINimum|MAXimum|DEFault (default is 1). Parameter data type is NRf+. Query does not support *points* = *value*.

The points value can be expressed by the following formula, using the step value set by the :SOURce:VOLTage:STEP command and the span value set by the :SOURce:VOLTage:<CENTer|SPAN> command.

points = *span*/*step* + 1 (where *step* is not 0)

points = 1 sets *step* = 0.

If *points* is changed, *span* works as a constant and *step* is changed. If *step* is changed, *span* works as a constant and *points* is changed. If *span* is changed, *points* works as a constant and *step* is changed.

The calculated points value is rounded down to an integer.

The sweep measurement is performed from the *start* value to the *stop* value given by the following formula, even if the specified stop value does not satisfy it.

stop = *start* + *step* × (*points* –1)

Query response *points* <newline>

points returns the present setting. If a parameter is specified, *points* returns the value assigned to DEF, MIN, or MAX. Response data type is NR1.

Example :SOUR:VOLT:POIN 51

:SOUR:VOLT:POIN?

:SOURce:VOLTage:RANGe

NOTE This command is available on B2985B/B2987B.

Sets the voltage output range. This command is effective when the automatic ranging function is off.

Syntax :SOURce[*c*]:VOLTage:RANGe *range*

:SOURce[*c*]:VOLTage:RANGe?

Parameter ***range*** *value* (see "Voltage Output Ranges" on page 72) |MINimum| MAXimum|DEFault (default is 20 V). Parameter data type is NRf+.

value for voltage output: See Table 2-9.

Query response *range* <newline>

range returns the present setting. Response data type is NR3.

Example :SOUR:VOLT:RANG 1E–6

:SOUR:VOLT:RANG?

:SOURce:VOLTage:RLIMit:STATe

NOTE This command is available on B2985B/B2987B.

Selects if 20 MW current-limiting resistor connects to HI of Voltage Output in series or not.

Syntax :SOURce[*c*]:VOLTage:RLIMit:STATe *mode*

:SOURce[*c*]:VOLTage:RLIMit:STATe?

Parameter ***mode*** Resistive limit ON or OFF. 1|ON|0|OFF (default). Parameter data type is boolean.

mode = 1 or ON connects current-limiting resistor.

mode = 0 or OFF does not connect current-limiting resistor.

Query response *response* <newline>

response returns 1 or 0, and indicates that the current-limiting resistor is used or not, respectively. Response data type is NR1.

Example :SOUR:VOLT:RLIM:STAT 1

:SOUR:VOLT:RLIM:STAT?

:SOURce:VOLTage:<STARt|STOP>

NOTE This command is available on B2985B/B2987B.

Sets the start or stop value for the voltage sweep output.

Syntax :SOURce[*c*]:VOLTage:<STARt|STOP> *data*

:SOURce[*c*]:VOLTage:<STARt|STOP>? [*data*]

For <STARt|STOP>, specify STARt for the sweep start value, or STOP for the sweep stop value.

Parameter ***data*** Sweep start or stop value. *value* (see "Voltage Output Ranges" on page 72) |MINimum|MAXimum|DEFault (default is 0.0). Parameter data type is NRf+. Query does not support *data* = *value*.

The start and stop values can be expressed by the following formula, using the center and span values set by the :SOURce:VOLTage:<CENTer|SPAN> command. So the last command setting is effective for these sweep parameters.

start = *center* – *span*/2

stop = *center* + *span*/2

Query response *data* <newline>

data returns the present setting. If a parameter is specified, *data* returns the value assigned to DEF, MIN, or MAX. Response data type is NR3.

Example :SOUR:VOLT:STOP 10

:SOUR:VOLT:STAR?

:SOURce:VOLTage:STEP

NOTE This command is available on B2985B/B2987B.

Sets the sweep step value of the voltage sweep output.

Syntax :SOURce[*c*]:VOLTage:STEP *step*

:SOURce[*c*]:VOLTage:STEP? [*step*]

Parameter

step	Sweep step value. *value* (see "Voltage Output Ranges" on page 72) |MINimum|MAXimum|DEFault (default is 0). Parameter data type is NRf+. Query does not support *step* = *value*.

The step value can be expressed by the following formula, using the points value set by the :SOURce:VOLTage:POINts command and the span value set by the :SOURce:VOLTage:<CENTer|SPAN> command.

step = *span*/(*points* – 1) (where *points* is not 1)

points = 1 sets *step* = 0.

If *points* is changed, *span* works as a constant and *step* is changed. If *step* is changed, *span* works as a constant and *points* is changed. If *span* is changed, *points* works as a constant and *step* is changed.

The calculated points value is rounded down to an integer.

The sweep measurement is performed from the *start* value to the *stop* value given by the following formula, even if the specified stop value does not satisfy it.

stop = *start* + *step* × (*points* –1)

Polarity of *step* and *span* must be the same. Different polarity causes an error.

Query response *step* <newline>

step returns the present setting. If a parameter is specified, *step* returns the value assigned to DEF, MIN, or MAX. Response data type is NR3.

Example :SOUR:VOLT:STEP 0.5

:SOUR:VOLT:STEP?

:SOURce:WAIT:AUTO

NOTE This command is available on B2985B/B2987B.

Enables or disables the initial wait time used for calculating the source wait time for the specified channel. The initial wait time is automatically set by the instrument and cannot be changed. See ::SOURce:WAIT[:STATe].

Syntax :SOURce[*c*]:WAIT:AUTO *mode*

:SOURce[*c*]:WAIT:AUTO?

Parameter ***mode*** 0|OFF|1|ON(default). Parameter data type is boolean.

mode = 1 or ON enables the initial wait time.

mode = 0 or OFF disables the initial wait time. The initial wait time is set to 0.

Query response *mode* <newline>

mode is 0 or 1, and indicates that the initial wait time is disabled or enabled, respectively. Response data type is NR1.

Example :SOUR:WAIT:AUTO 0

:SOUR:WAIT:AUTO?

:SOURce:WAIT:GAIN

NOTE This command is available on B2985B/B2987B.

Sets the gain value used for calculating the source wait time for the specified channel. See :SOURce:WAIT[:STATe].

Syntax :SOURce[*c*]:WAIT:GAIN *gain*

:SOURce[*c*]:WAIT:GAIN? [*gain*]

Parameter ***gain*** value (0 to 100)|MINimum|MAXimum|DEFault (default is 1). Parameter data type is NRf. Query does not support *gain = value*.

Query response *gain* <newline>

gain returns the present setting of the gain value. If a parameter is specified, *gain* returns the value assigned to DEF, MIN, or MAX. Response data type is NR3.

Example :SOUR:WAIT:GAIN 0.5

:SOUR:WAIT:GAIN?

:SOURce:WAIT:OFFSet

NOTE This command is available on B2985B/B2987B.

Sets the offset value used for calculating the source wait time for the specified channel. See :SOURce:WAIT[:STATe].

Syntax :SOURce[*c*]:WAIT:OFFSet *offset*

:SOURce[*c*]:WAIT:OFFSet? [*offset*]

Parameter ***offset*** value (0 to 1 seconds)|MINimum|MAXimum|DEFault (default is 0). Parameter data type is NRf. Query does not support *offset = value*.

Query response *offset* <newline>

offset returns the present setting of the offset value. If a parameter is specified, *offset* returns the value assigned to DEF, MIN, or MAX. Response data type is NR3.

Example :SOUR:WAIT:OFFS 0.5

:SOUR:WAIT:OFFS?

:SOURce:WAIT[:STATe]

NOTE This command is available on B2985B/B2987B.

Enables or disables the source wait time for the specified channel. The wait time is defined as the time the source channel cannot start output after the start of a DC output.

Syntax :SOURce[*c*]:WAIT[:STATe] *mode*

:SOURce[*c*]:WAIT[:STATe]?

Parameter ***mode*** 0|OFF|1|ON (default). Parameter data type is boolean.

mode = 0 or OFF disables the source wait time. The wait time is set to 0.

mode = 1 or ON enables the source wait time given by the following formula.

- :SOURce:WAIT:AUTO ON|1 condition:

  wait time = *gain* × initial wait time + *offset*

- :SOURce:WAIT:AUTO OFF|0 condition:

  wait time = *offset*

The initial wait time is automatically set by the instrument and cannot be changed.

gain and *offset* are set by the :SOURce:WAIT:GAIN and :SOURce:WAIT:GAIN commands, respectively.

Query response *mode* <newline>

mode is 0 or 1, and indicates that the measurement wait time is disabled or enabled, respectively. Response data type is NR1.

Example :SOUR:WAIT 0

STATus Subsystem

:STATus:<MEASurement|OPERation|QUEStionable>:CONDition?

Returns the value of the measurement, operation, or questionable status condition register. See Table 4-5 to 4-7 for the bit definitions. The register setting is not changed by this command.

Syntax :STATus:<MEASurement|OPERation|QUEStionable>:CONDition?

For <MEASurement|OPERation|QUEStionable>, specify MEASurement for the measurement status condition register, OPERation for the operation status condition register, or QUEStionable for the questionable status condition register.

Query response *value* <newline>

value returns the value of the specified register. It is the sum of the binary-weighted values for the set bits. Response data type is NR1 (decimal) or NDN (binary, octal, or hexadecimal) selected by the :FORMat:SREGister command.

Example :STAT:MEAS:COND?

:STAT:OPER:COND?

:STAT:QUES:COND?

Table 4-5 Questionable Status Condition Register Bit Definitions

Bit	Decimal value	Description	Definition
0 to 3		Not used	0 is returned.
4	16	Temperature Summary	Over temperature
5 to 7		Not used	0 is returned.
8	256	Calibration	Failed calibration
9	512	Self-test	Failed self-test

Bit	Decimal value	Description	Definition
10	1024	Interlock	Interlock circuit is open.
11	2048	Transition Event Lost	Lost arm or trigger transition event
12	4096	Acquire Event Lost	Lost arm or trigger acquire event
13 to 15		Not used	0 is returned.

Table 4-6 Measurement Status Condition Register Bit Definitions

Bit	Decimal value	Description	Definition
0	1	Limit Test Summary	Failed one or more limit tests.
1	2	Sample Buffer Available	Sample buffer has data.
2	4	Sample Buffer Full	Sample buffer is full.
3	8	Trace Buffer Available	Trace buffer has data.
4	16	Trace Buffer Full	Trace buffer is full.
5 to 15		Not used	0 is returned.

Table 4-7 Operation Status Condition Register Bit Definitions

Bit	Decimal value	Description	Definition
0	1	Calibration/Self-test Running	Self-calibration or Self-test is in progress.
1	2	Transition Idle	Trigger system for transition is in idle state.
2	4	Waiting for Transition Trigger	Trigger system is waiting for the transition trigger.
3	8	Waiting for Transition Arm	Trigger system is waiting for the transition arm.
4	16	Acquire Idle	Trigger system for acquire is in idle state.
5	32	Waiting for Acquire Trigger	Trigger system is waiting for the acquire trigger.
6	64	Waiting for Acquire Arm	Trigger system is waiting for the acquire arm.

Bit	Decimal value	Description	Definition
7 to 12		Not used	0 is returned.
13	8192	Instrument Locked	If a remote interface (GPIB, USB, or LAN) has a lock (see :SYSTem:LOCK:OWNer? command), this bit will be set. When a remote interface releases the lock (see :SYSTem:LOCK:NAME? command), this bit will be cleared.
14	16384	Program Running	Program is running. 0 is set during the program memory execution is stopped.
15	32768	Not used	0 is returned.

:STATus:<MEASurement|OPERation|QUEStionable>:ENABle

Sets the measurement, operation, or questionable status enable register. The enable register is a mask which allows true conditions in the event register to be reported in the summary bit.

Syntax :STATus:<MEASurement|OPERation|QUEStionable>:ENABle *mask*

:STATus:<MEASurement|OPERation|QUEStionable>:ENABle?

For <MEASurement|OPERation|QUEStionable>, specify MEASurement for the measurement status enable register, OPERation for the operation status enable register, or QUEStionable for the questionable status enable register.

Parameter *mask* Mask. 0 to 65535 (decimal). Default is 0. Parameter data type is NR1 or NDN.

mask is the sum of the binary-weighted values for the set bits.

Query response *mask* <newline>

mask returns the present setting of the specified enable register. Response data type is NR1 (decimal) or NDN (binary, octal, or hexadecimal) selected by the :FORMat:SREGister command.

Example :STAT:MEAS:ENAB 65535

:STAT:QUES:ENAB?

:STATus:<MEASurement|OPERation|QUEStionable>[:EVENt]?

Returns the value of the measurement, operation, or questionable status event register. The register setting is changed by this command.

Syntax :STATus:<MEASurement|OPERation|QUEStionable>[:EVENt]?

For <MEASurement|OPERation|QUEStionable>, specify MEASurement for the measurement status event register, OPERation for the operation status event register, or QUEStionable for the questionable status event register.

Query response *value* <newline>

value returns the present setting of the specified event register. It is the sum of the binary-weighted values for the set bits. Response data type is NR1 (decimal) or NDN (binary, octal, or hexadecimal) selected by the :FORMat:SREGister command.

Example :STAT:MEAS:EVEN?

:STAT:OPER:EVEN?

:STAT:QUES:EVEN?

:STATus:<MEASurement|OPERation|QUEStionable>:NTRansition

Sets the negative transition filter in the measurement, operation, or questionable status register. If you set a bit of the filter, a 1-to-0 transition of its register bit sets the corresponding bit of the event register.

Syntax :STATus:<MEASurement|OPERation|QUEStionable>:NTRansition *filter*

:STATus:<MEASurement|OPERation|QUEStionable>:NTRansition?

For <MEASurement|OPERation|QUEStionable>, specify MEASurement for the measurement status register, OPERation for the operation status register, or QUEStionable for the questionable status register.

Parameter

filter	Negative transition filter. 0 to 65535 (decimal). Default is 0. Parameter data type is NR1 or NDN.

filter is the sum of the binary-weighted values for the set bits.

Query response *filter* <newline>

filter returns the present setting of the negative transition filter in the specified register. Response data type is NR1 (decimal) or NDN (binary, octal, or hexadecimal) selected by the :FORMat:SREGister command.

Example :STAT:MEAS:NTR 0

:STAT:QUES:NTR?

:STATus:<MEASurement|OPERation|QUEStionable>:PTRansition

Sets the positive transition filter in the measurement, operation, or questionable status register. If you set a bit of the filter, a 0-to-1 transition of its register bit sets the corresponding bit of the event register.

Syntax :STATus:<MEASurement|OPERation|QUEStionable>:PTRansition *filter*

:STATus:<MEASurement|OPERation|QUEStionable>:PTRansition?

For <MEASurement|OPERation|QUEStionable>, specify MEASurement for the measurement status register, OPERation for the operation status register, or QUEStionable for the questionable status register.

Parameter

	filter	Positive transition filter. 0 to 65535 (decimal). Default is 32767. Parameter data type is NR1 or NDN.

filter is the sum of the binary-weighted values for the set bits.

Query response *filter* <newline>

filter returns the present setting of the positive transition filter in the specified register. Response data type is NR1 (decimal) or NDN (binary, octal, or hexadecimal) selected by the :FORMat:SREGister command.

Example :STAT:MEAS:PTR 32767

:STAT:QUES:PTR?

:STATus:PRESet

Sets all defined bits in the status system's PTR registers and clears the all bits in the NTR and Enable registers. The registers are returned to the default condition.

Syntax :STATus:PRESet

Example :STAT:PRES

SYSTem Subsystem

For the numeric suffix [*c*] and [*j*], see "Numeric Suffix" on page 27.

:SYSTem:AOUT

Selects the measurement mode of which results are applied to the analog output function.

Syntax :SYSTem:AOUT *function*

:SYSTem:AOUT?

Parameter

function	Measurement mode to be applied to the analog out function. IM(default)|QM|VM. Parameter data type is CPD. Specify IM for current measurement, QM for charge measurement, or VM for voltage measurement. On B2981B/B2983B, only IM can be specified for current measurement. If IM or QM is specified on B2985B/B2987B, the measurement results of current or charge which is presently-selected are outputted from the analog output terminal.

Query response *function* <newline>

function returns the present measurement mode setting for analog output function, IM, QM or VM. Response data type is CRD.

Example :SYST:AOUT IM

:SYST:AOUT?

:SYSTem:BATTery?

NOTE This command is available on B2983B/B2987B.

Returns a percentage of the remaining battery capacity.

Syntax :SYSTem:BATTery?

Query response *response* <newline>

response returns the percentage of the remaining battery capacity. Response data type is NR1.

Example :SYST:BATT?

:SYSTem:BATTery:CYCLes?

NOTE This command is available on B2983B/B2987B.

Returns the battery cycle count.

Syntax :SYSTem:BATTery:CYCLes?

Query response *response* <newline>

response returns the battery cycle count. Response data type is NR1.

Example :SYST:BATT:CYCL?

:SYSTem:BATTery:TEST?

NOTE This command is available on B2983B/B2987B.

Performs self-test on the battery and returns the result.

Syntax :SYSTem:BATTery:TEST?

Query response *result* <newline>

Returns the result of battery self-test. Response data type is NR1.

0: test passed
1: test failed

If battery self-test fail, a 1 is returned and an error is stored in the error queue. For a complete listing of the error messages related to self-test failures, see Chapter 5, "Error Messages."

Example :SYST:BATT:TEST?

:SYSTem:BEEPer[:IMMediate]

Generates a beep sound of the specified frequency and duration.

Syntax :SYSTem:BEEPer[:IMMediate] *frequency*, *time*

Parameter

frequency	Frequency, in Hz. 55 to 6640 Hz. Parameter data type is NRf.
time	Duration, in seconds. 0.05 to 12.75 seconds. Parameter data type is NRf+.

Example :SYST:BEEP 100,0.5

:SYSTem:BEEPer:STATe

Enables or disables the beeper. This command setting is not changed by power off or the *RST command.

Syntax :SYSTem:BEEPer:STATe *mode*

:SYSTem:BEEPer:STATe?

Parameter

mode	Beeper on or off. 0|OFF|1|ON. Parameter data type is boolean.

mode = 1 or ON enables the beeper.

mode = 0 or OFF disables the beeper.

Query response *mode* <newline>

mode returns 0 or 1, and indicates that the beeper is off or on, respectively. Response data type is NR1.

Example :SYST:BEEP:STAT 1

:SYST:BEEP:STAT?

:SYSTem:COMMunicate:ENABle

Enables or disables the remote interface GPIB, USB, or LAN, the remote service Sockets, Telnet, VXI-11, HiSLIP, or the built-in Web Interface. The setting is effective after rebooting the instrument. This command setting is not changed by power off or the *RST command.

Syntax :SYSTem:COMMunicate:ENABle *mode*, *interface*

:SYSTem:COMMunicate:ENABle? *interface*

Parameter

mode	Interface on or off. 1|ON|0|OFF. Parameter data type is boolean.
interface	Interface. GPIB|USB|LAN|SOCKets|TELNet|VXI11|HISLip| WEB. Parameter data type is CPD.

mode = 1 or ON enables the specified *interface*.

mode = 0 or OFF disables the specified *interface*.

Query response *mode* <newline>

mode returns 0 or 1, and indicates that the specified *interface* is off or on, respectively. Response data type is NR1.

Example :SYST:COMM:ENAB 0,USB

:SYST:COMM:ENAB? LAN

:SYSTem:COMMunicate:GPIB[:SELF]:ADDRess

Sets the GPIB address of the instrument. This command setting is not changed by power off or the *RST command.

Syntax :SYSTem:COMMunicate:GPIB[:SELF]:ADDRess *address*

:SYSTem:COMMunicate:GPIB[:SELF]:ADDRess?

Parameter

address	GPIB address, 0 to 30. Parameter data type is NR1.

Query response *address* <newline>

address returns the GPIB address of the instrument. Response data type is NR1.

Example :SYST:COMM:GPIB:ADDR 17

:SYST:COMM:GPIB:ADDR?

:SYSTem:COMMunicate:LAN:ADDRess

Sets the static LAN (IP) address of the instrument. The setting is enabled by the :SYSTem:COMMunicate:LAN:UPDate command. This command setting is not changed by power off or the *RST command.

Syntax :SYSTem:COMMunicate:LAN:ADDRess *address*

:SYSTem:COMMunicate:LAN:ADDRess? [CURRent|STATic]

Parameter ***address*** IP address of the instrument. It must be in the *A*.*B*.*C*.*D* format with 15 characters maximum. *A*, *B*, *C*, and *D* must be a number from 0 to 225. Parameter data type is SPD.

Query response *address* <newline>

address returns the static LAN (IP) address of the instrument. If the CURRent parameter is set, *address* returns the present setting. If the STATic parameter is set, *address* returns the reserved value for the next startup. Response data type is SRD.

Example :SYST:COMM:LAN:ADDR “192.168.100.100”

:SYST:COMM:LAN:ADDR?

:SYSTem:COMMunicate:LAN:BSTatus?

Returns the LAN boot status of the instrument.

Syntax :SYSTem:COMMunicate:LAN:BSTatus?

Query response *status* <newline>

status returns the following LAN boot status. Response data type is CRD.

LAN_AUTO_IP	The instrument booted with a local IP address.
LAN_DHCP	The instrument booted with a DHCP-assigned address.
LAN_FAULT	The instrument cannot detect a connection.
LAN_STATIC	The instrument booted with a static IP address.

Example :SYST:COMM:LAN:BST?

:SYSTem:COMMunicate:LAN:DHCP

Enables or disables the use of the Dynamic Host Configuration Protocol (DHCP). The setting is enabled by the :SYSTem:COMMunicate:LAN:UPDate command. This command setting is not changed by power off or the *RST command.

When DHCP is enabled, the instrument will try to obtain an IP address from a DHCP server. If a DHCP server finds the instrument, it will assign a dynamic IP address, subnet mask, and default gateway to the instrument. When DHCP is disabled or unavailable, the instrument will use the static IP address, subnet mask, and default gateway during power-on.

If a DHCP LAN address is not assigned by a DHCP server, a static IP address will be used after a timeout of approximately 2 minutes. For the instrument boot status, see the :SYSTem:COMMunicate:LAN:BSTatus? command.

Syntax :SYSTem:COMMunicate:LAN:DHCP *mode*

:SYSTem:COMMunicate:LAN:DHCP?

Parameter ***mode*** DHCP off or on. 0|OFF|1|ON. Parameter data type is boolean.

Query response *mode* <newline>

mode returns 0 or 1, and indicates that DHCP is off or on, respectively. Response data type is NR1.

Example :SYST:COMM:LAN:DHCP 0

:SYST:COMM:LAN:DHCP?

:SYSTem:COMMunicate:LAN:DNS

Sets the IP address of the DNS server. This command setting is not changed by power off or the *RST command.

Syntax :SYSTem:COMMunicate:LAN:DNS[*j*] *address*

:SYSTem:COMMunicate:LAN:DNS[*j*]? [CURRent|STATic]

Parameter ***address*** IP address of the DNS server. It must be in the *A*.*B*.*C*.*D* format with 15 characters maximum. *A*, *B*, *C*, and *D* must be a number from 0 to 255. Parameter data type is SPD.

Query response *address* <newline>

address returns the IP address of the DNS server. If the CURRent parameter is set, *address* returns the present setting. If the STATic parameter is set, *address* returns the reserved value for the next startup. Response data type is SRD.

Example :SYST:COMM:LAN:DNS “192.168.100.200”

:SYST:COMM:LAN:DNS2?

:SYSTem:COMMunicate:LAN:DOMain?

Returns the domain name of the network to which the instrument is connected.

Syntax :SYSTem:COMMunicate:LAN:DOMain?

Query response *domain_name* <newline>

domain_name returns the domain name of the network. Response data type is SRD.

Example :SYST:COMM:LAN:DOM?

:SYSTem:COMMunicate:LAN:<GATE|GATeway>

Sets the IP address of the default gateway. The setting is enabled by the :SYSTem:COMMunicate:LAN:UPDate command. This command setting is not changed by power off or the *RST command. For <GATE|GATeway>, specify GATE or GATeway.

Syntax :SYSTem:COMMunicate:LAN:<GATE|GATeway> *address*

:SYSTem:COMMunicate:LAN:<GATE|GATeway>? [CURRent|STATic]

Parameter ***address*** IP address of the default gateway. It must be in the *A*.*B*.*C*.*D* format with 15 characters maximum. *A*, *B*, *C*, and *D* must be a number from 0 to 225. Parameter data type is SPD.

Query response *address* <newline>

address returns the IP address of the default gateway. If the CURRent parameter is set, *address* returns the present setting. If the STATic parameter is set, *address* returns the reserved value for the next startup. Response data type is SRD.

Example :SYST:COMM:LAN:GATE “192.168.100.210”

:SYST:COMM:LAN:GATE?

:SYSTem:COMMunicate:LAN:<HNAMe|HOSTname>

Sets the host name of the instrument. The setting is enabled by the :SYSTem:COMMunicate:LAN:UPDate command. This command setting is not changed by power off or the *RST command.

Syntax :SYSTem:COMMunicate:LAN:<HNAMe|HOSTname> *hostname*

:SYSTem:COMMunicate:LAN:<HNAMe|HOSTname>? [CURRent|STATic]

Parameter ***hostname*** Host name. Up to 15 characters. Parameter data type is SPD.

Query response *hostname* <newline>

hostname returns the host name of the instrument. If the CURRent parameter is set, *hostname* returns the present setting. If the STATic parameter is set, *hostname* returns the reserved value for the next startup. Response data type is SRD.

Example :SYST:COMM:LAN:HNAM “A-B2981B-00001”

:SYST:COMM:LAN:HOST?

:SYSTem:COMMunicate:LAN:MAC?

Returns the MAC address of the instrument.

Syntax :SYSTem:COMMunicate:LAN:MAC?

Query response *mac_address* <newline>

mac_address returns the MAC address of the instrument. Response data type is SRD.

Example :SYST:COMM:LAN:MAC?

:SYSTem:COMMunicate:LAN:SMASk

Sets the static subnet mask. The setting is enabled by the :SYSTem:COMMunicate:LAN:UPDate command. This command setting is not changed by power off or the *RST command.

Syntax :SYSTem:COMMunicate:LAN:SMASk *subnet_mask*

:SYSTem:COMMunicate:LAN:SMASk? [CURRent|STATic]

Parameter ***subnet_mask*** Subnet mask. It must be in the *A.B.C.D* format with 15 characters maximum. *A*, *B*, *C*, and *D* must be a number from 0 to 255. Parameter data type is SPD.

Query response *subnet_mask* <newline>

subnet_mask returns the subnet mask. If the CURRent parameter is set, *subnet_mask* returns the present setting. If the STATic parameter is set, *subnet_mask* returns the reserved value for the next startup. Response data type is SRD.

Example :SYST:COMM:LAN:SMAS “255.255.255.0”

:SYST:COMM:LAN:SMAS?

:SYSTem:COMMunicate:LAN:TELNet:PROMpt

Sets the command prompt displayed during a Telnet session for establishing communication with the instrument. This command setting is not changed by power off or the *RST command.

The instrument uses LAN port 5024 for SCPI Telnet sessions, and 5025 for SCPI Socket sessions.

A Telnet session can typically be started as shown below from a host computer shell.

telnet *ip_address port*

Syntax :SYSTem:COMMunicate:LAN:TELNet:PROMpt *prompt*

:SYSTem:COMMunicate:LAN:TELNet:PROMpt?

Parameter ***prompt*** Command prompt. Up to 15 characters. Parameter data type is SPD.

Query response *prompt* <newline>

prompt returns the command prompt. Response data type is SRD.

Example :SYST:COMM:LAN:TELN:PROM “A-B2981B-00001> ”

:SYST:COMM:LAN:TELN:PROM?

:SYSTem:COMMunicate:LAN:TELNet:WMESsage

Sets the welcome message displayed during a Telnet session when starting communication with the instrument. This command setting is not changed by power off or the *RST command.

The instrument uses LAN port 5024 for SCPI Telnet sessions, and 5025 for SCPI Socket sessions.

Syntax :SYSTem:COMMunicate:LAN:TELNet:WMESsage *message*

:SYSTem:COMMunicate:LAN:TELNet:WMESsage?

Parameter ***message*** Welcome message. Up to 63 characters. Parameter data type is SPD.

Query response *message* <newline>

message returns the welcome message. Response data type is SRD.

Example :SYST:COMM:LAN:TELN:WMES “Welcome to A-B2981B-00001.”

:SYST:COMM:LAN:TELN:WMES?

:SYSTem:COMMunicate:LAN:UPDate

Disconnects all active LAN and Web Interface connections, updates the LAN setup, and restarts the LAN interface with the new setup. The new setup may change the IP address of the instrument.

Syntax :SYSTem:COMMunicate:LAN:UPDate

Example :SYST:COMM:LAN:UPD

:SYSTem:COMMunicate:LAN:WINS

Sets the IP address of the WINS server. This command setting is not changed by power off or the *RST command.

Syntax :SYSTem:COMMunicate:LAN:WINS[*j*] *address*

:SYSTem:COMMunicate:LAN:WINS[*j*]? [CURRent|STATic]

Parameter ***address*** IP address of the WINS server. It must be in the *A.B.C.D* format with 15 characters maximum. *A*, *B*, *C*, and *D* must be a number from 0 to 255. Parameter data type is SPD.

Query response *address* <newline>

address returns the IP address of the WINS server. If the CURRent parameter is set, *address* returns the present setting. If the STATic parameter is set, *address* returns the reserved value for the next startup. Response data type is SRD.

Example :SYST:COMM:LAN:WINS “192.168.100.150”

:SYST:COMM:LAN:WINS2?

:SYSTem:COMMunicate:<LAN|TCPip>:CONTrol?

Returns the control connection port number of the specified port.

Syntax :SYSTem:COMMunicate:<LAN|TCPip>:CONTrol?

Query response *port_number* <newline>

port_number returns the control connection port number of the specified port. Response data type is NR1.

Example :SYST:COMM:TCP:CONT?

:SYSTem:DATA:QUANtity?

Returns the number of data for the specified channel in the data buffer.

Syntax :SYSTem:DATA:QUANtity? [*chanlist*]

Parameter ***chanlist*** Channels. Parameter data type is channel list. (@1). See “Channel List Parameter” on page 27.

(@1) selects channel 1.

If this parameter is not specified, *chanlist* = (@1) is set.

Query response *response* <newline>

response returns the number of data. Response data type is NR1. They are separated by a comma.

Example :SYST:DATA:QUAN? (@1)

:SYSTem:DATE

Sets the date of the internal clock. This command setting is not changed by power off or the *RST command.

Syntax :SYSTem:DATE *year*, *month*, *day*

:SYSTem:DATE?

Parameter

year	Year. 4-digit integer. Parameter data type is NR1.
month	Month. Integer from 1 to 12. Parameter data type is NR1.
day	Day. Integer from 1 to 31. Parameter data type is NR1.

Query response *response* <newline>

response returns *year*, *month*, *day*. Each value is separated by a comma. Response data type is NR1.

Example :SYST:DATE 2021,1,1

:SYSTem:ERRor:ALL?

Reads and returns all items in the error/event queue, and clears the queue.

Syntax :SYSTem:ERRor:ALL?

Query response response <newline>

response returns *code*,message which contains the error/event code and message. Multiple responses are listed in the FIFO (first-in-first-out) order, separated by a comma. Data type of *code* is NR1 and message s is SRD.

If the queue is empty, the response is +0,“No error”.

Example :SYST:ERR:ALL?

:SYSTem:ERRor:CODE:ALL?

Reads all items in the error/event queue, returns all codes, and clears the queue.

Syntax :SYSTem:ERRor:CODE:ALL?

Query response code <newline>

code returns the error/event code. Multiple responses are listed in the FIFO (first-in-first-out) order, separated by a comma. Response data type is NR1.

If the queue is empty, the response is +0.

Example :SYST:ERR:CODE:ALL?

:SYSTem:ERRor:CODE[:NEXT]?

Reads and removes the top item in the error/event queue, and returns the top code.

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

Query response code <newline>

code returns the error/event code. Response data type is NR1.

If the queue is empty, the response is +0.

Example :SYST:ERR:CODE?

:SYSTem:ERRor:COUNt?

Returns the number of items in the error/event queue.

Syntax :SYSTem:ERRor:COUNt?

Query response response <newline>

response returns the number of items. Response data type is NR1.

If the queue is empty, the response is +0.

Example :SYST:ERR:COUN?

:SYSTem:ERRor[:NEXT]?

Reads and removes the top item in the error/event queue, and returns the top code and message.

Syntax :SYSTem:ERRor[:NEXT]?

Query response response <newline>

response returns *code*,message which contains the error/event code and message. Multiple responses are listed in the FIFO (first-in-first-out) order, separated by a comma. Data type of *code* is NR1 and message is SRD.

If the queue is empty, the response is +0,“No error”.

Example :SYST:ERR?

:SYSTem:HUMidity?

NOTE This command is available on B2985B/B2987B.

Returns the humidity measurement data from the humidity sensor.

Syntax :SYSTem:HUMidity?

Query response *response* <newline>

response returns the present humidity of the humidity sensor. If the measurement is failed, NaN is returned. Response data type is NR3.

Example :SYST:HUM?

:SYSTem:INTerlock:TRIPped?

Returns if the interlock circuit is close or open.

Syntax :SYSTem:INTerlock:TRIPped?

Query response *mode* <newline>

mode returns 0 or 1, and indicates that the interlock circuit is close or open, respectively. Response data type is NR1.

Example :SYST:INT:TRIP?

:SYSTem:LFRequency

Selects the line frequency. This command setting is not changed by power off or the *RST command.

Syntax :SYSTem:LFRequency *frequency*

:SYSTem:LFRequency?

Parameter ***frequency*** Line frequency. 50 (for 50 Hz, default)|60 (for 60 Hz). Parameter data type is NR1.

Query response *frequency* <newline>

frequency returns the present setting, 50 or 60. Response data type is NR1.

Example :SYST:LFR 60

:SYST:LFR?

:SYSTem:LFRequency:DETect?

Detects the power line frequency and returns the result.

Syntax :SYSTem:LFRequency:DETect?

Query response *response* <newline>

response returns the result of the power line frequency detection, 50 or 60. Response data type is NR1.

If the detection is failed, an error occurs and returns the setting value before command execution.

Example :SYST:LFR:DET?

:SYSTem:LFRequency:DETect:AUTO

Selects if auto-detection of power line frequency at power on is enabled or disabled. The setup by this command is not changed after power off or by the *RST command.

Syntax :SYSTem:LFRequency:DETect:AUTO *mode*

:SYSTem:LFRequency:DETect:AUTO?

Parameter ***mode*** Enables or disables the line frequency auto-detection at power on. 0|OFF|1|ON. Parameter data type is boolean.

mode = 1 or ON enables the line frequency auto-detection at power on.

mode = 0 or OFF disables the line frequency auto-detection at power on.

Query response *mode* <newline>

mode returns 0 or 1, and indicates that the line frequency auto-detection at power on is disabled or enabled, respectively. Response data type is NR1.

Example :SYST:LFR:DET:AUTO ON

:SYST:LFR:DET:AUTO?

:SYSTem:LOCK:NAME?

Returns the current I/O interface (the I/O interface in use by the querying computer).

Syntax :SYSTem:LOCK:NAME?

Query response *response* <newline>

response returns GPIB, USB, VXI11, or LAN <IP Address>, indicating the I/O interface being used by the querying computer.

Example :SYST:LOCK:NAME?

Remarks Use this command to determine the interface you are currently using. Then use the :SYSTem:LOCK:OWNer? command to determine which interface, if any, has the lock.

:SYSTem:LOCK:OWNer?

Returns the I/O interface that currently has a lock.

Syntax :SYSTem:LOCK:OWNer?

Query response *response* <newline>

response returns GPIB, USB, VXI11, or LAN <IP Address>, which indicates the I/O interface. If no interface has a lock, then NONE is returned.

Example :SYST:LOCK:OWN?

Remarks When a lock is active, Bit 13 in the Standard Operation Register will be set (see :STATus:<MEASurement|OPERation|QUEStionable>: CONDition? command). When the lock is released on all I/O interfaces, this bit will be cleared.

:SYSTem:LOCK:RELease

Decrements the lock count by one, and may release the I/O interface from which the command is executed.

Syntax :SYSTem:LOCK:RELease

Example :SYST:LOCK:REL

Remarks When a lock is active, Bit 13 in the Standard Operation Register will be set (see :STATus:<MEASurement|OPERation|QUEStionable>: CONDition? command). When the lock is released on all I/O interfaces, this bit will be cleared.

Note that for each successful lock request, a lock release is required. Two requests require two releases.

:SYSTem:LOCK:REQuest?

Requests a lock of the current I/O interface. This provides a mechanism by which you can lock the instrument's configuration or cooperatively share the instrument with other computers.

Syntax :SYSTem:LOCK:REQuest?

Query response *response* <newline>

response returns 1 if the lock request is granted, or 0 if denied.

Example :SYST:LOCK:REQ?

Remarks Lock requests can be nested, and each request increases the lock count by 1. For each request, you will need to issue a release from the same I/O interface (see :SYSTem:LOCK:RELease command).

Instrument locks are handled at the I/O interface level (GPIB, USB, LAN, etc.), and you are responsible for all coordination between threads and/or programs on that interface.

When a request is granted, only I/O sessions from the present interface will be allowed to change the state of the instrument. From other I/O interfaces, you can query the state of the instrument, but no measurement configuration changes or measurements will be allowed.

Locks from LAN sessions will be automatically released when a LAN disconnect is detected.

When a lock is granted, Bit 13 in the Standard Operation Register will be set (see :STATus:<MEASurement|OPERation|QUEStionable>: CONDition? command). In addition, the entire instrument front panel, including the Local key, will be locked down while a lock is in place ("KEYBOARD LOCKED" is displayed).

:SYSTem:PON

Specifies the power-on state.

The power-on state can be selected from the factory default reset condition (RST) and user conditions RCL0, RCL1, RCL2, RCL3, and RCL4 which can be defined by the *SAV 0, *SAV 1, *SAV 2, *SAV 3, and *SAV 4 commands, respectively.

Syntax :SYSTtem:PON *memory*

Parameter | *memory* | Power-on state, RST(default)|RCL0|RCL1|RCL2|RCL3|RCL4
Parameter data type is CPD.

Example :SYST:PON RCL0

:SYSTem:PRESet

Presets the instrument settings and the front panel display.

Syntax :SYSTem:PRESet

Example :SYST:PRESet

:SYSTem:SET

Sends or loads the instrument setup data.

Syntax :SYSTem:SET *data*

:SYSTem:SET?

Parameter *data* Instrument setup data. Parameter data type is a definite length arbitrary binary block.

Query response Response is a definite length arbitrary binary block.

:SYSTem:TEMPerature?

NOTE This command is available on B2985B/B2987B.

Returns the temperature measurement data from the currently-selected temperature sensor.

NOTE The temperature sensor is selected by :SYSTem:TEMPerature:SELect command.

NOTE The unit of temperature data is specified by :SYSTem:TEMPerature:UNIT command.

Syntax :SYSTem:TEMPerature?

Query response *response* <newline>

response returns the present temperature from the temperature sensor. If the measurement is failed, NaN is returned. Response data type is NR3.

Example :SYST:TEMP?

:SYSTem:TEMPerature:SELect

NOTE This command is available on B2985B/B2987B.

Selects the temperature sensor for temperature measurement.

Syntax :SYSTem:TEMPerature:SELect *sensor*

:SYSTem:TEMPerature:SELect?

Parameter *sensor* Temperature sensor. TC (default)|HSENsor. Parameter data type is CPD.

sensor = TC selects the thermocouple.

sensor = HSENsor selects the temperature sensor within the humidity sensor.

Query response *sensor* <newline>

sensor returns the present temperature sensor, TC or HSEN. Response data type is CRD.

Example :SYST:TEMP:SEL TC

:SYST:TEMP:SEL?

:SYSTem:TEMPerature:UNIT

NOTE This command is available on B2985B/B2987B.

Selects the unit for temperature measurement.

NOTE The system initial setting is Celsius.

Syntax :SYSTem:TEMPerature:UNIT *unit*

:SYSTem:TEMPerature:UNIT?

Parameter ***unit*** Temperature unit. C (default)|CEL|F|FAR|K. Parameter data type is CPD.

unit = C, CEL selects Celsius.

unit = F, FAR selects Fahrenheit.

unit = K selects Kelvin.

Query response *unit* <newline>

unit returns the present setting, C, F or K. Response data type is CRD.

Example :SYST:TEMP:UNIT K

:SYST:TEMP:UNIT?

:SYSTem:TIME

Sets the time of the internal clock. This command setting is not changed by power off or the *RST command.

Syntax :SYSTem:TIME *hour*, *minute*, *second*

:SYSTem:TIME?

Parameter

hour	Hour. Integer from 0 to 23. Parameter data type is NR1.
minute	Minute. Integer from 0 to 59. Parameter data type is NR1.
second	Second. Integer from 0 to 59. Parameter data type is NR1.

Query response *response* <newline>

response returns *hour*, *minute*, *second*. Each value is separated by a comma. Response data type is NR1.

Example :SYST:TIME 23,59,59

:SYSTem:TIME:TIMer:COUNt?

Returns the present count of the timer.

Syntax :SYSTem:TIME:TIMer:COUNt?

Query response *response* <newline>

response returns the present timer count. Response data type is NR3.

Example :SYST:TIME:TIM:COUN?

:SYSTem:TIME:TIMer:COUNt:RESet:AUTO

Enables or disables the automatic reset function of the timer. If this function is enabled, the timer count is reset when the initiate action occurs.

Syntax :SYSTem:TIME:TIMer:COUNt:RESet:AUTO *mode*

:SYSTem:TIME:TIMer:COUNt:RESet:AUTO?

Parameter

mode	Automatic reset function on or off. 0|OFF|1|ON (default). Parameter data type is boolean.

mode = 1 or ON enables the automatic reset function.

mode = 0 or OFF disables the automatic reset function.

Query response *mode* <newline>

mode returns 0 or 1, and indicates that the automatic reset function is off or on, respectively. Response data type is NR1.

Example :SYST:TIME:TIM:COUN:RES:AUTO 0

:SYST:TIME:TIM:COUN:RES:AUTO?

:SYSTem:TIME:TIMer:COUNt:RESet[:IMMediate]

Resets the timer count immediately.

Syntax :SYSTem:TIME:TIMer:COUNt:RESet[:IMMediate]

Example :SYST:TIME:TIM:COUN:RES

:SYSTem:<TIN|TOUT>:POLarity

Sets the polarity of the trigger input/output function for the specified BNC connector.

Syntax :SYSTem:<TIN|TOUT>:POLarity *polarity*

:SYSTem:<TIN|TOUT>:POLarity?

Parameter ***polarity*** Polarity of the trigger input/output function. NEG (default)|POS Parameter data type is CPD.

polarity = POS sets positive polarity.

polarity = NEG sets negative polarity.

Query response *polarity* <newline>

polarity returns POS or NEG. Response data type is CRD.

Example :SYST:TIN:POL NEG

:SYST:TOUT:POL?

:SYSTem:TOUT[:EDGE]:POSition

Selects the trigger output timing for the specified GPIO pin.

Syntax :SYSTem:TOUT[:EDGE]:POSition *position*

:SYSTem:TOUT[:EDGE]:POSition?

Parameter ***position*** Output trigger timing. BEFore|AFTer|BOTH (default). Parameter data type is CPD.

type = BEFore enables trigger output at the beginning of arm, trigger, and device actions (transient or acquire).

type = AFTer enables trigger output at the end of arm, trigger, and device actions (transient or acquire).

type = BOTH enables trigger output at both beginning and end of arm, trigger, and device actions (transient or acquire).

Query response *response* <newline>

response returns the present setting of output trigger timing, BEF, AFT or BOTH. Response data type is CRD.

Example :SYST:TOUT:POS BEF

:SYST:TOUT:POS?

:SYSTem:TOUT[:EDGE]:WIDTh

Sets the pulse width of the output trigger from the specified GPIO pin.

Syntax :SYSTem:TOUT[:EDGE]:WIDTh *width*

:SYSTem:TOUT[:EDGE]:WIDTh? [*width*]

Parameter ***width*** Pulse width. *value* (1E–5 to 1E–2, in seconds)| MINimum|MAXimum|DEFault (default is 0.1 ms). Parameter data type is NRf+. Query does not support *width* = *value*.

Query response *width* <newline>

width returns the present setting. If a parameter is specified, *width* returns the value assigned to DEF, MIN, or MAX. Response data type is NR3.

Example :SYST:TOUT:WIDT 1E–5

:SYST:TOUT:WIDT?

:SYSTem:TOUT:TYPE

Selects the output trigger type for the specified GPIO pin.

Syntax :SYSTem:TOUT:TYPE *type*

:SYSTem:TOUT:TYPE?

Parameter ***type*** Trigger type. EDGE (default)|LEVel. Parameter data type is CPD.

type = EDGE selects the Edge trigger.

type = LEVel selects the Level trigger.

Query response *response* <newline>

response returns the present setting of trigger type, EDGE or LEV. Response data type is CRD.

Example :SYST:TOUT:TYPE LEV

:SYST:TOUT:TYPE?

:SYSTem:VERSion?

Returns the version of the SCPI standard. This command setting is not changed by power off or the *RST command.

Syntax :SYSTem:VERSion?

Query response *response* <newline>

response returns the version of the SCPI standard. For example, *1999.0*. Response data type is NR2.

Example :SYST:VERS?

TRACe Subsystem

For the numeric suffix [*c*], see "Numeric Suffix" on page 27.

:TRACe:BIN:CENTer

Sets the center value of histogram on Histogram view.

Syntax :TRACe[*c*]:BIN:CENTer *center*

:TRACe[*c*]:BIN:CENTer?

Parameter ***center*** Center value. Parameter data type is NRf.

Query response *center* <newline>

center returns the present center value setting of histogram. Response data type is NR3.

Example :TRAC:BIN:CENT 1E–6

:TRAC:BIN:CENT?

:TRACe:BIN:COUNt

Sets the number of bins for histogram on Histogram view.

If the range of measurement results is over the range calculated from the center value set by :TRACe:BIN:CENTer, the bin width set by :TRACe:BIN:WIDTh, and the number of bins set by this command, the system increases the number of bins up to 100 to fit the range of measurement results. So the actual number of bins may be different from the setting by this command. To confirm the actual number of bins for present measurement results, use :TRACe:BIN:COUNt:ACTual?

Syntax :TRACe[*c*]:BIN:COUNt *count*

:TRACe[*c*]:BIN:COUNt? [*count*]

Parameter ***count*** Number of bins. *value* (0 to 100)|MINimum|MAXimum|DEFault (default is 10). Parameter data type is NR1. Query does not support *count* = *value*.

Query response *count* <newline>

count returns the present setting of number of bins. If a parameter is specified, *count* returns the value assigned to DEF, MIN, or MAX. Response data type is NR1.

Example :TRAC:BIN:COUN 20

:TRAC:BIN:COUN?

:TRACe:BIN:COUNt:ACTual?

Gets the actual number of bins for histogram on Histogram view.

Syntax :TRACe[*c*]:BIN:COUNt:ACTual?

Query response *count* <newline>

count returns the actual number of bins. Response data type is NR1.

Example :TRAC:BIN:COUN:ACT?

:TRACe:BIN:DATA:<CURRent|LIMit|MATH|RESistance|VOLTage>?

Returns the number of data in each bin on Histogram view.

Syntax :TRACe[*c*]:BIN:DATA<CURRent|LIMit|MATH|RESistance|VOLTage>?

For <CURRent|LIMit|MATH|RESistance|VOLTage>, specify CURRent for current measurement data in the trace buffer, LIMit for the limit test data in the trace buffer, MATH for calculation result data in the trace buffer, RESistance for resistance measurement data in the trace buffer, or VOLTage for voltage measurement data in the trace buffer.

NOTE B2981B/B2983B can specify CURRent, LIMit and MATH only.

Query response *response* <newline>

Returns the number of data within each bin in order from bin for smaller measurement data. Response data type is the list of NR1. Each value is separated by a comma.

Example :TRAC:BIN:DATA:CURR?

:TRACe:BIN:DATA:<CURRent|LIMit|MATH|RESistance|VOLTage>:OOB?

Returns the number of data that is out of bins range on Histogram view.

Syntax :TRACe[*c*]:BIN:DATA<CURRent|LIMit|MATH|RESistance|VOLTage>:OOB?

For <CURRent|LIMit|MATH|RESistance|VOLTage>, specify CURRent for current measurement data in the trace buffer, LIMit for the limit test data in the trace buffer, MATH for calculation result data in the trace buffer, RESistance for resistance measurement data in the trace buffer, or VOLTage for voltage measurement data in the trace buffer.

NOTE B2981B/B2983B can specify CURRent, LIMit and MATH only.

Query response *response* <newline>

response returns *lower*,*upper*. *lower* is the number of data that is lower than the lowest bin range and *upper* is the number of data that is higher than the highest bin range. Each value is separated by a comma. Response data type is NR1.

Example :TRAC:BIN:DATA:CURR:OOB?

:TRACe:BIN:WIDTh

Sets the bin width on Histogram view.

Syntax :TRACe[*c*]:BIN:WIDTh *width*

:TRACe[*c*]:BIN:WIDTh?

Parameter ***width*** Bin width. Parameter data type is NRf. Default is 0.004.

Query response *width* <newline>

width returns the present bin width setting of histogram. Response data type is NR3.

Example :TRAC:BIN:WIDT 1E–7

:TRAC:BIN:WIDT?

:TRACe:CLEar

Clears the trace buffer of the specified channel. This command is effective when the trace buffer control mode is set to NEV by the :TRACe:FEED:CONTrol command.

Syntax :TRACe[*c*]:CLEar

Example :TRAC:CLE

:TRACe:DATA?

Returns data in the trace buffer. The data placed in the buffer is specified by the :TRACe:FEED command.

Syntax :TRACe[*c*]:DATA? [*offset*[, *size*]]

Parameter

offset	Indicates the beginning of the data received. *n*|CURRent|STARt (default). Parameter data type is NR1 or CPD. *offset* = *n* specifies the *n+1*th data. *n* is an integer, 0 to maximum (depends on the buffer state). *offset* = CURR specifies the present data position. *offset* = STAR specifies the top of trace buffer. Same as *offset* = 0.
size	Number of data to be received. 1 to maximum (depends on the buffer state). Parameter data type is NR1. If this parameter is not specified, all data from *offset* is returned.

Query response *data* <newline>

Response data type is NR3. See "Data Output Format" on page 31.

Example :TRAC:DATA? 0,10

:TRACe:DATA:<CURRent|LIMit|MATH|RESistance|VOLTage>?

Returns specified measurement data in the trace buffer. The data stored in the buffer is specified by the :TRACe:FEED command.

Syntax :TRACe[*c*]:DATA:<CURRent|LIMit|MATH|RESistance|VOLTage>? [*offset*[, *size*]]

For <CURRent|LIMit|MATH|RESistance|VOLTage>, specify CURRent for current measurement data in the trace buffer, LIMit for the limit test data in the trace buffer, MATH for calculation result data in the trace buffer, RESistance for resistance measurement data in the trace buffer, or VOLTage for voltage measurement data in the trace buffer.

NOTE B2981B/B2983B can specify CURRent, LIMit and MATH only.

Parameter

offset	Specifies the beginning of the data to be received. *n*|CURRent|STARt (default). Parameter data type is NR1 or CPD. *offset* = *n* specifies the *n+1*th data. *n* is an integer, 0 to maximum (depends on the buffer state). *offset* = CURR specifies the present data position. *offset* = STAR specifies the top of trace buffer. Same as *offset* = 0.
size	Number of data to be received. 1 to maximum (depends on the buffer state). Parameter data type is NR1. If this parameter is not specified, all data from *offset* is returned.

Query response *data* <newline>

Response data type is NR3. See "Data Output Format" on page 31.

Example :TRAC:DATA:CURR? 0,10

:TRACe:FEED

Specifies the data placed in the trace buffer. This command is effective when the trace buffer control mode is set to NEV by the :TRACe:FEED:CONTrol command.

Syntax :TRACe[*c*]:FEED *type*

:TRACe[*c*]:FEED?

Parameter

type	Data type. LIMit|MATH|SENSe (default). Parameter data type is CPD.

type = SENS specifies the measurement result data, which contains all of the voltage measurement data, current measurement data, resistance measurement data, time data, status data, or source output setting data specified by the :FORMat:ELEMents:SENSe command.

type = LIM specifies the limit test data. The data contains the limit test data, time data, or status data specified by the :FORMat:ELEMents:CALCulate command. See :CALCulate:DATA? for the limit test data.

type = MATH specifies the calculation result data. The data contains the calculation result, time data, or status data specified by the :FORMat:ELEMents:CALCulate command. See :CALCulate:MATH:DATA? for more information.

Query response *type* <newline>

type returns the present setting of data type, MATH or SENS. Response data type is CRD.

Example :TRAC:FEED MATH

:TRAC:FEED?

:TRACe:FEED:CONTrol

Selects the trace buffer control.

Syntax :TRACe[*c*]:FEED:CONTrol *mode*

:TRACe[*c*]:FEED:CONTrol?

Parameter

mode	Trace buffer control mode. NEXT|NEVer (default). Parameter data type is CPD.

mode = NEV disables write operation to the trace buffer. The :TRACe:CLEar, :TRACe:FEED, and :TRACe:POINts commands can be used.

mode = NEXT enables write operation until buffer full. Buffer full changes *mode* to NEV automatically. No error occurs.

Query response *mode* <newline>

mode returns the present setting of the control mode, NEXT or NEV. Response data type is CRD.

Example :TRAC:FEED:CONT NEXT

:TRAC:FEED:CONT?

:TRACe:FREE?

Returns the available size (*available*) and the total size (*total*) of the trace buffer.

Syntax :TRACe[*c*]:FREE?

Query response *response* <newline>

response returns *available*,*total*. Each value is separated by a comma. Response data type is NR1.

Example :TRAC:FREE?

:TRACe:POINts

Sets the size of the trace buffer. This command is effective when the trace buffer control mode is set to NEV by the :TRACe:FEED:CONTrol command.

Syntax :TRACe[*c*]:POINts *points*

:TRACe[*c*]:POINts? [*points*]

Parameter

points	Size. *value* (1 to 100000)|MINimum|MAXimum|DEFault (default is 100000). Parameter data type is NR1. Query does not support *points* = *value*.

Query response *points* <newline>

points returns the present setting of the buffer size. If a parameter is specified, *points* returns the value assigned to DEF, MIN, or MAX. Response data type is NR1.

Example :TRAC:POIN 10000

:TRAC:POIN?

:TRACe:POINts:ACTual?

Returns the number of data in the trace buffer.

Syntax :TRACe[*c*]:POINts:ACTual?

Query response *points* <newline>

points returns the number of data in the trace buffer. Response data type is NR1.

Example :TRAC:POIN:ACT?

:TRACe:STATistic:DATA?

Returns the result of the statistical operation for the data stored in the trace buffer. Before executing this command, the statistical operation must be specified by the :TRACe:STATistic:FORMat command.

If the trace buffer is storing raw measurement data for multiple data types, the statistical operation is performed for all measurement data.

Statistical operation is not available for the TIME and STATus data.

Syntax :TRACe[*c*]:STATistic:DATA?

Query response *response* <newline>

response returns the result of the statistical operation. Response data type is NR3. See "Data Output Format" on page 31.

Example :TRAC:STAT:DATA?

:TRACe:STATistic:FORMat

Selects the statistical operation performed by the :TRACe:STATistic:DATA? command.

Syntax :TRACe[*c*]:STATistic:FORMat *operation*

:TRACe[*c*]:STATistic:FORMat?

Parameter

operation	Statistical operation. MINimum|MAXimum|SDEViation|PKPK|MEAN (default). Parameter data type is CPD.

operation = MEAN sets the operation for obtaining the mean value.

operation = SDEV sets the operation for obtaining the standard deviation.

operation = PKPK sets the operation for obtaining the peak to peak value.

operation = MIN sets the operation for obtaining the minimum value.

operation = MAX sets the operation for obtaining the maximum value.

Query response *operation* <newline>

operation returns the present setting of the statistical operation, MEAN, SDEV, PKPK, MIN, or MAX. Response data type is CRD.

Example :TRAC:STAT:FORM PKPK

:TRAC:STAT:FORM?

:TRACe:TSTamp:FORMat

Selects the rule for reading the timestamp data in the trace buffer.

Syntax :TRACe[*c*]:TSTamp:FORMat *rule*

:TRACe[*c*]:TSTamp:FORMat?

Parameter

rule	Rule for reading the timestamp data. DELTa|ABSolute (default). Parameter data type is CPD.

rule = ABS sets the returned data to the incremental value for the first timestamp data.

rule = DELT sets the returned data to the incremental value for the previous timestamp data.

Query response *rule* <newline>

rule returns the present setting of the rule, DELT or ABS. Response data type is CRD.

Example :TRAC:TST:FORM DELT

:TRAC:TST:FORM?

TRIGger Subsystem

For the numeric suffix [*c*], see "Numeric Suffix" on page 27.

:ABORt<:ACQuire|:TRANsient|[:ALL]>

Aborts the specified device action for the specified channel. Trigger status is changed to idle.

Syntax :ABORt<:ACQuire|:TRANsient|[:ALL]> [*chanlist*]

For <:ACQuire|:TRANsient|[:ALL]>, specify :ACQuire for measurement, :TRANsient for source output, or :ALL for both device actions.

NOTE TRANsient is not available on B2981B/B2983B.

Parameter ***chanlist*** Channels. Parameter data type is channel list. (@1). See "Channel List Parameter" on page 27.

(@1) selects channel 1.

If this parameter is not specified, *chanlist* = (@1) is set.

Example :ABOR:ACQ (@1)

:ARM<:ACQuire|:TRANsient|[:ALL]>[:IMMediate]

Sends an immediate arm trigger for the specified device action to the specified channel. When the status of the specified device action is initiated, the arm trigger causes a layer change from arm to trigger.

Syntax :ARM<:ACQuire|:TRANsient|[:ALL]>[:IMMediate] [*chanlist*]

For <:ACQuire|:TRANsient|[:ALL]>, specify :ACQuire for measurement, :TRANsient for source output, or :ALL for both device actions.

NOTE TRANsient is not available on B2981B/B2983B.

Parameter *chanlist* Channels. Parameter data type is channel list. (@1). See "Channel List Parameter" on page 27.

(@1) selects channel 1.

If this parameter is not specified, *chanlist* = (@1) is set.

Example :ARM:ACQ (@1)

:ARM<:ACQuire|:TRANsient|[:ALL]>[:LAYer]:BYPass

Enables or disables a bypass for the event detector in the arm layer.

Syntax :ARM[*c*]<:ACQuire|:TRANsient|[:ALL]>[:LAYer]:BYPass *bypass*

:ARM[*c*]<:ACQuire|:TRANsient>[:LAYer]:BYPass?

For <:ACQuire|:TRANsient|[:ALL]> and <:ACQuire|:TRANsient>, specify :ACQuire for measurement, :TRANsient for source output, or :ALL for both device actions.

NOTE TRANsient is not available on B2981B/B2983B.

Parameter *bypass* Bypass setting. ONCE|OFF (default). Parameter data type is CPD.

bypass = OFF disables the bypass.

bypass = ONCE enables the bypass, but only for the first passage.

Query response *response* <newline>

response returns the present setting of the bypass, OFF or ONCE. Response data type is CRD.

Example :ARM:BYP ONCE

:ARM:ACQ:BYP?

:ARM<:ACQuire|:TRANsient|[:ALL]>[:LAYer]:COUNt

Sets the arm count for the specified device action.

Syntax :ARM[*c*]<:ACQuire|:TRANsient|[:ALL]>[:LAYer]:COUNt *arm_count*

:ARM[*c*]<:ACQuire|:TRANsient>[:LAYer]:COUNt? [*arm_count*]

:ARM[*c*][:ALL][:LAYer]:COUNt? *arm_count*

For <:ACQuire|:TRANsient|[:ALL]>, specify :ACQuire for measurement, :TRANsient for source output, or :ALL for both device actions.

NOTE TRANsient is not available on B2981B/B2983B.

Parameter

arm_count	Arm count. *value* (1 to 100000 or 2147483647)|INFinity|MINimum|MAXimum|DEFault (default is 1). Parameter data type is NRf+. *value* = 2147483647 indicates infinity. Query does not support *arm_count* = *value* and INFinity. *Arm count* × *Trigger count* must be less than 100001.

Query response *response* <newline>

response returns the present setting of arm count. If a parameter is specified, *response* returns the value assigned to DEF, MIN, MAX, or INF. Response data type is NR1. If the arm count is set to infinity, *response* returns 2147483647.

Example

```
:ARM:COUN 10
:ARM:ACQ:COUN?
```

:ARM<:ACQuire|:TRANsient|[:ALL]>[:LAYer]:DELay

Sets the arm delay for the specified device action.

Syntax

:ARM[*c*]<:ACQuire|:TRANsient|[:ALL]>[:LAYer]:DELay *delay*

:ARM[*c*]<:ACQuire|:TRANsient>[:LAYer]:DELay? [*delay*]

:ARM[*c*][:ALL][:LAYer]:DELay? *delay*

For <:ACQuire|:TRANsient|[:ALL]>, specify :ACQuire for measurement, :TRANsient for source output, or :ALL for both device actions.

NOTE TRANsient is not available on B2981B/B2983B.

Parameter

delay	Arm delay, in seconds. *value* (0 to 100000)|MINimum|MAXimum|DEFault (default is 0). Parameter data type is NRf+. Query does not support *delay* = *value*.

Query response *response* <newline>

response returns the present setting of arm delay. If a parameter is specified, *response* returns the value assigned to DEF, MIN, or MAX. Response data type is NR3.

Example :ARM:DEL 0.1

:ARM:ACQ:DEL?

:ARM<:ACQuire|:TRANsient|[:ALL]>[:LAYer]:SOURce:LAN

Specifies one or more LXI triggers used for the arm source for the specified device action.

Syntax :ARM[*c*]<:ACQuire|:TRANsient|[:ALL]>[:LAYer]:SOURce:LAN *lan_id*{,*lan_id*}

:ARM[*c*]<:ACQuire|:TRANsient>[:LAYer]:SOURce:LAN?

For <:ACQuire|:TRANsient|[:ALL]> and <:ACQuire|:TRANsient>, specify :ACQuire for measurement, :TRANsient for source output, or :ALL for both device actions.

NOTE TRANsient is not available on B2981B/B2983B.

Parameter

lan_id	LAN ID of the LXI trigger. LAN0|LAN1|LAN2|LAN3|LAN4|LAN5|LAN6|LAN7. All is selected as default. Parameter data type is CPD.

Query response *response* <newline>

response returns the present setting, LAN0 through LAN7. Response data type is CRD. Multiple responses are separated by a comma.

Example :ARM:SOUR:LAN LAN7

:ARM:ACQ:SOUR:LAN?

:ARM<:ACQuire|:TRANsient|[:ALL]>[:LAYer]:SOURce[:SIGNal]

Selects the arm source for the specified device action.

Syntax :ARM[*c*]<:ACQuire|:TRANsient|[:ALL]>[:LAYer]:SOURce[:SIGNal] *source*

:ARM[*c*]<:ACQuire|:TRANsient>[:LAYer]:SOURce[:SIGNal]?

For <:ACQuire|:TRANsient|[:ALL]> and <:ACQuire|:TRANsient>, specify :ACQuire for measurement, :TRANsient for source output, or :ALL for both device actions.

NOTE TRANsient is not available on B2981B/B2983B.

Parameter

source Arm source. AINT (default)|BUS|TIMer|INT1|INT2|LAN|EXT1|EXT2|EXT3|EXT4|EXT5|EXT6|EXT7|TIN. Parameter data type is CPD.

source = AINT (automatic internal) automatically selects the arm source most suitable for the present operating mode by using internal algorithms.

source = BUS selects the remote interface trigger command such as the group execute trigger (GET) and the *TRG command.

source = TIMer selects a signal internally generated every interval set by the :ARM<:ACQuire|:TRANsient|[:ALL]>[:LAYer]:TIMer command.

source = INT1 or INT2 selects a signal from the internal bus 1 or 2, respectively.

source = LAN selects the LXI trigger specified by the :ARM<:ACQuire|:TRANsient|[:ALL]>[:LAYer]:SOURce:LAN command.

source = EXT*n* selects a signal from the GPIO pin *n*, which is an input port of the Digital I/O D-sub connector on the rear panel. *n* = 1 to 7.

source = TIN selects the BNC Trigger In.

Query response

response <newline>

response returns the present setting of arm source, AINT, BUS, TIM, INT1, INT2, LAN, or EXT1 through EXT7. Response data type is CRD.

Example

:ARM:SOUR AINT

:ARM:ACQ:SOUR?

:ARM<:ACQuire|:TRANsient|[:ALL]>[:LAYer]:TIMer

Sets the interval of the TIMer arm source for the specified device action.

Syntax

:ARM[*c*]<:ACQuire|:TRANsient|[:ALL]>[:LAYer]:TIMer *interval*

:ARM[*c*]<:ACQuire|:TRANsient>[:LAYer]:TIMer? [*interval*]

:ARM[*c*][:ALL][:LAYer]:TIMer? *interval*

For <:ACQuire|:TRANsient|[:ALL]>, specify :ACQuire for measurement, :TRANsient for source output, or :ALL for both device actions.

NOTE TRANsient is not available on B2981B/B2983B.

Parameter

interval Interval, in seconds. *value* (see below)|MINimum|MAXimum|DEFault (default is 1E–4).
Parameter data type is NRf+. Query does not support *interval* = *value*.

The following values are available for *value*.

For B2981B and B2983B:
ACQuire: 1E–5 to 1E+5
ALL: 1E–5 to 1E+5

For B2985B and B2987B:
ACQuire: 1E–5 to 1E+5
TRANsient: 1E–4 to 1E+5
ALL: 1E–4 to 1E+5

Query response *response* <newline>

response returns the present setting of interval. If a parameter is specified, *response* returns the value assigned to DEF, MIN, or MAX. Response data type is NR3.

Example

```
:ARM:TIM 2E–4
:ARM:ACQ:TIM?
```

:ARM<:ACQuire|:TRANsient|[:ALL]>[:LAYer]:TOUTput:SIGNal

Selects the trigger output for the status change between the idle state and the arm layer. Multiple trigger output ports can be set.

Syntax

:ARM[*c*]<:ACQuire|:TRANsient|[:ALL]>[:LAYer]:TOUTput:SIGNal *output*{,output}

:ARM[*c*]<:ACQuire|:TRANsient>[:LAYer]:TOUTput:SIGNal?

For <:ACQuire|:TRANsient|[:ALL]> and <:ACQuire|:TRANsient>, specify :ACQuire for measurement, :TRANsient for source output, or :ALL for both device actions.

NOTE TRANsient is not available on B2981B/B2983B.

Parameter *output* Trigger output port. EXT1 (default)|EXT2|EXT3|EXT4|EXT5|EXT6|EXT7|LAN|INT1|INT2|TOUT. Parameter data type is CPD.

output = INT1 or INT2 selects the internal bus 1 or 2, respectively.

output = LAN selects a LAN port.

output = EXT*n* selects the GPIO pin *n*, which is an output port of the Digital I/O D-sub connector on the rear panel. *n* = 1 to 7.

output = TOUT selects the BNC Trigger Out.

Query response *response* <newline>

response returns the present setting, INT1, INT2, LAN, or EXT1 through EXT7. Response data type is CRD. Multiple responses are separated by a comma.

Example :ARM:TOUT:SIGN EXT1

:ARM:ACQ:TOUT:SIGN?

:ARM<:ACQuire|:TRANsient|[:ALL]>[:LAYer]:TOUTput[:STATe]

Enables or disables the trigger output for the status change between the idle state and the arm layer.

Syntax :ARM[*c*]<:ACQuire|:TRANsient|[:ALL]>[:LAYer]:TOUTput[:STATe] *mode*

:ARM[*c*]<:ACQuire|:TRANsient>[:LAYer]:TOUTput[:STATe]?

For <:ACQuire|:TRANsient|[:ALL]> and <:ACQuire|:TRANsient>, specify :ACQuire for measurement, :TRANsient for source output, or :ALL for both device actions.

NOTE TRANsient is not available on B2981B/B2983B.

Parameter *mode* Trigger output ON or OFF. 1|ON|0|OFF (default). Parameter data type is boolean.

mode = 1 or ON enables the trigger output.

mode = 0 or OFF disables the trigger output.

Query response *response* <newline>

response returns 1 or 0, and indicates that the trigger output is on or off, respectively. Response data type is NR1.

Example :ARM:TOUT 1

:ARM:ACQ:TOUT:STAT?

:IDLE<:ACQuire|:TRANsient|[:ALL]>?

Checks the status of the specified device action for the specified channel, and waits until the status is changed to idle.

Syntax :IDLE[*c*]<:ACQuire|:TRANsient|[:ALL]>?

For <:ACQuire|:TRANsient|[:ALL]>, specify :ACQuire for measurement, :TRANsient for source output, or :ALL for both device actions.

NOTE TRANsient is not available on B2981B/B2983B.

Query response *response* <newline>

response returns 1 if the specified device action is in the idle state. Response data type is NR1.

Example :IDLE:ACQ

:INITiate[:IMMediate]<:ACQuire|:TRANsient|[:ALL]>

Initiates the specified device action for the specified channel. Trigger status is changed from idle to initiated.

Syntax :INITiate[:IMMediate]<:ACQuire|:TRANsient|[:ALL]> [*chanlist*]

For <:ACQuire|:TRANsient|[:ALL]>, specify :ACQuire for measurement, :TRANsient for source output, or :ALL for both device actions.

NOTE TRANsient is not available on B2981B/B2983B.

Parameter ***chanlist*** Channels. Parameter data type is channel list. (@1). See "Channel List Parameter" on page 27.

(@1) selects channel 1.

If this parameter is not specified, *chanlist* = (@1) is set.

Example :INIT:ACQ (@1)

:TRIGger<:ACQuire|:TRANsient|[:ALL]>:BYPass

Enables or disables a bypass for the event detector in the trigger layer.

Syntax :TRIGger[*c*]<:ACQuire|:TRANsient|[:ALL]>:BYPass *bypass*

:TRIGger[*c*]<:ACQuire|:TRANsient>:BYPass?

For <:ACQuire|:TRANsient|[:ALL]> and <:ACQuire|:TRANsient>, specify :ACQuire for measurement, :TRANsient for source output, or :ALL for both device actions.

NOTE TRANsient is not available on B2981B/B2983B.

Parameter

bypass	Bypass setting. ONCE|OFF (default). Parameter data type is CPD.

bypass = OFF disables the bypass.

bypass = ONCE enables the bypass, but only for the first passage.

Query response *response* <newline>

response returns the present setting of the bypass, OFF or ONCE. Response data type is CRD.

Example :TRIG:BYP ONCE

:TRIG:TRAN:BYP?

:TRIGger<:ACQuire|:TRANsient|[:ALL]>:COUNt

Sets the trigger count for the specified device action.

Syntax :TRIGger[*c*]<:ACQuire|:TRANsient|[:ALL]>:COUNt *trigger_count*

:TRIGger[*c*]<:ACQuire|:TRANsient>:COUNt? [*trigger_count*]

:TRIGger[*c*][:ALL]:COUNt? *trigger_count*

For <:ACQuire|:TRANsient|[:ALL]>, specify :ACQuire for measurement, :TRANsient for source output, or :ALL for both device actions.

NOTE TRANsient is not available on B2981B/B2983B.

Parameter ***trigger_count*** Trigger count. *value* (1 to 100000 or 2147483647)|INFinity|MINimum|MAXimum|DEFault (default is 1). Parameter data type is NRf+. *value* = 2147483647 indicates infinity.

Query does not support *trigger_count* = *value* and INFinity.

Arm count × *Trigger count* must be less than 100001.

Query response *response* <newline>

response returns the present setting of trigger count. If a parameter is specified, *response* returns the value assigned to DEF, MIN, or MAX. Response data type is NR1.

Example :TRIG:COUN 10

:TRIG:TRAN:COUN?

:TRIGger<:ACQuire|:TRANsient|[:ALL]>:DELay

Sets the trigger delay for the specified device action.

Syntax :TRIGger[*c*]<:ACQuire|:TRANsient|[:ALL]>:DELay *delay*

:TRIGger[*c*]<:ACQuire|:TRANsient>:DELay? [*delay*]

:TRIGger[*c*][:ALL]:DELay? *delay*

For <:ACQuire|:TRANsient|[:ALL]>, specify :ACQuire for measurement, :TRANsient for source output, or :ALL for both device actions.

NOTE TRANsient is not available on B2981B/B2983B.

Parameter ***delay*** Trigger delay, in seconds. *value* (0 to 100000)|MINimum|MAXimum|DEFault (default is 0). Parameter data type is NRf+. Query does not support *delay* = *value*.

Query response *response* <newline>

response returns the present setting of trigger delay. If a parameter is specified, *response* returns the value assigned to DEF, MIN, or MAX. Response data type is NR3.

Example :TRIG:DEL 0.1

:TRIG:TRAN:DEL?

:TRIGger<:ACQuire|:TRANsient|[:ALL]>[:IMMediate]

Sends an immediate trigger for the specified device action to the specified channel. When the status of the specified device action is initiated, the trigger causes the specified device action.

Syntax :TRIGger<:ACQuire|:TRANsient|[:ALL]>[:IMMediate] [*chanlist*]

For <:ACQuire|:TRANsient|[:ALL]>, specify :ACQuire for measurement, :TRANsient for source output, or :ALL for both device actions.

NOTE TRANsient is not available on B2981B/B2983B.

Parameter ***chanlist*** Channels. Parameter data type is channel list. (@1). See "Channel List Parameter" on page 27.

(@1) selects channel 1.

If this parameter is not specified, *chanlist* = (@1) is set.

Example :TRIG:ACQ (@1)

:TRIGger<:ACQuire|:TRANsient|[:ALL]>:SOURce:LAN

Specifies one or more LXI triggers used for the trigger source for the specified device action.

Syntax :TRIGger[*c*]<:ACQuire|:TRANsient|[:ALL]>:SOURce:LAN *lan_id*{,*lan_id*}

:TRIGger[*c*]<:ACQuire|:TRANsient>:SOURce:LAN?

For <:ACQuire|:TRANsient|[:ALL]> and <:ACQuire|:TRANsient>, specify :ACQuire for measurement, :TRANsient for source output, or :ALL for both device actions.

NOTE TRANsient is not available on B2981B/B2983B.

Parameter *lan_id* LAN ID of the LXI trigger. LAN0|LAN1|LAN2|LAN3|LAN4|LAN5|LAN6|LAN7. All is selected as default. Parameter data type is CPD.

Query response *response* <newline>

response returns the present setting, LAN0 through LAN7. Response data type is CRD. Multiple responses are separated by a comma.

Example :TRIG:SOUR:LAN LAN7

:TRIG:TRAN:SOUR:LAN?

:TRIGger<:ACQuire|:TRANsient|[:ALL]>:SOURce[:SIGNal]

Selects the trigger source for the specified device action.

Syntax :TRIGger[*c*]<:ACQuire|:TRANsient|[:ALL]>:SOURce[:SIGNal] *source*

:TRIGger[*c*]<:ACQuire|:TRANsient>:SOURce[:SIGNal]?

For <:ACQuire|:TRANsient|[:ALL]> and <:ACQuire|:TRANsient>, specify :ACQuire for measurement, :TRANsient for source output, or :ALL for both device actions.

NOTE TRANsient is not available on B2981B/B2983B.

Parameter *source* Trigger source. AINT (default)|BUS|TIMer|INT1|INT2|LAN|EXT1|EXT2|EXT3|EXT4|EXT5|EXT6|EXT7|TIN. Parameter data type is CPD.

source = AINT (automatic internal) automatically selects the trigger source most suitable for the present operating mode by using internal algorithms.

source = BUS selects the remote interface trigger command such as the group execute trigger (GET) and the *TRG command.

source = TIMer selects a signal internally generated every interval set by the :TRIGger<:ACQuire|:TRANsient|[:ALL]>:TIMer command.

source = INT1 or INT2 selects a signal from the internal bus 1 or 2, respectively.

source = LAN*n* selects a LXI trigger specified by the :TRIGger<:ACQuire|:TRANsient|[:ALL]>:SOURce:LAN command.

source = EXT*n* selects a signal from the GPIO pin *n*, which is an input port of the Digital I/O D-sub connector on the rear panel. *n* = 1 to 7.

source = TIN selects the BNC Trigger In.

Query response *response* <newline>

response returns the present setting of trigger source, AINT, BUS, TIM, INT1, INT2, LAN, or EXT1 through EXT7. Response data type is CRD.

Example

```
:TRIG:SOUR EXT1
:TRIG:TRAN:SOUR:SIGN?
```

:TRIGger<:ACQuire|:TRANsient|[:ALL]>:TIMer

Sets the interval of the TIMer trigger source for the specified device action.

Syntax

```
:TRIGger[c]<:ACQuire|:TRANsient|[:ALL]>:TIMer interval
:TRIGger[c]<:ACQuire|:TRANsient>:TIMer? [interval]
:TRIGger[c][:ALL]:TIMer? interval
```

For <:ACQuire|:TRANsient|[:ALL]>, specify :ACQuire for measurement, :TRANsient for source output, or :ALL for both device actions.

NOTE TRANsient is not available on B2981B/B2983B.

Parameter

interval Interval, in seconds. *value* (see below)|MINimum|MAXimum|DEFault (default is 1E–4). Parameter data type is NRf+. Query does not support *interval* = *value*.

The following values are available for *value*.

For B2981B and B2983B:
ACQuire: 1E–5 to 1E+5
ALL: 1E–5 to 1E+5

For B2985B and B2987B:
ACQuire: 1E–5 to 1E+5
TRANsient: 1E–4 to 1E+5
ALL: 1E–4 to 1E+5

Query response *response* <newline>

response returns the present setting of interval. If a parameter is specified, *response* returns the value assigned to DEF, MIN, or MAX. Response data type is NR3.

Example :TRIG:TIM 2E-4

:TRIG:TRAN:TIM?

:TRIGger<:ACQuire|:TRANsient|[:ALL]>:TOUTput:SIGNal

Selects the trigger output for the status change between the arm layer and the trigger layer. Multiple trigger output ports can be set.

Syntax :TRIGger[*c*]<:ACQuire|:TRANsient|[:ALL]>:TOUTput:SIGNal *output*{,output}

:TRIGger[*c*]<:ACQuire|:TRANsient>:TOUTput:SIGNal?

For <:ACQuire|:TRANsient|[:ALL]> and <:ACQuire|:TRANsient>, specify :ACQuire for measurement, :TRANsient for source output, or :ALL for both device actions.

NOTE TRANsient is not available on B2981B/B2983B.

Parameter ***output*** Trigger output port. EXT1 (default)|EXT2|EXT3|EXT4|EXT5|EXT6|EXT7| LAN|INT1|INT2|TOUT. Parameter data type is CPD.

output = INT1 or INT2 selects the internal bus 1 or 2, respectively.

output = LAN selects a LAN port.

output = EXT*n* selects the GPIO pin *n*, which is an output port of the Digital I/O D-sub connector on the rear panel. *n* = 1 to 7.

output = TOUT selects the BNC Trigger Out.

Query response *response* <newline>

response returns the present setting, INT1, INT2, LAN, or EXT1 through EXT7. Response data type is CRD. Multiple responses are separated by a comma.

Example :TRIG:TOUT:SIGN EXT3

:TRIG:TRAN:TOUT:SIGN?

:TRIGger<:ACQuire|:TRANsient|[:ALL]>:TOUTput[:STATe]

Enables or disables the trigger output for the status change between the arm layer and the trigger layer.

Syntax :TRIGger[*c*]<:ACQuire|:TRANsient|[:ALL]>:TOUTput[:STATe] *mode*

:TRIGger[*c*]<:ACQuire|:TRANsient>:TOUTput[:STATe]?

For <:ACQuire|:TRANsient|[:ALL]> and <:ACQuire|:TRANsient>, specify :ACQuire for measurement, :TRANsient for source output, or :ALL for both device actions.

NOTE TRANsient is not available on B2981B/B2983B.

Parameter *mode* Trigger output ON or OFF. 1|ON|0|OFF (default). Parameter data type is boolean.

mode = 1 or ON enables the trigger output.

mode = 0 or OFF disables the trigger output.

Query response *response* <newline>

response returns 1 or 0, and indicates that the trigger output is on or off, respectively. Response data type is NR1.

Example :TRIG:TOUT 1

:TRIG:TRAN:TOUT:STAT?

5 Error Messages

This chapter shows the error code/messages returned from Keysight B2980 when any error occurred during a SCPI program is executed.

Table 5-1 Error Category

Error range	Error category	Standard event status register bit
0	No error	
–100 to –199	Command error	bit5
–200 to –299	Execution error	bit4
–300 to –399	Device-dependent error	bit3
–400 to –499	Query error	bit2
1 to 32767	B2980 specific error	bit3

Negative error numbers (command error, execution error, device-dependent error, query error) are standard SCPI errors.

Positive error numbers are B2980 specific errors, not standard SCPI errors.

When B2980 is in the remote control state, the occurrence of an error (except for error number 0 or emergency error) sets the corresponding bit in the standard event status register. An emergency error sets the corresponding bit in the emergency status register.

If an error occurs, the error number and message are placed in the error queue, which can be read by the `:SYSTem:ERRor?` query command. Then the front-panel ERR indicator turns on. Errors are cleared by reading them. When all errors are read from the queue, the errors are cleared and the ERR indicator turns off. Errors are retrieved in the FIFO (first-in-first-out) order. The first error returned is the first error that was stored.

The error queue is also cleared by the common command *CLS, and when power is turned on. The error queue is not cleared by the *RST command. For these commands, see Chapter 3, "Common Commands."

If more errors have occurred than can fit in the buffer, the last error stored in the queue (the most recent error) is replaced with –350, "Error queue overflow." No additional errors are stored until removing errors from the queue. If no errors have occurred when reading the error queue, the instrument responds with +0, "No error."

No Error

This message indicates that Keysight B2980 has no errors.

Error 0 `No error`

The error queue is completely empty. Every error/event in the queue has been read or the queue was purposely cleared by power-on, `*CLS`, and so on.

Command Error

If syntax of SCPI command is *not* valid, a -1*XX* error occurs.

Error -100 `Command error`

Generic syntax error that cannot be determined more specifically.

Error -101 `Invalid character`

An invalid character for the type of a syntax element was received; for example, a header containing an ampersand.

Error -102 `Syntax error`

An unrecognized command or data type was received; for example, a string was received when B2980 does not accept strings.

Error -103 `Invalid separator`

An illegal character was received when a separator was expected; for example, the semicolon was omitted after a program message unit.

Error -104 `Data type error`

An improper data type was received; for example, numeric data was expected but string data was received.

Error -105 `GET not allowed`

A group execute trigger was received within a program message.

Error -108 `Parameter not allowed`

Too many parameters for the command were received.

Error -109 `Missing parameter`

Fewer parameters were received than required for the command.

Error -110 `Command header error`

An error was detected in the header. This error message is reported if B2980 cannot determine the more specific header errors -111 through -114.

Error -111 `Header separator error`

An illegal character for a header separator was received; for example, no white space between the header and parameter.

Error -112 `Program mnemonic too long`

A keyword in the command header contains more than twelve characters.

Error -113 `Undefined header`

An undefined command header was received; for example, `*XYZ`.

Error -114 `Header suffix out of range`

The value of a numeric suffix attached to a program mnemonic is out of range; for example, `:INPut2:STATe` specifies illegal channel number 2.

Error -120 `Numeric data error`

Numeric (including the non-decimal numeric types) data error. This error message is reported when B2980 cannot determine the more specific errors -121 through -128.

Error -121 `Invalid character in number`

An invalid character for the data type was received; for example, an alpha-character was received when the type was decimal numeric.

Error -123 `Exponent too large`

The magnitude of the exponent was larger than 32000.

Error -124 `Too many digits`

The mantissa of a decimal numeric data contained more than 255 digits excluding leading zeros.

Error -128 `Numeric data not allowed`

Numeric data is not allowed in this position for this command.

Error -130 `Suffix error`

An error was detected in the suffix. This error message is reported if B2980 cannot determine the more specific suffix errors -131 through -138.

Error -131 `Invalid suffix`

The suffix does not follow the correct syntax or the suffix is inappropriate.

Error -134 `Suffix too long`

The suffix contains more than 12 characters.

Error -138 `Suffix not allowed`

A suffix was received after a numeric parameter that does not allow suffixes.

Error -140 `Character data error`

An error was detected in a character parameter. This error message is reported if B2980 cannot determine the more specific errors -141 through -148.

Error -141 `Invalid character data`

Either the character parameter contains an invalid character or the particular element received is not valid for the command header.

Error -144 `Character data too long`

The character parameter contains more than 12 characters.

Error -148 `Character data not allowed`

A character parameter is not allowed for this position.

Error -150 `String data error`

An error was detected in a string parameter. This error is reported if B2980 cannot determine a more specific error -151 and -158.

Error -151 `Invalid string data`

An invalid string parameter data was received; for example, an END message was received before the terminal quote character.

Error -158 `String data not allowed`

A string parameter data was received but was not allowed at this point.

Error -160 `Block data error`

An error was detected in a block data. This error is reported if B2980 cannot determine more specific errors -161 and -168.

Error -161 `Invalid block data`

An invalid block data was received; for example, an END message was received before the length was satisfied.

Error -168 `Block data not allowed`

A legal block data was received but was not allowed at this point.

Error -170 `Expression error`

An error was detected in an expression. This error is reported if B2980 cannot determine more specific errors -171 and -178.

Error -171 `Invalid expression`

The expression was invalid; for example, unmatched parentheses or an illegal character.

Error -178 `Expression data not allowed`

An expression was received but was not allowed at this point.

Execution Error

Keysight B2980 reports -2*XX* errors when it is unable to perform a valid programming command.

Error -200 `Execution error`

Generic execution error for B2980 that cannot be determined more specifically.

Error -220 `Parameter error;` *message*

Invalid parameter was specified. Set appropriate value.

Error -221 `Settings conflict;` *message*

A specified parameter setting could not be executed due to the present device state. Check the settings specified by *message* and set appropriate value.

Error -222 `Data out of range;` *message*

Interpreted value of the program was out of range as defined by B2980. Check the B2980 settings specified by *message* and set appropriate value.

Error -223 `Too much data`

Too many parameters were sent. Reduce number of list data.

Error -224 `Illegal parameter value;` *message*

Illegal parameter value was sent. Set appropriate parameter value.

Error -230 `Data corrupt or stale`

Possibly invalid data; new reading started but not completed since last access.

Error -231 `Data questionable`

Measurement accuracy is suspect.

Error -232 `Invalid format`

The data format or structure is inappropriate.

Error -233 `Invalid version`

The version of the data format is incorrect to the instrument.

Error -240 `Hardware error`

A hardware problem in B2980. This error message is reported if B2980 cannot detect the more specific error -241.

Error -241 `Hardware missing; To recover channel, execute *TST?`

A program command or query could not be executed because of missing hardware; for example, an option was not installed. Execute the `*TST?` command to recover or unlock channel.

Device-Dependent Error

-3*XX* errors indicate that Keysight B2980 has detected an error that is not a command error, a query error, or an execution error; some device operations did not properly complete, possibly due to an abnormal hardware or firmware condition. These codes are also used for self-test response errors.

Error -300 `Device-specific error`

Generic device-dependent error for B2980 that cannot be determined more specifically.

Error -310 `System error`

Some error, termed “system error” by B2980, has occurred.

Error -311 `Memory error`

An error was detected in B2980’s memory.

Error -313 `Calibration memory lost; Calibration data has been lost, Calibration data is initialized`

`Calibration memory lost; Nonvolatile data saved by the *CAL? command has been lost`

Non-volatile data related to the `*CAL?` command has been lost.

Error -315 `Configuration memory lost`

Non-volatile configuration data saved by B2980 has been lost.

Error -321 `Out of memory`

Too many data was sent at a time.

Error -350 `Queue overflow`

This code is entered into the queue instead of the code that caused the error. This code indicates that there is no room in the queue and an error occurred but was not recorded.

Query Error

If the output queue control of Keysight B2980 detects one of following problems, a -4*XX* error occurs.

- An attempt was made to read data from the output queue when no output data is present or pending.
- Data in the output queue has been lost.

Error -400 `Query error`

Generic query error for B2980 that cannot be determined more specifically.

Error -410 `Query INTERRUPTED`

A condition causing an INTERRUPTED query error occurred; for example, a query followed by DAB or GET before a response was completely sent.

Error -420 `Query UNTERMINATED`

A condition causing an UNTERMINATED query error occurred; for example, B2980 was addressed to talk and an incomplete program message was received.

Error -430 `Query DEADLOCKED`

A condition causing a DEADLOCKED query error occurred; for example, both input buffer and output buffer are full and B2980 cannot continue.

Error -440 `Query UNTERMINATED after indefinite response`

A query was received in the same program message after a query requesting an indefinite length response was executed.

B2980 Specific Error

Positive error numbers are Keysight B2980 specific errors, not standard SCPI errors. Consult service for errors 111 to 140 and 142.

Error 111 `Self-calibration failed; Voltage burden,` *item*

Failed the voltage burden self-calibration specified by *item*.

Error 112 `Self-calibration failed; Current offset,` *item*

Failed the current offset self-calibration specified by *item*.

Error 113 `Self-calibration failed; Current gain,` *item*

Failed the current gain self-calibration specified by *item*.

Error 114 `Self-calibration failed; Charge offset,` *item*

Failed the charge offset self-calibration specified by *item*.

Error 115 `Self-calibration failed; Charge gain,` *item*

Failed the charge gain self-calibration specified by *item*.

Error 116 `Self-calibration failed; Voltage offset,` *item*

Failed the voltage offset self-calibration specified by *item*.

Error 117 `Self-calibration failed; Voltage gain,` *item*

Failed the voltage gain self-calibration specified by *item*.

Error 121 `Self-test failed; CPU communication,` *item*

Failed the CPU communication test specified by *item*.

Error 131 `Self-test failed; Module communication,` *item*

Failed the module communication test specified by *item*.

Error 133 `Self-test failed; Trigger count,` *item*

Failed the trigger count test specified by *item*.

Error 134 `Self-test failed; I ADC,` *item*

Failed the current ADC test specified by *item*.

Error 135 `Self-test failed; V ADC,` *item*

Failed the voltage ADC test specified by *item*.

Error 136 `Self-test failed; I measure,` *item*

Failed the current measurement test specified by *item*.

Error 137 `Self-test failed; Q measure,` *item*

Failed the charge measurement test specified by *item*.

Error 138 `Self-test failed; Burden DAC,` *item*

Failed the burden DAC test specified by *item*.

Error 140 `Self-test failed; Temperature sensor,` *item*

Failed the temperature sensor test specified by *item*.

Error 141 `Self-test skipped; To recover module, execute *TST?`

Error 142 `Self-test failed; SDRAM`

Error 143 `Self-test failed; Voltage source,` *item*

Failed the voltage source test specified by *item*.

Error 144 `Self-test failed; Battery,` *item*

Failed the battery test specified by *item*.

Error 201 `Not able to perform requested operation`

Error 202 `Not allowed; Instrument locked by another I/O session`

The requested operation is not allowed because the instrument is locked by another I/O session. The instrument must be unlocked.

Error 203 `Not able to execute while instrument is measuring`

Error 210 `Operation is not completed`

Operation is still in progress. Wait for operation complete.

Error 211 `Cannot switch low sense terminal with output on`

Output relay must be off to switch low sense terminal.

Error 212 `Output relay must be on`

(Available for B2985B/B2987B)

Error 213 `Output relay must be off`

(Available for B2985B/B2987B)

Error 214 `Display must be enabled`

Display is currently disabled. Set remote display on.

Error 215 `Input relay must be on`

Error 216 `Input relay must be off`

Error 241 `Cannot detect power line frequency during battery operation`

Error 242 `Failed to detect power line frequency`

Error 290 `Not able to recall state: it is empty`

Error 291 `State file size error`

Error 292 `State file corrupt`

Error 301 `Emergency; IM over current detected`

Overcurrent status was detected. Input switch is opened. For B2985B/B2987B, voltage output changed to 0 V and the switch is opened. Execute the *TST? command.

Error 302 `Emergency; IM/QM over current detected`

Overcurrent status was detected. Input switch is opened. For B2985B/B2987B, voltage output changes to 0 V and the switch is opened. Execute the *TST? command.

Error 303 `Emergency; VM overload detected, check VM input connection`

(Available for B2985B/B2987B)
Voltage measure overload was detected. Check voltage measurement connection.

Error 304 `Emergency; VM guard abuse detected, check VM input connection`

(Available for B2985B/B2987B)
Voltage measure guard abuse status was detected. Guard of voltage measurement connects to COM. Execute the *TST? command.

Error 305 `Emergency; High temperature at meter unit detected`

High temperature status was detected at meter unit. Input switch is opened. For B2985B/B2987B, voltage output changes to 0 V and the switch is opened. Execute the *TST? command.

Error 306 `Emergency; Power supply for meter unit stopped`

Power supply for meter unit stopped. Input switch is opened. For B2985B/B2987B, voltage output changes to 0 V and the switch is opened. Execute the *TST? command.

Error 307 `Emergency; Power supply for source unit stopped`

(Available for B2985B/B2987B)
Power supply for source unit stopped. Voltage output changes to 0 V and the switch is opened. Execute the *TST? command.

Error 308 `Emergency; High voltage PS emergency detected`

(Available for B2985B/B2987B)
High voltage PS emergency was detected. Voltage output changes to 0 V and the switch is opened. Execute the *TST? command.

Error 311 `Emergency; VF 20V range overtemperature detected`

(Available for B2985B/B2987B)
Overtemperature status was detected at VF 20V range. Voltage output changes to 0 V and the switch is opened. Execute the *TST? command.

Error 312 `Emergency; High temperature at source unit detected`

(Available for B2985B/B2987B)
High temperature status at source unit was detected. Voltage output changes to 0 V and the switch is opened. Execute the *TST? command.

Error 313 `Emergency; Power supply for meter unit was turned off`

Power supply for meter unit was turned off because emergency status was detected. Meter and source units were disabled. Execute the *TST? command.

Error 314 `Emergency; Power supply for source unit was turned off`

(Available for B2985B/B2987B)
Power supply for source unit was turned off because emergency status was detected. source unit was disabled. Execute the *TST? command.

Error 331 `Emergency; Interlock open detected`

(Available for B2985B/B2987B)
Interlock open was detected. Voltage output changes to 0 V and the switch is opened. Execute the *TST? command. Do not open interlock circuit while module is in high voltage state.

Error 351 `Emergency; Internal communication failure detected by Module`

Internal communication failure was detected. Input switch is opened. For B2985B/B2987B, voltage output changed to 0 V and the switch is opened. Execute the *TST? command.

Error 352 `Emergency; Watchdog timer expired`

Watchdog timer expired status was detected. Input switch is opened. For B2985B/B2987B, voltage output changed to 0 V and the switch is opened. Execute the *TST? command.

Error 353 `Emergency; Communication to source unit was failed`

(Available for B2985B/B2987B)
Internal communication failure was detected between measure and source unit. Voltage output changes to 0 V and the switch is opened. Execute the *TST? command.

Error 356 `Emergency; Sense data FIFO overflow detected`

Sense data FIFO overflow was detected. Input switch is opened. For B2985B/B2987B, voltage output changed to 0 V and the switch is opened. Execute the *TST? command.

Error 361 `Emergency; Internal communication failure detected by CPU`

Internal communication failure was detected by CPU. Input switch is opened. For B2985B/B2987B, voltage output changed to 0 V and the switch is opened. Execute the *TST? command.

Error 362 `Emergency; Internal command queue overflow detected`

Internal command queue overflow was detected. Input switch is opened. For B2985B/B2987B, voltage output changed to 0 V and the switch is opened. Execute the *TST? command.

Error 363 `Emergency; Sense data was not received for acquire trigger`

Sense data was not received for acquire trigger. Input switch is opened. For B2985B/B2987B, voltage output changed to 0 V and the switch is opened. Execute the *TST? command.

Error 364 `Emergency; Unexpected sense data was received`

Unexpected sense data was received. Input switch is opened. For B2985B/B2987B, voltage output changed to 0 V and the switch is opened. Execute the *TST? command.

Error 365 `Emergency; Sense data was not received in Timer period`

Data communication failure. Input switch is opened. For B2985B/B2987B, voltage output changed to 0 V and the switch is opened. Execute the *TST? command.

Error 366 `Emergency; Timestamp FIFO overflow detected`

Data communication failure. Input switch is opened. For B2985B/B2987B, voltage output changed to 0 V and the switch is opened. Execute the *TST? command.

Error 600 `Some or all licenses from license file(s) could not be installed`

Error 700 `ProgramMemory; Program size overflow`

Program memory cannot save the program. Reduce program size.

Error 701 `ProgramMemory; Invalid variable`

Appropriate variable name must be specified.

Error 702 `ProgramMemory; Invalid variable number`

Appropriate variable name must be specified.

Error 703 `ProgramMemory; Query command is not supported`

Memory program cannot contain query command.

Error 704 `ProgramMemory; Program is not selected`

Appropriate program name must be specified.

Error 705 `ProgramMemory; Cannot execute program while another program is running`

Another program is running. Execute the program after it is stopped.

Error 706 `ProgramMemory; Cannot execute program while this program is running`

This program is running. Execute the program after it is stopped.

Error 707 `ProgramMemory; Cannot step program while program is running`

Program is running. Step execution is effective when program is paused or stopped.

Error 708 `ProgramMemory; Cannot continue program while program is running`

Program is running. Program continue is effective when program is paused.

Error 709 `ProgramMemory; Cannot continue program while program is stopped`

Program is stopped. Program continue is effective when program is paused.

Error 710 `ProgramMemory; Program line is too long`

Program memory cannot save the program. Reduce program line.

Error 711 `ProgramMemory; Variable length is too long`

Variable contains too many data. Reduce variable length.

Error 712 `ProgramMemory; Unsupported command is used in program`

Memory program cannot contain the specified command.

Error 713 `ProgramMemory; Cannot set multiple INIT commands in program line`

A program line cannot contain multiple INIT commands.

Error 714 `ProgramMemory; Invalid character in program line`

Program line contains invalid character. Use appropriate characters.

Error 715 `ProgramMemory; Invalid character in program name`

Appropriate program name must be specified.

Error 716 `ProgramMemory; Program count overflow`

Program memory cannot save the program. Delete dispensable program.

Error 801 `Calculate; Expression list full`

Cannot save the expression. Delete dispensable expression.

Error 802 `Calculate; Expression cannot be deleted`

Cannot delete the specified expression. Specify erasable expression.

Error 803 `Calculate; Missmatched parenthesis`

Number of open and close parentheses must be the same.

Error 804 `Calculate; Not a number of data handle`

Expression contains invalid floating point number or symbol. Enter appropriate expression. Available symbols are VOLT, CURR, CHAR, RES, TIME, SOUR, TEMP and HUM.

Error 805 `Calculate; Mismatched brackets`

Number of open and close brackets must be the same.

Error 806 `Calculate; Entire expression not parsed`

Expression is not correct. Enter appropriate expression.

Error 807 `Calculate; Not an operator or number`

Expression contains not an operator or not a number. Enter appropriate expression.

Error 811 `Calculate; Error parsing value`

Expression contains invalid floating point number. Enter appropriate expression.

Error 812 `Calculate; Invalid data handle index`

Vector expression contains invalid index value of an array. Enter appropriate expression.

Error 813 `Calculate; Divided by zero`

Denominator must not be zero. Enter appropriate expression.

Error 814 `Calculate; Log of zero`

Expression cannot contain log 0. Enter appropriate expression.

Error 815 `Calculate; Invalid binary format string is used`

Data contains invalid binary format string. Enter appropriate expression.

Error 816 `Calculate; Invalid hex format string is used`

Data contains invalid hex format string. Enter appropriate expression.

Error 817 `Calculate; Invalid channel number is used`

Expression contains invalid channel number. Enter appropriate expression.

Error 818 `Calculate; Null expression`

Expression is not defined. Enter appropriate expression.

Error 819 `Calculate; Null expression in parentheses`

Expression contains empty parentheses. Enter appropriate expression.

Error 820 `Calculate; Null expression in brackets`

Expression contains empty brackets. Enter appropriate expression.

Error 822 `Calculate; Missmatched trigger counts`

Trigger count of grouped channels must be the same.

Error 823 `Calculate; Missmatched vector lengths`

Vector length of grouped channels must be the same.

Error 824 `Calculate; Invalid character in math name`

Appropriate math expression name must be specified.

Error 861 `Trace; Illegal with storage active`

Storage device must be idle to perform the requested operation.

Error 862 `Trace; No trace data`

Trace buffer must contain data to perform the requested operation.

Error 870 `Macro file size error`

Macro file size error. Reduce file size.

Error 871 `Cannot create state data on non-volatile memory`

Error 872 `Cannot create data on non-volatile memory`

Error 873 `Cannot save list sweep data`

Error 900 `Internal system error`

Error 950 `Unsupported parameter`

Specified parameter is not supported by this model.

Error 951 `Unsupported command`

Specified command is not supported by this model.

Edition 1, April 2021

B2980-90130
www.keysight.com

KEYSIGHT
TECHNOLOGIES