# DC Electronic Load

PEL-3000AE

---

**PROGRAMMING MANUAL**

**GW INSTEK**

# Table of Contents

# INTERFACE OVERVIEW

This chapter describes basic configuration of IEEE488.2 based remote control.

# Front Panel Overview



Air inlet  LCD Display  Function keys  Power key

Main/Local
FUNC/File
Help/Utility
Short
Load On/Off
Scroll wheel

Input terminals  Sense-, Sense+ terminals  USB Port, Preset, A/B and Shift keys  Number pad, Clear/ Lock and Enter keys

# Rear Panel Overview



J1 Frame control ports (Analog control connector)  Trigger out port  Trigger in port

RS232/RS485 port
LAN port

USB device port

GPIB port (optional)  Power socket and switch  Exhaust fan

# Interface Configuration

## Configure RS232/RS485

| RS232C Configuration | Connector | RJ-45 |
|---|---|---|
| | Baud Rate | 2400/ 4800/ 9600/ 19200/ 38400/ 57600/ 115200 |
| | Data Bits | 7bits/ 8bits |
| | Stop Bit | 1bit/ 2bits |
| | Parity | None/ Odd/ Even |

| Operation | 1. Connect an RS232 or RS485 series cable from the PC to the Remote IN port on the real panel. |  |
|---|---|---|
| | Connect the other end of the cable to the PC. | |
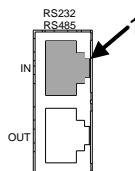
Operation

Utility

1. Press ( Shift ) + ( Help ) > *Interface[F3]* and set the *Interface* setting to *RS232 or RS485*.

2. Set the *Baud Rate*, *Stop Bit* and *Parity* settings.

## Set the UART settings

Overview

The PEL-3000AE series uses the IN & OUT ports for UART communication coupled with RS232 (GW Instek Part number: GTL-259) or RS485 adapters (GW Instek part number: GTL-260).

The pin outs for the adapters are shown below.

RS232 cable with DB9 & RJ-45 shielded connectors from GTL-259 connection kit

| DB-9 Connector | | Remote IN Port | | Remarks |
|---|---|---|---|---|
| Pin No. | Name | Pin No. | Name | |
| Housing | Shield | Housing | Shield | |
| 2 | RX | 7 | TX | Twisted pair |
| 3 | TX | 8 | RX | |
| 5 | SG | 1 | SG | |



Connection diagram



GTL-259
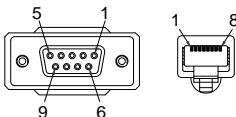
RS485 cable with DB9 & RJ-45 shielded connectors from GTL-260 connection kit

| DB-9 Connector | | Remote IN Port | | Remarks |
|---|---|---|---|---|
| Pin No. | Name | Pin No. | Name | |
| Housing | Shield | Housing | Shield | |
| 9 | TXD - | 6 | RXD - | Twisted pair |
| 8 | TXD + | 3 | RXD + | |
| 1 | SG | 1 | SG | |
| 5 | RXD - | 5 | TXD - | Twisted pair |
| 4 | RXD + | 4 | TXD + | |

**5**
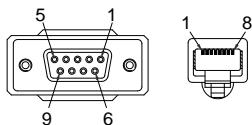
| Connection diagram |  |

GTL-260

| Diagram of Intermediate connector |  |
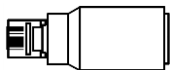
Intermediate connector from GTL-259 or GTL-260 connection kit.

| Intermediate connector | | | | | |
|---|---|---|---|---|---|
| 8 Pin (Male) | | | 8 Pin (Female) | | |
| Pin No. | Name | | Pin No. | Name | Remarks |
| Housing | Shield | ↔ | Case | Shield | |
| 1 | SG | ↔ | 1 | SG | |
| 6 | TXD - | ↔ | 6 | TXD - | Internal paralleled by 120 ohm |
| 3 | TXD + | ↔ | 3 | TXD + | |
| 5 | RXD - | ↔ | 5 | RXD - | Internal paralleled by 120 ohm |
| 4 | RXD + | ↔ | 4 | RXD + | |

| Diagram of End terminal connector |  |

End terminal connector from GTL-259 or GTL-260 connection kit.

| End terminal connector | |
|---|---|
| 8 Pin Connector | |
| Pin No. | Remarks |
| 3 | Internal shorted |
| 7 | |
| 4 | Internal shorted |
| 8 | |

## Multiple Unit Connection

The PEL-3000A/AE can have up to 16 units daisy-chained together using the 8 pin connectors (IN OUT ports) on the rear panel. The first unit in the chain is remotely connected to a PC using RS485. Each subsequent unit is daisy-chained to the next using a RS485 local bus. The OUT port of the first unit must be connected to intermediate connector and the OUT port of the last unit must be connected to end terminal connector.



Each unit is assigned a unique address and can then be individually controlled from the host PC.

Operation
1. Connect the first unit's IN port to a PC using RS485 cable with DB9 & RJ-45 connectors.

2. Plug in intermediate connector to the OUT port on the first unit then using the slave serial link cable (black plug) to connect intermediate connector to the IN port of the second unit.

   Terminate the OUT port of the last unit with the end terminal connector included in the GTL-260 connection kit.
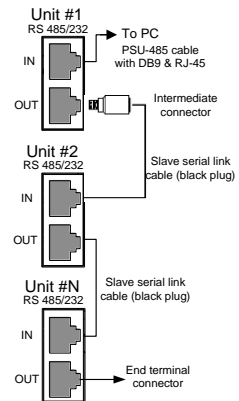
3. Power up all units.

4. Press ( Shift ) + ( Help )^Utility > *Interface[F3]* and set the *Interface* setting to UART> *Mode* and set the Mode to *RS485*.

5. Set the addresses and mode of all units using UART menu. It must be a unique address identifier and mode select is RS485.

| 11/May/2021 | RS485 LOAD |
|---|---|
| Mode | RS485 |
| Baud Rate | 19200 |
| Date Bit | 8 Bit |
| Stop Bit | 1 |
| Address | 01 |

| System Info | Load | Interface | Time Set | Other |
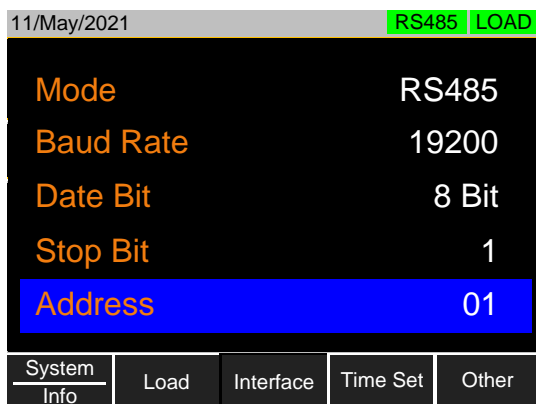
6. Multiple units can be operated using SCPI commands now. See the programming manual or see the function check below for usage details.

## RS232 or RS485/USB Remote Control Function Check

| | |
|---|---|
| Functionality check | Invoke a terminal application such as Realterm. |
| | For RS-232C, set the COM port, baud rate, stop bit, data bit and parity accordingly. |
| | The USB connection emulates a COM port on the PC. To check the COM settings in Windows, see the Device Manager. For example, for Win 7 go to the Control panel → Hardware and Sound → Device Manager. |
| ⚠ Note | If you are not familiar with using a terminal application to send/receive remote commands from the serial port or via a USB connection, please page 15 (Using Realterm to Establish a Remote Connection) for more information. |
| | Run this query command via the terminal after the instrument has been configured for RS-232C/USB remote control (page 4). |
| | *idn? |
| | This should return the Manufacturer, Model number, Serial number, and Firmware version in the following format. |
| | • *GW-INSTEK,PEL-303XAE, XXXXXXXXXXXX, V.X.X.X.X* |
| | Manufacturer: GW-INSTEK |
| | Model number : PEL-303XAE |
| | Serial number : XXXXXXXXXXXX |
| | Firmware version : V.X.X.X |
| ⚠ Note | For further details, please see the programming manual, available on the GW Instek web site @ www.gwinstek.com. |

## Using Realterm to Establish a Remote Connection

| | |
|---|---|
| Background | Realterm is a terminal program that can be used to communicate with a device attached to the serial port of a PC or via an emulated serial port via USB. |
| | The following instructions apply to version 2.0.0.70. Even though Realterm is used as an example to establish a remote connection, any terminal program can be used that has similar functionality. |

<div></div>

| | |
|---|---|
| ⚠ Note | Realterm can be downloaded on Sourceforge.net free of charge. |
| | For more information please see http://realterm.sourceforge.net/ |

| | |
|---|---|
| Operation | 1. Download Realterm and install according to the instructions on the Realterm website. |
| | 2. Connect the PEL-3000AE via USB (page 4) or via RS232 (page 4). |
| | 3. If using RS232, make note of the configured baud rate, stop bits and parity. |
| | 4. Go to the Windows device manager and find the COM port number for the connection. For example, go to the Start menu > Control Panel > Hardware and Sound >Device Manager |
| | Double click the *Ports* icon to reveal the connected serial port devices and the COM port for the each connected device. |
| | If using USB, the baud rate, stop bit and parity settings can be viewed by right-clicking connected device and selecting the *Properties* option. |
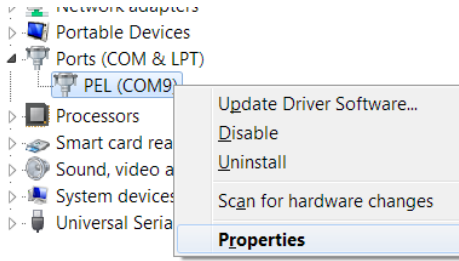
5.  Start Realterm on the PC as an administrator.
    Click:
     Start menu>All Programs>RealTerm>realterm

    Tip: to run as an administrator, you can right
    click the Realterm icon in the Windows Start
    menu and select the *Run as Administrator*
    option.

6.  After Realterm has started, click on the *Port*
    tab.

    Enter the *Baud*, *Parity*, *Data bits*, *Stop bits* and
    *Port* number configuration for the connection.

    The *Hardware Flow Control*, *Software Flow
    Control* options can be left at the default
    settings.

7.  Press *Open* to connect to the PEL-3000AE.
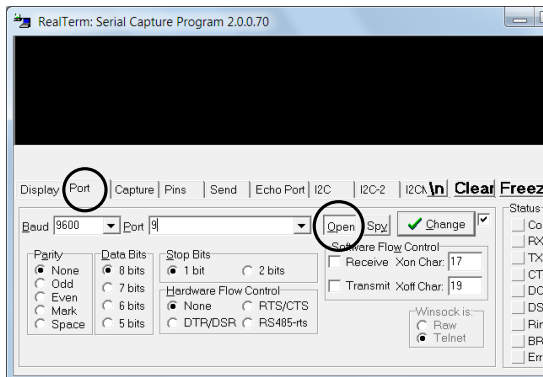
8. Click on the *Send* tab.

   In the *EOL* configuration, check on the *+CR* and *+LF* check boxes.

   Enter the query:
   *\*idn?*

   Click on *Send ASCII*.
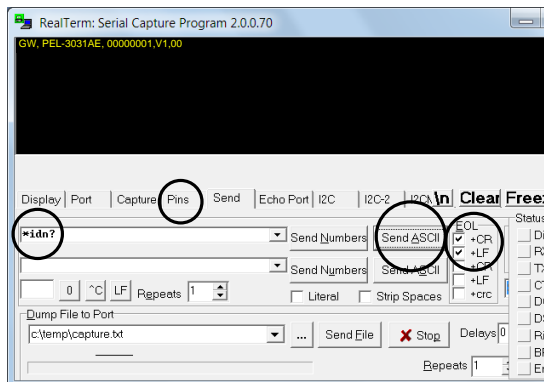


9. The terminal display will return the following:

   *GW,PEL-303XAE, XXXXXXXXXXXX, V.X.X.X.X*

   (manufacturer, model, serial number, version)

10. If Realterm fails to connect to the PEL-3000AE, please check all the cables and settings and try again.

## Configure to USB Remote Interface

| USB configuration | PC side connector | Type A, host |
|---|---|---|
| | PEL-3000AE side connector | Rear panel Type B, slave |
| | Speed | 2.0 (full speed) |
| | USB Class | USB CDC ACM |

| ⚠ Note | Before USB can be used for remote control, it is necessary to install the PEL-3000AE USB device driver, located on the accompanying User Manual CD. |
|---|---|
| | Supported OS: <br> 32 bit(x86): Windows 2000/XP/Vista/7/8 <br> 64 bit(x64): Windows XP/Vista/7/8 |

| Operation | 1. Connect the USB cable to the rear panel USB B port. |
|---|---|
| | Utility <br> 2. Press ( Shift ) + ( Help ) > *Interface[F3]* and set the *Interface* setting to *USB*. |

## Configure GPIB Interface

To use GPIB, the optional GPIB port must be installed. See the user manual for installation details.

| Operation | 1. Ensure the PEL-3000AE is off before proceeding. |
|---|---|
| | 2. Connect a GPIB cable from a GPIB controller to the GPIB port on the PEL-3000AE. |
| | 3. Turn the PEL-3000AE on. |
| | Utility <br> 4. Press ( Shift ) + ( Help ) > *Interface[F3]* and set the *Interface* setting to *GPIB*. |

5. Set the GPIB address.

GPIB address      0~30

| GPIB constraints | • *Maximum 15 devices altogether, 20m cable length, 2m between each device*<br>• *Unique address assigned to each device*<br>• *At least 2/3 of the devices turned On*<br>• *No loop or parallel connection* |
|---|---|

Pin Assignment
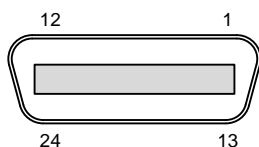


| Pin | Signal | Pin | Signal |
|---|---|---|---|
| 1~4 | Data I/O 1~4 | 13~16 | Data I/O 5~8 |
| 5 | EOI | 17 | REN |
| 6 | DAV | 18 | Ground (DAV) |
| 7 | NRFD | 19 | Ground (NRFD) |
| 8 | NDAC | 20 | Ground (NDAC) |
| 9 | IFC | 21 | Ground (IFC) |
| 10 | SRQ | 22 | Ground (SRQ) |
| 11 | ATN | 23 | Ground (ATN) |
| 12 | SHIELD Ground | 24 | Single GND |

## USB Remote Control Function Check

| Functionality check | Invoke a terminal application such as Realterm.<br><br>The USB connection emulates a COM port on the PC. To check the COM port settings in Windows, see the Device Manager. For example, for Win 7 go to the Control panel → Hardware and Sound → Device Manager. |
|---|---|
| ⚠ Note | If you are not familiar with using a terminal application to send/receive remote commands from a serial port or via a USB connection, please see page 15 (Using Realterm to Establish a Remote Connection) for more information. |

Run this query command via the terminal after the instrument has been configured for
USB remote control (page 4).

*idn?

This should return the Manufacturer, Model number, Serial number, and Firmware version in the following format.

- *GW*,PEL-303XAE, *XXXXXXXXXXXX, V.X.X.X.X*

Manufacturer: GW

Model number : PEL-303XAE

Serial number : XXXXXXXXXXXX

Firmware version : V.X.X.X

⚠️ Note  For further details, please see the programming manual, available on the GW Instek web site @ www.gwinstek.com.

## Using Realterm to Establish a Remote Connection

Background  Realterm is a terminal program that can be used to communicate with a device attached to the serial port of a PC or via an emulated serial port via USB.

The following instructions apply to version 2.0.0.70. Even though Realterm is used as an example to establish a remote connection, any terminal program can be used that has similar functionality.

⚠️ Note  Realterm can be downloaded on Sourceforge.net free of charge.

For more information please see http://realterm.sourceforge.net/

Operation

1. Download Realterm and install according to the instructions on the Realterm website.

2. Connect the PEL-3000AE via USB (page 4).

3. Go to the Windows device manager and find the COM port number for the connection. For example, go to the Start menu > Control Panel > Hardware and Sound >Device Manager.

   Double click the *Ports* icon to reveal the connected serial port devices and the COM port for the each connected device.

   If using USB, the baud rate, stop bit and parity settings can be viewed by right-clicking connected device and selecting the *Properties* option.
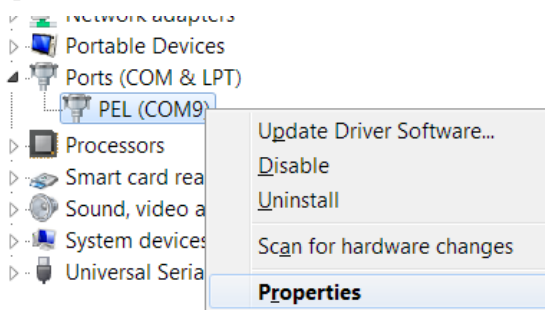


4. Start Realterm on the PC as an administrator. Click:
    Start menu>All Programs>RealTerm>realterm

   Tip: to run as an administrator, you can right click the Realterm icon in the Windows Start menu and select the *Run as Administrator* option.

5. After Realterm has started, click on the *Port* tab.

   Enter the *Baud*, *Parity*, *Data bits*, *Stop bits* and *Port* number configuration for the connection.

The *Hardware Flow Control*, *Software Flow Control* options can be left at the default settings.

Press *Open* to connect to the PEL-3000AE.
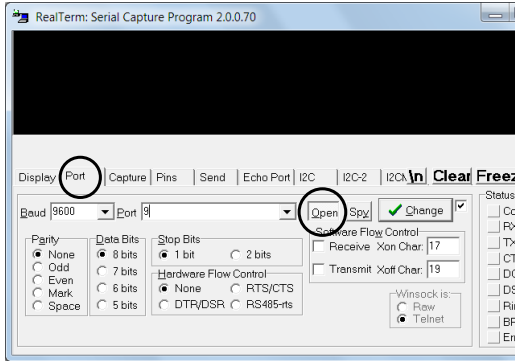


6.  Click on the *Send* tab.

    In the *EOL* configuration, check on the *+CR* and *+LF* check boxes.

    Enter the query:
    *\*idn?*

    Click on *Send ASCII*.
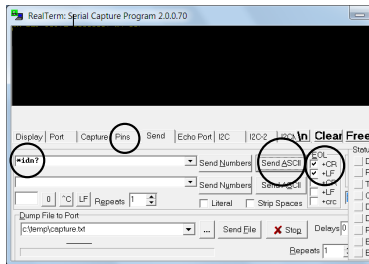


7.  The terminal display will return the following:

    *GW,PEL-303XAE, XXXXXXXXXXXX, V.X.X.X.X*

    (manufacturer, model, serial number, version)

8. If Realterm fails to connect to the PEL-3000AE, please check all the cables and settings and try again.

## GPIB Function Check

| | |
|---|---|
| Functionality check | Please use the National Instruments Measurement & Automation Controller software to confirm GPIB/LAN functionality.<br><br>See the National Instrument website, http://www.ni.com for details. |
| ⚠ Note | For further details, please see the programming manual, available on the GW Instek web site @ www.gwinstek.com. |
| Operation | 1. Start the NI Measurement and Automation Explorer (MAX) program. Using Windows, press:<br><br>*Start>All Programs>National Instruments>Measurement & Automation* |
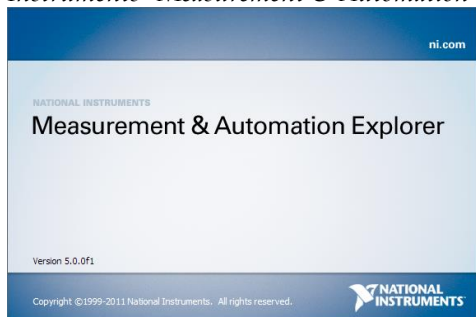


2. From the Configuration panel access;

   My System>Devices and Interfaces>GPIB0

3. Press the *Scan for Instruments* button.

4. In the *Connected Instruments* panel the PEL-3000AE should be detected as *Instrument 0* with

the address the same as that configured on the PEL-3000AE.

5.  Double click the *Instrument 0* icon.
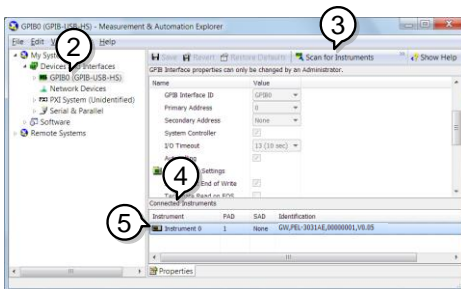


6.  Click on the Attributes tab at the bottom of the screen.

7.  Click on *Communicate with Instrument*.

8.  In the *NI-488.2 Communicator* window, ensure *\*IND?* is written in the *Send String*: text box.

    Click on the *Query* button to send the *\*IDN?* query to the instrument.

9.  The *String Received* text box will display the query return:

    *GW*,PEL-303XAE, *XXXXXXXXXXXX, V.X.X.X.X*

    (manufacturer, model, serial number, version)
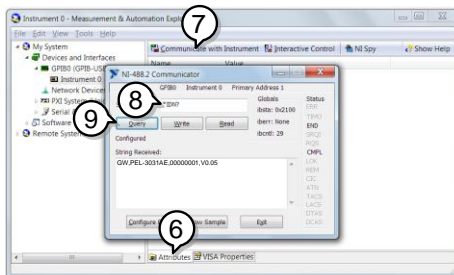


10. The function check is complete.

# COMMAND OVERVIEW

The Command overview chapter lists all PEL-3000AE commands in functional order as well as alphabetical order. The command syntax section shows you the basic syntax rules you have to apply when using commands.

# Command Syntax

| Compatible Standard | IEEE488.2 | Partial compatibility |
|---|---|---|
| | SCPI, 1999 | Partial compatibility |

| Command Structure | SCPI (Standard Commands for Programmable Instruments) commands follow a tree-like structure, organized into nodes. Each level of the command tree is a node. Each keyword in a SCPI command represents each node in the command tree. Each keyword (node) of a SCPI command is separated by a colon (:). |
|---|---|

For example, the diagram below shows an SCPI sub-structure and a command example.

```
┌─────────────────┐
│ [:MODE]:CRANge  │
└─────────────────┘

         :MODE
    ┌──────┼──────────┐
:CRANge  :VRANge  :DYNamic
```
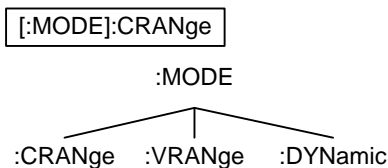
| Command types | There are a number of different instrument commands and queries. A command sends instructions or data to the unit and a query receives data or status information from the unit. |
|---|---|

Command types

| Simple | A single command with/without a parameter |
|---|---|
| Example | :CONFigure:SHORt HOLD |

| | |
|---|---|
| Query | A query is a simple or compound command followed by a question mark (?). A parameter (data) is returned. |
| Example | :CONFigure:SHORt? |
| Compound | Two or more commands on the same command line. Compound commands are separated with either a semi-colon (;) or a semi-colon and a colon (;:). |
| | A semi-colon is used to join two related commands, with the caveat that the last command must begin at the last node of the first command. |
| | A semi-colon and colon are used to combine two commands from different nodes. |
| Example | CONFigure:VON MAX;:CONFigure:VDELay MIN |

Command Forms    Commands and queries have two different forms, long and short. The command syntax is written with the short form of the command in capitals and the remainder (long form) in lower case.

The commands can be written in capitals or lower-case, just so long as the short or long forms are complete. An incomplete command will not be recognized.

Below are examples of correctly written commands.

| | |
|---|---|
| Long form | :CURRent:LEVel?<br>:CURRENT:LEVEL?<br>:current:level? |
| Short form | :CURR:LEV?<br>:curr:lev? |

| | |
|---|---|
| Square Brackets | Commands that contain square brackets indicate that the contents are optional. The function of the command is the same with or without the square bracketed items, as shown below<br><br>For example for the query:<br>"[:CONFigure]:GNG [:PASS]?"<br><br>Both ":CONFigure:GNG:PASS?" and ":GNG?" are both valid forms. |

| | |
|---|---|
| Command Format | :CURRent:Set   1.00A<br><br>     1     2 3  4<br><br>1.  Command header   3.  Parameter 1<br>2.  Space               4.  Unit or suffix. |

| Common Unit Parameters | Type | Description | Example |
|---|---|---|---|
| | <Boolean> | boolean logic | 0, 1 |
| | <NR1> | integers | 0, 1, 2, 3 |
| | <NR2> | decimal numbers | 0.1, 3.14, 8.5 |
| | <NR3> | floating point | 4.5e-1, 8.25e+1 |
| | <NRf> | any of NR1, 2, 3 | 1, 1.5, 4.5e-1 |

| [MIN] (Optional parameter) | For commands, this will set the setting to the lowest value. This parameter can be used in place of any numerical parameter where indicated. |
|---|---|
| | For queries, it will return the lowest possible value allowed for the particular setting. |
| [MAX] (Optional parameter) | For commands, this will set the setting to the highest value. This parameter can be used in place of any numerical parameter where indicated. |
| | For queries, it will return the highest possible value allowed for the particular setting. |
| Unit Suffixes (Optional parameters) | Unit suffixes can be optionally used with most NRf type input parameters. |
| [A] | Amps | 1.00A |
| [%] | Percentage | 10% |
| [V] | Volts | 5.00V |
| [W] | Watts | 3.00W |
| [ms] | milliseconds | 20ms |
| [mV] | Millivolts | 150mV |
| [s] | Seconds | 5s |
| [mS] | Reciprocal of 1000 ohms | 20mS |
| [OHM] | Ohm | 50OHM |
| [mA/us] | Millamps/ microsecond | 100mA/us |
| [Hz] | Hertz | 1000Hz |

| ⚠ Note | For [OHM] return values, an infinite resistance (open) will be returned as $9.9e^{37}$. |
|---|---|

| Message Terminator | LF | Line feed code (0x0A) |
|---|---|---|
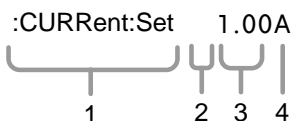
# Command List

## Common Commands

## *CLS

Set →

| | |
|---|---|
| Description | Clears the error queue. |
| Syntax | *CLS |

## *ESE

Set →
→ Query
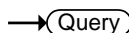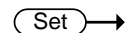
| | |
|---|---|
| Description | Queries or sets the Standard Event Status Enable register. The Standard Event Status Enable register determines which events can set the Event Summary bit (ESB) in the Status Byte Register. Any bits that are set to 1 enable the corresponding event. Each event is represented by a bit in the Standard Event Status Enable register. |
| Syntax | *ESE <NR1> |
| Query Syntax | *ESE? |

| Parameter/ | <NR1> (bit weight) | Bit number / Description |
|---|---|---|
| Return parameter | 1 | 0/ OPC |
| | 4 | 2/ QYE |
| | 8 | 3/ DDE |

| 16 | 4/ EXE |
| 32 | 5/ CME |

| Example | *ESE 8 |
|---|---|
| | Sets bit 3 of the ESE register. |

| Query example | *ESE? |
|---|---|
| | >12 |
| | Bits 2 and 3 are set in the Standard Event Status Enable register. |

## *ESR?                                    → Query

| Description | Reads the Standard Event Status register. This command will also clear the Standard Event Status register. |
|---|---|

| Query Syntax | *ESR? |
|---|---|

| Return parameter | <NR1> (bit weight) | Bit number / Description |
|---|---|---|
| | 4 | 3/ QYE |
| | 8 | 4/ DDE |
| | 16 | 5/ EXE |
| | 32 | 6/ CME |

| Query example | *ESR? |
|---|---|
| | >48 |
| | Bits 4 and 5 are set in the Standard Event register. |

## *IDN?                                    → Query

| Description | Queries the manufacturer, model number, serial number, and firmware version of the instrument. |
|---|---|

| Query Syntax | *IDN? |
|---|---|

| Return parameter | <string> | Returns the instrument identification as a string in the following format: |
|---|---|---|
| | | GW-INSTEK, PEL-303XAE, XXXXXXXXXXXX, V.X.X.X |
| | | Manufacturer: GW-INSTEK |
| | | Model number : PEL-303XAE |
| | | Serial number : XXXXXXXX<br>Firmware version : V.X.X.X |

## *OPC

Set ⟶
⟶ Query

| Description | This command sets the OPC (Operation Command Bit) bit (bit 0) of the Standard Event Status Register after the instrument has completed all pending operations. The query will return the status of the OPC bit. |
|---|---|
| Syntax | *OPC |
| Query Syntax | *OPC? |
| Return parameter | <NR1> | |
| | 1 | Operation complete |
| Query Example | *OPC?<br>>1 | |
| | Indicates that all pending operations are complete. | |

## *RCL

Set ⟶

| Description | The Recall Instrument State command restores the instrument settings from a previously saved memory setting. |
|---|---|
| Syntax | *RCL <NR1> |
| Parameter | <NR1> | |
| | 1~256 | Memory number 1 to 256 |
| Example | *RCL 20 | |
| | Recall setting memory 20. | |

## *RST                                    Set ⟶

| | |
|---|---|
| Description | Resets the unit. This is command forces the ABORt, and *CLS |
| Syntax | *RST |

## *SAV                                    Set ⟶

| | |
|---|---|
| Description | The Save Instrument State command saves the instrument settings to one of the memory setting slots. |
| Syntax | *SAV <NR1> |
| Parameter | <NR1> |
| | 1~256 | Memory number 1 to 256 |
| Example | *SAV 20 |
| | Saves the current setting to memory 20. |

## *SRE                                    Set ⟶
                                          ⟶ Query

| | |
|---|---|
| Description | Queries or sets the Service Request Enable register. The Service Request Enable register determines which events in the Status Byte register can set the Master Summary bit (MSB) in the Status Byte Register. Any bits that are set to 1 will cause the MSS bit to be set. |
| Syntax | *SRE <NR1> |
| Query Syntax | *SRE? |

| Parameter/ Return parameter | <NR1> (bit weight) | Bit number / Description |
|---|---|---|
| | 1 | 0/Not used |
| | 2 | 1/ERR |
| | 4 | 2/ CSUM |
| | 8 | 3/ QUES |
| | 16 | 4/ MAV |
| | 32 | 5/ ESB |
| | 64 | 6/ RQS_MSS |

| 128 | 7/ OPER |
|-----|---------|

| ⚠ Note | Bit 1 and 2 cannot be set, however bit 1 (ERR) can be returned. |
|---|---|

| Example | *SRE 8 |
|---|---|
| | Sets bit 3 of the Service Request Enable register. |

| Query example | *SRE? |
|---|---|
| | >12 |
| | Bits 2 and 3 are set in the Service Request Enable register. |

## *STB?                                                    → Query

| Description | Reads the Status Byte register. This command will not clear the Status Byte register. |
|---|---|
| | If the Master Summary Status bit (MSS) is set, it indicates that there is a reason for a service request. |

| Query Syntax | *STB? |
|---|---|

| Return parameter | <NR1> (bit weight) | Bit number / Description |
|---|---|---|
| | 1 | 0/Not used |
| | 2 | 1/ERR |
| | 4 | 2/ CSUM |
| | 8 | 3/ QUES |
| | 16 | 4/ MAV |
| | 32 | 5/ ESB |
| | 64 | 6/ RQS_MSS |
| | 128 | 7/ OPER |

| ⚠ Note | Bit 1 and 2 cannot be set, however bit 2 (ERR) can be returned. |
|---|---|

| Query example | *STB? |
|---|---|
| | >36 |
| | Bits 2 and 5 are set in the Status Byte register. |

## *TRG                                    ⸨ Set ⸩→

| | |
|---|---|
| Description | This command triggers the unit. |
| Syntax | *TRG |

## *TST?                                    →⸨Query⸩

| | |
|---|---|
| Description | This command is a standard SCPI self-test command. The PEL-3000AE does not perform any self-tests so will always return 0 (pass) for this command. |
| Query Syntax | *TST? |
| Return parameter | <NR1> |
| | 0 | Pass |
| Query example | *TST?<br>>0 |

## *WAI                                    ⸨ Set ⸩→

| | |
|---|---|
| Description | Wait command. Prevents new operations from executing until all pending operations have finished. |
| Syntax | *WAI |

# Trigger Commands

## Trigger States

```
                    ┌──────────────┐
                    │  PEL-303XAE  │
                    │  power on    │
                    └──────┬───────┘
                           │      ┌──────────┐
                           │◄─────│ *RST,    │
                           ▼      │ ABORt    │
                    ┌──────────────┐ └──────────┘
          ┌────────►│  Idle state  │
          │         └──────┬───────┘
          │                │      ┌───────────────────────┐
          │                │◄─────│ :INITate[:IMMediate], │
          │                ▼      │ :INITiate:CONTinuous ON│
          │         ┌──────────────┐ └───────────────────────┘
        ┌─┼────────►│ Trigger Wait │
        │ │         │   state      │
        │ │         └──────┬───────┘
        │ │                │      ┌──────────────┐
        │ │                │◄─────│ *TRG,        │
        │ │                ▼      │ manual trigger│
        │ │         ┌──────────────┐ └──────────────┘
        │ │         │  Triggered   │
        │ │         └──────┬───────┘
        │ │                │
        │ │  OFF    ╱────────────╲
        └─┼────────│ :INIT:CONT  │
          │         ╲────────────╱
          │               │ ON
          └───────────────┘
```
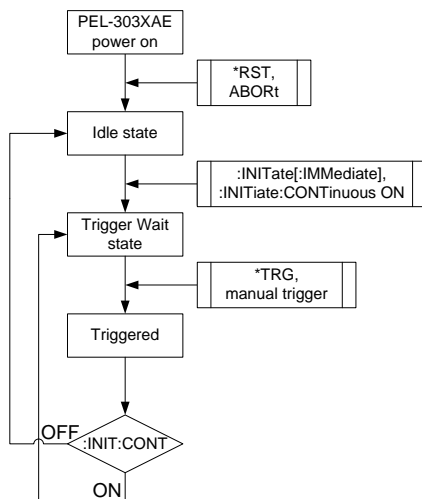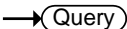
## :ABORt                    ⬭ Set ⟶

| | |
|---|---|
| Description | Clears the trigger wait status and returns to the idle state. |
| Query Syntax | :ABORt |

Set ⟶
:INPut[:STATe]:TRIGgered                        ⟶ Query

| Description | Turn ON/OFF the trigger input. | |
|---|---|---|
| Syntax | :INPut[:STATe]:TRIGgered {<Boolean>|OFF | ON } | |
| Query Syntax | :INPut[:STATe]:TRIGgered? | |
| Parameter | OFF or 0 | Trigger input = OFF |
| | ON or 1 | Trigger input = ON |
| Return parameter | 0 | Trigger input = OFF |
| | 1 | Trigger input = ON |
| Example | :INPut:TRIGgered ON | |
| | Turns on the trigger input. | |
| Query example | :INPut:TRIGgered? | |
| | >0 | |
| | Indicates that the trigger input is turned off. | |

Set ⟶
:INITiate[:IMMediate]

| Description | Sets the trigger to the wait state. If the trigger is activated, the trigger will automatically go to the idle state. |
|---|---|
| Query Syntax | :INITiate[:IMMediate] |

Set ⟶
:INITiate:CONTinuous                            ⟶ Query

| Description | Sets or queries whether the trigger will go back to the wait state or go to the idle state after a trigger. When the :INITiate:CONTinuous function is set to OFF, the trigger state can only be returned to idle if the *RST or ABORt command is used | |
|---|---|---|
| Syntax | :INITiate:CONTinuous {<Boolean>|OFF | ON } | |
| Query Syntax | :INITiate:CONTinuous? | |
| Parameter | OFF or 0 | Continuous = OFF |
| | ON or 1 | Continuous = ON |

| Return parameter | 0 | Continuous = OFF |
| --- | --- | --- |
| | 1 | Continuous = ON |

| Example | :INITiate:CONTinuous ON |
| --- | --- |
| | Configures the trigger to go to the wait state after a trigger. |

| Query example | :INITiate:CONTinuous? |
| --- | --- |
| | >1 |
| | Indicates that the trigger will to go to the wait state after a trigger. |

## :CONDuctance[:VA]:TRIGgered  ⟨ Set ⟩⟶

| Description | Sets the conductance value for when the trigger is activated. |
| --- | --- |
| Syntax | :CONDuctance[:VA]:TRIGgered {<NR2>[ mS ] |MINimum | MAXimum } |
| Parameter | <NR2>[ mS ] | Conductance in millisiemens(mS). |
| | MINIMUM or MIN | MINIMUM value. |
| | MAXIMUM or MAX | MAXIMUM value. |
| Example | :CONDuctance:TRIGgered MAX |
| | Sets the conductance to the maximum value. |

## :CURRent[:VA]:TRIGgered  ⟨ Set ⟩⟶

| Description | Sets the current generated when the trigger is activated. |
| --- | --- |
| Syntax | :CURRent[:VA]:TRIGgered {<NR2>[ A ] |MINimum | MAXimum } |
| Parameter | <NR2>[ A ] | Current in amps. |
| | MINIMUM or MIN | MINIMUM value. |
| | MAXIMUM or MAX | MAXIMUM value. |
| Example | :CURRent:TRIGgered MAX |
| | Sets the current generated to the maximum value. |

## :RESistance[:VA]:TRIGgered     ⟨ Set ⟩⟶

| | |
|---|---|
| Description | Sets the resistance for when the trigger is activated. |
| Syntax | :RESistance[:VA]:TRIGgered {<NR2>[OHM] \|MINimum \| MAXimum } |

| Parameter | <NR2>[OHM] | Resistance in ohms. |
|---|---|---|
| | MINIMUM or MIN | MINIMUM value. |
| | MAXIMUM or MAX | MAXIMUM value. |

| | |
|---|---|
| Example | :RESistance:TRIGgered MAX |
| | Sets the resistance to the maximum value. |

## :TRIGger[:Delay]:Time     ⟨ Set ⟩⟶   ⟶⟨ Query ⟩

| | |
|---|---|
| Description | The command determines how long to delay any action after a trigger is received. Equivalent to using the "Trig In Delay" setting on the front panel. |
| Syntax | :TRIGger[:Delay]:Time <NR2>\|MINimum\|MAXimum |
| Query Syntax | :TRIGger[:Delay]:Time? [MINimum\|MAXimum] |

| Parameter/ | <NR2> | 0 ~ 0.005s (0 ~ 5000µs) |
|---|---|---|
| Return Parameter | MINimum | Minimum delay time |
| | MAXimum | Maximum delay time |

| | |
|---|---|
| Example | :TRIG:T MAX |
| | Sets the trigger in delay to 5ms. |
| Query example | :TRIG:T?<br>>0.0050000 |
| | Returns the delay time in seconds. |

## :TRIGger[:PULSe]:WIDTh     ⟨ Set ⟩⟶   ⟶⟨ Query ⟩

| | |
|---|---|
| Description | Sets the trigger output signal's pulse width. |
| Syntax | :TRIGger[:PULSe]:WIDTh <NR2>\|MINimum\|MAXimum |

| Query Syntax | :TRIGger[:PULSe]:WIDTh? [MINimum|MAXimum] | |
|---|---|---|
| Parameter/ Return Parameter | <NR2> | 0.0000025~0.005s (2.5µs ~ 5000µs) |
| | MINimum | Minimum pulse width |
| | MAXimum | Maximum pulse width |
| Example | :TRIG:WIDT MAX | |
| | Sets the trigger pulse width to 5ms. | |
| Query example | :TRIG:WIDT?<br>>0.0050000 | |
| | Returns the pulse width of the trigger output. | |

## Input State Commands

Set ⟶
⟶ Query

## :INPut

| Description | Sets or queries the status of the load. | |
|---|---|---|
| Syntax | :INPut {<Boolean>|OFF | ON } | |
| Query Syntax | :INPut? | |
| Parameter | OFF or 0 | Load  = OFF |
| | ON or 1 | Load = ON |
| Return parameter | 0 | Load = OFF |
| | 1 | Load = ON |
| Example | :INPut ON | |
| | Turns the load on. | |
| Query example | :INPut? | |
| | >0 | |
| | Indicates that the load is off. | |

Set ⟶
⟶ Query

## :INPut:SHORt

| Description | Shorts or opens the input terminals or queries their status. | |
|---|---|---|
| Syntax | :INPut:SHORt {<Boolean>|OFF | ON } | |
| Query Syntax | :INPutSHORt? | |
| Parameter | OFF or 0 | Short  = OFF |
| | ON or 1 | Short = ON |
| Return parameter | 0 | Short  = OFF |
| | 1 | Short = ON |
| Example | :INPut:SHORt ON | |
| | Shorts the input terminals. | |

| Query example | :INPut:SHORt? |
|---|---|
| | >0 |
| | Indicates that the short function is off. |

Set →
→ Query
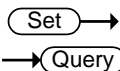
## :INPut:MODE

| Description | Selects the load function mode. |
|---|---|
| Syntax | :INPut:MODE{ LOAD\|PROG\|NSEQ\|FSEQ } |
| Query Syntax | :INPut:MODE? |
| Parameter/ Return parameter | LOAD | Sets the load to generator as a CC/CV/CP/CR load. |
| | PROG | Sets the function mode to program. |
| | NSEQ | Sets the function mode to Normal Sequence mode. |
| | FSEQ | Sets the function mode to Fast Sequence mode. |
| Example | :INPut:MODE PROG |
| | Puts the load generator into Program mode. |
| Query example | :INPut:MODE? |
| | >PROG |
| | Indicates that the current function mode is Program. |

## Measurement Commands

## :MEASure:CURRent?                              ➔ Query

| Description | This command returns the load current. | |
|---|---|---|
| Query Syntax | :MEASure:CURRent? | |
| Return parameter | <NR2> | Load current in amps |
| Query example | :MEASure:CURRent?<br>>0.79860 | |
| | Returns the load current in amps. | |

## :MEASure:VOLTage?                              ➔ Query

| Description | This command returns the load voltage. | |
|---|---|---|
| Query Syntax | :MEASure:VOLTage? | |
| Return parameter | <NR2> | Load voltage in volts |
| Query example | :MEASure:VOLTage?<br>>1.49900 | |
| | Returns the load voltage in volts. | |

## :MEASure:POWer?                                ➔ Query

| Description | This command returns the power. | |
|---|---|---|
| Query Syntax | :MEASure:POWer? | |
| Return parameter | <NR2> | Power in watts |
| Query example | :MEASure:POWer?<br>>1.19695 | |
| | Returns the power in watts. | |

## :MEASure:ETIMe?                    → ( Query )

| | |
|---|---|
| Description | Returns the amount of time the load has been on for in seconds (elapsed time). |
| Query Syntax | :MEASure:ETIMe? |
| Return parameter | <NR2>                    Elapsed time in seconds |
| Query example | :MEASure:ETIMe?<br>>316.0<br><br>Returns the elapsed time in seconds. |

## Fetch Subsystem

## :FETCh:VOLTage?                                    ➔ Query

| Description | This query returns the real-time voltage of the load input. |
|---|---|
| Query syntax | :MODE? |
| Return parameter | <NR2> | Voltage in volts |
| Query example | :FETC:VOLT?<br>>11.2 |

The load has a voltage of 11.2 volts at the input.

## :FETCh:CURRent?                                    ➔ Query

| Description | This query returns the real-time current of the load input. |
|---|---|
| Query syntax | :FETCh:CURRent? <NR2> |
| Return parameter | <NR2> | Current in amps |
| Query example | :FETC:CURR?<br>>1.2 |

The load has a current of 1.2 amps at the input.

## :FETCh:POWer?                                    ➔ Query

| Description | This query returns the real-time power of the load input. |
|---|---|
| Query syntax | :FETCh:POWer? <NR2> |

| Return parameter | <NR2> | Power in watts |
|---|---|---|

| Query example | :FETC:POW?<br>>1.2 |
|---|---|
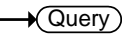
The load is at 1.2 watts.

## Configure Subsystem Commands

Set ⟶
[:CONFigure]:OCP                                ⟶ Query

| Description | Sets or queries the OCP trip settings. The OCP limit can be set to a specific value or the trip setting can be set to either limit the current or to turn the load off. |
|---|---|
| Syntax | [:CONFigure]:OCP {<NRf>[ A ] | MINimum | MAXimum | LIMit | LOFF | OFF } |
| Query Syntax | [:CONFigure]:OCP? |

| Parameter | <NRf>[A] | Current limit value. |
|---|---|---|
| | MINIMUM or MIN | Minimum current limit value. |
| | MAXIMUM or MAX | MAXimum current limit value. |
| | LIMIT or LIM | Limit the load |
| | LOFF | Turn the load off |
| | OFF | Turn the unit off |
| Return parameter | Returns a string with OCP setting followed by the OCP value. | |
| Example1 | :OCP LIM | |
| | Sets the OCP setting to limit. | |
| Example2 | :OCP 63.000 | |
| | Sets the OCP value to 63A. | |
| Query example | :OCP?<br>>LIMIT, 63.000 | |
| | The OCP setting is LIMIT and the OCP value is 64.000A. | |

## [:CONFigure]:OPP

Set →
→ Query
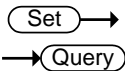
| Description | Sets or queries the OPP trip settings. The OPP limit can be set to a specific value or the trip setting can be set to either limit the power or to turn the load off. | |
|---|---|---|
| Syntax | [:CONFigure]:OPP {<NRf> [W] | MINimum | MAXimum | LIMit | LOFF | OFF } | |
| Query Syntax | [:CONFigure]:OPP? | |
| Parameter | <NRf>[W] | Power limit value. |
| | MINIMUM or MIN | MINIMUM power limit value. |
| | MAXIMUM or MAX | MAXIMUM power limit value. |
| | LIMIT or LIM | Limit the load |
| | LOFF | Turn the load off |
| | OFF | Turn the unit off |

| Return parameter | Returns a string with OPP setting followed by the OPP value. |
|---|---|
| Example1 | :OPP LIMIT |
| | Sets the OCP setting to limit. |
| Example2 | :OPP 10.000 |
| | Sets the OPP value to 10W. |
| Query example | :OPP?<br>>LIMIT, 10.000 |
| | The OPP setting is LIMIT and the OPP value is 10.000W. |

## [:CONFigure]:UVP

Set →
→ Query

| Description | Sets or queries the UVP trip settings. The UVP can also be cleared with this command. |
|---|---|
| Syntax | [:CONFigure]:UVP {<NRf>[ V ] |MINimum | MAXimum } |
| Query Syntax | [:CONFigure]:UVP? |

| Parameter | <NRf>[V] | voltage limit value. |
|---|---|---|
| | MINIMUM or MIN | MINIMUM value. |
| | MAXIMUM or MAX | MAXIMUM value. |

| Return parameter | Returns the UVP level (<NR2>) |
|---|---|
| Example1 | :UVP 10.00 |
| | Sets the UVP setting to 10V. |
| Query example | :UVP?<br>> 10.0000 |
| | The UVP setting is 10.0000V. |

## [:CONFigure]:OVP

Set →
→ Query

| Description | Sets or queries the OVP trip settings. The OVP can also be cleared with this command. |
|---|---|

| Syntax | [:CONFigure]:OVP {<NRf>[ V ] | MINimum| MAXimum} | |
|---|---|---|
| Query Syntax | [:CONFigure]:OVP? | |
| Parameter | <NRf>[V] | voltage limit value. |
| | MINIMUM or MIN | MINIMUM value. |
| | MAXIMUM or MAX | MAXIMUM value. |
| Return parameter | Returns the OVP level (<NR2>). | |
| Example1 | :OVP 10.00 | |
| | Sets the OVP setting to 10V. | |
| Query example | :OVP? | |
| | > 10.0000 | |
| | The OVP setting is 10.0000V. | |

Set ⟶

⟶ Query
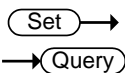
# [:CONFigure]:SSTart

| Description | Sets or queries the Soft Start time setting. | |
|---|---|---|
| Syntax | [:CONFigure]:SSTart {<NRf>[s] | MINimum| MAXimum | OFF } | |
| Query Syntax | [:CONFigure]:SSTart? | |
| Parameter | <NRf>[s] | The soft start time in seconds. |
| | MINIMUM or MIN | Minimum time = 0 second |
| | MAXIMUM or MAX | Maximum time |
| | OFF | OFF = 0 second |
| Return parameter | <NR2> | Returns the soft start time in seconds. |
| | OFF | Off |
| Example | :SSTart OFF | |
| | Turns the soft start function off. | |
| Query example | :SSTart? | |
| | >OFF | |
| | The soft start function is off. | |

Set →
→ Query

## [:CONFigure]:VON

| Description | Sets or queries the Von voltage settings. | |
|---|---|---|
| Syntax | [:CONFigure]:VON {<NRf>[ V ] \| MINimum\| MAXimum \| LON \| LOFF } | |
| Query Syntax | [:CONFigure]:VON? | |
| Parameter | {<NRf>[ V ] | The Von voltage level (default unit is V) |
| | MINIMUM or MIN | Minimum Von voltage level |
| | MAXIMUM or MAX | Maximum Von voltage level |
| | LON | Latch on |
| | LOFF | Latch off |
| Return parameter | <ASCII string> | Returns the Von latch settings. |
| Example | :VON 10.0V | |
| | Sets the Von voltage to 10.0 volts. | |
| Query example | :VON?<br> >Latch OFF, 10.000 | |
| | The Von voltage level is 10V. | |

Set →
→ Query

## [:CONFigure]:VDELay

| Description | Sets or queries the Von Delay settings in seconds. | |
|---|---|---|
| Syntax | [:CONFigure]:VDELay {<NRf>[s] \| MINimum \| MAXimum \| OFF } | |
| Query Syntax | [:CONFigure]:VDELay? | |
| Parameter | <NRf>[s] | The delay time in seconds |
| | OFF | Disable the delay time |
| | MINIMUM or MIN | Minimum delay time |
| | MAXIMUM or MAX | Maximum delay time |
| Return parameter | <NR2> | Returns the delay time in seconds |

| Example | :VDELay 1.5 ms |
|---|---|
| | Sets the delay time to 1.5ms. |
| | :VDELay 0.0015 s |
| | Sets the delay time to 1.5ms. |
| Query example | :VDELay?<br> >0.0015 |
| | The delay time is 1.5ms. |

Set →

## [:CONFigure]:CNTime
→ Query
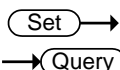
| Description | Turns the Count Time timer function on or off. |
|---|---|
| Syntax | [:CONFigure]:CNTime {<Boolean> |OFF | ON } |
| Query Syntax | [:CONFigure]:CNTime? |
| Parameter/<br>Return parameter | OFF or 0 | Turns the Count Time timer off. |
| | ON or 1 | Turns the Count Time timer on |
| Example | [:CONFigure]:CNTime ON |
| | Turns the Count Time timer on. |
| Query example | [:CONFigure]:CNTime?<br> >ON |
| | Count Time is turned on. |

Set →
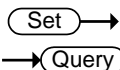
## [:CONFigure]:COTime
→ Query

| Description | Sets or queries the load cutoff time. A cutoff time of 0 seconds is the equivalent of disabling the cutoff time. |
|---|---|
| Syntax | [:CONFigure]:COTime {<NRf>[s] | MINimum | MAXimum | OFF } |
| Query Syntax | [:CONFigure]:COTime? |
| Parameter | <NRf>[s] | Cut off time in seconds (1~3599999) |
| | OFF | Turns the cutoff time off. |

| | MINIMUM or MIN | Sets the cutoff time to the maximum |
| | MAXIMUM or MAX | Sets the cutoff time to the minimum |
| Return parameter | <NR1> | Returns the cutoff time |
| Example | :COTime MAX | |
| | Sets the cutoff time to the maximum. | |
| Query example | :COTime? >3599999 | |
| | The cutoff time is set to 3599999 seconds. | |

Set ⟶

[:CONFigure]:CRUNit          ⟶ Query

| Description | Sets or queries the CR mode setting units. | |
| Syntax | [:CONFigure]:CRUNit {OHM\|MHO} | |
| Query Syntax | [:CONFigure]:CRUNit? | |
| Parameter/ Return parameter | OHM MHO | Set the units to ohms. Set the units to mho (conductance) |
| Example | :CRUNit OHM | |
| | Sets the CR mode units to ohms. | |
| Query example | :CRUNit? >OHM | |
| | The CR mode units are ohms. | |

Set ⟶

:CONFigure:DYNamic          ⟶ Query

| Description | Sets the display units for when dynamic mode switching is used. Units can be selected from Percent and Value. See the user manual for further details. This command will also configure whether to switch between each level using timers or a set duty cycle. |
| Syntax | :CONFigure:DYNamic { VALue \| PERCent \| TIME \| FDUTy } |

| Query Syntax | :CONFigure:DYNamic? | |
|---|---|---|
| Parameter | VALUE or VAL | Set the units to Value. |
| | PERCENT or PERC | Set the units to Percent. |
| | TIME | Use timers for timing. |
| | FDUTY or FUDT | Use duty cycle for timing. |
| Return parameter | <ASCII string> | Return a string containing the unit mode and the timing mode. |
| Example | :CONFigure:DYNamic VALue | |
| | Sets the dynamic mode units to value. | |
| Query example | :CONFigue:DYNamic? | |
| | > Value,T1,T2 | |
| | The dynamic mode becomes a value setup and timer setup. | |

## :CONFigure:MEMory

Set →
→ Query
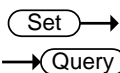
| Description | This command configures the how the files are recalled *in local mode* (using the front panel interface). By default when you try to recall a file or setting from memory, a message will appear asking you to press the Enter key to confirm each time you wish to recall. This command enables (SAFety) or disables this feature (DIRect). | |
|---|---|---|
| Syntax | :CONFigure:MEMory {SAFety \| DIRect } | |
| Query Syntax | :CONFigure:MEMory? | |
| Parameter | SAFETY or SAF | Safety setting. |
| | DIRECT or DIR | Directly recall the chosen file. |
| Return parameter | Safety | Safety setting. |
| | Direct | Directly setting. |
| Example | :CONFigure:MEMory SAFety | |
| | Enables the safety setting. | |

Query example          :CONFigure:MEMory?

>Safety

The safety setting is enabled.

Set →
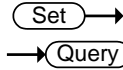
## :CONFigure:SHORt                                      → Query

| Description | Configures the short key. | |
|---|---|---|
| Syntax | :CONFigure:SHORt { TOGGle \| HOLD } | |
| Query Syntax | :CONFigure:SHORt? | |
| Parameter | HOLD | Sets the button configuration to hold |
| | TOGGLE or TOGG | Sets the button configuration to toggle |
| Return parameter | Toggle | Toggle |
| | Hold | Hold |
| Example | :CONFigure:SHORt TOGGle Sets the Short key configuration to toggle. | |
| Query example | :CONFigure:SHORt? >Toggle | |
| | The Short key is configured to toggle. | |

Set →
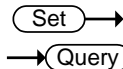
## :CONFigure:SHORt:SAFety                               → Query

| Description | Turns the Short Safety function on/off. The short safety function requires the load to already be turned on before the load can be shorted using the Short key or :INPut:SHORt command. | |
|---|---|---|
| Syntax | :CONFigure:SHORt:SAFety {<bool>\|OFF\|ON} | |
| Query Syntax | :CONFigure:SHORt:SAFety? | |
| Parameter | OFF \| 0 | Turns Short Safety off. |
| | ON \| 1 | Turns Short Safety on. |
| Return parameter | OFF | Short Safety is off. |
| | ON | Short Safety is on. |

| Example | :CONF:SHOR:SAF OFF |
| --- | --- |
| | Turns Short Safety off. |

| Query example | :CONF:SHOR:SAF? |
| --- | --- |
| | >OFF |
| | Short Safety is turned off. |

Set →
→ Query

# [:CONFigure]:GNG:SPECtest

| Description | Enables/Disables Go-NoGo testing (SPEC test = ON/SPEC test = OFF). |
| --- | --- |
| Syntax | [:CONFigure]:GNG:SPECtest {<Boolean>|OFF | ON } |
| Query Syntax | [:CONFigure]:GNG:SPECtest? |
| Parameter/ | OFF or 0 | SPEC test = OFF |
| Return parameter | ON or 1 | SPEC test = ON |
| Example | :GNG:SPECtest ON |
| | Turns Go-NoGo testing on. |
| Query example | :GNG:SPECtest? |
| | >OFF |
| | Indicates that Go-NoGo testing is off. |

Set →
→ Query

# [:CONFigure]:GNG:DTIMe

| Description | Sets or queries the Go-NoGo delay time. |
| --- | --- |
| Syntax | [:CONFigure]:GNG:DTIMe {<NRf>[s] | MINimum |MAXimum } |
| Query Syntax | [:CONFigure]:GNG:DTIMe? |
| Parameter | <NRf>[s] | Sets the Go-NoGo delay time in seconds (0.0~1.0) with 0.1 second resolution. |
| | MINIMUM or MIN | Minimum delay time |
| | MAXimum or MAX | Maximum delay time |
| Return parameter | <NR2> | Returns the delay time in seconds. |

| Example | :GNG:DTIMe 0.5 |
| --- | --- |
| | Sets the delay time to 0.5 seconds. |

| Query example | :GNG:DTIMe?<br>>0.5<br>The delay time is 0.5 seconds. |
| --- | --- |

Set →

[:CONFigure]:GNG:MODE                           → Query

| Description | Sets or queries the entry mode for the Go-NoGo settings. The entry mode determines whether the Go-NoGo limits are set as values or as a percentage value from a center reference value. |
| --- | --- |
| Syntax | [:CONFigure]:GNG:MODE {PERCent \| VALue } |
| Query Syntax | [:CONFigure]:GNG:MODE? |
| Parameter | PERCENT or PERC | Sets the entry mode to %. |
| | VALUE or VAL | Sets the entry mode to value |
| Return parameter | Percent | % entry mode |
| | Value | Value entry mode |
| Example | :GNG:MODE PERCent | |
| | Sets the entry mode to %. | |
| Query example | :GNG:MODE?<br>>Percent<br>The entry mode is %. | |

[:CONFigure]:GNG[:PASS]                         → Query

| Description | Queries the Go-NoGo test result(s). This command can be used for all test modes (CC, CV, CR, CP). |
| --- | --- |
| Query Syntax | [:CONFigure]:GNG[:PASS]? |
| Return parameter | NG | No Go (fail) |
| | GO | Go (Pass) |
| Query example | :GNG?<br>>GO<br>Returns the Go-NoGo test result. | |

Set ⟶

## [:CONFigure]:GNG:H

⟶ Query

| | |
|---|---|
| Description | Sets or queries the high voltage/current limit value. If the entry mode is set to value, the high voltage/current limit value units are in volts/amps. If the entry mode is set to percent, the high voltage/current limit value units are in percent. |
| Syntax | [:CONFigure]:GNG:H <NRf> |
| Query Syntax | [:CONFigure]:GNG:H? |

| Parameter | <NRf> | Sets the high voltage/current limit value in volts/amps or in percent. |
|---|---|---|
| Return parameter | <NR2> | Returns the high voltage/current limit value in volts/amps or as %. |

| | |
|---|---|
| Example | :GNG:H 100.0 |
| | Sets the high voltage limit value to 100%. |
| Query example | :GNG:H?<br>>100.0<br>Returns the high voltage limit value as 100.0%. |

Set ⟶

## [:CONFigure]:GNG:L

⟶ Query

| | |
|---|---|
| Description | Sets or queries the low voltage/current limit value. If the entry mode is set to value, the low voltage/current limit value units are in volts/amps. If the entry mode is set to percent, the low voltage/current limit value units are in percent. |
| Syntax | [:CONFigure]:GNG:L <NRf> |
| Query Syntax | [:CONFigure]:GNG:L? |

| Parameter | <NRf> | Sets the low voltage/current limit value in volts/amps or in percent. |
|---|---|---|
| Return parameter | <NR2> | Returns the low voltage/current limit value in volts/amps or as %. |

| Example | :GNG:L 10.0 |
|---|---|
| | Sets the low voltage limit value to 10%. |

| Query example | :GNG:L? |
|---|---|
| | >10.0 |
| | Returns the low voltage limit value as 10.0%. |

# [:CONFigure]:GNG:C

Set →
→ Query

| Description | Sets or queries the center voltage/current limit value. The center voltage limit value is used as the center reference value when the entry mode is set to percent ([:CONFigure]:GNG:Mode=PERCent). |
|---|---|

| Syntax | [:CONFigure]:GNG:C <NRf> |
|---|---|

| Query Syntax | [:CONFigure]:GNG:C? |
|---|---|

| Parameter | <NRf> | Sets the center voltage/current limit value in volts/amps. |
|---|---|---|
| Return parameter | <NR2> | Returns the center voltage/current limit value in volts/amps. |

| Example | :GNG:C 10.0 |
|---|---|
| | Sets the center voltage/current limit value to 10V or A. |

## Step Resolution Commands

## [:CONFigure]:STEP:CCH

Set →
→ Query

| Description | Configures the step resolution for CC High Range. | |
|---|---|---|
| ⚠ Note | The step resolution setting will be automatically rounded to the closest multiple of the base resolution. | |
| Syntax | [:CONFigure]:STEP:CCH {<NRf>[ A ] | MINimum | MAXimum } | |
| Query Syntax | [:CONFigure]:STEP:CCH? | |
| Parameter | <NRf>[ A ]<br>MINIMUM or MIN<br>MAXIMUM or MAX | Step resolution.<br>Minimum step resolution<br>Maximum step resolution |
| Return parameter | <ASCII string> | Returns the range and the step resolution. |
| Example | :STEP:CCH 0.002A | |
| | Sets the step resolution to 0.002A. | |
| Query example | :STEP:CCH?<br>> CCH:0.002 | |
| | Returns the step resolution (0.002A). | |

## [:CONFigure]:STEP:CCL

Set →
→ Query
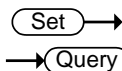
| Description | Configures the step resolution for CC low Range. | |
|---|---|---|
| ⚠ Note | The step resolution setting will be automatically rounded to the closest multiple of the base resolution. | |
| Syntax | [:CONFigure]:STEP:CCL {<NRf>[ A ] | MINimum | MAXimum } | |
| Query Syntax | [:CONFigure]:STEP:CCL? | |
| Parameter | <NRf>[ A ]<br>MINIMUM or MIN<br>MAXIMUM or MAX | Step resolution.<br>Minimum step resolution<br>Maximum step resolution |
| Return parameter | <ASCII string> | Returns the range and the step resolution. |
| Example | :STEP:CCL 0.02mA<br><br>Sets the step resolution to 0.02mA. | |
| Query example | :STEP:CCL?<br>> CCL:0.00002<br><br>Returns the step resolution (0.00002A). | |

## [:CONFigure]:STEP:CC

→ Query

| Description | Returns the step resolution for each CC Mode range as a string. | |
|---|---|---|
| Query Syntax | [:CONFigure]:STEP:CC? | |
| Return parameter | <ASCII string> | Returns the CCH and CCL step resolution settings. |
| Query example | :STEP:CC?<br>>CCH:0.002, CCL:0.0002<br><br>Returns the CC mode step resolution for each range. | |

Set →
→ Query

# [:CONFigure]:STEP:CRH

| | |
|---|---|
| Description | Configures the step resolution for CR High Range. |
| ⚠ Note | The step resolution setting will be automatically rounded to the closest multiple of the base resolution. |
| Syntax | [:CONFigure]:STEP:CRH {<NRf>[mS] \| MINimum \| MAXimum } |
| Query Syntax | [:CONFigure]:STEP:CRH? |

| Parameter | <NRf>[mS] | Step resolution (Unit: mS) |
|---|---|---|
| | MINIMUM or MIN | Minimum step resolution |
| | MAXIMUM or MAX | Maximum step resolution |
| Return parameter | <ASCII string> | Returns the range and the step resolution. |

| | |
|---|---|
| Example | :STEP:CRH 2 |
| | Sets the step resolution to 2 mS. |
| Query example | :STEP:CRH? |
| | >CRH:2.00000 |
| | Returns the step resolution (2 mS). |

Set →
→ Query

# [:CONFigure]:STEP:CRL

| | |
|---|---|
| Description | Configures the step resolution for CR Low Range. |
| ⚠ Note | The step resolution setting will be automatically rounded to the closest multiple of the base resolution. |
| Syntax | [:CONFigure]:STEP:CRL {<NRf>[mS] \| MINimum \| MAXimum } |
| Query Syntax | [:CONFigure]:STEP:CRL? |

| Parameter | <NRf>[mS] | Step resolution (Unit: mS) |
|---|---|---|
| | MINIMUM or MIN | Minimum step resolution |
| | MAXIMUM or MAX | Maximum step resolution |
| Return parameter | <ASCII string> | Returns the range and the step resolution. |

| Example | :STEP:CRL 20 |
|---|---|
| | Sets the step resolution to 20mS. |
| Query example | :STEP:CRL?<br>>CRL:20.0 |
| | Returns the step resolution (20 mS). |

## [:CONFigure]:STEP:CR  ➡ Query

| Description | Returns the step resolution for each CR Mode range as a string. |  |
|---|---|---|
| Query Syntax | [:CONFigure]:STEP:CR? |  |
| Return parameter | <ASCII string> | Returns the CRH and CRL step resolution settings (in mS). |
| Query example | :STEP:CR?<br>> CRH:2.00000, CRL:20.0 |  |
| | Returns the CR mode step resolution for each range (mS). |  |

## [:CONFigure]:STEP:CVH  Set ➡  ➡ Query

| Description | Configures the step resolution for CV High Range. |  |
|---|---|---|
| ⚠ Note | The step resolution setting will be automatically rounded to the closest multiple of the base resolution. |  |
| Syntax | [:CONFigure]:STEP:CVH{<NRf>[ V ] \| MINimum \| MAXimum} |  |
| Query Syntax | [:CONFigure]:STEP:CVH? |  |
| Parameter | <NRf>[ V ]<br>MINIMUM or MIN<br>MAXIMUM or MAX | Step resolution.<br>Minimum step resolution<br>Maximum step resolution |
| Return parameter | <ASCII string> | Returns the range and the step resolution. |
| Example | :STEP:CVH 0.5V |  |
| | Sets the step resolution to 0.5V. |  |

| Query example | :STEP:CVH? |
| --- | --- |
| | > CVH:0.500 |
| | Returns the step resolution (0.5V). |

## [:CONFigure]:STEP:CVL

Set →
→ Query

| Description | Configures the step resolution for CV Low Range. | |
| --- | --- | --- |
| ⚠ Note | The step resolution setting will be automatically rounded to the closest multiple of the base resolution. | |
| Syntax | [:CONFigure]:STEP:CVL{<NRf>[ V ] \| MINimum \| MAXimum} | |
| Query Syntax | [:CONFigure]:STEP:CVL? | |
| Parameter | <NRf>[ V ] | Step resolution. |
| | MINIMUM or MIN | Minimum step resolution |
| | MAXIMUM or MAX | Maximum step resolution |
| Return parameter | <ASCII string> | Returns the range and the step resolution. |
| Example | :STEP:CVL 0.05V | |
| | Sets the step resolution to 0.001V. | |
| Query example | :STEP:CVL? | |
| | > CVL:0.0500 | |
| | Returns the step resolution (0.05V). | |

## [:CONFigure]:STEP:CV

→ Query

| Description | Returns the step resolution for each CV Mode range as a string. | |
| --- | --- | --- |
| Query Syntax | [:CONFigure]:STEP:CV? | |
| Return parameter | <ASCII string> | Returns the CVH and CVL step resolution settings. |
| Query example | :STEP:CV? | |
| | >CVH:0.500, CVL:0.0500 | |
| | Returns the CV mode step resolution for each range. | |

## [:CONFigure]:STEP:CPH

Set →
→ Query

| | |
|---|---|
| Description | Configures the step resolution for CP High Range. |
| ⚠ Note | The step resolution setting will be automatically rounded to the closest multiple of the base resolution. |
| Syntax | [:CONFigure]:STEP:CPH {<NRf>[ W ] | MINimum | MAXimum } |
| Query Syntax | [:CONFigure]:STEP:CPH? |

| Parameter | <NRf>[ W ] | Step resolution. |
|---|---|---|
| | MINIMUM or MIN | Minimum step resolution |
| | MAXIMUM or MAX | Maximum step resolution |
| Return parameter | <ASCII string> | Returns the range and the step resolution. |

| | |
|---|---|
| Example | :STEP:CPH 1 |
| | Sets the step resolution to 0.01W. |
| Query example | :STEP:CPH? |
| | >CPH:1.00 |
| | Returns the step resolution (1W). |

## [:CONFigure]:STEP:CPL

Set →
→ Query

| | |
|---|---|
| Description | Configures the step resolution for CP Low Range. |
| ⚠ Note | The step resolution setting will be automatically rounded to the closest multiple of the base resolution. |
| Syntax | [:CONFigure]:STEP:CPL {<NRf>[ W ] | MINimum | MAXimum } |
| Query Syntax | [:CONFigure]:STEP:CPL? |

| Parameter | <NRf>[ W ] | Step resolution. |
|---|---|---|
| | MINIMUM or MIN | Minimum step resolution |
| | MAXIMUM or MAX | Maximum step resolution |
| Return parameter | <ASCII string> | Returns the range and the step resolution. |

| Example | :STEP:CPL 0.1 |
|---|---|
| | Sets the step resolution to 0.0001W. |

| Query example | :STEP:CPL?<br>>CPL:0.100 |
|---|---|
| | Returns the step resolution (0. 1W). |

## [:CONFigure]:STEP:CP  → Query

| Description | Returns the step resolution for each CP Mode range as a string. | |
|---|---|---|
| Query Syntax | [:CONFigure]:STEP:CP? | |
| Return parameter | \<ASCII string> | Returns the CPH and CPL step resolution settings. |
| Query example | :STEP:CP?<br>> CPH:1.00, CPL:0.100 | |
| | Returns the CP mode step resolution for each range. | |

## External Control Commands

Set →
→ Query

## [:CONFigure]:EXTernal[:CONTrol]

| Description | Configures the unit for external control or disables external control. |
|---|---|
| Syntax | [:CONFigure]:EXTernal[:CONTrol] {OFF \| VOLTage \| RESistance \| RINV } |
| Related Commands | [:CONFigure]:EXTernal[:CONTrol]? |
| Parameter | OFF — Disables external control |
| | VOLTAGE or VOLT — Sets the unit to external voltage control |
| | RESISTANCE or RES — Sets the unit to external resistance control |
| | RINV — Sets the unit to external resistance (inverted) control |
| Return Parameter | <ASCII string> — Returns the external control configuration |
| Example | :EXTernal VOLT |
| | Turns external control voltage control on. |
| Query example | :EXTernal? |
| | >voltage |
| | The unit uses external voltage control. |

Set →
→ Query

## [:CONFigure]:EXTernal:LOADonin

| Description | The LOADonin settings determine whether the load is turned on when an external switch is closed (LOW) or open (HIGH). |
|---|---|

| | | |
|---|---|---|
| Syntax | [:CONFigure]:EXTernal:LOADonin {OFF | HIGH |LOW } | |
| Related Commands | [:CONFigure]:EXTernal:LOADonin? | |
| Parameter | OFF<br>HIGH<br>LOW | LoadOnIN = OFF<br>LoadOnIN = OPEN<br>LoadOnIN = CLOSE |
| Return Parameter | <ASCII string> | Returns the Loadonin configuration as a string. |
| Example | :EXTernal:LOADonin OFF<br><br>Turns external control off. | |
| Query example | :EXTernal:LOADonin?<br>> OFF<br><br>External control is off | |

## Mode Subsystem Commands

## :MODE

Set →
→ Query

| Description | Sets the operating mode. | |
|---|---|---|
| Syntax | :MODE {CC \| CR \| CV \| CP \| CCCV \| CRCV \| CPCV } | |
| Query Syntax | :MODE? | |
| Parameter/Return parameter | CC | CC mode |
| | CR | CR mode |
| | CV | CV mode |
| | CP | CP mode |
| | CCCV | CC + CV mode |
| | CRCV | CR + CV mode |
| | CPCV | CP + CV mode |
| Example | :MODE CC | |
| | Sets the mode to CC mode. | |
| Query example | :MODE?<br>>CC | |
| | Returns the operating mode (CC mode). | |

## [:MODE]:CRANge

Set →
→ Query

| Description | Configures the current range for all the applicable operating modes. | |
|---|---|---|
| Syntax | [:MODE]:CRANge {HIGH \| LOW } | |
| Query Syntax | [:MODE]:CRANge? | |
| Parameter | HIGH | High range |
| | LOW | Low range |
| Return parameter | High | High range |

| | | |
|---|---|---|
| | Low | Low range |
| Example | :CRANge LOW | |
| | Sets the current range to Low. | |
| Query example | :CRANge?<br>>Low | |
| | The current range is set to Low. | |

Set →
→ Query

## [:MODE]:VRANge

| | | |
|---|---|---|
| Description | Configures the voltage range for all the applicable operating modes. | |
| Syntax | [:MODE]:VRANge {HIGH \| LOW } | |
| Query Syntax | [:MODE]:VRANge? | |
| Parameter | HIGH<br>LOW | High range<br>Low range |
| Return parameter | High<br>Low | High range<br>Low range |
| Example | :VRANge LOW | |
| | Sets the voltage range to Low. | |
| Query example | :VRANge?<br>>LOW | |
| | The voltage range is set to Low. | |

Set →
→ Query

## [:MODE]:DYNamic

| | | |
|---|---|---|
| Description | Sets or queries the switching mode. Dynamic is the default mode. | |
| Syntax | :MODE:DYNamic{ DYNamic \| STATic } | |
| Query Syntax | :MODE:DYNamic? | |
| Parameter | DYNAMIC or DYN<br>STATIC or STAT | Set to Dynamic mode<br>Set to Static mode |
| Return parameter | <ASCII string> | Returns the switching mode. |

| | |
|---|---|
| Example | :MODE:DYNamic STAT |
| | Set the switching mode to dynamic |
| Query example | :MODE:DYNamic?<br>> STATIC |
| | The switching mode is set to static mode. |

## Current Subsystem Commands

Set →
→ Query
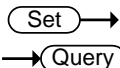
## :CURRent:RECall

| Description | Sets or queries whether A Value or B Value is the currently active value in CC static mode. | |
|---|---|---|
| Syntax | :CURRent:RECall {<bool> \| A \| B } | |
| Query Syntax | :CURRent:RECall? | |
| Parameter | A, 0 | A Value |
| | B, 1 | B Value |
| Return parameter | A | A Value |
| | B | B Value |
| Example | :CURRent:RECall 0 | |
| | Sets the current setting to A Value. | |

Set →
→ Query
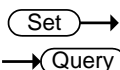
## :CURRent[:VA]

| Description | Sets or queries the CC mode "A Value" (static mode) current. This command is applicable to static modes. |
|---|---|

| ⚠️ Note | A different current value can be set for each current range (High/Low). |  |
|---|---|---|
|  | The [:VA] node can only be omitted when in static mode. |  |
| Syntax | :CURRent[:VA] {<NRf>[ A ] \| MINimum \| MAXimum } |  |
| Query Syntax | :CURRent[:VA]? |  |
| Parameter | <NRf>[A]<br>MINIMUM or MIN<br>MAXIMUM or MAX | "A Value" current value<br>Minimum current level<br>Maximum current level |
| Return parameter | <NR2> | Returns the "A Value" current value. |
| Example | :CURRent:VA MIN |  |
|  | Sets the current value to the minimum. |  |
| Query example | :CURRent:VA?<br>>1.0A |  |
|  | Returns the "A Value" current setting. |  |

Set ⟶
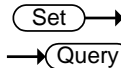
⟶ Query

## :CURRent:VB

| Description | Sets or queries the CC mode "B Value" (static mode) current. This command is applicable to static modes. |  |
|---|---|---|
| ⚠️ Note | A different current value can be set for each current range (High/Low). |  |
| Syntax | :CURRent:VB {<NRf>[ A ] \| MINimum \| MAXimum } |  |
| Query Syntax | :CURRent:VB? |  |
| Parameter | <NRf>[A]<br>MINIMUM or MIN<br>MAXIMUM or MAX | "B Value" current value<br>Minimum current level<br>Maximum current level |
| Return parameter | <NR2> | Returns the "B Value" current value. |
| Example | :CURRent:VB MIN |  |
|  | Sets the current value to the minimum. |  |

| Query example | :CURRent:VB? |
|---|---|
| | >1.0A |
| | Returns the "B Value" current setting. |

Set →
→ Query

## :CURRent:SRATe

| Description | Sets or queries the current slew rate for CC static mode. | |
|---|---|---|
| Syntax | :CURRent:SRATe {<NRf>\| MINimum \| MAXimum } | |
| Query Syntax | :CURRent:SRATe? | |
| Parameter | <NRf> | Sets the slew rate in mA/us |
| | MINIMUM or MIN | Set to the highest slew rate |
| | MAXIMUM or MAX | Set to the lowest slew rate |
| Return parameter | <NR2> | Returns the slew rate in mA/us. |
| Example | :CURRent:SRATe MIN | |
| | Sets the slew rate to the minimum. | |
| Query example | :CURRent:SRATe? | |
| | >10 | |
| | Returns the slew rate as 10mA/us. | |

Set →
→ Query

## :CURRent:L1

| Description | Sets or queries the CC mode "Level 1" current. This command is only applicable to dynamic modes. | |
|---|---|---|
| ⚠Note | A different current value can be set for each range (High/Low). | |
| Syntax | :CURRent:L1 {<NRf>[ A ] \| MINimum \| MAXimum } | |
| Query Syntax | :CURRent:L1? | |
| Parameter | <NRf>[A] | "Level1" current value |
| | MINIMUM or MIN | Minimum current level |
| | MAXIMUM or MAX | Maximum current level |

| Return parameter | \<NR2> | Returns the "Level1" current value. |
|---|---|---|

| Example | :CURRent:L1 MIN |
|---|---|
| | Sets the current value to the minimum. |

| Query example | :CURRent:L1? |
|---|---|
| | >1.000A |
| | Returns the "Level1" current setting. |

## :CURRent:L2

Set →
→ Query

| Description | Sets or queries the CC mode "Level2" current. This command is only applicable to dynamic modes. |
|---|---|

| ⚠ Note | A different current value can be set for each range (High/Low). |
|---|---|

| Syntax | :CURRent:L2 {\<NRf>[ A ] | MINimum | MAXimum } |
|---|---|

| Query Syntax | :CURRent:L2? |
|---|---|

| Parameter | \<NRf>[A] | "Level2" current value |
|---|---|---|
| | MINIMUM or MIN | Minimum current level |
| | MAXIMUM or MAX | Maximum current level |

| Return parameter | \<NR2> | Returns the "Level2" current value. |
|---|---|---|

| Example | :CURRent:L2 MIN |
|---|---|
| | Sets the current value to the minimum. |

| Query example | :CURRent:L2? |
|---|---|
| | >1.000A |
| | Returns the "Level2" current setting. |
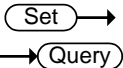
## :CURRent:SET

Set →
→ Query

| Description | Sets or queries the CC dynamic mode "Set" value when the dynamic units are set to Percent. |
|---|---|
| | This command is only applicable in dynamic mode. |

| Syntax | :CURRent:SET{\<NRf>[ A ] | MINimum | MAXimum } |
|---|---|

| Query Syntax | :CURRent:SET? | |
|---|---|---|
| Related Commands | :CURRent:LEVel | |
| Parameter | <NRf>[A]<br>MINIMUM or MIN<br>MAXIMUM or MAX | "Set" current value<br>Minimum current level<br>Maximum current level |
| Return parameter | <NR2> | Returns the "Set" current value. |
| Example | :CURRent:SET MIN | |
| | Sets the current value to the minimum. | |
| Query example | :CURRent:SET?<br>>1.000A | |
| | Returns the "Set" current setting. | |

## :CURRent:LEVel

Set →
→ Query
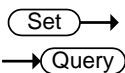
| Description | Sets or queries the CC mode % level (percentage of the Set current value) when the dynamic mode units are set to Percent. | |
|---|---|---|
| Syntax | :CURRent:LEVel {<NRf>\| MINimum \| MAXimum } | |
| Query Syntax | :CURRent:LEVel? | |
| Related Commands | :CURRent:SET | |
| Parameter | <NRf><br>MINIMUM or MIN<br>MAXIMUM or MAX | "% level" current value<br>Minimum % current level<br>Maximum % current level |
| Return parameter | <NR2> | Returns the "% level" current value. |
| Example | :CURRent:LEVel MIN | |
| | Sets the % level current value to the minimum. | |
| Query example | :CURRent:LEVel?<br>>50.00 | |
| | Returns the "% Level" as 50 percent. | |

Set →
→ Query

## :CURRent:RISE

| | |
|---|---|
| Description | Sets the rising current slew rate for CC dynamic mode. |
| Syntax | :CURRent:RISE {<NRf> \| MINimum \| MAXimum } |
| Query Syntax | :CURRent:RISE? |
| Related Commands | :CURRent:FALL |
| Parameter | <NRf>                 Rising current slew rate<br>MINIMUM or MIN    Minimum slew rate<br>MAXIMUM or MAX   Maximum slew rate |
| Return parameter | <NR2>                Returns the rising current slew rate in mA/us. |
| Example | :CURRent:RISE MIN |
| | Sets the rising slew rate to the minimum. |
| Query example | :CURRent:RISE?<br>>2500 |
| | Returns the rising slew rate as 2500mA/us. |

Set →
→ Query

## :CURRent:FALL

| | |
|---|---|
| Description | Sets the falling current slew rate for CC dynamic mode. |
| Syntax | :CURRent:FALL {<NRf> \| MINimum \| MAXimum } |
| Query Syntax | :CURRent:FALL? |
| Related Commands | :CURRent:FALL |
| Parameter | <NRf>                 Falling current slew rate<br>MINIMUM or MIN    Minimum slew rate<br>MAXIMUM or MAX   Maximum slew rate |
| Return parameter | <NR2>                Returns the falling current slew rate in mA/us. |

| Example | :CURRent:FALL MIN |
| --- | --- |
| | Sets the falling slew rate to the minimum. |

| Query example | :CURRent:FALL? <br> >2500 |
| --- | --- |
| | Returns the falling slew rate as 5000mA/us. |

## :CURRent:T1

Set →
→ Query
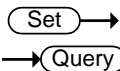
| Description | Sets the T1 timer settings for the CC dynamic mode. |
| --- | --- |
| Syntax | :CURRent:T1 {<NRf>[s] \| MINimum \| MAXimum} |
| Query Syntax | :CURRent:T1? |
| Related Commands | :CURRent:T2 |

| Parameter | <NRf>[s] | T1 timer setting in seconds |
| --- | --- | --- |
| | MINIMUM or MIN | Minimum time |
| | MAXIMUM or MAX | Maximum time |
| Return parameter | <NR2> | Returns the T1 timer time in seconds. |

| Example | :CURRent:T1 0.200 |
| --- | --- |
| | Sets the T1 timer time to 200ms. |

| Query example | :CURRent:T1? <br> >0.200000 |
| --- | --- |
| | Returns the T1 timer time as 200ms. |

## :CURRent:T2

Set →
→ Query
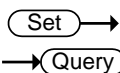
| Description | Sets the T2 timer settings for the CC dynamic mode. |
| --- | --- |
| Syntax | :CURRent:T2 {<NRf>[s] \| MINimum \| MAXimum} |
| Query Syntax | :CURRent:T2? |

| Related Commands | :CURRent:T1 | |
|---|---|---|
| Parameter | <NRf>[s]<br>MINIMUM or MIN<br>MAXIMUM or MAX | T2 timer setting in seconds<br>Minimum time<br>Maximum time |
| Return parameter | <NR2> | Returns the T2 timer time in seconds. |
| Example | :CURRent:T2 0.200 | |
| | Sets the T2 timer time to 200ms. | |
| Query example | :CURRent:T2?<br>>0.200000 | |
| | Returns the T2 timer time as 200ms. | |

## :CURRent:FREQuency

Set →

→ Query
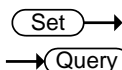
| Description | Sets the CC dynamic mode switching frequency. This command along with the ":CURRent:DUTY" command is used when the dynamic time setting is set to "Freq./Duty". | |
|---|---|---|
| | This command is only applicable for dynamic mode. | |
| Syntax | :CURRent:FREQuency {<NRf> \| MINimum \| MAXimum} | |
| Query Syntax | :CURRent:FREQuency? | |
| Related Commands | :CURRent:DUTY | |
| Parameter | <NRf><br><br>MINIMUM or MIN<br>MAXIMUM or MAX | Sets the switching frequency in hertz.<br>Minimum frequency<br>Maximum frequency |
| Return parameter | <NR2> | Returns the frequency in hertz |
| Example | :CURRent:FREQuency 10000 | |
| | Sets frequency to 10kHz. | |

Query example          :CURRent:FREQuency?
                       >10000.0

                       Returns the switching frequency (10kHz).

## :CURRent:DUTY

Set →
→ Query

Description            Sets the positive duty cycle for the CC dynamic
                       mode switching frequency. This command along
                       with the ":CURRent:FREQuency" command is
                       used when the dynamic time setting is set to
                       "Freq./Duty".

                       This command is only applicable for dynamic
                       mode.

Syntax                 :CURRent:DUTY {<NRf> | MINimum | MAXimum}

Query Syntax           :CURRent:DUTY?

Related                :CURRent:FREQuency
Commands

| Parameter | <NRf> | Sets the duty as a percentage. |
|---|---|---|
| | MINIMUM or MIN | Minimum duty |
| | MAXIMUM or MAX | Maximum duty |
| Return parameter | <NR2> | Returns the duty as a percentage |

Example                :CURRent:DUTY 50

                       Sets the duty to 50%.

Query example          :CURRent:DUTY?
                       >50.0

                       Returns the duty (50%).

## :CURRent:CV:RESPonse

Set →
→ Query

Description            Sets or returns the +CV response speed for CC
                       Mode. Slow is the default response speed.

Syntax                 :CURRent:CV:RESPonse { FAST | SLOW }

| Query Syntax | :CURRent:CV:RESPonse? | |
|---|---|---|
| Parameter | FAST | Fast response |
| | SLOW | Slow response |
| Return parameter | <ASCII string> | Returns +CV response speed as a string. |
| Example | :CURRent:CV:RESPonse FAST | |
| | Sets the response to fast. | |
| Query example | :CURRent:CV:RESPonse?<br>> FAST | |
| | The +CV response is FAST. | |

## Resistance Subsystem Commands

## :RESistance:RECall

Set →
→ Query

| Description | Sets or queries whether A Value or B Value is the currently active value in CC static mode. | |
|---|---|---|
| ⚠Note | This command is only applicable when the CR unit is set to ohms. | |
| Syntax | :RESistance:RECall {<bool> \| A \| B } | |
| Query Syntax | :RESistance:RECall? | |
| Parameter | A, 0 | A Value |
| | B, 1 | B Value |
| Return parameter | A | A Value |
| | B | B Value |

| Example | :RESistance:RECall 0 |
|---|---|
| | Sets the resistance setting to A Value. |

## :CONDuctanceRECall

Set ⟶
⟶ Query

| Description | Sets or queries whether A Value or B Value is the currently active value in CC static mode when the units are set to Siemens. |
|---|---|
| ⚠️Note | This command is only applicable when the CR unit is set to Siemens. |
| Syntax | :CONDuctance:RECall {<bool> | A | B } |
| Query Syntax | :CONDuctance:RECall? |
| Parameter | A, 0            A Value<br>B, 1            B Value |
| Return parameter | A             A Value<br>B             B Value |
| Example | :CONDuctance:RECall 0 |
| | Sets the conductance setting to A Value. |

## :CONDuctance[:VA]

Set ⟶
⟶ Query

| Description | Sets or queries the CR mode "A Value" conductance. This command is only applicable to static mode. |
|---|---|
| ⚠️Note | A different value can be set for each current range (High/Low).<br><br>The optional command node [:VA] can only be omitted when in static mode.<br><br>This command is only applicable when the CR unit is set to Siemens. |
| Syntax | :CONDuctance[:VA] {<NRf>[mS] | MINimum | MAXimum } |
| Query Syntax | :CONDuctance[:VA]? |

| Parameter | <NRf>[ mS ] | "A Value" conductance value (Unit: millisiemens) |
| --- | --- | --- |
| | MINIMUM or MIN | Minimum conductance level |
| | MAXIMUM or MAX | Maximum conductance level |
| Return parameter | <NR2> | Returns the "A Value" conductance value |
| Example | :CONDuctance:VA MIN | |
| | Sets the conductance value to the minimum. | |
| Query example | :CONDuctance:VA? > 60000 | |
| | Returns the "A Value" conductance setting(mS). | |

## :CONDuctance:VB

Set →
→ Query
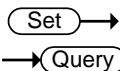
| Description | Sets or queries the CR mode "B Value" conductance. This command is applicable to static mode only. |
| --- | --- |
| ⚠ Note | A different value can be set for each current range (High/Low). |
| | This command is only applicable when the CR unit is set to Siemens. |
| Syntax | :CONDuctance:VB {<NRf>[mS] \| MINimum \| MAXimum } |
| Query Syntax | :CONDuctance:VB? |

| Parameter | <NRf>[mS] | "B Value" conductance value |
| --- | --- | --- |
| | MINIMUM or MIN | Minimum conductance level |
| | MAXIMUM or MAX | Maximum conductance level |
| Return parameter | <NR2> | Returns the "B Value" conductance value |
| Example | :CONDuctance:VB MIN | |
| | Sets the conductance value to the minimum. | |

| Query example | :CONDuctance:VB?<br>> 60000 |
|---|---|
| | Returns the "B Value" conductance setting(mS). |

Set ⟶
:RESistance[:VA]                              ⟶ Query

| Description | Sets or queries the CR mode "A Value" resistance. This command is only applicable to static mode. |
|---|---|
| ⚠ Note | A different value can be set for each current range (High/Low). |
| | The optional command node [:VA] can only be omitted when in static mode. |
| | This command is only applicable when the CR unit is set to ohms. |
| Syntax | :RESistance[:VA] {<NRf>[OHM] \| MINimum \| MAXimum } |
| Query Syntax | :RESistance[:VA]? |
| Parameter | <NRf>[OHM]             "A Value" resistance value<br>MINIMUM or MIN        Minimum resistance level<br>MAXIMUM or MAX       Maximum resistance level |
| Return parameter | <NR2>                     Returns the "A Value" resistance value |
| Example | :RESistance:VA 500 |
| | Sets the resistance value to the minimum. |
| Query example | :RESistance:VA?<br>> 500.00 |
| | Returns the "A Value" resistance setting(OHM). |

Set ⟶
:RESistance:VB                               ⟶ Query

| Description | Sets or queries the CR mode "B Value" resistance. This command is applicable to static mode only. |
|---|---|

| Note | A different value can be set for each current range (High/Low). |  |
|---|---|---|
| | This command is only applicable when the CR unit is set to ohms. | |
| Syntax | :RESistance:VB {<NRf>[OHM] \| MINimum \| MAXimum } | |
| Query Syntax | :RESistance:VB? | |
| Parameter | <NRf>[OHM]<br>MINIMUM or MIN<br>MAXIMUM or MAX | "B Value" resistance value<br>Minimum resistance level<br>Maximum resistance level |
| Return parameter | <NR2> | Returns the "B Value" resistance value |
| Example | :RESistance:VB 500 | |
| | Sets the resistance value to the minimum. | |
| Query example | :RESistance:VB?<br>> 500.00 | |
| | Returns the "B Value" resistance setting (OHM). | |

Set ⟶
## :RESistance:SRATe
⟶ Query
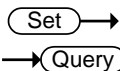
| Description | Sets or queries the current slew rate for CR static mode. |  |
|---|---|---|
| Note | This command is only applicable when the CR unit is set to ohms. | |
| Syntax | :RESistance:SRATe {<NRf> \| MINimum \| MAXimum} | |
| Query Syntax | :RESistance:SRATe? | |
| Parameter | <NRf><br><br>MINIMUM or MIN<br>MAXIMUM or MAX | Sets the slew rate in mA/us<br>Set to the lowest slew rate<br>Set to the highest slew rate |
| Return parameter | <NR2> | Returns the slew rate in mA/us |
| Example | :RESistance:SRATe MIN | |
| | Sets the slew rate to the minimum. | |

| Query example | :RESistance:SRATe?
>10 |
|---|---|

Returns the slew rate as 10mA/us.

Set →
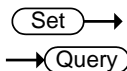
## :CONDuctance:L1
→ Query

| Description | Sets or queries the CR mode "Level1" conductance. This command is only applicable to dynamic mode. |
|---|---|
| ⚠Note | A different value can be set for each current range (High/Low). |
| | This command is only applicable when the CR unit is set to Siemens. |
| Syntax | :CONDuctance:L1 {<NRf>[mS] | MINimum | MAXimum } |
| Query Syntax | :CONDuctance:L1? |
| Parameter | <NRf>[mS] | "level1" conductance value (Unit: millisiemens) |
| | MINIMUM or MIN | Minimum conductance level |
| | MAXIMUM or MAX | Maximum conductance level |
| Return parameter | <NR2> | Returns the "level1" conductance value |
| Example | :CONDuctance:L1 MIN |
| | Sets the conductance value to the minimum. |
| Query example | :CONDuctance:L1?
> 2 |
| | Returns the "level1" conductance setting(2mS). |

Set →
:CONDuctance:L2 → Query

| Description | Sets or queries the CR mode "level2" conductance. This command is applicable to dynamic mode only. | |
|---|---|---|
| ⚠Note | A different value can be set for each current range (High/Low). | |
| | This command is only applicable when the CR unit is set to Siemens. | |
| Syntax | :CONDuctance:L2 {<NRf>[mS] \| MINimum \| MAXimum } | |
| Query Syntax | :CONDuctance:L2? | |
| Parameter | <NRf> | "level2" conductance value (Unit: millisiemens) |
| | MINIMUM or MIN | Minimum conductance level |
| | MAXIMUM or MAX | Maximum conductance level |
| Return parameter | <NR2> | Returns the "Level2" conductance value |
| Example | :CONDuctance:L2 MIN | |
| | Sets the conductance value to the minimum. | |
| Query example | :CONDuctance:L2? <br> > 2 | |
| | Returns the "Level2" conductance setting(mS). | |

Set →
:CONDuctance:SET → Query

| Description | Sets or queries the CR dynamic mode Set value when the dynamic units are set to Percent. | |
|---|---|---|
| | This command is only applicable in dynamic mode. | |
| ⚠Note | This command is only applicable when the CR unit is set to Siemens. | |

| Syntax | :CONDuctance:SET {<NRf>[mS] \| MINimum \| MAXimum } |  |
| --- | --- | --- |
| Query Syntax | :CONDuctance:SET? |  |
| Related Commands | :CONDuctance:LEVel |  |
| Parameter | <NRf> | "Set" conductance value (Unit: mS) |
|  | MINIMUM or MIN | Minimum conductance level |
|  | MAXIMUM or MAX | Maximum conductance level |
| Return parameter | <NR2> | Returns the "Set" conductance value (Unit: mS) |
| Example | :CONDuctance:SET MIN |  |
|  | Sets the conductance to the minimum. |  |
| Query example | :CONDuctance:SET?<br>> 0 |  |
|  | Returns the "Set" conductance setting in mS. |  |

## :RESistance:L1

Set ⟶
⟶ Query
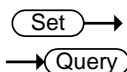
| Description | Sets or queries the CR mode "Level1" resistance. This command is only applicable to dynamic mode. |  |
| --- | --- | --- |
| ⚠Note | A different value can be set for each current range (High/Low). |  |
|  | This command is only applicable when the CR unit is set to ohms. |  |
| Syntax | :RESistance:L1 {<NRf>[OHM] \| MINimum \| MAXimum } |  |
| Query Syntax | :RESistance:L1? |  |
| Parameter | <NRf>[OHM] | "level1" resistance value (Unit: OHM) |

| | MINIMUM or MIN | Minimum resistance level |
|---|---|---|
| | MAXIMUM or MAX | Maximum resistance level |
| Return parameter | <NR2> | Returns the "level1" resistance value |

| Example | :RESistance:L1 MIN |
|---|---|
| | Sets the resistance value to the minimum. |

| Query example | :RESistance:L1? |
|---|---|
| | > 0 |
| | Returns the "level1" resistance setting(OHM). |

## :RESistance:L2

<div style="text-align:right">Set →<br>→ Query</div>
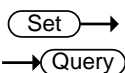
| Description | Sets or queries the CR mode "level2" resistance. This command is applicable to dynamic mode only. |
|---|---|
| ⚠ Note | A different value can be set for each current range (High/Low). |
| | This command is only applicable when the CR unit is set to ohms. |
| Syntax | :RESistance:L2 {<NRf>[OHM] \| MINimum \| MAXimum } |
| Query Syntax | :RESistance:L2? |
| Parameter | <NRf> | "level2" resistance value |
| | MINIMUM or MIN | Minimum resistance level |
| | MAXIMUM or MAX | Maximum resistance level |
| Return parameter | <NR2> | Returns the "level2" resistance value |

| Example | :RESistance:L2 MIN |
|---|---|
| | Sets the resistance value to the minimum. |

| Query example | :RESistance:L2? |
|---|---|
| | > 0 |
| | Returns the "level2" resistance setting.(OHM) |

Set ⟶

⟶ Query

## :RESistance:SET

| Description | Sets or queries the CR dynamic mode Set value when the dynamic units are set to Percent. |
| --- | --- |
| | This command is only applicable in dynamic mode. |
| ⚠ Note | This command is only applicable when the CR unit is set to ohms. |
| Syntax | :RESistance:SET {<NRf>[OHM] \| MINimum \| MAXimum } |
| Query Syntax | :RESistance:SET? |
| Related Commands | :RESistance:LEVel |
| Parameter | <NRf> | "Set" resistance value (Unit: OHM) |
| | MINIMUM or MIN | Minimum resistance level |
| | MAXIMUM or MAX | Maximum resistance level |
| Return parameter | <NR2> | Returns the "Set" resistance value. |
| Example | :RESistance:SET MIN | |
| | Sets the resistance to the minimum. | |
| Query example | :RESistance:SET?<br>> 0 | |
| | Returns the "Set" resistance setting in OHM. | |

Set ⟶

⟶ Query

## :RESistance:LEVel

| Description | Sets or queries the CR mode % level (percentage of the Set conductance value) when the dynamic mode units are set to Percent. |
| --- | --- |
| Syntax | :RESistance:LEVel {<NRf> \| MINimum \| MAXimum } |
| Query Syntax | :RESistance:LEVel? |

| Related Commands | :RESistance:SET | |
|---|---|---|
| Parameter | <NRf> | "% level" conductance value |
| | MINIMUM or MIN | Minimum % conductance level |
| | MAXIMUM or MAX | Maximum % conductance level |
| Return parameter | <NR2> | Returns the "% level" conductance value. |
| Example | :RESistance:LEVel MIN | |
| | Sets the % level conductance value to the minimum. | |
| Query example | :RESistance:LEVel?<br>>50.00 | |
| | Returns the "% Level" as 50 percent. | |
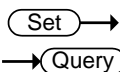
## :RESistance:RISE

Set →
→ Query

| Description | Sets the rising current slew rate for CR dynamic mode. | |
|---|---|---|
| Syntax | :RESistance:RISE {<NRf> \| MINimum \| MAXimum } | |
| Query Syntax | :RESistance:RISE? | |
| Related Commands | :RESistance:FALL | |
| Parameter | <NRf> | Rising current slew rate [mA/us] |
| | MINIMUM or MIN | Minimum slew rate |
| | MAXIMUM or MAX | Maximum slew rate |
| Return parameter | <NR2> | Returns the rising current slew rate in mA/us. |
| Example | :RESistance:RISE MIN | |
| | Sets the rising slew rate to the minimum. | |
| Query example | :RESistance:RISE?<br>>2500 | |
| | Returns the rising slew rate as 50mA/us. | |

Set →
→ Query

## :RESistance:FALL

| Description | Sets the falling current slew rate for CR dynamic mode. | |
|---|---|---|
| Syntax | :RESistance:FALL {<NRf> \| MINimum \| MAXimum } | |
| Query Syntax | :RESistance:FALL? | |
| Related Commands | :RESistance:RISE | |
| Parameter | <NRf> | Falling current slew rate [ mA/us] |
| | MINIMUM or MIN | Minimum slew rate |
| | MAXIMUM or MAX | Maximum slew rate |
| Return parameter | <NR2> | Returns the falling current slew rate in mA/us. |
| Example | :RESistance:FALL MIN | |
| | Sets the falling slew rate to the minimum. | |
| Query example | :RESistance:FALL? >2500 | |
| | Returns the falling slew rate as 50mA/us. | |

Set →
→ Query

## :RESistance:T1

| Description | Sets the T1 timer settings for the CR dynamic mode. | |
|---|---|---|
| Syntax | :RESistance:T1 {<NRf>[s] \| MINimum \| MAXimum} | |
| Query Syntax | :RESistance:T1? | |
| Related Commands | :RESistance:T2 | |
| Parameter | <NRf>[s] | T1 timer setting in seconds |
| | MINIMUM or MIN | Minimum time |
| | MAXIMUM or MAX | Maximum time |
| Return parameter | <NR2> | Returns the T1 timer time in seconds. |

| Example | :CURRent:T1 0.200 |
| --- | --- |
| | :CURRent:T1 0.200s |
| | :CURRent:T1 200 ms |
| | Sets the T1 timer time to 200ms. |
| Query example | :CURRent:T1? <br> >0.200000 |
| | Returns the T1 timer time as 200ms. |

## :RESistance:T2

Set →
→ Query
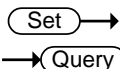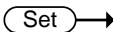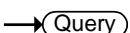
| Description | Sets the T2 timer settings for the CR dynamic mode. | |
| --- | --- | --- |
| Syntax | :RESistance:T2 {<NRf>[s] \| MINimum \| MAXimum} | |
| Query Syntax | :RESistance:T2? | |
| Related Commands | :RESistance:T1 | |
| Parameter | <NRf>[s] | T2 timer setting in seconds |
| | MINIMUM or MIN <br> MAXIMUM or MAX | Minimum time <br> Maximum time |
| Return parameter | <NR2> | Returns the T2 timer time in seconds. |
| Example | :RESistance:T2 0.200 | |
| | Sets the T2 timer time to 200ms. | |
| Query example | :RESistance:T2? <br> >0.200000 | |
| | Returns the T2 timer time as 200ms. | |

:RESistance:FREQuency

Set →
→ Query

| | |
|---|---|
| Description | Sets the CR dynamic mode switching frequency. This command along with the ":RESistance:DUTY" command is used when the dynamic time setting is set to "Freq./Duty".<br><br>This command is only applicable for dynamic mode. |
| Syntax | :RESistance:FREQuency {<NRf> \| MINimum \| MAXimum} |
| Query Syntax | :RESistance:FREQuency? |
| Related Commands | :RESistance:DUTY |

| Parameter | <NRf><br><br>MINIMUM or MIN<br>MAXIMUM or MAX | Sets the switching frequency in hertz.<br>Minimum frequency<br>Maximum frequency |
|---|---|---|
| Return parameter | <NR2> | Returns the frequency in hertz. |

| | |
|---|---|
| Example | :RESistance:FREQuency 10000<br><br>Sets frequency to 10kHz. |
| Query example | :RESistance:FREQuency?<br>>10000.0<br><br>Returns the switching frequency (10kHz). |

:RESistance:DUTY

Set →
→ Query

| | |
|---|---|
| Description | Sets the positive duty cycle for the CR dynamic mode switching frequency. This command along with the ":RESistance:FREQuency" command is used when the dynamic time setting is set to "Freq./Duty".<br><br>This command is only applicable for dynamic mode. |

| Syntax | :RESistance:DUTY {<NRf> \| MINimum \| MAXimum} | |
|---|---|---|
| Query Syntax | :RESistance:DUTY? | |
| Related Commands | :RESistance:FREQuency | |
| Parameter | <NRf> | Sets the duty as a percentage. |
| | MINIMUM or MIN | Minimum duty |
| | MAXIMUM or MAX | Maximum duty |
| Return parameter | <NR2> | Returns the duty as a percentage. |
| Example | :RESistance:DUTY 50 | |
| | Sets the duty to 50%. | |
| Query example | :RESistance:DUTY?<br>>50.0 | |
| | Returns the duty (50%). | |

## :RESistance:CV:RESPonse

Set →
→ Query
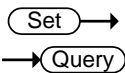
| Description | Sets or returns the +CV response speed for CR Mode. Slow is the default response speed. | |
|---|---|---|
| Syntax | :RESistance:CV:RESPonse { FAST \| SLOW } | |
| Query Syntax | :RESistance:CV:RESPonse? | |
| Parameter | FAST | Fast response |
| | SLOW | Slow response |
| Return parameter | <ASCII string> | Returns +CV response speed as a string. |
| Example | :RESistance:CV:RESPonse FAST | |
| | Sets the response to fast. | |
| Query example | :RESistance:CV:RESPonse?<br>> FAST | |
| | The +CV response is FAST. | |

## Voltage Subsystem Commands

Set ⟶

⟶ Query

### :VOLTage:RECall

| | | |
|---|---|---|
| Description | Sets or queries whether A Value or B Value is the currently active value in CV mode. | |
| Syntax | :VOLTage:RECall {<bool> | A | B } | |
| Query Syntax | :VOLTage:RECall? | |
| Parameter | A, 0 | A Value |
| | B, 1 | B Value |
| Return parameter | A | A Value |
| | B | B Value |
| Example | :VOLTage:RECall 0 | |
| | Sets the voltage setting to A Value. | |

Set ⟶

⟶ Query

### :VOLTage[:VA]

| | | |
|---|---|---|
| Description | Sets or queries the CV mode "A Value" voltage or the +CV voltage value. | |
| ⚠ Note | The same value applies for each current range (High/Low). | |
| | The optional command node [:VA] can only be omitted when in static mode. | |
| Syntax | :VOLTage[:VA] {<NRf>[ V ] | MINimum | MAXimum } | |
| Query Syntax | :VOLTage[:VA]? | |
| Parameter | <NRf>[ V ] | "A Value" voltage value |
| | MINIMUM or MIN | Minimum voltage level |
| | MAXIMUM or MAX | Maximum voltage level |

| Return parameter | <NR2> | Returns the "A Value" voltage value. |
|---|---|---|

| Example | :VOLTage:VA MIN |
|---|---|
| | Sets the voltage value to the minimum. |

| Query example | :VOLTage:VA? |
|---|---|
| | >15.0000 |
| | Returns the "A Value" voltage setting (15). |

## :VOLTage:VB

Set →
→ Query
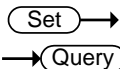
| Description | Sets or queries the CV mode "B Value". |
|---|---|
| ⚠ Note | The same value applies for each current range (High/Low). |
| Syntax | :VOLTage:VB {<NRf>[ V ] | MINimum | MAXimum } |
| Query Syntax | :VOLTage:VB? |

| Parameter | <NRf>[ V ] | "B Value" voltage value |
|---|---|---|
| | MINIMUM or MIN | Minimum voltage level |
| | MAXIMUM or MAX | Maximum voltage level |

| Return parameter | <NR2> | Returns the "B Value" voltage value. |
|---|---|---|

| Example | :VOLTage:VB MIN |
|---|---|
| | Sets the voltage value to the minimum. |

| Query example | :VOLTage:VB? |
|---|---|
| | >15.0000 |
| | Returns the "B Value" voltage setting. |

## :VOLTage:RESPonse

Set →
→ Query
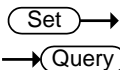
| Description | Sets or returns the response speed for CV Mode. Slow is the default response speed. |
|---|---|
| Syntax | :VOLTage:RESPonse { FAST | NORMal | SLOW } |
| Query Syntax | :VOLTage:RESPonse? |

| Parameter | FAST | Fast response |
|---|---|---|
| | NORMal | Normal response |

| | SLOW | Slow response |
|---|---|---|
| Return parameter | <ASCII string> | Returns response speed as a string. |
| Example | :VOLTage:RESPonse FAST | |
| | Sets the response to fast. | |
| Query example | :VOLTage:RESPonse?<br>> FAST | |
| | The response is FAST. | |

## Power Subsystem Commands

## :POWer:RECall

Set →
→ Query
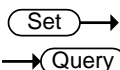
| Description | Sets or queries which power level is active (CP A Value or CP B Value). | |
|---|---|---|
| Syntax | :POWer:RECall {<bool> \| A \| B } | |
| Query Syntax | :POWer:RECall? | |
| Parameter | A, 0 | A Value |
| | B, 1 | B Value |
| Return parameter | A | A Value |
| | B | B Value |
| Example | :POWer:RECall 0 | |
| | Sets the power setting to A Value. | |

## :POWer[:VA]

Set →
→ Query
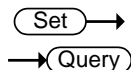
| Description | Sets or queries the CP mode "A Value" power. | |
|---|---|---|
| ⚠ Note | A different "A Value" can be set for different current ranges. | |
| Syntax | :POWer[:VA] {<NRf>[ W ] \| MINimum \| MAXimum } | |
| Query Syntax | :POWer[:VA]? | |
| Parameter | <NRf>[ W ] | "A Value" power value |
| | MINIMUM or MIN | Minimum power level |
| | MAXIMUM or MAX | Maximum power level |
| Return parameter | <NR2> | Returns the "A Value" power value. |

| | |
|---|---|
| Example | :POWer:VA MAX |
| | Sets the power value to the minimum. |
| Query example | :POWer:VA?<br>>306.00 |
| | Returns the "A Value" power setting (306W). |

:POWer:VB

Set ⟶

⟶ Query

| | | |
|---|---|---|
| Description | Sets or queries the CP mode "B Value" power. This command is only applicable to static mode. | |
| ⚠ Note | A different "B Value" can be set to different current ranges. | |
| Syntax | :POWer:VB {<NRf>[ W ] \| MINimum \| MAXimum } | |
| Query Syntax | :POWer:VB? | |
| Parameter | <NRf>[W]<br>MINIMUM or MIN<br>MAXIMUM or MAX | "B Value" power value<br>Minimum power level<br>Maximum power level |
| Return parameter | <NR2> | Returns the "B Value" power value. |
| Example | :POWer:VB 10 | |
| | Sets the power value to 10W. | |
| Query example | :POWer:VB?<br>>10.00 | |
| | Returns the "B Value" power setting. | |

:POWer:RESPonse

Set ⟶

⟶ Query

| | | |
|---|---|---|
| Description | Sets or returns the response speed for CP Mode. Slow is the default response speed. | |
| Syntax | :POWer:RESPonse { FAST \| NORMal \| SLOW } | |
| Query Syntax | :POWer:RESPonse? | |
| Parameter | FAST<br>NORMal<br>SLOW | Fast response<br>Normal response<br>Slow response |

| Return parameter | <ASCII string> | Returns response speed as a string. |
|---|---|---|

| Example | :POWer:RESPonse FAST |
|---|---|
| | Sets the response to fast. |

| Query example | :POWer:RESPonse? |
|---|---|
| | > FAST |
| | The response is FAST. |

Set ⟶

⟶ Query
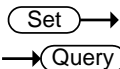
## :POWer:CV:RESPonse

| Description | Sets or returns the +CV response speed for CP Mode. Slow is the default response speed. |
|---|---|

| Syntax | :POWer:CV:RESPonse { FAST \| SLOW } |
|---|---|

| Query Syntax | :POWer:CV:RESPonse? |
|---|---|

| Parameter | FAST | Fast response |
|---|---|---|
| | SLOW | Slow response |

| Return parameter | <ASCII string> | Returns +CV response speed as a string. |
|---|---|---|

| Example | :POWer:CV:RESPonse FAST |
|---|---|
| | Sets the response to fast. |

| Query example | :POWer:CV:RESPonse? |
|---|---|
| | > FAST |
| | The +CV response is FAST. |

## Function Commands

## :PROGram:STATe

Set →
→ Query

| Description | Sets or queries the state of the program function. | |
|---|---|---|
| Syntax | :PROGram:STATe {ON\|OFF\|PAUSe\|CONTinue\|NEXT} | |
| Query Syntax | :PROGram:STATe? {{ON,STOP\|RUN\|PAUSe}\|OFF} | |
| Parameter | ON | Turn program on |
| | OFF | Program off |
| | PAUse | Program pause |
| | RUN | Program running |
| | NEXT | Next step in the program |
| | CONTinue | Program continue |
| Return Parameter | ON,STOP | Program is on, stopped |
| | ON,PAUSE | Program is paused |
| | ON,RUN | Program is running |
| | OFF | Program is off |
| Example | :PROGram:STATe ON | |
| | Turns "Program" on. | |
| Query example | :PROGram:STATe? | |
| | >OFF | |
| | "Program" is off. | |

Set →

→ Query

## :PROGram

| | |
|---|---|
| Description | Configures the parameters for the Program sequence. This remote command is equivalent to all the settings in the "Timing Edit for Program" menu when entering the FUNC>PROGRAM menu. |
| Syntax | :PROGram {<NR1>,<NR1>,<NR1>,SKIP\|AUTO\| MANual,<NR2>,<NR2>\|OFF,<NR2>\|OFF,<NR2>\|OFF} |
| Query Syntax | :PROGram? |
| Related Commands | :PROGram:STATe |

| Parameter | Parameters (In order) | Description |
|---|---|---|
| | <NR1> | Program number (1~16) |
| | <NR1> | Step number (1~16) |
| | <NR1> | Memory number (1~256) |
| | SKIP\|AUTO\|MANual | Run parameter: |
| | <NR2> | On-Time (0.1~60)s |
| | <NR2>\|OFF | Off-Time (0.1~60)s |
| | <NR2>\|OFF | P/F Time (0.0~119.9)s |
| | <NR2>\|OFF | Short-Time (0.1 ~ On-Time)s |
| Return parameter | <string> | Returns the program settings in the following order: Program: <NR1>; Start: <NR1>, Step:XX, Memory: <NR1>, Run: Skip\|Automatic\|Manual, On-Time:<NR2>, Off-Time: <NR2>\|OFF, P/F-Time: <NR2>\|OFF, Short-Time: <NR2>\|OFF |

| Example | :PROGram: 1,1,1,AUTO,2.0,OFF,1.0,OFF |
|---|---|
| | Sets the program settings as follows: Program no. = 1, Step no. = 1, memory number = M001, Run = AUTO, On-Time = 2.0 secs, Off-Time = OFF, P/F Time = 1.0 secs, Short-Time = OFF. |
| Query example | :PROGram?<br>>Start:1, Step:1, Memory:1, Run:Manual, On-Time:2.0, Off-Time:0.0, P/F-Time:0.0, Short-Time:0.0 |
| | Returns the program settings for program number 1, at step 1. |

## :PROGram[:RECall]:DEFault　　　　　　　⟨ Set ⟩⟶

| Description | The default settings for the program sequence function. |
|---|---|
| Syntax | :PROGram[:RECall]:DEFault |
| Example | :PROGram:DEFault |

## :PROGram:STARt　　　　　　　⟨ Set ⟩⟶<br>　　　　　　　⟶⟨ Query ⟩

| Description | Sets or queries the program number for the program sequence. |
|---|---|
| Syntax | :PROGram:STARt {<NR1>} |
| Query Syntax | :PROGram:STARt? |
| Parameter | <NR1> | (1~16) program number |
| Return Parameter | <string> | Start:<NR1> |
| Example | :PROGram:STARt 1 |
| | Sets the program number to 1. |
| Query example | :PROGram:STARt?<br>>Start:1 |
| | Returns the program number (1). |

Set ⭢

:PROGram:STEP ⭢ Query

| Description | Sets or queries the step number for the program sequence. | |
|---|---|---|
| Syntax | :PROGram:STEP {<NR1>} | |
| Query Syntax | :PROGram:STEP? | |
| Parameter | <NR1> | (1~16) step number |
| Return Parameter | <string> | Step:<NR1> |
| Example | :PROGram:STEP 1 | |
| | Sets the step number to 1. | |
| Query example | :PROGram:STEP?<br>>Step:1 | |
| | Returns the step number (1). | |

Set ⭢

:PROGram:MEMory ⭢ Query

| Description | Sets or queries the memory number for the program sequence. | |
|---|---|---|
| Syntax | :PROGram:MEMory{<NR1>} | |
| Query Syntax | :PROGram:MEMory? | |
| Parameter | <NR1> | (1~256) memory number |
| Return Parameter | <string> | Memory:M <NR1> |
| Example | :PROGram:MEMory 1 | |
| | Sets the memory number to 1. | |
| Query example | :PROGram:MEMory?<br>>Memory:M 1 | |
| | Returns the memory number (1). | |

Set ⭢

:PROGram:RUN ⭢ Query

| Description | Sets or queries the "Run" state for the selected program step in the program sequence. |
|---|---|

| Syntax | :PROGram:RUN{SKIP|AUTO|MANual} | |
|---|---|---|
| Query Syntax | :PROGram:RUN? { Run:Skip|Automatic|Manual} | |
| Parameter | SKIP | Sets the run state to skip |
| | AUTO | Sets the run state to automatic |
| | MANual | Sets the run state to manual |
| Return Parameter | Run:Skip | Run state = Skip |
| | Run:Automatic | Run state = Automatic |
| | Run:Manual | Run state = Manual |
| Example | :PROGram:RUN SKIP | |
| | Skips the selected step. | |
| Query example | :PROGram:RUN? >Run:Skip | |
| | Returns the "Run" state (skip). | |

## :PROGram:ONTime

Set →
→ Query
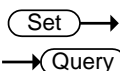
| Description | Sets or queries the On-Time for the selected step number.  Resolution 0.1 seconds. | |
|---|---|---|
| Syntax | :PROGram:ONTime {<NR2>[s]} | |
| Query Syntax | :PROGram:ONTime? | |
| Parameter | <NR2>[s] | (0.1~60) On-Time in seconds Turns the On-Time off. |
| Return Parameter | <string> | On-Time: <NR2> |
| Example | :PROGram:ONTime 1 | |
| | Sets the On-Time to 1 second. | |
| Query example | :PROGram:ONTime? >On-Time:1.0 | |
| | Returns the On-Time (1 second). | |

## :PROGram:OFFTime

Set →
→ Query
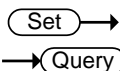
| Description | Sets or queries the Off-Time for the selected step number. |
|---|---|

| Syntax | :PROGram:OFFTime {<NR2>[s]|OFF} | |
|---|---|---|
| Query Syntax | :PROGram:OFFTime? | |
| Parameter | <NR2>[s] | (0~60) Off-Time in seconds |
| | OFF | Turns the Off-Time off. |
| Return Parameter | <string> | Off-Time:<NR2> (a time of 0.0 is equivalent to "OFF") |
| Example | :PROGram:OFFTime 1 | |
| | Sets the Off-Time to 1 second. | |
| Query example | :PROGram:OFFTime? >Off-Time:1.0 | |
| | Returns the Off-Time (1 second). | |

## :PROGram:PFTime

Set →
→ Query
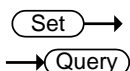
| Description | Sets or queries the Pass/Fail-Time for the selected step number. | |
|---|---|---|
| Syntax | :PROGram:PFTime {<NR2>[s]|OFF} | |
| Query Syntax | :PROGram:PFTime? | |
| Parameter | <NR2>[s] | (0 ~ (On-time + Off time - 0.1)) (in seconds) |
| | OFF | Turns the P/F-Time off. |
| Return Parameter | <string> | P/F-Time:<NR2> |
| Example | :PROGram:PFTime 1 | |
| | Sets the P/F-Time to 1 second. | |
| Query example | :PROGram:PFTime? > P/F-Time:0.0 | |
| | Returns the P/F-Time (1 second). | |

## :PROGram:STIMe

Set →
→ Query
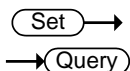
| Description | Sets or queries the Short-Time for the selected step number. |
|---|---|

| Syntax | :PROGram:STIMe {<NR2>[s]|OFF} | |
|---|---|---|
| Query Syntax | :PROGram:STIMe? | |
| Parameter | <NR2>[s] | (0~On-Time) (in seconds) |
| | OFF | Turns the Short-Time off. |
| Return Parameter | <string> | Short-Time:<NR2> |
| Example | :PROGram:STIMe 1 | |
| | Sets the Short-Time to 1 second. | |
| Query example | :PROGram:STIMe? | |
| | >Short-Time:1.0 | |
| | Returns the Short-Time (1 second). | |

## [:PROGram]:CHAin:STARt

Set →
→ Query

| Description | Sets or queries the first program number for the chain. | |
|---|---|---|
| Syntax | [:PROGram]:CHAin:STARt {<NR1>} | |
| Query Syntax | [:PROGram]:CHAin:STARt? | |
| Parameter | <NR1> | (1~16) Program number |
| Return Parameter | <string> | P1 ~ P16 |
| Example | :CHAin:STARt 1 | |
| | Sets program 1 as the start of the program chain. | |
| Query example | :CHAin:STARt? | |
| | >P1 | |
| | Returns the program number that starts the chain. | |

## [:PROGram]:CHAin

Set →
→ Query

| Description | Sets or queries which programs are added to program chain and the order of those programs in the program chain. |
|---|---|

| Syntax | [:PROGram]:CHAin<br>{<NR1>\|OFF, <NR1>\|OFF, <NR1>\|OFF, <NR1>\|OFF,<br><NR1>\|OFF, <NR1>\|OFF, <NR1>\|OFF, <NR1>\|OFF,<br><NR1>\|OFF, <NR1>\|OFF, <NR1>\|OFF, <NR1>\|OFF,<br><NR1>\|OFF, <NR1>\|OFF, <NR1>\|OFF, <NR1>\|OFF} | |
|---|---|---|
| Query Syntax | [:PROGram]:CHAin? | |
| Parameter | Parameters in order: | Description |
| | <NR1>\|OFF | (1~16) Program no. that follows P01. |
| | <NR1>\|OFF | (1~16) Program no. that follows P02. |
| | <NR1>\|OFF | (1~16) Program no. that follows P03. |
| | <NR1>\|OFF | (1~16) Program no. that follows P04. |
| | <NR1>\|OFF | (1~16) Program no. that follows P05. |
| | <NR1>\|OFF | (1~16) Program no. that follows P06. |
| | <NR1>\|OFF | (1~16) Program no. that follows P07. |
| | <NR1>\|OFF | (1~16) Program no. that follows P08. |
| | <NR1>\|OFF | (1~16) Program no. that follows P09. |
| | <NR1>\|OFF | (1~16) Program no. that follows P10. |
| | <NR1>\|OFF | (1~16) Program no. that follows P11. |
| | <NR1>\|OFF | (1~16) Program no. that follows P12. |
| | <NR1>\|OFF | (1~16) Program no. that follows P13. |
| | <NR1>\|OFF | (1~16) Program no. that follows P14. |
| | <NR1>\|OFF | (1~16) Program no. that follows P15. |
| | <NR1>\|OFF | (1~16) Program no. that follows P16. |
| Return Parameter | <string> | P1->XX;P2->XX;P3->XX;P4->XX;<br>P5->XX;P6->XX;P7->XX;P8->XX;<br>P9->XX;P10->XX;P11->XX;P12->XX;<br>P13->XX;P14->XX;P15->XX;P16->XX<br><br>Where XX = P1 ~ P16 or OFF |
| Example | :CHAin 4,OFF,OFF,5,6,OFF,OFF,OFF,OFF,OFF,OFF,<br>OFF,OFF, OFF,OFF,OFF | |
| | Configures the program chain as follows: | |
| | P01→P04→P05→P06→OFF | |

| | |
|---|---|
| Query example | :CHAin?<br>>P1->P4;P2->Off;P3->Off;P4->P5;P5->P6;P6->Off;<br>  P7->Off;P8->Off;P9->Off;P10->Off;P11->Off;<br>  P12->Off;P13->Off;P14->Off;P15->Off;P16->Off |
| | Returns the program chain configuration. |

## [:PROGram]:CHAin:P2P

Set →
→ Query

| | |
|---|---|
| Description | Sets or queries a single link in the program chain. |
| Syntax | [:PROGram]:CHAin:P2P {<NR1>,<NR1>\|OFF} |
| Query Syntax | [:PROGram]:CHAin:P2P? {P1\|P2\|P3\|P4\|P5\|P6\|P7\|P8\|P9<br>\|P10\|P11\|P12\|P13\|P14\|P15\|P16} |

| Parameter | Parameters in order: | Description |
|---|---|---|
| | <NR1> | (1~16) Program number of single link. |
| | <NR1>\|OFF | (1~16) Program number that follows. |
| Query Parameter | P1 ~ P16 | The program number of the first link of the chain that you are inquiring about. |
| Return Parameter | PX->PX\|Off | Returns the program number that follows or Off. |

| | |
|---|---|
| Example | :CHAin:P2P 4,6 |
| | Configures the program number 6 to follow after program number 4: |
| | P04→P06 |
| Query example | :CHAin:P2P? P4<br>>P4->6 |
| | Indicates that program 6 follows from program 4 in the chain. |

## [:PROGram]:CHAin[:RECall]:DEFault

Set →

| | |
|---|---|
| Description | Recalls the default settings for the program chain. |
| Syntax | [:PROGram]:CHAin[:RECall]:DEFault |
| Example | :CHAin:DEFault |

## :PROGram:SAVE

<span style="float:right;">Set ⟶</span>

| | |
|---|---|
| Description | Saves the program chain. |
| Syntax | :PROGram:SAVE |
| Example | :PROG:SAVE |

## :NSEQuence:STATe

<span style="float:right;">Set ⟶<br>⟶ Query</span>

| | | |
|---|---|---|
| Description | Sets or queries the state of the Normal Sequence function. | |
| Syntax | :NSEQuence:STATe {OFF\|ON\|PAUSe\|CONTinue\|NEXT} | |
| Query Syntax | :NSEQuence:STATe? {ON,STOP\|RUN\|PAUSe}\|OFF | |
| Parameter | OFF | Normal sequence Off |
| | ON | Normal sequence On |
| | PAUSe | Pause sequence |
| | CONTinue | Continue sequence |
| | NEXT | Go to next sequence |
| Return Parameter | ON,STOP | Sequence has stopped |
| | ON, RUN | Sequence is running |
| | ON, PAUSE | Sequence is paused |
| | OFF | Sequence is off |
| Example | :NSEQuence:STATe ON | |
| | Turns "Normal Sequence" on. | |
| Query example | :NSEQuence:STATe?<br>>OFF | |
| | "Normal Sequence" is off. | |

Set →

→ Query
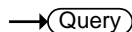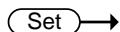
## :NSEQuence

| | |
|---|---|
| Description | Configures the parameters for the Normal Sequence. This remote command is equivalent to the all the settings in the "Timing Edit for Normal Sequence" menu when entering the FUNC>Sequence>Normal Sequence menu. |
| ⚠ Note | CR mode unit = mS |
| Syntax | :NSEQuence<br>{ <NR1>,<NR1>,<string>,CC\|CR\|CV\|CP,ILVL\|IHVL\|<br>ILVH\|IHVH,<NR1>\|INFinity,<bool>\|OFF\|ON,<NR2>,<<br>NR1>\|OFF} |
| Query Syntax | :NSEQuence? |
| Related Commands | :NSEQuence:STATe |

| Parameter | Parameter in order: | Description |
|---|---|---|
| | <NR1> | Starting sequence number |
| | <NR1> | Current sequence number |
| | <string> | Memo message. (need to enclose in quotes) |
| | CC\|CR\|CV\|CP | Mode |
| | ILVL\|IHVL\|ILVH\| IHVH | Range |
| | <NR1>\|INFinity | Sets the loop number |
| | <bool>\|ON\|OFF | Last Load 0=Off, 1=On. Sets the load condition after the end of the sequence. |
| | <NR2> | Last value: Sets the value of the load condition if Last Load = On |
| | <NR1>\|OFF | Chain: Sets the next sequence in the chain. |

| Return parameter | <string> | Returns the normal sequence settings in the following order: Start:<NR1>, Seq No: <NR1>, Memo:<string>, Mode:CC|CR|CV|CP, Range: ILVL|IHVL|ILVH|IHVH, Loop:<NR1>, Last Load: OFF|ON, Last:<NR2>, Chain:<NR1> |
|---|---|---|
| Example | :NSEQuence: 1,1,"hi",CC,ILVL,1,ON,0.000,2 | |
| | Sets the normal sequence configuration to that shown below. | |
| Query example | :NSEQuence:? > Start:1, Seq No:1, Memo: HI, Mode:CC, Range:IHVL, Loop:1, Last Load:ON, Last:0.000, Chain:2 | |
| | Returns the normal sequence settings. | |

## :NSEQuence:STARt

Set →
→ Query

| Description | Sets or queries the starting sequence number. | |
|---|---|---|
| Syntax | :NSEQuence:STARt {<NR1>} | |
| Query Syntax | :NSEQuence:STARt? | |
| Parameter | <NR1> | (1~10) sequence number |
| Return Parameter | <string> | Start:<NR1> |
| Example | :NSEQuence:STARt 1 | |
| | Sets the starting sequence number to 1. | |
| Query example | :NSEQuence:STARt? >Start:1 | |
| | Returns the starting sequence number (1). | |

Set →
→ Query

## :NSEQuence:NUMBer

| | | |
|---|---|---|
| Description | Sets or queries the current sequence number for the normal sequence. | |
| Syntax | :NSEQuence:NUMBer {<NR1>} | |
| Query Syntax | :NSEQuence:NUMBer? | |
| Parameter/ | <NR1> | (1~10) sequence number |
| Return Parameter | <string> | Seq No:<NR1> |
| Example | :NSEQuence:NUMBer 1 | |
| | Sets the sequence number to 1. | |
| Query example | :NSEQuence:NUMBer?<br>>Seq No:1 | |
| | Returns the sequence number (1). | |

Set →
→ Query

## :NSEQuence:MEMO

| | | |
|---|---|---|
| Description | Sets or queries the memo string for the normal sequence. | |
| Syntax | :NSEQuence:MEMO {<string>} | |
| Query Syntax | :NSEQuence:MEMO? | |
| Parameter | <string> | ASCII character string max of 12 characters. First character must be an alpha character. String must be enclosed in quotes. |
| Return Parameter | <string> | Memo: <string> |
| Example | :NSEQuence:MEMO "HI" | |
| | Sets the memo string to "HI". | |
| Query example | :NSEQuence:MEMO?<br>>Memo: HI | |
| | Returns the memo string. | |

Set →

→ Query

:NSEQuence:MODE

| Description | Sets or queries the mode used for the normal sequence. | |
| --- | --- | --- |
| Syntax | :NSEQuence:MODE {CC\|CR\|CV\|CP} | |
| Query Syntax | :NSEQuence:MODE? | |
| Parameter | CC | Constant current mode |
| | CR | Constant resistance mode |
| | CV | Constant voltage mode |
| | CP | Constant power mode |
| Return Parameter | <string> | Mode: CC\|CR\|CV\|CP |
| Example | :NSEQuence:MODE  CC | |
| | Sets the mode to CC. | |
| Query example | :NSEQuence:MODE?<br>>Mode: CC | |
| | Returns the mode. | |

Set →

→ Query

:NSEQuence:RANGe

| Description | Sets or queries the range used for the normal sequence. | |
| --- | --- | --- |
| Syntax | :NSEQuence:RANGe { ILVL\|IHVL\|ILVH\|IHVH } | |
| Query Syntax | :NSEQuence:RANGe? | |
| Parameter | ILVL | Low I range, low V range |
| | IHVL | High I range, low V range |
| | ILVH | Low I range, high V range |
| | IHVH | High I range, high V range |
| Return Parameter | <string> | Returns the range in the following format:<br>Range: ILVL |
| Example | :NSEQuence:RANGe IHVL | |
| | Sets the range to IHVL. | |

| Query example | :NSEQuence:RANGe?<br>>Range: IHVL |
|---|---|
| | Returns the range. |

Set →
:NSEQuence:LOOP                                    → Query

| Description | Sets or queries the number of loops for the sequence. |
|---|---|
| Syntax | :NSEQuence:LOOP {<NR1>\|INFinity} |
| Query Syntax | :NSEQuence:LOOP? |
| Parameter | <NR1><br>INFinity | (1~9999) Sets the number of loops<br>Sets the number of loops to infinite. |
| Return Parameter | <string> | Returns the number of loops in the following format:<br>Loop: <NR1>\|Infinity |
| Example | :NSEQuence:LOOP 1 | |
| | Sets the number of loops to 1. | |
| Query example | :NSEQuence:LOOP?<br>>Loop: 1 | |
| | Returns the number of loops(1). | |

Set →
:NSEQuence:LLOad                                   → Query

| Description | Sets or queries the Last Load state. |
|---|---|
| Syntax | :NSEQuence:LLOad {<bool>\|ON\|OFF} |
| Query Syntax | :NSEQuence:LLOad? |
| Parameter | ON, 1<br>OFF, 0 | Turns Last Load on.<br>Turns Last Load off. |
| Return Parameter | <string> | Returns the Last Load state in the following format:<br>Last Load:ON\|OFF |
| Example | :NSEQuence:LLOad ON | |
| | Turns Last Load on. | |

Query example   :NSEQuence:LLOad?
                >Last Load: ON

                Returns the last load state.

Set →
## :NSEQuence:LAST                                      → Query

| | |
|---|---|
| Description | Sets or queries the value of Last Load when Last Load = ON. The unit depends on the Mode. |

⚠Note | CR mode unit = mS

Syntax | :NSEQuence:LAST{<NR2>}

Query Syntax | :NSEQuence:LAST?

| Parameter | <NR2> | current value(CC), resistance value(CR), voltage value(CV), power value(CP). |
|---|---|---|
| Return Parameter | <string> | Returns the value for the Last Load in the following format: Last:<NR2> |

Example | :NSEQuence:LAST 1.000

Sets the value of Last Load to 1.000.

Query example   :NSEQuence:LAST?
                >Last: 1.000

                Returns the value of Last Load.

Set →
## :NSEQuence:CHAin                                     → Query

| | |
|---|---|
| Description | Sets or queries the next sequence in the chain. |

Syntax | :NSEQuence:CHAin {<NR1>|OFF}

Query Syntax | :NSEQuence:CHAin?

| Parameter | <NR1> | The sequence number of the next sequence in the chain. |
|---|---|---|
| | OFF | End the chain at the current sequence. |

| Return Parameter | <string> | Returns the next sequence in the chain in the following format: Chain:<NR1>|OFF |
|---|---|---|
| Example | :NSEQuence:CHAin 2 | |
| | Sets the next sequence in the chain as S02. | |
| Query example | :NSEQuence:CHAin?<br>>Chain: 2 | |
| | Returns the next sequence in the chain (S02). | |

Set →→
→ Query
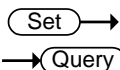
## :NSEQuence:EDIT

| Description | Configures the parameters for the Data Edit for Normal Sequence menu, equivalent to entering the FUNC>Sequence>Normal Sequence>Data Edit menu. |
|---|---|
| ⚠ Note | CR mode unit = mS |
| Syntax | :NSEQuence:EDIT<br>{<NR1>,<NR1>,<NR2>,<NR1>,<NR1>,<NR1>,<NR1><br>,<bool>|OFF|ON,<bool>|OFF|ON,<bool>|OFF|ON,<br><bool>|OFF|ON} |
| Query Syntax | :NSEQuence:EDIT? |
| Related Commands | :NSEQuence: |

| Parameter | Parameters in order: | Description |
|---|---|---|
| | <NR1> | Current step number |
| | <NR1> | Total number of steps |
| | <NR2> | Value |
| | <NR1> | Time: hours |
| | <NR1> | Time: minutes |
| | <NR1> | Time: seconds |
| | <NR1> | Time: milliseconds |
| | <bool>|OFF|ON | LOAD setting |
| | <bool>|OFF|ON | TRIG OUT setting |
| | <bool>|OFF|ON | RAMP setting |
| | <bool>|OFF|ON | PAUSE setting |

| Return parameter | <string> | Returns the settings for the Data Edit for Normal Sequence menu: Step: <NR1>/<NR1>, Value: <NR1>, Time: <NR1>H: <NR1>M: <NR1>S: <NR1>ms, LOAD:ON\|OFF, TRIG OUT:ON\|OFF, RAMP: ON\|OFF, PAUSE: ON\|OFF |
|---|---|---|
| Example | :NSEQuence:EDIT 3,12,0.1,0,1,0,0,ON,OFF,OFF,ON | |
| | Configures the Data Edit for Normal Sequence menu to the settings shown below. | |
| Query example | :NSEQuence:EDIT? <br> > Step:3/12, Value:0.10000, Time:0H:0M:1S:0ms, LOAD:ON, TRIG OUT:OFF, RAMP:OFF, PAUSE:ON | |

## :NSEQuence:EDIT:POINt

Set →
→ Query

| Description | Sets or queries the current point in the sequence. | |
|---|---|---|
| Syntax | :NSEQuence:EDIT:POINt {<NR1>} | |
| Query Syntax | :NSEQuence:EDIT:POINt? | |
| Related Commands | :NSEQuence:EDIT:END? | |
| Parameter/ Return parameter | <NR1> | 1~1000*. The current point in the sequence. *Depends on the number of steps in the sequence. |
| Example | :NSEQuence:EDIT:POINt 10 | |
| | Sets the current point to 10. | |
| Query example | :NSEQuence:EDIT:POINt? <br> >10 | |
| | Returns the current point in the sequence. | |

## :NSEQuence:EDIT:END?        ➝ Query

| | |
|---|---|
| Description | Returns the number of points in the sequence. |
| Query Syntax | :NSEQuence:EDIT:END? |
| Return parameter | <NR1>                     1~1000 |
| Query example | :NSEQuence:EDIT:END? <br> > 20 <br><br> Indicates that there are 20 steps in the sequence. |

## :NSEQuence[:DELet]:ALL        Set ➝

| | |
|---|---|
| Description | Deletes all the data points for the normal sequence. |
| Syntax | :NSEQuence[:DELet]:ALL |
| Example | :NSEQuence:ALL |

## :NSEQuence:SAVE        Set ➝

| | |
|---|---|
| Description | Saves the data points for the normal sequence. |
| Syntax | :NSEQuence:SAVE |
| Example | :NSEQuence:SAVE |

## :FSEQuence:STATe        Set ➝ <br> ➝ Query

| | | |
|---|---|---|
| Description | Sets or queries the state of the Fast Sequence function. | |
| Syntax | :FSEQuence:STATe {ON | OFF} | |
| Query Syntax | :FSEQuence:STATe? | |
| Parameter | ON <br> OFF | Turn fast sequence on <br> Turn fast sequence off |
| Return Parameter | ON,STOP <br> RUN,STOP <br> OFF | Fast sequence is on <br> Fast sequence is running <br> Fast sequence is off |

| Example | :FSEQuence:STATe ON |
|---|---|
| | Turns "Fast Sequence" on. |

| Query example | :FSEQuence:STATe? |
|---|---|
| | >OFF |
| | "Fast Sequence" is off. |

Set →

## :FSEQuence

→ Query

| Description | Configures the parameters for the Fast Sequence. This remote command is equivalent to all the settings in the "Timing Edit for Fast Sequence" menu when entering the FUNC>Sequence>Fast Sequence menu. |
|---|---|

| ⚠ Note | CR mode unit = mS |
|---|---|

| Syntax | :FSEQuence {<string>,CC|CR, ILVL|IHVL|ILVH|IHVH, <NR1>|INFinity,<NR2>,<bool>|OFF|ON,<NR2>, <NR1>} |
|---|---|

| Query Syntax | :FSEQuence? |
|---|---|

| Related Commands | :FSEQuence:STATe |
|---|---|

| Parameter | Parameter in order: | Description |
|---|---|---|
| | <string> | Memo message. (Need to enclose in quotes) |
| | CC|CR | Mode |
| | ILVL|IHVL|ILVH|IHVH | Range |
| | <NR1>|INFinity | Number of loops |
| | <NR2> | Time base in seconds |
| | <bool> ON|OFF | Last Load 0=Off, 1=On. Sets the load condition after the end of the sequence. |
| | <NR2> | Last value: Sets the value of the load condition for Last Load = On |
| | <NR1> | Repeat Step from (0001~1000) per loop |

| Return parameter | \<string> | Returns the fast sequence settings in the following order: FSeq:ON\|OFF; Memo:\<string>, Mode: CC\|CR, Range: ILVL\|IHVL\|ILVH\|IHVH, Loop: \<NR1>\|Infinity, Time Base:\<NR2>, Last Load:ON\|OFF, Last:\<NR2>, RPTSTEP:\<NR1> |
|---|---|---|
| Example | :FSEQuence HI,CC,ILVL,5,0.3,1,0.7,11 | |
| | Sets the fast sequence configuration to that shown below. | |
| Query example | :FSEQuence? >Memo: HI, Mode:CC, Range:ILVL, Loop:5, Time Base:300.00, Last Load:OFF, Last:0.70000, RPTSTEP:11 | |
| | Returns the fast sequence settings. | |

## :FSEQuence:MEMO

Set →
→ Query

| Description | Sets or queries the memo string for the fast sequence. | |
|---|---|---|
| Syntax | :FSEQuence:MEMO {\<string>} | |
| Query Syntax | :FSEQuence:MEMO? | |
| Parameter | \<string> | ASCII character string max of 12 characters. First character must be an alpha character. The string must be enclosed in quotes. |
| Return Parameter | \<string> | Memo:\<string> |
| Example | :FSEQuence:MEMO "HI" | |
| | Sets the memo string to "HI". | |

| Query example | :FSEQuence:MEMO? |
|---|---|
| | >Memo: HI |

Returns the memo string.

Set →
→ Query

## :FSEQuence:MODE

| Description | Sets or queries the mode used for the fast sequence. |
|---|---|
| Syntax | :FSEQuence:MODE {CC\|CR} |
| Query Syntax | :FSEQuence:MODE? |
| Parameter | CC | Constant current mode |
| | CR | Constant resistance mode |
| Return Parameter | <string> | Mode: CC\|CR |
| Example | :FSEQuence:MODE  CC |
| | Sets the mode to CC. |
| Query example | :FSEQuence:MODE? |
| | >Mode: CC |
| | Returns the mode. |

Set →
→ Query

## :FSEQuence:RANGe

| Description | Sets or queries the range used for the fast sequence. |
|---|---|
| Syntax | :FSEQuence:RANGe { ILVL\|IHVL\|ILVH\|IHVH } |
| Query Syntax | :FSEQuence:RANGe? |
| Parameter | ILVL | Low I range, low V range |
| | IHVL | High I range, low V range |
| | ILVH | Low I range, high V range |
| | IHVH | High I range, high V range |
| Return Parameter | <string> | Returns the range in the following format: Range: ILVL |
| Example | :FSEQuence:RANGe IHVL |
| | Sets the range to IHVL. |

| Query example | :FSEQuence:RANGe?<br>>Range: IHVL |
| --- | --- |
| | Returns the range. |

Set →
→ Query

## :FSEQuence:LOOP

| Description | Sets or queries the number of loops for the sequence. | |
| --- | --- | --- |
| Syntax | :FSEQuence:LOOP {<NR1>\|INFinity} | |
| Query Syntax | :FSEQuence:LOOP? | |
| Parameter | <NR1><br>INFinity | (1~9999) Sets the number of loops<br>Sets the number of loops to infinite. |
| Return Parameter | <string> | Loop:<NR1> |
| Example | :FSEQuence:LOOP 1 | |
| | Sets the number of loops to 1. | |
| Query example | :FSEQuence:LOOP?<br>>Loop:1 | |
| | Returns the number of loops(1). | |

Set →
→ Query

## :FSEQuence:TBASe

| Description | Sets or queries the Time Base. | |
| --- | --- | --- |
| Syntax | :FSEQuence:TBASe {<NR2>[s]} | |
| Query Syntax | :FSEQuence:TBASe? | |
| Parameter | <NR2>[s] | 0.000025 ~ 0.6s |
| Return Parameter | <string> | Returns the Time Base in the following format:<br>Time Base:<NR2> |
| Example | :FSEQuence:TBASe 0.3 | |
| | Sets the time base to 300ms. | |
| Query example | :FSEQuence:TBASe?<br>>Time Base:0.30000 | |
| | Returns the time base. | |

Set ⟶
⟶ Query

## :FSEQuence:LLOad

| Description | Sets or queries the Last Load state. | |
|---|---|---|
| Syntax | :FSEQuence:LLOad {<bool>ON\|OFF} | |
| Query Syntax | :FSEQuence:LLOad? | |
| Parameter | ON, 1<br>OFF, 0 | Turns Last Load on.<br>Turns Last Load off. |
| Return Parameter | <string> | Last Load:ON\|OFF |
| Example | :FSEQuence:LLOad ON | |
| | Turns Last Load on. | |
| Query example | :FSEQuence:LLOad?<br>>Last Load: ON | |
| | Returns the last load state. | |

Set ⟶
⟶ Query

## :FSEQuence:LAST

| Description | Sets or queries the value of Last Load when Last Load = ON. | |
|---|---|---|
| ⚠ Note | CR mode unit = mS | |
| Syntax | :FSEQuence:LAST{<NR2>} | |
| Query Syntax | :FSEQuence:LAST? | |
| Parameter | <NR2> | Current value(CC), resistance value(CR) |
| Return Parameter | <string> | Last:<NR2> |
| Example | :FSEQuence:LAST 1.000 | |
| | Sets the value of Last Load to 1.000. | |
| Query example | :FSEQuence:LAST?<br>>Last: 1.000 | |
| | Returns the value of Last Load. | |

Set →
→ Query

## :FSEQuence:RPTStep

| Description | Sets or queries the last step number per loop. |
|---|---|
| Syntax | :FSEQuence:RPTStep {<NR1>} |
| Query Syntax | :FSEQuence:RPTStep? |
| Parameter | <NR1> | 1~1000 |
| Return Parameter | <string> | RPTSTEP:<NR1> |
| Example | :FSEQuence:RPTStep 11 |
| | Sets the RPTSTEP to 11. |
| Query example | :FSEQuence:RPTStep?<br>> RPTSTEP: 11 |
| | Returns the RPTSTEP number. |

Set →
→ Query
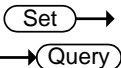
## :FSEQuence:EDIT

| Description | Configures the parameters for the Data Edit for Fast Sequence menu, equivalent to entering the FUNC>Sequence>Fast Sequence>Edit Sequence menu. |
|---|---|
| ⚠ Note | CR mode unit = mS |
| Syntax | :FSEQuence:EDIT {<NR1>,<NR1>,<NR2>,<bool>\|OFF\|ON} |
| Query Syntax | :FSEQuence:EDIT? |
| Related Commands | :FSEQuence: |

| Parameter | Parameters in order:<br><NR1><br><NR1><br><NR2><br><bool>\|OFF\|ON | Description<br>Current step number<br>Total number of steps<br>Value<br>TRIG OUT setting |
|---|---|---|
| Return parameter | <string> | Returns the settings for the Data Edit for Fast Sequence menu:<br>Step:<NR1>/<NR1>,<br>Value:<NR2>, TRIG<br>OUT:ON\|OFF |
| Example | :FSEQuence:EDIT 4,11,0.1,OFF | |
| | Configures the Data Edit for Normal Sequence menu to the settings shown below. | |
| Query example | :FSEQuence:EDIT?<br>> Step:0004/0011; Value:0.10000, TRIG OUT:OFF | |

## :FSEQuence:EDIT:POINt

Set →
→ Query

| Description | Sets or queries the current point in the sequence. |
|---|---|
| Syntax | :FSEQuence:EDIT:POINt {<NR1>} |
| Query Syntax | :FSEQuence:EDIT:POINt? |
| Related Commands | :FSEQuence:EDIT:END? |
| Parameter/<br>Return parameter | <NR1> | 1~1000*. The current point in the sequence. *Depends on the number of steps in the sequence. |
| Example | :FSEQuence:EDIT:POINt 10 | |
| | Sets the current point to 10. | |
| Query example | :FSEQuence:EDIT:POINt?<br>>10 | |
| | Returns the current point in the sequence. | |

## :FSEQuence:EDIT:END?      →( Query )

| | |
|---|---|
| Description | Returns the number of points in the sequence. |
| Query Syntax | :FSEQuence:EDIT:END? |
| Return parameter | <NR1>                  1~1000 |
| Query example | :FSEQuence:EDIT:END? <br> > 20 |
| | Indicates that there are 20 steps in the sequence. |

## :FSEQuence[:DELet]:ALL      ( Set )→

| | |
|---|---|
| Description | Deletes all the data points for the fast sequence. |
| Syntax | :FSEQuence[:DELet]:ALL |
| Example | :FSEQuence:ALL |

## :FSEQuence[:EDIT]:FILL      ( Set )→ <br> →( Query )

| | |
|---|---|
| Description | The FILL function is used to evenly fill up the current or resistance value settings from a starting step to a finishing step. This function is equivalent to entering the FUNC>Sequence>Fast Sequence>Edit Sequence>Fill menu. |
| ⚠Note | CR mode unit = mS |
| Syntax | :FSEQuence[:EDIT]:FILL <br> {<NR2>,<NR2>,<NR1>,<NR1>} |
| Query Syntax | :FSEQuence[:EDIT]:FILL? |
| Related Commands | :FSEQuence:EDIT |

| Parameter | Parameters in order:<br><NR2><br><NR2><br><NR1><br><NR1> | Description<br>Start_Value<br>End_Value<br>Start_Step<br>End_Step |
|---|---|---|
| Return parameter | <string> | Returns the settings for the Fill Edit for Fast Sequence menu:<br>Start Value:<NR2>, End Value:<NR2>, Start Step:<NR1>, End Step:<NR1> |
| Example | :FSEQuence:FILL 0.1,0.5,1,159<br><br>Fills the values of step 1 to step 159 gradually from 0.1A to 0.5A. | |
| Query example | :FSEQuence:FILL?<br>>Start Value:0.10000, End Value:0.50000, Start Step:1, End Step:159 | |

## :FSEQuence:SAVE                                    ⟨Set⟩⟶

| Description | Saves the data points for the fast sequence. |
|---|---|
| Syntax | :FSEQuence:SAVE |
| Example | :FSEQuence:SAVE |

## Utility Commands

## :UTILity:SYSTem?                                    → Query

| | | |
|---|---|---|
| Description | Returns the manufacturer, model, serial number and firmware version number. | |
| Query Syntax | :UTILity:SYSTem? | |
| Return parameter | <string> | model name, serial number, firmware version |
| Query example | :UTILity:SYSTem?<br>>GW,PEL-303XAE,00000001,V1.00 | |
| | Returns the system information. | |
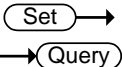
## :UTILity:LOAD                                     Set →<br>→ Query

| | |
|---|---|
| Description | Sets or queries the Auto Load and Auto Load On settings. This is equivalent to the settings in the Shift+UTILITY>Load menu. |
| Syntax | :UTILity:LOAD {OFF\|ON\|LOAD\|PROG\|NSEQ\|FSEQ} |

| Query Syntax | :UTILity:LOAD? | |
|---|---|---|
| Parameter | OFF | Turns Auto Load off. |
| | ON | Turns Auto Load on. This will turn the load on automatically at start up. The type of load function that is automatically turned on is configured below. |
| | LOAD | Configure the load to automatically turn on in CC, CR, CV or CP mode. |
| | PROG | Configure the load to automatically turn on a program sequences. |
| | NSEQ | Configure the load to automatically turn on a normal sequence. |
| | FSEQ | Configure the load to automatically turn on a fast sequence. |
| Return Parameter | <string> | Load:On\|Off, Load On:Load\|Prog\|NSeq\|FSeq |
| Example | :UTILity:LOAD ON FSEQ | |
| | Configures the unit to automatically turn fast sequences on at start up (with the load on). | |
| Query example | :UTILity:LOAD?<br>> Load:On, Load On:Prog | |
| | The unit is configured to automatically turn Programs on at start up (with the load on). | |

Set →

→ Query

## :UTILity:LOAD:MODE

| Description | Sets or queries the Load Off (Mode) setting. This is equivalent to the settings in the Shift+UTILITY>Load menu. |
|---|---|
| Syntax | :UTILity:LOAD:MODE {<bool>\|OFF\|ON} |

| Query Syntax | :UTILity:LOAD:MODE? | |
|---|---|---|
| Parameter | OFF, 0 | Sets Load Off (Mode) to OFF. |
| | ON, 1 | Sets Load Off (Mode) to ON. |
| Return Parameter | OFF | Load Off (Mode) is OFF. |
| | ON | Load Off (Mode) is ON. |
| Example | :UTILity:LOAD:MODE ON | |
| | Turns Load Off (Mode) to on. | |
| Query example | :UTILity:LOAD:MODE?<br>> Off | |
| | Load Off (Mode) is off. | |

## :UTILity:LOAD:RANGe

Set →
→ Query
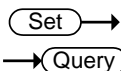
| Description | Sets or queries the Load Off (Range) setting. This is equivalent to the settings in the Shift+UTILITY>Load menu. | |
|---|---|---|
| Syntax | :UTILity:LOAD:RANGe {<bool>\|OFF\|ON} | |
| Query Syntax | :UTILity:LOAD:RANGe? | |
| Parameter | OFF, 0 | Sets Load Off (Range) to OFF. |
| | ON, 1 | Sets Load Off (Range) to ON. |
| Return Parameter | OFF | Load Off (Range) is OFF. |
| | ON | Load Off (Range) is ON. |
| Example | :UTILity:LOAD:RANGe ON | |
| | Turns Load Off (Range) to on. | |
| Query example | :UTILity:LOAD:RANGe?<br>> Off | |
| | Load Off (Range) is off. | |

Set →
→ Query

## :UTILity:INTerface

| | |
|---|---|
| Description | Sets or queries the remote interface. Settings only take effect after a restart. |
| Syntax | :UTILity:INTerface {USB | GPIB} |
| Query Syntax | :UTILity:INTerface? |
| Parameter/ Return Parameter | USB — USB remote interface; GPIB — GPIB remote interface |
| Example | :UTILity:INTerface USB<br>Sets the interface to USB. |
| Query example | :UTILity:INTerface?<br>>USB<br>The current remote interface is USB. |

Set →
→ Query

## :UTILity:TIME

| | |
|---|---|
| Description | Sets or queries the date and time for the unit. |
| Syntax | :UTILity:TIME <NR1>,<NR1>,<NR1>,<NR1>,<NR1> |
| Query Syntax | :UTILity:TIME? |

| Parameter | Parameters in order | Description |
|---|---|---|
| | <NR1> | Month |
| | <NR1> | Day |
| | <NR1> | Year |
| | <NR1> | Hour |
| | <NR1> | Minute |
| Return Parameter | <string> | Month:<NR1>, Day: <NR1>, Year: <NR1>, Hour: <NR1>, Minute: <NR1> |

| | |
|---|---|
| Example | :UTILity:TIME 10,2,2013,8,26<br>Sets the date to October 2, 2013. Sets the time to 8:26. |
| Query example | :UTILity:TIME?<br>> Month:10, Day:2, Year:2013, Hour:8, Minute:26 |

Set ⟶

⟶ Query

## :UTILity:KNOB

| | |
|---|---|
| Description | The Knob Type setting determines if values are updated immediately as they are edited or if they are only updated after the Enter key is pressed. |
| | The Updated setting is applicable for when the load is already on and the user wishes to change the set values (current, voltage, etc.) in real-time. |
| | The Old setting will only update the values after the Enter key is pressed. |
| Syntax | :UTILity:KNOB {OLD|UPDated} |
| Query Syntax | :UTILity:KNOB? |
| Parameter/ Return Parameter | OLD                Old setting<br>UPDated       Updated setting |
| Example | :UTILity:KNOB OLD |
| | Sets the knob type to OLD. |
| Query Example | :UTILity:KNOB? |
| | >Old |

Set ⟶

⟶ Query

## :UTILity:SPEaker

| | |
|---|---|
| Description | Turns the speaker on or off. |
| ⚠️Note | This will not turn the Go/NoGo alarm tone settings. |
| Syntax | :UTILity:SPEaker {<bool>|ON|OFF} |
| Query Syntax | :UTILity:SPEaker? |
| Parameter | ON, 1           Speaker on<br>OFF, 0         Speaker off |
| Return Parameter | On              Speaker on<br>Off            Speaker off |
| Example | :UTILity:SPEaker ON |
| | Turns the speaker on. |

| Query Example | :UTILity:SPEaker? |
|---|---|
| | >On |

## :UTILity:ALARm

Set →
→ Query

| | |
|---|---|
| Description | Enables/disables the alarm tone for the unit. |
| Syntax | :UTILity:ALARm {<bool>\|ON\|OFF} |
| Query Syntax | :UTILity:ALARm? |

| Parameter | ON, 1 | Alarm on |
|---|---|---|
| | OFF, 0 | Alarm off |
| Return Parameter | On | Alarm on |
| | Off | Alarm off |

| | |
|---|---|
| Example | :UTILity:ALARm ON |
| | Enables the alarm tone. |
| Query Example | :UTILity:ALARm? |
| | >On |

## :UTILity:UNReg

Set →
→ Query

| | |
|---|---|
| Description | Enables/disables the UnReg alarm tone for the unit. |
| Syntax | :UTILity:UNReg {<bool>\|ON\|OFF} |
| Query Syntax | :UTILity:UNReg? |

| Parameter | ON, 1 | Alarm on |
|---|---|---|
| | OFF, 0 | Alarm off |
| Return Parameter | On | Alarm on |
| | Off | Alarm off |

| | |
|---|---|
| Example | :UTILity:UNReg ON |
| | Enables the alarm tone. |
| Query Example | :UTILity:UNReg? |
| | >On |

Set →

## :UTILity:GNG

→ Query

| | | |
|---|---|---|
| Description | Enables/disables the Go/NoGo alarm tone for the unit. | |
| Syntax | :UTILity:GNG {<bool>\|ON\|OFF} | |
| Query Syntax | :UTILity:GNG? | |
| Parameter | ON, 1 | Go/NoGo alarm tone on |
| | OFF, 0 | Go/NoGo alarm tone off |
| Return Parameter | On | Go/NoGo alarm tone on |
| | Off | Go/NoGo alarm tone off |
| Example | :UTILity:GNG ON | |
| | Enables the Go/NoGo alarm tone. | |
| Query Example | :UTILity:GNG? | |
| | >On | |

Set →

## :UTILity:CONTrast

→ Query

| | | |
|---|---|---|
| Description | Sets or queries the display contrast level. | |
| Syntax | :UTILity:CONTrast {NR1} | |
| Query Syntax | :UTILity:CONTrast? | |
| Parameter/ Return Parameter | <NR1> | 3~13 (low ~ high) |
| Example | :UTILity:CONTrast 8 | |
| | Sets the contrast to 8. | |
| Query Example | :UTILity:CONTrast? | |
| | >8 | |

Set →

## :UTILity:BRIGhtness

→ Query

| | |
|---|---|
| Description | Sets or queries the display contrast level. |
| Syntax | :UTILity:BRIGhtness {NR1} |

| Query Syntax | :UTILity:BRIGhtness? | |
|---|---|---|
| Parameter/ Return Parameter | <NR1> | 50~90 (low ~ high) |
| Example | :UTILity:BRIGhtness 70 | |
| | Sets the contrast to 70. | |
| Query Example | :UTILity:BRIGhtness? | |
| | >70 | |

## :UTILity:LANGuage

Set →
→ Query

| Description | Sets or queries the language. | |
|---|---|---|
| ⚠️ Note | English is the only available language. | |
| Syntax | :UTILity:LANGuage {ENGLish} | |
| Query Syntax | :UTILity:LANGuage? | |
| Parameter/ Return Parameter | ENGLish | English language settings |
| Example | :UTILity:LANGuage? ENGLish | |
| | Sets the language to English. | |
| Query Example | :UTILity:LANGuage? | |
| | >English | |

## :UTILity:REMote

Set →

| Description | Turns the remote control on or off. | |
|---|---|---|
| Syntax | :UTILity:REMote {OFF\|0\|ON\|1} | |
| Parameter | OFF/0 | Turns Remote control off |
| | ON/1 | Turns remote control on |
| Example | :UTIL:REM 1 | |
| | Turns remote control on. | |

## :UTILity:REMote:MODE

⟨ Set ⟩⟶

| | |
|---|---|
| Description | Sets the remote mode to fast or normal. When in fast mode, the panel interface is deactivated with an interface time of no more than 10ms. Normal mode has an interface time of 30~130ms. In normal mode the display interface continues to update the screen in real-time. |
| Syntax | :UTILity:REMote:MODE {NORMAL|0|FAST|1} |
| Parameter | NORMAL/0      NORMAL |
| | FAST/1      FAST |
| Example | :UTIL:REM:MODE 1 |
| | Turns remote mode to fast. |

## File Commands

## :MEMory:SAVE                               ⟨ Set ⟩⟶

| | | |
|---|---|---|
| Description | Saves Memory Data to internal memory to one of 256 slots. | |
| Syntax | :MEMory:SAVE {<NR1>} | |
| Parameter | <NR1> | M001 ~ M256 |
| Example | :MEMory:SAVE 1 <br> Save Memory Data to M001. | |

## :MEMory:RECall                             ⟨ Set ⟩⟶

| | | |
|---|---|---|
| Description | Recalls Memory Data from internal memory to one of 256 slots. | |
| Syntax | :MEMory:RECall {<NR1>} | |
| Parameter | <NR1> | M001 ~ M256 |
| Example | :MEMory:RECall 1 <br> Recall Memory Data from M001. | |

## :PRESet:SAVE
<span style="float:right">⟨ Set ⟩→</span>

| | |
|---|---|
| Description | Saves Preset Data to internal memory to one of 9 slots. |
| Syntax | :PRESet:SAVE {<NR1>} |
| Parameter | <NR1>      P1 ~ P9 |
| Example | :PRESet:SAVE 1 <br> Save Preset Data to P1. |

## :PRESet:RECall
<span style="float:right">⟨ Set ⟩→</span>

| | |
|---|---|
| Description | Recalls Preset Data from internal memory to one of 9 slots. |
| Syntax | :PRESet:RECall {<NR1>} |
| Parameter | <NR1>      P1 ~ P9 |
| Example | :PRESet:RECall 1 <br> Recall Preset Data from P1. |

## :SETup:SAVE
<span style="float:right">⟨ Set ⟩→</span>

| | |
|---|---|
| Description | Saves Setup Data to internal memory to one of 100 slots. |
| Syntax | :SETup:SAVE {<NR1>} |
| Parameter | <NR1>      S001 ~ S100 |
| Example | :SETup:SAVE 1 <br> Save Setup Data to S001. |

## :SETup:RECall
<span style="float:right">⟨ Set ⟩→</span>

| | |
|---|---|
| Description | Recalls Setup Data from internal memory to one of 100 slots. |
| Syntax | :SETup:RECall {<NR1>} |
| Parameter | <NR1>      S001 ~ S100 |

| Example | :SETup:RECall 1 |
|---|---|
| | Recall Setup Data from S001. |

## :FACTory[:RECall]                    ⟨ Set ⟩⟶

| Description | Recalls the default settings. See the user manual for the default settings. |
|---|---|
| Syntax | :FACTory[:RECall] |
| Example | :FACTory |

## :USER[:DEFault]:SAVE                    ⟨ Set ⟩⟶

| Description | Saves the current settings as the user default. |
|---|---|
| Syntax | :USER[:DEFault]:SAVE |
| Example | :USER:SAVE |
| | Saves the current settings as the user default. |

## :USER[:DEFault]:RECall                    ⟨ Set ⟩⟶

| Description | Recalls the user default settings. |
|---|---|
| Syntax | :USER[:DEFault]:RECall |
| Example | :USER:RECall |
| | Recalls the user default settings. |

## SCPI Register Commands

## :SYSTem:ERRor?                                          ➜ (Query)

| Description | Return the error message in the error queue. If there is no error, "+0, "No error."" Will be returned. | |
|---|---|---|
| Query Syntax | :SYSTem:ERRor? | |
| Return parameter | \<string> | Error code, followed by error description. |
| Query example | :SYSTem:ERRor?<br>> +0, "No error."<br><br>Returns the system information. | |

## :STATus:PRESet                    ⟨ Set ⟩→

| Description | The :STATus:PRESet command resets the enable registers NTR/PTR registers from Operation Status, Questionable Status and CSummary Status registers. |
|---|---|

| Preset Value | Register | Preset |
|---|---|---|
| | Operation Status Enable | All bits set to 1 |
| | Operation Status PTR | All bits set to 1 |
| | Operation Status NTR | All bits set to 0 |
| | Questionable Status Enable | All bits set to 0 |
| | Questionable Status PTR | All bits set to 1 |
| | Questionable Status NTR | All bits set to 0 |
| | CSummary Status Enable | All bits set to 0 |
| | CSummary Status PTR | All bits set to 1 |
| | CSummary  Status NTR | All bits set to 0 |

| Syntax | :STATus:PRESet |
|---|---|

## :STATus:QUEStionable:CONDition?         →⟨ Query ⟩

| Description | Returns the status of the Questionable Status Condition register. See page 160 for details. |
|---|---|

| Query Syntax | :STATus:QUEStionable:CONDition? |
|---|---|

| Return parameter | <NR1> | Condition | <NR1> | Condition |
|---|---|---|---|---|
| | 1 | OV | 512 | UV |
| | 2 | OC | 1024 | EXT |
| | 8 | OP | 2048 | REV |
| | 16 | OT | All other bits | Not used |

| Query example | :STATus:QUEStionable:CONDition? |
|---|---|
| | >1 |
| | Indicates an over voltage (OV) error has occurred. |

Set →
## :STATus:QUEStionable:ENABle
→ Query

| Description | Sets which events are enabled in the Questionable Status Enable register. The mask values are the bit weights of the events. See page 160 for details. | | | |
|---|---|---|---|---|
| Syntax | :STATus:QUEStionable:ENABle <NR1> | | | |
| Query Syntax | :STATus:QUEStionable:ENABle? | | | |
| Parameter/ Return Parameter | <NR1> | Condition | <NR1> | Condition |
| | 1 | OV | 512 | UV |
| | 2 | OC | 1024 | EXT |
| | 8 | OP | 2048 | REV |
| | 16 | OT | All other bits | Not used |
| Example | :STATus:QUEStionable:ENABle 11 | | | |
| | Events OV, OC and OP (Bits 1, 2, 3) are enabled in the Questionable Status Enable register. | | | |
| Query example | :STATus:QUEStionable:ENABle? >11 | | | |
| | Indicates that OV, OC and OP bits are enabled in the Questionable Status Enable register. | | | |

## :STATus:QUEStionable[:EVENt]?
→ Query

| Description | Returns the status of the Questionable Status Event register. The register is cleared upon reading. See page 160 for details. | | | |
|---|---|---|---|---|
| Query Syntax | :STATus:QUEStionable[:EVENt]? | | | |
| Return parameter | <NR1> | Condition | <NR1> | Condition |
| | 1 | OV | 512 | UV |
| | 2 | OC | 1024 | EXT |
| | 8 | OP | 2048 | REV |
| | 16 | OT | All other bits | Not used |

| Query example | :STATus:QUEStionable:EVENt?<br>> 1 |
|---|---|
| | Indicates an over voltage (OV) event has occurred. |

## :STATus:QUEStionable:NTRansition   Set → → Query

| Description | Determines whether a negative transition (NTR 1→0) in the Questionable Status Condition register will set the corresponding event in the Questionable Status Event register. |
|---|---|
| | The mask values are the bit weights of the Questionable Status NTR filters. See page 160 for details. |
| Syntax | :STATus:QUEStionable:NTRansition <NR1> |
| Query Syntax | :STATus:QUEStionable:NTRansition? |

| Parameter/<br>Return Parameter | <NR1> | Condition | <NR1> | Condition |
|---|---|---|---|---|
| | 1 | OV | 512 | UV |
| | 2 | OC | 1024 | EXT |
| | 8 | OP | 2048 | REV |
| | 16 | OT | All other bits | Not used |

| Example | :STATus:QUEStionable:NTRansition 11 |
|---|---|
| | Configures the OV, OC and OP bits (Bits 1, 2, 3) in the Questionable Status Condition register to set the corresponding bits in the Questionable Status Enable register on a negative transition |

| Query example | :STATus:QUEStionable:NTRansition?<br>>11 |
|---|---|
| | OV, OC and OP bits are configured to NTR in the Questionable Status NTR filter. |

| | Set ⟶ |
|---|---|
| :STATus:QUEStionable:PTRansition | ⟶ Query |

| | |
|---|---|
| Description | Determines whether a positive transition (PTR 0→1) in the Questionable Status Condition register will set the corresponding event in the Questionable Status Event register. |
| | The mask values are the bit weights of the Questionable Status PTR filters. See page 160 for details. |
| Syntax | :STATus:QUEStionable:PTRansition <NR1> |
| Query Syntax | :STATus:QUEStionable:PTRansition? |

| Parameter/ Return Parameter | <NR1> | Condition | <NR1> | Condition |
|---|---|---|---|---|
| | 1 | OV | 512 | UV |
| | 2 | OC | 1024 | EXT |
| | 8 | OP | 2048 | REV |
| | 16 | OT | All other bits | Not used |

| | |
|---|---|
| Example | :STATus:QUEStionable:PTRansition 11 |
| | Configures the OV, OC and OP bits (Bits 1, 2, 3) in the Questionable Status Condition register to set the corresponding bits in the Questionable Status Enable register on a positive transition. |
| Query example | :STATus:QUEStionable:PTRansition? >11 |
| | OV, OC and OP bits are configured to PTR in the Questionable Status PTR filter. |

| | Query |
|---|---|
| :STATus:OPERation:CONDition? | ⟶ Query |

| | |
|---|---|
| Description | Returns the status of the Operation Status Condition register. See page 161 for details. |
| Query Syntax | :STATus: OPERation:CONDition? |

| Return parameter | <NR1> | Condition | <NR1> | Condition |
|---|---|---|---|---|
| | 1 | CAL | All other bits | |
| | 32 | WTG | | |

| Query example | :STATus:OPERation:CONDition? |
| | >32 |
| | Indicates that the unit is waiting for a trigger. |
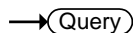
## :STATus:OPERation:ENABle

Set ⟶

⟶ Query

| Description | Sets which events are enabled in the Operation Status Enable register. The mask values are the bit weights of the events. See page 161 for details. |
| --- | --- |
| Syntax | :STATus: OPERation:ENABle <NR1> |
| Query Syntax | :STATus: OPERation:ENABle? |

| Parameter/ Return Parameter | <NR1> | Condition | <NR1> | Condition |
| --- | --- | --- | --- | --- |
| | 1 | CAL | All other | Not used |
| | 32 | WTG | bits | |

| Example | :STATus:OPERation:ENABle 32 |
| --- | --- |
| | WTG is enabled in the OPERation Status Enable register. |

| Query example | :STATus:OPERation:ENABle? |
| --- | --- |
| | >32 |
| | Indicates that the WTG bit is enabled in the Operation Status Enable register. |

## :STATus:OPERation[:EVENt]?

⟶ Query

| Description | Returns the status of the Operation Status Event register. The register is cleared upon reading. See page 161 for details. |
| --- | --- |
| Query Syntax | :STATus:OPERation[:EVENt]? |

| Return parameter | <NR1> | Condition | <NR1> | Condition |
| --- | --- | --- | --- | --- |
| | 1 | CAL | All other | Not used |
| | 32 | WTG | bits | |

| Query example | :STATus:OPERation:EVENt? |
| --- | --- |
| | > 1 |
| | Indicates that unit is currently calibrating. |

Set ⟶

⟶ Query

## :STATus:OPERation:NTRansition

| Description | Determines whether a negative transition (NTR 1→0) in the Operation Status Condition register will set the corresponding event in the Operation Status Event register. |
| --- | --- |
| | The mask values are the bit weights of the Operation Status NTR filters. See page 161 for details. |
| Syntax | :STATus:OPERation:NTRansition <NR1> |
| Query Syntax | :STATus:OPERation:NTRansition? |

| Parameter/ Return Parameter | <NR1> | Condition | <NR1> | Condition |
| --- | --- | --- | --- | --- |
| | 1 | CAL | All other bits | Not used |
| | 32 | WTG | | |

| Example | :STATus:OPERation:NTRansition 1 |
| --- | --- |
| | Configures the CAL bit (Bit 1) in the Operation Status Condition register to set the corresponding bit in the Questionable Status Enable register on a negative transition |
| Query example | :STATus:OPERation:NTRansition? >1 |
| | The CAL bit is configured to NTR in the Operation Status NTR filter. |

Set ⟶

⟶ Query

## :STATus:OPERation:PTRansition

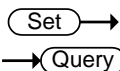| Description | Determines whether a positive transition (PTR 0→1) in the Operation Status Condition register will set the corresponding event in the Operation Status Event register. |
| --- | --- |
| | The mask values are the bit weights of the Operation Status PTR filters. See page 161 for details. |

| Syntax | :STATus:OPERation:PTRansition <NR1> | | | |
|---|---|---|---|---|
| Query Syntax | :STATus:OPERation:PTRansition? | | | |
| Parameter/ Return Parameter | <NR1> 1 32 | Condition CAL WTG | <NR1> All other bits | Condition Not used |
| Example | :STATus:OPERation:PTRansition 32 | | | |
| | Configures the WTG bit (Bit 5) in the Operation Status Condition register to set the corresponding bit in the Operation Status Enable register on a positive transition. | | | |
| Query example | :STATus:OPERation:PTRansition? >32 | | | |
| | The WTG bit is configured to PTR in the Operation Status PTR filter. | | | |

## :STATus:CSUMmary:CONDition? → Query

| Description | Returns the status of the CSummary Status Condition register. See page 159 for details. | | | |
|---|---|---|---|---|
| Query Syntax | :STATus:CSUMmary:CONDition? | | | |
| Return parameter | <NR1> 1 2 4 8 | Condition CC CR CV CP | <NR1> 256 All other bits | Condition PRUN Not used |
| Query example | :STATus:CSUMmary:CONDition? >1 | | | |
| | Indicates the unit is in CC mode. | | | |

## :STATus:CSUMmary:ENABle Set → → Query

| Description | Sets which events are enabled in the CSummary Status Enable register. The mask values are the bit weights of the events. See page 159 for details. |
|---|---|

| Syntax | :STATus:CSUMmary:ENABle <NR1> | | | |
|---|---|---|---|---|
| Query Syntax | :STATus:CSUMmary:ENABle? | | | |
| Parameter/<br>Return Parameter | <NR1><br>1<br>2<br>4<br>8 | Condition<br>CC<br>CR<br>CV<br>CP | <NR1><br>256<br>All other<br>bits | Condition<br>PRUN<br>Not used |
| Example | :STATus:CSUMmary:ENABle 1 | | | |
| | Enables the CC bit in the CSummary Status Enable register. | | | |
| Query example | :STATus:CSUMmary:ENABle?<br>>1 | | | |
| | Indicates that the CC bit is enabled in the CSummary Status Enable register. | | | |

## :STATus:CSUMmary[:EVENt]?                    ➔ Query

| Description | Returns the status of the CSummary Status Event register. The register is cleared upon reading. See page 159 for details. | | | |
|---|---|---|---|---|
| Query Syntax | :STATus:CSUMmary[:EVENt]? | | | |
| Return parameter | <NR1><br>1<br>2<br>4<br>8 | Condition<br>CC<br>CR<br>CV<br>CP | <NR1><br>256<br>All other<br>bits | Condition<br>PRUN<br>Not used |
| Query example | :STATus:CSUMmary:EVENt?<br>> 1 | | | |
| | Indicates that the unit is in CC mode. | | | |

## :STATus:CSUMmary:NTRansition                    ➔ Set<br>➔ Query

| Description | Determines whether a negative transition (NTR 1→0) in the CSummary Status Condition register will set the corresponding event in the CSummary Status Event register. |
|---|---|

The mask values are the bit weights of the CSummary Status NTR filters. See page 159 for details.

| Syntax | :STATus:CSUMmary:NTRansition <NR1> |
|---|---|
| Query Syntax | :STATus:CSUMmary:NTRansition? |

| Parameter/ Return Parameter | <NR1> | Condition | <NR1> | Condition |
|---|---|---|---|---|
| | 1 | CC | 256 | PRUN |
| | 2 | CR | All other bits | Not used |
| | 4 | CV | | |
| | 8 | CP | | |

| Example | :STATus:CSUMmary:NTRansition 3 |
|---|---|
| | Configures the CC and CR bits (Bits 1, 2) in the CSUMmary Status Condition register to set the corresponding bits in the CSUMmary Status Enable register on a negative transition. |

| Query example | :STATus:CSUMmary:NTRansition?<br>>3 |
|---|---|
| | The CC and CR bits are configured to NTR in the CSummary Status NTR filter. |
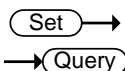
## :STATus:CSUMmary:PTRansition

⟶ Set ⟶
⟶ Query

| Description | Determines whether a positive transition (PTR 0→1) in the CSUMmary Status Condition register will set the corresponding event in the CSUMmary Status Event register. |
|---|---|
| | The mask values are the bit weights of the CSummary Status PTR filters. See page 159 for details. |

| Syntax | :STATus:CSUMmary:PTRansition <NR1> |
|---|---|
| Query Syntax | :STATus:CSUMmary:PTRansition? |

| Parameter/ Return Parameter | <NR1> | Condition | <NR1> | Condition |
|---|---|---|---|---|
| | 1 | CC | 256 | PRUN |
| | 2 | CR | All other bits | Not used |
| | 4 | CV | | |
| | 8 | CP | | |

| Example | :STATus:CSUMmary:PTRansition 12 |
|---|---|
| | Configures the CV and CP bits (Bits 3, 4) in the CSummary Status Condition register to set the corresponding bits in the CSummary Status Enable register on a positive transition. |
| Query example | :STATus:CSUMmary:PTRansition?<br>>12 |
| | The CV and CP bits are configured to PTR in the CSummary Status PTR filter. |

# Status Registers

To program the PEL-3000AE Series effectively, the Status Register structure needs to be understood. This chapter explains in detail the structure of the status registers.

Overview         The status registers are used to determine the status of the electronic load. The status registers maintain the status of the protection conditions, load conditions and channel conditions of the load modules.

The PEL-3000AE series have a number of register groups:

CSummary Registers

Questionable Status Registers

Standard Event Status Registers

Status Byte Register

Service Request Generation Register

Each register group consists of a Condition, EVENT and ENABLE register. They also have PTR and NTR (positive and negative transition) filters.

The structure of the status registers is shown on the next page.

## Status Registers



**CSummary Status**

| Mode | CC | CR | CV | CP | N.U. | PRUN | N.U. | |
|---|---|---|---|---|---|---|---|---|
| Bit position | 0 | 1 | 2 | 3 | 4-7 | 8 | 6-15 | |
| CONDITION | 1 | 2 | 4 | 8 | | 256 | | BIT WEIGHT |
| PTR/NTR | 1 | 2 | 4 | 8 | | 256 | | |
| EVENT | 1 | 2 | 4 | 8 | | 256 | | |
| ENABLE | 1 | 2 | 4 | 8 | | 256 | | |

**Questionable Status Registers**

| Query | OV | OC | N.U. | OP | OT | N.U. | UV | EXT | REV | N.U. | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit position | 0 | 1 | 2 | 3 | 4 | 5-8 | 9 | 10 | 11 | 12-15 | |
| CONDITION | 1 | 2 | | 8 | 16 | 32 | 512 | 1024 | 2048 | | BIT WEIGHT |
| PTR/NTR | 1 | 2 | | 8 | 16 | 32 | 512 | 1024 | 2048 | | |
| EVENT | 1 | 2 | | 8 | 16 | 32 | 512 | 1024 | 2048 | | |
| ENABLE | 1 | 2 | | 8 | 16 | 32 | 512 | 1024 | 2048 | | |

OR

OR

**Output Queue**

| Data | Data | Data |
|---|---|---|

**Standard Event Status Registers**

| Message | OPC | RQC | QUE | DDE | EXE | CME | URQ | PON | |
|---|---|---|---|---|---|---|---|---|---|
| Bit position | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | |
| EVENT | 1 | 2 | 4 | 8 | 16 | 32 | 64 | 128 | BIT WEIGHT |
| ENABLE | 1 | 2 | 4 | 8 | 16 | 32 | 64 | 128 | |

OR

**Error Queue**

| Data | Data | Data |
|---|---|---|

**Operation Status Register**

| Mode | CAL | N.U. | WTG | N.U. | |
|---|---|---|---|---|---|
| Bit position | 0 | 1-4 | 5 | 6-15 | |
| CONDITION | 1 | | 32 | | BIT WEIGHT |
| PTR/NTR | 1 | | 32 | | |
| EVENT | 1 | | 32 | | |
| ENABLE | 1 | | 32 | | |

OR

**Status Byte Register**

| Status | N.U. | ERR | CSUM | QUES | MAV | ESB | MSS | OPER | |
|---|---|---|---|---|---|---|---|---|---|
| Bit position | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | |
| STATUS BYTE | | 2 | 4 | 8 | 16 | 32 | 64 | 128 | BIT WEIGHT |

Service Request Generation

**Service Request Generation Register**

| Service | N.U. | ERR | CSUM | QUES | MAV | ESB | | OPER | |
|---|---|---|---|---|---|---|---|---|---|
| Bit position | 0 | 1 | 2 | 3 | 4 | 5 | | 7 | |
| Request Generation | | 2 | 4 | 8 | 16 | 32 | | 128 | BIT WEIGHT |

OR

## CSummary Status Registers

Overview
The CSummary Status registers indicate the current operation mode of the load generator.

| CSummary Status Registers | | | | | | |
|---|---|---|---|---|---|---|
| Bit Position | 9-15 | 8 | 4-7 | 3 | 2 | 1 | 0 |
| Condition | 0 | PRUN | 0 | CP | CV | CR | CP |
| PTR/NTR | 0 | PRUN | 0 | CP | CV | CR | CP |
| EVENT | 0 | PRUN | 0 | CP | CV | CR | CP |
| ENABLE | 0 | PRUN | 0 | CP | CV | CR | CP |
| Bit weight | | 256 | 16 | 8 | 4 | 2 | 1 |

Event Bits
CP: The unit has switched to CP mode.
CR: The unit has switched to CR mode.
CV: The unit has switched to CV mode.
CP: The unit has switched to CP mode.
PRUN: A program is running.

Condition Register
The condition register indicates the status of the electronic load. The condition register can only be changed by a change in the condition of the electronic load. Reading the condition register does not change the state of the condition register.

PTR/NTR Filters
The PTR/NTR (Positive/Negative transition) register determines the type of transition conditions that will trigger an event. Only the CSummary Status Register, Questionable Status Register and Operation Status Register can be transition programmed.

Event Register
The Event Register indicates if an event has been triggered according to the transition settings from the PTR/NTR Register.

Enable Register    The Enable register determines which status
                   event(s) are enabled. Any status events (CC, CV,
                   CR, CP, PRUN) that are enabled will set the
                   corresponding bit in the CSummary Event
                   Register.

## Questionable Status Registers

Overview    The Questionable Status Registers will show if any
            faults or errors have occurred.

| Questionable Status Register | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| Bit Position | 12-15 | 11 | 10 | 9 | 5-8 | 4 | 3 | 2 | 1 | 0 |
| Condition | 0 | REV | EXT | UV | 0 | OT | OP | 0 | OC | OV |
| PTR/NTR | 0 | REV | EXT | UV | 0 | OT | OP | 0 | OC | OV |
| EVENT | 0 | REV | EXT | UV | 0 | OT | OP | 0 | OC | OV |
| ENABLE | 0 | REV | EXT | UV | 0 | OT | OP | 0 | OC | OV |
| Bit weight | | 2048 | 1024 | 512 | | 16 | 8 | 4 | 2 | 1 |

Events Bits    OV: Over voltage condition occurred.
               OC: Over current condition occurred.
               OP: Over power condition occurred.
               OT: Over temperature condition occurred.
               UV: Under voltage condition occurred.
               EXT: External problem occurred. (Same a Para
               alarm event: UnReg, R.OCP, OHP condition
               occurred.)
               REV: A reverse voltage condition has occurred.

Condition      The condition register indicates the status of the
Register       protection systems. The condition register can only
               be changed by a change in the condition of the
               protection systems. Reading the condition register
               does not change the state of the condition register.

PTR/NTR Filters    The PTR/NTR (Positive/Negative transition) register determines the type of transition conditions that will trigger an event. Only the CSummary Status Register, Questionable Status Register and Operation Status Register can be transition programmed.

Event Register     The Event Register indicates if an event has been triggered according to the transition settings from the PTR/NTR Register.

Enable Register    The Enable register determines which status event(s) are enabled. Any status events (OV, OC, OP, OT, UV, EXT or REV) that are enabled will set the corresponding bit in the Questionable Event Register.

## Operation Status Registers

Overview           The Operation Status Registers will show the operating state of electronic load (waiting for a trigger or in calibration mode).

| Operation Status Registers | | | |
|---|---|---|---|
| Bit Position | 5 | 1-4 | 0 |
| Condition | WTG | 0 | CAL |
| PTR/NTR | WTG | 0 | CAL |
| EVENT | WTG | 0 | CAL |
| ENABLE | WTG | 0 | CAL |
| Bit weight | 32 | | 1 |

Events Bits        CAL: Indicates that the unit is currently in calibration mode.
                   WTG: Indicates that the unit is waiting for a trigger.

| Condition Register | The condition register indicates the operating state of the unit. The condition register can only be changed by a change in the condition of operating state. Reading the condition register does not change the state of the condition register. |
|---|---|
| PTR/NTR Filters | The PTR/NTR (Positive/Negative transition) register determines the type of transition conditions that will trigger an event. Only the CSummary Status Register, Questionable Status Register and Operation Status Register can be transition programmed. |
| Event Register | The Event Register indicates if an event has been triggered according to the transition settings from the PTR/NTR Register. |
| Enable Register | The Enable register determines which status event(s) are enabled. Any status events (CAL or WTG) that are enabled will set the corresponding bit in the Operation Event Register. |

## Standard Event Status Registers

| Overview | The Standard Event Status Registers indicate any programming errors that occur. The Standard Event Status Register group comprises of the Event and Enable registers. |
|---|---|

| Standard Event Status Registers | | | | | | | |
|---|---|---|---|---|---|---|---|
| Bit Position | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| EVENT | PON | URQ | CME | EXE | DDE | QUE | RQC | OPC |
| ENABLE | PON | URQ | CME | EXE | DDE | QUE | RQC | OPC |
| Bit weight | 128 | 64 | 32 | 16 | 8 | 4 | 2 | 1 |

Events Bits   OPC: The operation complete bit is set when all selected pending operations are complete. This bit is set in response to the *OPC command.
RQC: Not used.
QUE: The query error bit is set when there is an error reading the output queue. This can be caused by trying to read the output queue when there is no data in it.
DDE: The Device Dependent Error indicates a memory error/lost memory or failure of the self-test.
EXE: The Execution bit indicates an execution error due to one of the following:
-Illegal command parameter
-Parameter out of range
-Invalid parameter
-Command didn't execute due to an overriding operation condition.
CME: The Command Error bit can be set when a syntax error has occurred, an unidentifiable header is received or when a <GET> command is received within a program message. (Group Execute Trigger) as defined in IEEE 488.1.
URQ: Not used.
PON: Not used.

Event Register   The Event Register will be set to 0 when read.

Enable Register   The Enable Register determines which events will set the ESB Bit (bit 5) in the Status Byte Register.

## Status Byte Register

Overview
The Status Byte register consolidates the status events of all the status registers. The Status Byte register can be read with the *STB? query or a serial poll and can be cleared with the *CLS command.

| Status Byte Register | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| Bit Position | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| Condition | OPER | MSS | ESB | MAV | QUES | CSUM | ERR | 0 |
| Bit weight | 128 | 64 | 32 | 16 | 8 | 4 | 2 | 1 |

Status Bits
ERR: The ERR bit is set when there is a message in the error queue.
CSUM: The CSUM bit is set when an event has occurred in the CSummary Status Register group.
QUES: The QUES bit is set when an event has occurred in the Questionable Status Register group.
MAV: The MAV bit is set when there is outstanding data in the Output Queue.
ESB: The Event Status bit is set if an enabled event in the Standard Event Status Register group has occurred.
MSS & RQS: The Master Summary Status is used with the *STB? query. When the *STB? query is read, the MSS bit is not cleared. The Request Service bit is cleared when it is polled during a serial poll.
OPER: The OPER bit is set when if an enabled event in the Operation Status Register has occurred.

## Service Request Register

Overview     The Service Request Generation Register
             determines which events in the Status Byte
             Register will generate Service Requests. It is
             essentially the Status Byte Enable Register. The bit
             events are the same as the Status Byte Register,
             minus the MSS/RQS bit.

| Service Request Generation Register (Status Byte Enable) | | | | | | | |
|---|---|---|---|---|---|---|---|
| Bit Position | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| Condition | OPER | 0 | ESB | MAV | QUES | CSUM | ERR | 0 |
| Bit weight | 128 | 64 | 32 | 16 | 8 | 4 | 2 | 1 |

# Error Messages

The following error messages may be encountered when reading the error queue.

| Error Code and string | Description |
| --- | --- |
| Command Errors | |
| 0 NoError | No error |
| -100 Command Error | This is the generic syntax error for devices that cannot detect more specific errors. |
| -101 Invalid character | A syntactic element contains a character which is invalid for that type. |
| -102 Syntax error | An unrecognized command or data type was encountered. |
| -103 Invalid separator | The parser was expecting a separator and encountered an illegal character. |
| -104 Data type error | The parser recognized a data element different than the one allowed. |
| -105 GET not allowed | A Group Execute Trigger was received within a program message. |
| -108 Parameter not allowed | More parameters were received than expected for the header. |
| -109 Missing parameter | Fewer parameters were received than required for the header. |
| -110 Command header error | An error was detected in the header. |

| -111 Header separator error | A character which is not a legal header separator was encountered while parsing the header. |
| --- | --- |
| -112 Program mnemonic too long | The header contains more than twelve characters. |
| -113 Undefined header | The header is syntactically correct, but it is undefined for this specific device. |
| -114 Header suffix out of range | The value of a numeric suffix attached to a program mnemonic. |
| -115 Unexpected number of parameters | The number of parameters received does not correspond to the number of parameters expected. |
| -120 Numeric data error | This error is generated when parsing a data element which appears to be numeric, including the nondecimal numeric types. |
| -121 Invalid character in number | An invalid character for the data type being parsed was encountered. |
| -123 Exponent too large | The magnitude of the exponent was larger than 32000. |
| -124 Too many digits | The mantissa of a decimal numeric data element contained more than 255 digits excluding leading zeros. |
| -128 Numeric data not allowed | A legal numeric data element was received, but the device does not accept one in this position for the header |
| -130 Suffix error | This error, as well as errors -131 through -139, are generated when parsing a suffix. |

| | |
|---|---|
| -131 Invalid suffix | The suffix does not follow the syntax described in IEEE 488.2 or the suffix is inappropriate for this device. |
| -134 Suffix too long | The suffix contained more than 12 characters. |
| -138 Suffix not allowed | A suffix was encountered after a numeric element which does not allow suffixes. |
| -140 Character data error | This error is generated when parsing a character data element. |
| -141 Invalid character data | Either the character data element contains an invalid character or the particular element received is not valid for the header. |
| -144 Character data too long | The character data element contains more than twelve characters |
| -148 Character data not allowed | A legal character data element was encountered where prohibited by the device. |
| -150 String data error | This error is generated when parsing a string data element. |
| -151 Invalid string data | A string data element was expected, but was invalid for some reason. |
| -158 String data not allowed | A string data element was encountered but was not allowed by the device at this point in parsing. |
| -160 Block data error | This error is generated when parsing a block data element. |
| -161 Invalid block data | A block data element was expected, but was invalid for some reason. |

| | |
|---|---|
| -168 Block data not allowed | A legal block data element was encountered but was not allowed by the device at this point in parsing. |
| -170 Expression error | This error is generated when parsing an expression data element. |
| -171 Invalid expression | The expression data element was invalid. |
| -178 Expression data not allowed | A legal expression data was encountered but was not allowed by the device at this point in parsing. |
| -180 Macro error | This error is generated when defining a macro or executing a macro. |
| -181 Invalid outside macro definition | Indicates that a macro parameter placeholder ($<number) was encountered outside of a macro definition. |
| -183 Invalid inside macro definition | Indicates that the program message unit sequence, sent with a *DDT or *DMC command, is syntactically invalid. |
| -184 Macro parameter error | Indicates that a command inside the macro definition had the wrong number or type of parameters. |

| Execution Errors | |
| --- | --- |
| -200 Execution error | This is the generic syntax error for devices that cannot detect more specific errors. This code indicates only that an Execution Error as defined in IEEE 488.2 has occurred. |
| -201 Invalid while in local | Indicates that a command is not executable while the device is in local due to a hard local control |
| -202 Settings lost due to rtl | Indicates that a setting associated with a hard local control was lost when the device changed to LOCS from REMS or to LWLS from RWLS. |
| -203 Command protected | Indicates that a legal password-protected program command or query could not be executed because the command was disabled. |
| -210 Trigger error | |
| -211 Trigger ignored | Indicates that a GET, *TRG, or triggering signal was received and recognized by the device but was ignored because of device timing considerations. |
| -212 Arm ignored | Indicates that an arming signal was received and recognized by the device but was ignored. |
| -213 Init ignored | Indicates that a request for a measurement initiation was ignored as another measurement was already in progress. |

| -214 Trigger deadlock | Indicates that the trigger source for the initiation of a measurement is set to GET and subsequent measurement query is received. The measurement cannot be started until a GET is received, but the GET would cause an INTERRUPTED error. |
|---|---|
| -215 Arm deadlock | Indicates that the arm source for the initiation of a measurement is set to GET and subsequent measurement query is received. The measurement cannot be started until a GET is received, but the GET would cause an INTERRUPTED error. |
| -220 Parameter error | Indicates that a program data element related error occurred. |
| -221 Settings conflict | Indicates that a legal program data element was parsed but could not be executed due to the current device state. |
| -222 Data out of range | Indicates that a legal program data element was parsed but could not be executed because the interpreted value was outside the legal range as defined by the device. |
| -223 Too much data | Indicates that a legal program data element of block, expression, or string type was received that contained more data than the device could handle due to memory or related device-specific requirements. |

| | |
|---|---|
| -224 Illegal parameter value | Used where an exact value, from a list of possibilities, was expected. |
| -225 Out of memory. | The device has insufficient memory to perform the requested operation. |
| -226 Lists not same length. | Attempted to use LIST structure having individual LIST's of unequal lengths. |
| -230 Data corrupt or stale | Possibly invalid data; new reading started but not completed since last access. |
| -231 Data questionable | Indicates that measurement accuracy is suspect. |
| -232 Invalid format | Indicates that a legal program data element was parsed but could not be executed because the data format or structure is inappropriate. |
| -233 Invalid version | Indicates that a legal program data element was parsed but could not be executed because the version of the data is incorrect to the device. |
| -240 Hardware error | Indicates that a legal program command or query could not be executed because of a hardware problem in the device. |
| -241 Hardware missing | Indicates that a legal program command or query could not be executed because of missing device hardware. |
| -250 Mass storage error | Indicates that a mass storage error occurred. |

| -251 Missing mass storage | Indicates that a legal program command or query could not be executed because of missing mass storage. |
|---|---|
| -252 Missing media | Indicates that a legal program command or query could not be executed because of a missing media. |
| -253 Corrupt media | Indicates that a legal program command or query could not be executed because of corrupt media. |
| -254 Media full | Indicates that a legal program command or query could not be executed because the media was full. |
| -255 Directory full | Indicates that a legal program command or query could not be executed because the media directory was full. |
| -256 File name not found | Indicates that a legal program command or query could not be executed because the file name on the device media was not found. |
| -257 File name error | Indicates that a legal program command or query could not be executed because the file name on the device media was in error. |
| -258 Media protected | Indicates that a legal program command or query could not be executed because the media was protected. |
| -260 Expression error | Indicates that a expression program data element related error occurred. |

| | |
|---|---|
| -261 Math error in expression | Indicates that a syntactically legal expression program data element could not be executed due to a math error. |
| -270 Macro error | Indicates that a macro-related execution error occurred. |
| -271 Macro syntax error | Indicates that that a syntactically legal macro program data sequence, according to IEEE 488.2 could not be executed due to a syntax error within the macro definition. |
| -272 Macro execution error | Indicates that a syntactically legal macro program data sequence could not be executed due to some error in the macro definition. |
| -273 Illegal macro label | Indicates that the macro label defined in the *DMC command was a legal string syntax, but could not be accepted by the device. |
| -274 Macro parameter error | Indicates that the macro definition improperly used a macro parameter placeholder. |
| -275 Macro definition too long | Indicates that a syntactically legal macro program data sequence could not be executed because the string or block contents were too long for the device to handle. |
| -276 Macro recursion error | Indicates that a syntactically legal macro program data sequence could not be executed because the device found it to be recursive. |

| | |
|---|---|
| -277 Macro redefinition not allowed | Indicates that a syntactically legal macro label in the *DMC command could not be executed because the macro label was already defined. |
| -278 Macro header not found | Indicates that a syntactically legal macro label in the *GMC? query could not be executed because the header was not previously defined. |
| -280 Program error | Indicates that a downloaded program-related execution error occurred. |
| -281 Cannot create program | Indicates that an attempt to create a program was unsuccessful. A reason for the failure might include not enough memory. |
| -282 Illegal program name | The name used to reference a program was invalid. |
| -283 Illegal variable name | An attempt was made to reference a nonexistent variable in a program. |
| -284 Program currently running | Certain operations dealing with programs may be illegal while the program is running. |
| -285 Program syntax error | Indicates that a syntax error appears in a downloaded program. |
| -286 Program runtime error | |
| -290 Memory use error | Indicates that a user request has directly or indirectly caused an error related to memory or <data_handle>s, this is not the same as "bad" memory. |
| -291 Out of memory | |

-292 Referenced name does not
exist

-293 Referenced name already exists

| -294 Incompatible type | Indicates that the type or structure of a memory item is inadequate. |
|---|---|

## Device Specific Errors

| -300 Device-specific error | This is the generic device-dependent error for devices that cannot detect more specific errors. |
|---|---|
| -310 System error | Indicates that some error, termed "system error" by the device, has occurred. |
| -311 Memory error | Indicates some physical fault in the device's memory, such as parity error. |
| -312 PUD memory lost | Indicates that the protected user data saved by the *PUD command has been lost. |
| -313 Calibration memory lost | Indicates that nonvolatile calibration data used by the *CAL? command has been lost. |
| -314 Save/recall memory lost | Indicates that the nonvolatile data saved by the *SAV? command has been lost. |
| -315 Configuration memory lost | Indicates that nonvolatile configuration data saved by the device has been lost. |

| | |
|---|---|
| -320 Storage fault | [Indicates that the firmware detected a fault when using data storage. This error is not an indication of physical damage or failure of any mass storage element. |
| -321 Out of memory | An internal operation needed more memory than was available. |
| -330 Self-test failed | |
| -340 Calibration failed | |
| -350 Queue overflow | A specific code entered into the queue in lieu 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. |
| -360 Communication error | This is the generic communication error. |
| -361 Parity error in program message | Parity bit not correct when data received. |
| -362 Framing error in program message | A stop bit was not detected when data was received. |
| -363 Input buffer overrun | Software or hardware input buffer on serial port overflows with data caused by improper or nonexistent pacing. |
| -365 Time out error | This is a generic device-dependent error. |

| Query Errors | |
|---|---|
| -400 Query error | This is the generic query error. |
| -410 Query INTERRUPTED | Indicates that a condition causing an INTERRUPTED Query error occurred. |

| | |
|---|---|
| -420 Query UNTERMINATED | Indicates that a condition causing an UNTERMINATED Query error occurred. |
| -430 Query DEADLOCKED | Indicates that a condition causing an DEADLOCKED Query error occurred. |
| -440 Query UNTERMINATED after indefinite response | Indicates that a query was received in the same program message after a query requesting an indefinite response was executed |

### Power On Event Commands

| | |
|---|---|
| -500 Power on | The instrument has detected an off to on transition in its power supply. |

### User Request Event

| | |
|---|---|
| -600 User request | The instrument has detected the activation of a user request local control. |

### Request Control Event

| | |
|---|---|
| -700 Request control | The instrument requested to become the active IEEE 488.1 controller-in-charge. |

### Operation Complete Event

| | |
|---|---|
| -800 Operation complete | The instrument has completed all selected pending operations in accordance with the IEEE 488.2 synchronization protocol. |