WT500
Power Analyzer Communication Interface
USER'S MANUAL

YOKOGAWA
Yokogawa Test & Measurement Corporation

IM 760201-17E
3rd Edition

Thank you for purchasing the WT500 Power Analyzer.
This Communication Interface User's Manual describes the functions of the USB, GP-IB, and Ethernet interfaces and communication commands. To ensure correct use, please read this manual thoroughly before beginning operation. After reading the manual, keep it in a convenient location for quick reference whenever a question arises during operation.

List of Manuals

The following manuals, including this one, are provided as manuals for this instrument. Please read all manuals.

Manual Title	Manual No.	Description
WT500 Power Analyzer User's Manual	IM 760201-01E	Explains all functions and procedures of the WT500 excluding the communication functions.
WT500 Power Analyzer Communication Interface User's Manual (CD)	IM 760201-17E	This manual. Explains the functions for controlling the WT500 using communication commands.
WT500 Power Analyzer Pulse Output Function (/P14 Optional) User's Manual	IM 760201-51E	Included with models with the /P14 option. Explains the features of the Pulse Output Function option and how to use them.
WT500 Power Analyzer Pulse Output Function (/P17 Optional) User's Manual	IM 760201-52E	Included with models with the /P17 option. Explains the features of the Pulse Output Function option and how to use them.
WT500 Power Analyzer	IM 760201-92	Document for China

The "E" in the manual number is the language code.

Contact information of Yokogawa offices worldwide is provided on the following sheet.

Document No.	Description
PIM 113-01Z2	List of worldwide contacts

Note

- The contents of this manual are subject to change without prior notice as a result of continuing improvements to the instrument's performance and functions. The figures given in this manual may differ from those that actually appear on your screen.
- Every effort has been made in the preparation of this manual to ensure the accuracy of its contents. However, should you have any questions or find any errors, please contact your nearest YOKOGAWA dealer.
- Copying or reproducing all or any part of the contents of this manual without the permission of YOKOGAWA is strictly prohibited.
- The TCP/IP software of this product and the document concerning the TCP/IP software have been developed/created by YOKOGAWA based on the BSD Networking Software, Release 1 that has been licensed from University of California.

USB Interface and Ethernet Interface

- The items below are needed on the PC to use the communication functions via the USB interface.
  - DL/WT series library (TMCTL)
  - USB connection device driver between the PC and WT500
- The item below is needed on the PC to use the communication functions via the Ethernet interface.
  - DL/WT series library (TMCTL)

The library and driver above can be downloaded from the following Web page.

http://www.yokogawa.com/tm/

3rd Edition: October 2017 (YMI)

Trademarks

- Microsoft, Internet Explorer, MS-DOS, Windows, Windows NT, and Windows Vista are either registered trademarks or trademarks of Microsoft Corporation in the United States and/or other countries.
- Adobe and Acrobat are registered trademarks or trademarks of Adobe Systems Incorporated.
- For purposes of this manual, the ® and TM symbols do not accompany their respective registered trademark or rtrademark names.
- Other company and product names are registered trademarks or trademarks of their respective holders.

Revisions

- 1st Edition June 2008
- 2nd Edition December 2015
- 3rd Edition October 2017

Structure of the Manual

This User's Manual consists of the following sections:

Chapter 1 USB Interface
Describes the functions and specifications of the USB interface.

Chapter 2 GP-IB Interface (Option)
Describes the functions and specifications of the GP-IB interface.

Chapter 3 Ethernet Interface (Option)
Describes the functions and specifications of the Ethernet interface.

Chapter 4 Before Programming
Describes the syntax used to transmit commands.

Chapter 5 Communication Commands
Describes all the commands one by one.

Chapter 6 Status Reports
Describes the status byte, various registers, queues, and other information.

Appendix
Describes reference material such as an ASCII character code table.

Index

Symbols and Notation Used in This Manual

Unit and Note

Type	Symbol	Meaning	
Unit	k	1000	Example: 100 kHz
	K	1024	Example: 640 KB (file data size)
Note	***Note***	Calls attention to information that is important for proper operation of the instrument.	

Subheadings

On pages that describe operating procedures, the following symbols, displayed characters, and terminology are used to distinguish the procedures from their explanations:

Procedure Follow the numbered steps. All procedures are written with inexperienced users in mind; experienced users may not need to carry out all the steps.

Explanation This subsection describes the setup parameters and the limitations on the procedures.

Displayed Characters and Terminology Used in the Procedural Explanations

Panel Keys and Soft keys

Bold characters used in the procedural explanations indicate characters that are marked on the panel keys or the characters of the soft keys or menus displayed on the screen.

SHIFT + Panel Key

SHIFT + key means you will press the SHIFT key to turn ON the SHIFT key followed by the operation key. The setup menu marked in purple below the panel key that you pressed appears on the screen.

Symbols Used in the Syntax

Symbols which are used in the syntax descriptions in Chapter 6 are shown below. These symbols are referred to as BNF (Backus-Naur Form) symbols. For details on the data, see pages 4-5 and 4-6.

Symbol	Meaning	Example	Example of Input
`< >`	Defined value.	`ELEMent<x> <x> = 1 to 4`	`->ELEMENT2`
`{ }`	Select from values given in { }.	`SCALING:{VT|CT|SFACtor}?`	`->SCALING:VT?`
`|`	Exclusive OR		
`[ ]`	Can be omitted.	`NUMeric[:NORMal]:VALue?`	`->NUMERIC:VALUE?`

Contents

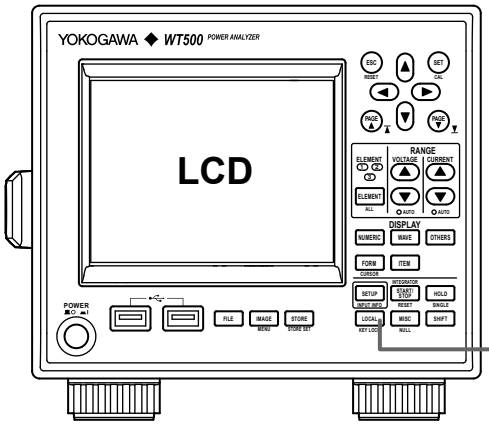
Chapter 6 Status Reports

Appendix

Index

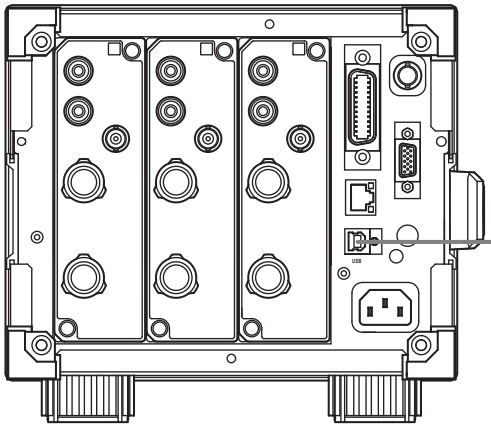
1.1 Names of Parts

Front Panel



LOCAL key
Press this key to clear the remote mode (controlled via communications) and enter the local mode in which key operations are enabled.

Rear Panel



USB connector for connecting to a PC
A connector used to connect the WT500 to the controller (such as a PC) using a USB cable. For the connection procedure, see page 1-3.

1.2 USB Interface Functions and Specifications

You can control the WT500 from a PC using the USB interface. YOKOGAWA's dedicated USB connection device driver and library software (TMCTL) must be installed on the PC.

Reception Function

You can specify the same settings as those specified by front panel key operations.
Receives output requests for measured and computed data, setup data, and error codes.

Transmission Function

Outputs measured and computed data.
Outputs panel setup parameters and the status byte.
Outputs error codes that have occurred.

Switching between Remote and Local Modes

When Switching from Local to Remote Mode

Remote mode is activated when the `:COMMunicate:REMote ON` command is received from a controller while local mode is active.

- The REMOTE indicator is turned ON.
- All keys except the LOCAL key are disabled.
- Settings entered in local mode are retained even when the WT500 switches to remote mode.

When Switching from Remote to Local Mode

Pressing the LOCAL key when the WT500 is in the remote mode causes the instrument to switch to the local mode. However, this is not possible when the `:COMMunicate:LOCKout ON` command is received from the PC while Local Lockout mode is active.
Local mode is activated when the `:COMMunicate:REMote OFF` command is received regardless of Local Lockout.

- The REMOTE indicator is turned OFF.
- Key operations are enabled.
- Settings entered in remote mode are retained even when the WT500 switches to local mode.

Note

The USB interface cannot be used simultaneously with other communication interfaces (GP-IB and Ethernet).

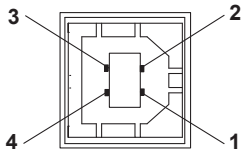
USB Interface Specifications

Electrical and mechanical specifications: Conforms to USB Rev.1.1

Connector:	Type B connector (receptacle)
Number of ports:	1
Power supply:	Self-powered
Protocol:	USB-TMC
PC system supported:	A controller such as a PC running Windows 98 SE, Windows Me, Windows 2000, or Windows XP that is equipped with a USB port as standard (a separate device driver is required for the connection with a PC)

1.3 Connection via the USB Interface

Connector and Signal Names



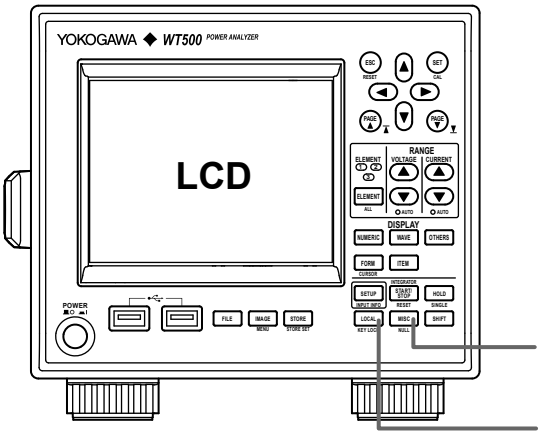
Pin No.	Signal Name	
1	V_{BUS}:	+5 V
2	D–:	–Data
3	D+:	+Data
4	GND:	Ground

Precautions to Be Taken When Making Connections

- Connect the USB cable by inserting the connector firmly into the USB connector.
- When connecting multiple devices using USB hubs, connect the WT500 to the USB hub that is closest to the controller.
- Do not connect or disconnect the USB cable after the power is turned ON until the WT500 boots up completely (until the WT500 is ready for operation, approximately 20 to 30 s). If you do, the WT500 may malfunction.

2.1 Names and Functions of Parts

Front Panel

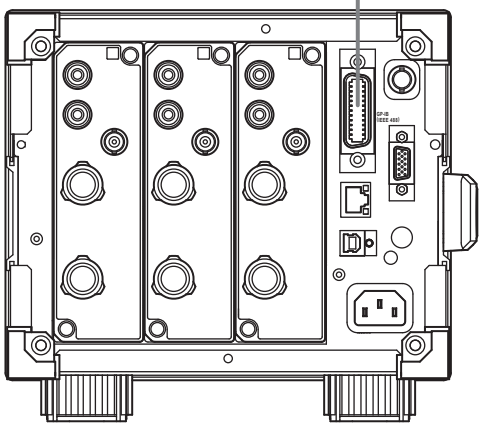


MISC key
Press this key to configure communications.

LOCAL key
Press this key to clear the remote mode (controlled via communications) and enter the local mode in which key operations are enabled.

Rear Panel

GP-IB connector
Connector used to connect the WT500 to the controller (PC) using a GP-IB cable.



2.2 GP-IB Interface Functions

GP-IB Interface Functions

Listener Capability

- All of the information that you can set with the panel keys can be set through the GP-IB interface except for turning ON/OFF the power and setting the communication parameters.
- Receives commands from a controller requesting the output of setup parameters, measured/computed data, waveform data, and other information.
- Also receives status report commands.

Talker Capability

- Outputs setup parameters, measured/computed data, waveform data, and other information.

Note

Talk-only, listen-only, and controller functions are not available on this instrument.

Switching between Remote and Local Modes

When Switching from Local to Remote Mode

Receiving a REN (Remote Enable) message from the controller when the instrument is in the local mode causes the instrument to switch to the remote mode.

- The REMOTE indicator is turned ON.
- All keys except the LOCAL key are disabled.
- Settings entered in local mode are retained even when the WT500 switches to remote mode.

When Switching from Remote to Local Mode

Pressing LOCAL key in remote mode puts the instrument in local mode. However, this act is invalid if the instrument has been set to Local Lockout mode (see page 2-6) by the controller.

- The REMOTE indicator is turned OFF.
- Key operations are enabled.
- Settings entered in remote mode are retained even when the WT500 switches to local mode.

Note

The GP-IB interface cannot be used simultaneously with other communication interfaces (USB, or Ethernet).

2.3 GP-IB Interface Specifications

GP-IB Interface Specifications

Supported device:	National Instruments • AT-GPIB • PCI-GPIB and PCI-GPIB+ • PCMCIA-GPIB and PCMCIA-GPIB + NI-488.2M driver version 1.60 or later
Electrical and mechanical specifications:	Conforms to IEEE St'd 488-1978
Functional specifications:	See table below.
Protocol:	Conforms to IEEE St'd 488.2-1992
Code used:	ISO (ASCII) code
Mode:	Addressable mode
Address setting:	The address can be set in the range from 0 to 30.
Clear remote mode:	Remote mode can be cleared by pressing LOCAL key except when the instrument has been set to Local Lockout mode by the controller.

Functional specifications

Function	Subset Name	Description
Source handshaking	SH1	Full source handshaking capability.
Acceptor handshaking	AH1	Full acceptor handshaking capability.
Talker	T6	Basic talker capability, serial polling, untalk on MLA (My Listen Address), and no talk-only capability.
Listener	L4	Basic listener capability, unlisten on MTA (My Talk Address), and no listen-only capability.
Service request	SR1	Full service request capability
Remote local	RL1	Full remote/local capability
Parallel polling	PP0	No parallel polling capability
Device clear	DC1	Full device clear capability
Device trigger	DT1	Full device trigger capability
Controller	C0	No controller capability
Electrical characteristics	E1	Open collector

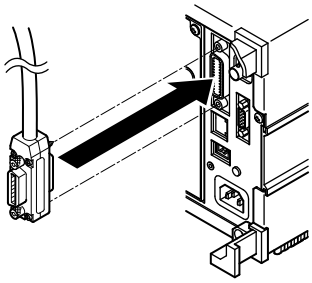
2.4 Connecting the GP-IB Cable

GP-IB Cable

The GP-IB connector used on this instrument is a 24-pin connector that conforms to the IEEE St'd 488-1978. Use a GP-IB cable that conforms to this standard.

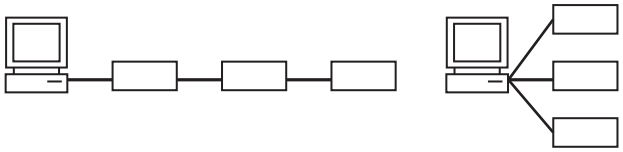
Connection Procedure

Connect the cable as shown below.



Precautions to Be Taken When Making Connections

- Firmly tighten the screws on the GP-IB cable connector.
- Use an NI (National Instruments) model GP-IB port (or card) on the PC side. For details, see section 2.3.
- If a converter is used along the communication cable connecting the WT and PC (for example, a GP-IB-to-USB converter), malfunctions can occur. For details, consult with your Yokogawa dealer or representative.
- Multiple cables can be used to connect multiple devices. However, no more than 15 devices including the controller can be connected on a single bus.
- When connecting multiple devices, each device must have its own unique address.
- Use a cable of length 2 m or less for connecting the devices.
- Make sure the total cable length does not exceed 20 m.
- When communicating, have at least two-thirds of the devices turned ON.
- To connect multiple devices, wire them in a daisy-chain or star configuration as shown below. You can also mix these configurations. Loop configuration is not allowed.



- To connect a BNC cable to the external clock input/external start signal output connector (EXT CLK) when a GP-IB cable is connected to the GP-IB connector, first remove the GP-IB cable. Then connect the BNC cable. Finally, reconnect the GP-IB cable.

CAUTION

When connecting or disconnecting communication cables, make sure to turn OFF the PC and the WT500. Otherwise, erroneous operation or damage to the internal circuitry may result.

French

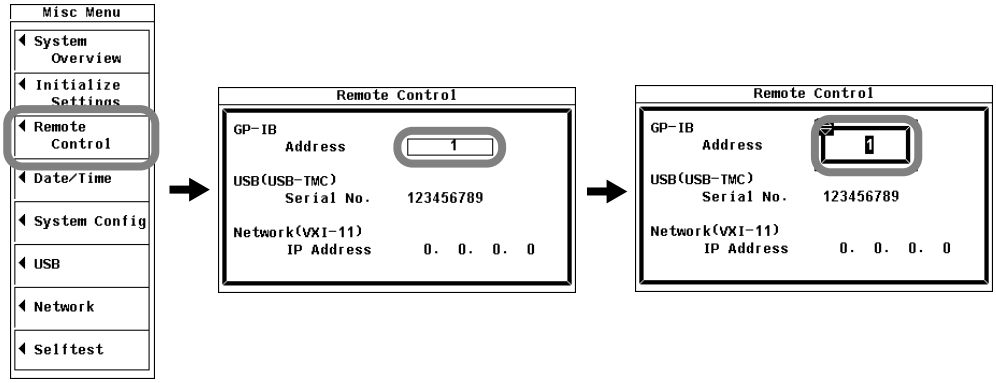
ATTENTION

Lors de la connexion ou déconnexion des câbles de communication, veiller à mettre le PC et cet instrument hors tension. À défaut, des dysfonctionnements et/ou un endommagement du circuit interne risquent de se produire.

2.5 Setting the GP-IB Control

Procedure

1. Press MISC to display the Misc Menu.
2. Press the cursor keys to select the Remote Control.
3. Press SET. The Remote Control dialog box appears.
4. Press the cursor keys to select GP-IB Address.
5. Press SET. The GP-IB Address dialog box appears.
6. Press the cursor keys to select the address.
7. Press SET to confirm the address.



Explanation

Enter the following settings when using a controller to set information that can be specified through key operation on the WT500 or when outputting setup parameters or output waveform display data to the controller.

Address

Set the address of the WT500 within the following range for the addressable mode.
0 to 30

Each device that can be connected via GP-IB has a unique address within the GP-IB system. This address is used to distinguish the device from others. Therefore, when you connect the WT500 to a PC, for example, make sure to assign a unique address to the WT500.

Note

Do not change the address while the controller or other devices are using the GP-IB system.

2.6 Responses to Interface Messages

Responses to Interface Messages

Responses to a Uni-Line Message

- **IFC (Interface Clear)**
  Clears the talker and listener functions. Stops output if data are being output.
- **REN (Remote Enable)**
  Switches between the remote and local modes.

  IDY (Identify) is not supported.

Responses to a Multi-Line Message (Address Command)

- **GTL (Go To Local)**
  Switches to the local mode.
- **SDC (Selected Device Clear)**
  - Clears the program message (command) being received and the output queue (see page 6-8).
  - *OPC and *OPC? commands in execution are void.
  - The *WAI and COMMunicate:WAIT commands are immediately terminated.
- **GET (Group Execute Trigger)**
  Same operation as the *TRG command.

  PPC (Parallel Poll Configure) and TCT (Take Control) are not supported.

Responses to a Multi-Line Message (Universal Command)

- **LLO (Local Lockout)**
  Disables LOCAL on the front panel to prohibit switching to the local mode.
- **DCL (Device Clear)**
  Same operation as the SDC message.
- **SPE (Serial Poll Enable)**
  Sets the talker function on all devices on the bus to serial polling mode. The controller polls the devices in order.
- **SPD (Serial Poll Disable)**
  Clears the serial polling mode of the talker function on all devices on the bus.

  PPU (Parallel Poll Unconfigure) is not supported.

What Is an Interface Message

Interface messages are also referred to as interface commands or bus commands. They are commands that are issued by the controller. They are classified as follows:

Uni-Line Messages

A single control line is used to transmit uni-line messages. The following three types are available.

- IFC (Interface Clear)
- REN (Remote Enable)
- IDY (Identify)

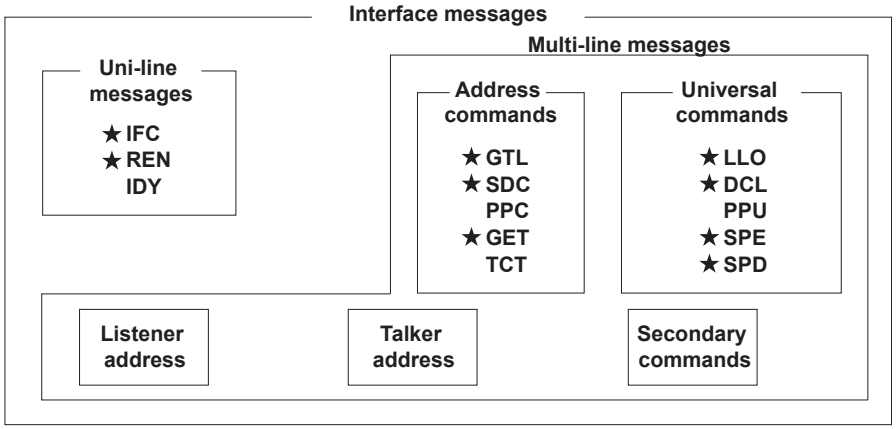
Multi-Line Messages

Eight data lines are used to transmit multi-line messages. The messages are classified as follows:

- **Address Commands**
  These commands are valid when the instrument is designated as a listener or as a talker. The following five types are available.
  - Commands that are valid on an instrument that is designated as a listener
    - GTL (Go To Local)
    - SDC (Selected Device Clear)
    - PPC (Parallel Poll Configure)
    - GET (Group Execute Trigger)
  - Commands that are valid on an instrument that is designated as a talker
    TCT (Take Control)

- **Universal Commands**
  These commands are valid on all instruments regardless of the listener and talker designations. The following five types are available.
  - LLO (Local Lockout)
  - DCL (Device Clear)
  - PPU (Parallel Poll Unconfigure)
  - SPE (Serial Poll Enable)
  - SPD (Serial Poll Disable)

In addition, listener address, talker address, and secondary commands are also considered interface messages.



The WT1500 supports interface messages marked with a ★.

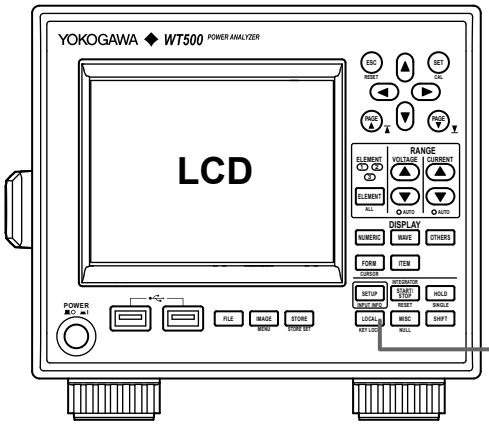
Note

The differences between SDC and DCL

In multi-line messages, SDC messages are those that require talker or listener designation and DCL messages are those that do not require the designation. Therefore, SDC messages are directed at a particular instrument while DCL messages are directed at all instruments on the bus.

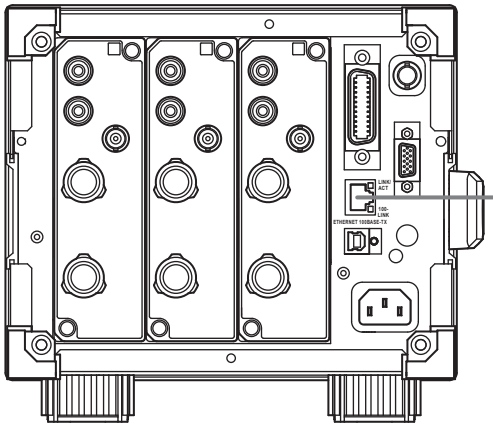
3.1 Names and Functions of Parts

Front Panel



LOCAL key
Press this key to clear the remote mode (controlled via communications) and enter the local mode in which key operations are enabled.

Rear Panel



Ethernet port (100BASE-TX)
Port used to connect to the controller (PC) via the network or by aone-to-one connection. For the connection procedure, see section 11.2 in the User's Manual IM760201-01E.

3.2 Ethernet Interface Functions and Specifications

You can control the WT500 from a PC using the Ethernet interface. YOKOGAWA's dedicated library software (TMCTL) must be installed on the PC.

Receiving Function

You can specify the same settings as those specified by front panel key operations.
Receives output requests for measured and computed data, setup parameters of the panel, and error codes.

Sending Function

Outputs measured and computed data.
Outputs panel setup parameters and the status byte.
Outputs error codes that have occurred.

Switching between Remote and Local Modes

When Switching from Local to Remote Mode

If the WT500 receives a "`:COMMunicate:REMote ON`" command from the PC when it is in the local mode, it switches to the remote mode.

- The REMOTE indicator is turned ON.
- All keys except the LOCAL key are disabled.
- Settings entered in local mode are retained even when the WT500 switches to remote mode.

When Switching from Remote to Local Mode

Pressing LOCAL key in remote mode puts the instrument in local mode. However, this is void when the WT500 has received a "`:COMMunicate:LOCKout ON`" command from the PC (local lockout condition). When the WT500 receives a "`:COMMunicate:REMote OFF`" command from the PC, the WT500 switches to the local mode regardless of the local lockout condition.

- The REMOTE indicator is turned OFF.
- Key operations are enabled.
- Settings entered in remote mode are retained even when the WT500 switches to local mode.

Note

The Ethernet interface cannot be used simultaneously with other communication interfaces (USB or GP-IB).

Ethernet Interface Specifications

Electrical and mechanical specifications: Conforms to IEEE 802.3.
Number of simultaneous connections: 1
Protocol: VXI-11
For details on other specifications, see chapter 11 in the *User's Manual IM760201-01E*.

Connecting the WT500 and the PC

For the procedure of connecting the WT500 to a PC, see section 11.2 in the *User's Manual IM760201-01E*.

Entering TCP/IP Settings

You must enter TCP/IP settings to control the WT500 from a PC using the Ethernet interface. For the setup procedure, see section 11.3 in the *User's Manual IM760201-01E*.

4.1 Messages

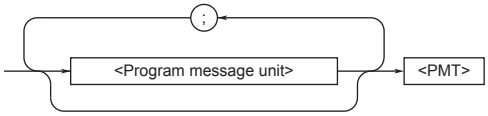
Messages and Queries

Messages are used to exchange information between the controller and the instrument. Messages sent from the controller to the WT500 are called program messages. Program messages that request a response to be sent from the WT500 to the controller are called queries. Messages that the controller receives from the WT500 are called response messages.

If a query is included in a program message, the WT500 sends a response message after receiving the program message. A single response message is always returned in response to a single program message.

Program Messages

The program message format is shown below.



<Program Message Unit>

A program message consists of one or more program message units; each unit corresponds to one command. The instrument executes the received commands in order.

Each program message unit is separated by a semicolon (`;`).

For details regarding the format of the program message unit, see the next section.

Example

```
:INPut:CFACtor 3;INDependent OFF<PMT>
```

Unit Unit

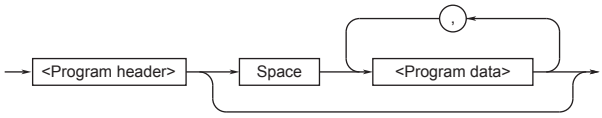
<PMT>

PMT is a program message terminator. The following three types are available.

- **NL (New Line)**
  Same as LF (Line Feed). ASCII code “0AH.”
- **^END**
  The END message (EOI signal) as defined in the IEEE488.1. (The data byte that is sent with the END message is the last data of the program message.)
- **NL^END**
  NL with an END message attached. (NL is not included in the program message.)

Program Message Unit Format

The program message unit format is shown below.



<Program Header>

The program header indicates the command type. For details, see page 4-3.

<Program Data>

If certain conditions are required in executing a command, program data is added. A space (ASCII code “20H”) separates the program data from the header. If there are multiple sets of program data, they are separated by commas (`,`).

For details, see page 4-5.

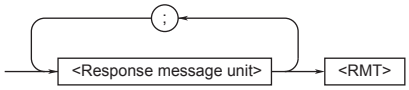
Example

```
:INPut:CFACtor 3<PMT>
```

Header Data

Response Message

The response message format is shown below.



<Response Message Unit>

A response message consists of one or more response message units; each response message unit corresponds to one response.

Response message units are separated by a semicolon (`;`).

For details regarding the format of the response message unit, see the next section.

Example

```
:INPUT:CFACTOR 3;INDEPENDENT 0<RMT>
```

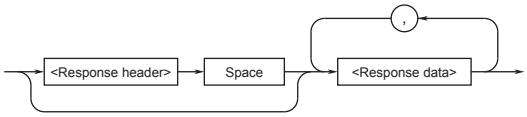
Unit Unit

<RMT>

A response message terminator. It is NL^END.

Response Message Unit Format

The response message unit format is shown below.



<Response Header>

A response header sometimes precedes the response data. A space separates the data from the header. For details, see page 4-4.

<Response Data>

Response data contains the content of the response. If there are multiple sets of response data, they are separated by commas (,). For details, see page 4-5.

Example

```
100.00E-03<RMT>  :DISPLAY:MODE WAVE<RMT>
```

Data — Header — Data

If there are multiple queries in a program message, responses are made in the same order as the queries. In most cases, a single query returns a single response message unit, but there are a few queries that return multiple units. The first response message unit always corresponds to the first query, but the n^{th} response unit may not necessarily correspond to the n^{th} query. Therefore, if you want to make sure that every response is retrieved, divide the program messages into individual messages.

Precautions to Be Taken when Transferring Messages

- If a program message that does not contain a query is sent, the next program message can be sent at any time.
- If a program message that contains a query is sent, a response message must be received before the next program message can be sent. If the next program message is sent before the response message is received in its entirety, an error occurs. The response message that was not received is discarded.
- If the controller tries to receive a response message when there is none, an error occurs. If the controller tries to receive a response message before the transmission of the program message is complete, an error occurs.
- If a program message containing multiple message units is sent, and the message contains incomplete units, the instrument attempts to execute the ones that are believed to be complete. However, these attempts may not always be successful. In addition, if the message contains queries, the responses may not be returned.

Deadlock

The instrument can store in its buffer program and response messages of length 1024 bytes or more (The number of available bytes varies depending on the operating conditions). When both the transmit and receive buffers become full at the same time, the instrument can no longer continue its communication operation. This state is called a deadlock. In this case, operation can be resumed by discarding the program message.

Deadlock will not occur if the program message (including the <PMT>) is kept below 1024 bytes. Furthermore, deadlock never occurs if a program message does not contain a query.

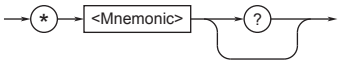
4.2 Commands

Commands

There are three types of commands (program headers) that are sent from the controller to the instrument. They differ in their program header formats.

Common Command Header

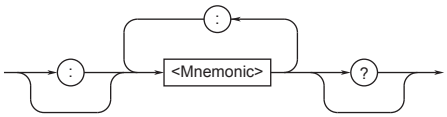
Commands that are defined in the IEEE 488.2-1992 are called common commands. The header format of a common command is shown below. An asterisk (*) is always placed in the beginning of a command.



Common command example *CLS

Compound Header

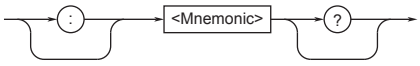
Dedicated commands used by the instrument are classified and arranged in a hierarchy according to their functions. The format of a compound header is shown below. A colon (:) must be used to specify a lower hierarchy.



Compound header example :DISPlay:MODE

Simple Header

These commands are functionally independent and do not have a hierarchy. The format of a simple header is shown below.



Simple header example :HOLD

Note

A <mnemonic> is a character string made up of alphanumeric characters.

When Concatenating Commands

Command Group

A command group is a group of commands that have common compound headers arranged in a hierarchy. A command group may contain sub-groups.

Example Group of commands related to integration measurement

```
:INTEGrate?
:INTEGrate:MODE
:INTEGrate:TIMer
:INTEGrate:RTIMe?
:INTEGrate:RTIMe:STARt
:INTEGrate:RTIMe:END
:INTEGrate:ACAL
:INTEGrate:WPTYpe
:INTEGrate:QMODe
:INTEGrate:STARt
:INTEGrate:STOP
:INTEGrate:RESet
:INTEGrate:STATe?
```

When Concatenating Commands of the Same Group

The instrument stores the hierarchical level of the command that is currently being executed, and performs analysis on the assumption that the next command sent will also belong to the same level. Therefore, common header sections can be omitted for commands belonging to the same group.

Example
```
:INTEGrate:MODE NORMal;
ACAL ON <PMT>
```

When Concatenating Commands of Different Groups

If the following command does not belong to the same group, a colon (:) is placed in front of the header.

Example
```
:INTEGrate:MODE NORMal;:
DISPlay:MODE NUMeric<PMT>
```

When Concatenating Simple Headers

If a simple header follows another command, a colon (:) is placed in front of the simple header.

Example
```
:INTEGrate:MODE NORMal;:
HOLD ON<PMT>
```

When Concatenating Common Commands

Common commands that are defined in the IEEE 488.2-1992 are independent of hierarchy. Colons (:) are not needed before a common command.

Example
```
:INTEGrate:MODE NORMal;*CLS;
ACAL ON<PMT>
```

When Separating Commands with <PMT>

If a terminator is used to separate two commands, each command is a separate message. Therefore, the common header must be specified for each command even when commands belonging to the same command group are being concatenated.

Example
```
:INTEGrate:MODE NORMal<PMT>:
INTEGrate:ACAL ON<PMT>
```

Upper-Level Query

An upper-level query is a query that is made by appending a question mark to a command higher in the group. The controller can receive all of the settings in a group collectively by executing a highest-level query. Some query groups which are comprised of more than three hierarchical levels can output all the lower level settings.

Example
```
:INTEGrate?<PMT> ->
:INTEGRATE:MODE NORMAL;
TIMER 0,0,0;ACAL 0;
WPTYPE CHARGE;QMODE DC<RMT>
```

The response to an upper-level query can be transmitted as a program message back to the instrument. In this way, the settings that existed when the upper-level query was made can be restored. However, some upper-level queries do not return setup information that is not currently in use. It is important to remember that not all the group's information is necessarily returned as part of a response.

Header Interpretation Rules

The instrument interprets the header that is received according to the rules below.

- Mnemonics are not case sensitive.
  Example "`CURSor`" can also be written as "`cursor`" or "`CUrsor`."
- The lower-case section of the header can be omitted.
  Example "`CURSor`" can also be written as "`CURSO`" or "`CURS`."
- The question mark (`?`) at the end of a header indicates that it is a query. The question mark (`?`) cannot be omitted.
  Example: The shortest abbreviation for `CURSor?` is `CURS?`.
- If the <x> (value) at the end of a mnemonic is omitted, it is interpreted as a 1.
  Example If "`ELEMent<x>`" is written as "`ELEM`," it means "`ELEMent1`."
- The section enclosed by braces ([]) can be omitted.
  Example "`[:INPut]:SCALing[:STATe][:ALL] ON`" can be written as "`SCAL ON`."
  However, the last section enclosed by braces ([]) cannot be omitted in an upper-level query.
  Example "`SCALing?`" and "`SCALing:STATe?`" are different queries.

4.3 Responses

When the controller sends a message unit that has a question mark (?) in its program header (query), the instrument returns a response message to the query. A response message is returned in one of the following two forms.

- **Response Consisting of a Header and Data**
  If the response can be used as a program message without any change, it is returned with a command header attached.
  Example `:DISPlay:MODE?<PMT>`
  `-> :DISPLAY:MODE WAVE<RMT>`

- **Response Consisting of Data Only**
  If the response cannot be used as a program message unless changes are made to it (query-only command), only the data section is returned. However, there are query-only commands that return responses with the header attached.
  Example `[:INPut]:POVer?<PMT> -> 0<RMT>`

When You Wish to Return a Response without a Header

Responses that return both header and data can be set so that only the data section is returned. The "`COMMunicate:HEADer`" command is used to do this.

Abbreviated Form

Normally, the lower-case section is removed from a response header before the response is returned to the controller. Naturally, the full form of the header can also be used. For this, the "`COMMunicate:VERBose`" command is used. The sections enclosed by braces (`[]`) are also omitted in the abbreviated form.

4.4 Data

Data

A data section comes after the header. A space must be included between the header and the data. The data contains conditions and values. Data is classified as below.

Data	Description
<Decimal>	A value expressed as a decimal number (Example: VT ratio setting -> `[:INPUt]:SCALing:VT: ELEMent1 100`)
<Voltage><Current> <Time><Frequency>	A physical value (Example: Voltage range setting ->`[:INPUt]:VOLTage:RANGE: ELEMent1 100V`)
<Register>	Register value expressed as binary, octal, decimal or hexadecimal. (Example: Extended event register value ->`:STATUS:EESE #HFE`)
<Character Data>	Predefined character string (mnemonic). Can be selected from { }. (Example: Trigger mode selection ->`:DISPlay:WAVE:TRIGger: MODE {AUTO|NORMal}`)
<Boolean>	Indicates ON and OFF. Set using ON, OFF or a value (Example: Data hold ON ->`:HOLD ON`)
<String data>	An arbitrary character string (Example: User-defined function ->`:MEASure:FUNCtion1: EXPRession "URMS(E1)"`)
<Block data>	Arbitrary 8-bit data (Example: Response to acquired waveform data ->`#40012ABCDEFGHIJKL`)

<Decimal>

<Decimal> indicates a value expressed as a decimal number, as shown in the table below. Decimal values are given in the NR form as specified in the ANSI X3.42-1975.

Symbol	Meaning	Example		
<NR1>	Integer	125	-1	+1000
<NR2>	Fixed-point number	125.0	-.90	+001.
<NR3>	Floating-point number	125.0E+0	-9E-1	+.1E4
<NRf>	Any of the forms <NR1> to <NR3> is allowed.			

- The instrument can receive decimal values that are sent from the controller in any of the forms, <NR1> to <NR3>. This is represented by <NRf>.
- For response messages that the instrument returns to the controller, the form (<NR1> to <NR3> to be used) is determined by the query. The same form is used regardless of the size of the value.
- For the <NR3> format, the "+" sign after the "E" can be omitted. However, the "−" sign cannot be omitted.
- If a value outside the setting range is entered, the value is normalized so that it is just inside the range.
- If a value has more significant digits than the available resolution, the value is rounded.

<Voltage>, <Current>, <Time>, and <Frequency>

<Voltage>, <Current>, <Time>, and <Frequency> indicate data that have physical dimensions. <Multiplier> or <Unit> can be attached to the <NRf> form that was described earlier. It is expressed in one of the following forms.

Form	Example
<NRf><Multiplier><Unit>	5MV
<NRf><Unit>	5E-3V
<NRf>	5E-3

<Multiplier>

<Multipliers> which can be used are indicated below.

Symbol	Word	Multiplier
EX	Exa	10^{18}
PE	Peta	10^{15}
T	Tera	10^{12}
G	Giga	10^{9}
MA	Mega	10^{6}
K	Kilo	10^{3}
M	Milli	10^{-3}
U	Micro	10^{-6}
N	Nano	10^{-9}
P	Pico	10^{-12}
F	Femto	10^{-15}

<Unit>

<Units> that can be used are indicated below.

Symbol	Word	Meaning
V	Volt	Voltage
A	Ampere	Current
S	Second	Time
HZ	Hertz	Frequency
MHZ	Megahertz	Frequency

- <Multiplier> and <Unit> are not case sensitive.
- "U" is used to indicate micro "μ".
- "MA" is used for Mega to distinguish it from Milli. However, "MA" is interpreted as milliampere for current. In addition, megahertz is expressed as "MHZ." Therefore, the "M (Milli)" multiplier cannot be used for frequencies.
- If both <Multiplier> and <Unit> are omitted, the default unit (V, A, S, or HZ) is used.
- Response messages are always expressed in the <NR3> form. Response messages are returned using the default unit without the <Multiplier> or <Unit>.

<Register>

<Register> indicates an integer, and can be expressed in hexadecimal, octal, or binary as well as a decimal number. It is used when each bit of the value has a particular meaning. It is expressed in one of the following forms.

Form	Example
<NRf>	1
#H<Hexadecimal value made up of the digits 0 to 9 and A to F>	#H0F
#Q<Octal value made up of the digits 0 to 7>	#Q777
#B<Binary value made up of the digits 0 and 1>	#B001100

- <Register> is not case sensitive.
- Response messages are always expressed as <NR1>.

<Character Data>

<Character Data> is a specified string of character data (a mnemonic). It is mainly used to indicate options and is chosen from the character strings given in { }. For interpretation rules, refer to “Header Interpretation Rules” on page 4-4.

Form	Example
{AUTO|NORMal}	AUTO

- As with the header, the “COMMunicate:VERBose” command can be used to select whether to return the response in the full form or in the abbreviated form.
- The “COMMunicate:HEADer” setting does not affect the character data.

<Boolean>

<Boolean> is data that indicates ON or OFF. It is expressed in one of the following forms.

Form	Example
{ON|OFF|<NRf>}	ON OFF 1 0

- When <Boolean> is expressed in the <NRf> form, “OFF” is selected if the rounded integer value is 0, and ON for all other cases.
- A response message is always returned with a 1 if the value is ON and 0 if the value is OFF.

<String Data>

<String data> is not a specified character string like <Character data>. It is an arbitrary character string. The character string must be enclosed in single quotation marks (') or double quotation marks (").

Form	Example
<String data>	'ABC' "IEEE488.2-1992"

- If a character string contains a double quotation mark ("), the double quotation mark is replaced by two concatenated double quotation marks (""). This rule also applies to a single quotation mark within a character string.
- A response message is always enclosed in double quotation marks (").
- <String data> is an arbitrary character string. Therefore the instrument assumes that the remaining program message units are part of the character string if no single (') or double quotation mark (") is encountered. As a result, no error is detected if a quotation mark is omitted.

<Block Data>

<Block data> is arbitrary 8-bit data. It is only used in response messages on the WT500. The syntax is as follows:

Form	Example
#N<N-digit decimal number><Data byte sequence>	#40012ABCDEFGHIJKL

- #N
  Indicates that the data is <Block data>. “N” indicates the number of succeeding data bytes (digits) in ASCII code characters.
- <N-digit decimal number>
  Indicates the number of bytes of data (example: 0012 = 12 bytes).
- <Data byte sequence>
  Expresses the actual data (example: ABCDEFGHIJKL).
- Data is comprised of 8-bit values (0 to 255). This means that the ASCII code “0AH,” which stands for “NL,” can also be a code used for data. Hence, care must be taken when programming the controller.

4.5 Synchronization with the Controller

Overlap Commands and Sequential Commands

There are two types of commands, overlap commands and sequential commands. In the case of overlap commands, the execution of the next command may start before the execution of the previous command is completed.

For example, if the next program message is transmitted when specifying the voltage range and querying the result, the response always returns the most recent setting (100 V in this case).

```
:INPut:VOLTage:RANGe:ELEMent1 100V;
ELEMent?<PMT>
```

This is because the next command is forced to wait until the processing of "`INPut:VOLTage:RANge:ELEMent1`" itself is completed. This type of command is called a sequential command.

On the contrary, let us assume that you send the next program message when you wish to load a file and query the voltage range of the result.

```
:FILE:LOAD:SETup "FILE1";:INPut:VOLTage:
RANGe:ELEMent1?
```

In this case, "`:INPut:VOLTage:RANGe:ELEMent1?`" is executed before the loading of the file is completed, and the voltage range that is returned is the value before the file is loaded.

The act of executing the next command before the processing of itself is completed such as with "`FILE:LOAD:SETup`" is called an overlap operation. A command that operates in this way is called an overlap command.

In such case, the overlap operation can be prevented by using the methods below.

Synchronizing with Overlap Commands

Using the *WAI Command

The `*WAI` command holds the subsequent commands until the overlap command is completed.

Example
```
:COMMunicate:OPSE
#H0040;:FILE:LOAD:SETup
"FILE1";*WAI;:INPut:VOLTage:
RANGe:ELEMent1?<PMT>
```

"`COMMunicate:OPSE`" is a command used to select the "`*WAI`" target. Here, media access is specified.

Because "`*WAI`" is executed immediately before "`:INPut:VOLTage:RANGe:ELEMent1?`," "`:INPut:VOLTage:RANGE:ELEMent1?`" is not executed until the loading of the file is completed.

Using the COMMunicate:OVERlap command

The `COMMunicate:OVERlap` command enables (or disables) overlap operation.

Example
```
:COMMunicate:OVERlap
#HFFBF;:FILE:
LOAD:SETup
"FILE1";:INPut:VOLTage:
RANGe:ELEMent1?<PMT>
```

"`COMMunicate:OVERlap #HFFBF`" enables overlap operation on commands other than media access.

Because the overlap operation of file loading is disabled, "`FILE:LOAD:SETup`" operates in the same way as a sequential command. Therefore, "`:INPut:VOLTage:RANGe:ELEMent1?`" is not executed until the loading of the file is completed.

Using the *OPC Command

The *OPC command sets the OPC bit, bit 0 of the standard event register (see page 6-5), to 1 when the overlap operation is completed.

Example
```
:COMMunicate:OPSE #H0040;
*ESE 1;*ESR?;*SRE 32;:FILE:LOAD:
SETup "FILE1";*OPC<PMT>
(Read the response to *ESR?)
(Wait for a service request)
:INPut:VOLTage:RANGe:
ELEMent1?<PMT>
```

"COMMunicate:OPSE" is a command used to select the "*OPC" target. Here, media access is specified. "*ESE 1" and "*SRE 32" indicate that a service request is generated only when the OPC bit is 1."*ESR?" clears the standard event register.In the example above, ":INPut:VOLTage:RANGe:ELEMent1?" is not executed until a service request is generated.

Using the *OPC? Query

The *OPC? query generates a response when an overlap operation is completed.

Example
```
:COMMunicate:OPSE #H0040;:FILE:
LOAD:SETup "FILE1";*OPC?<PMT>
(Read the response to *OPC?)
:INPut:VOLTage:RANGe:
ELEMent?<PMT>
```

"COMMunicate:OPSE" is a command used to select the "*OPC?" target. Here, media access is specified.Because "*OPC?" does not generate a response until the overlap operation is completed, the loading of the file will have been completed by the time the response to "*OPC?" is read.

Note

Most commands are sequential commands. Overlap commands are indicated as overlap commands in chapter 5. All other commands are sequential commands.

Achieving Synchronization without Using Overlap Commands

Even for sequential commands, synchronization is sometimes required to correctly query the measured data. If you wish to query the most recent numeric data on every time measured data is updated, for example, sending the ":NUMeric[:NORMal]:VALue?" command at an arbitrary timing can cause data that is the same as the previous data to be received. This is because the WT500 returns the current measured data regardless of whether the measured data has been updated since the previous query.In this case, the following method must be used to synchronize with the end of the updating of the measured data.

Using the STATus:CONDition? query

The "STATus:CONDition?" query is used to query the contents of the condition register (page 6-7). You can determine whether the measured data is being updated by reading bit 0 of the condition register. If bit 0 of the condition register is 1, the measured data is being updated. If it is 0, the measured data can be queried.

Using the Extended Event Register

The changes in the condition register can be reflected in the extended event register (page 6-7).

Example
```
:STATus:FILTer1 FALL;:STATus:
EESE 1;EESR?;*SRE 8<PMT>
(Read the response to STATus:EESR?)
Loop
(Wait for a service request)
:NUMeric[:NORMal]:VALue?<PMT>
(Read the response to
:NUMeric[:NORMal]:VALue?)
:STATus:EESR?<PMT>
(Read the response to :STATus:EESR?)
(Return to LOOP)
```

The "`STATus:FILTer1 FALL`" command sets the transition filter so that bit 0 (`FILTer1`) of the extended event register is set to `1` when bit 0 of the condition register changes from `1` to `0`.
The "`:STATus:EESE 1`" command is used to reflect only bit 0 of the extended event register to the status byte.
The "`STATus:EESR?`" command is used to clear the extended event register.
The "`*SRE 8`" command is used to generate a service request solely on the cause of the extended event register.
The "`:NUMeric[:NORMal]:VALue?`" command is not executed until a service request is generated.

Using the COMMunicate:WAIT command

The "`COMMunicate:WAIT`" command halts communications until a specific event is generated.

Example
```
:STATus:FILTer1 FALL;:STATus:
EESR?<PMT>
(Read the response to STATus:EESR?)
Loop
COMMunicate:WAIT 1<PMT>
:NUMeric[:NORMal]:VALue?<PMT>
(Read the response to
:NUMeric[:NORMal]:VALue?)
:STATus:EESR?<PMT>
(Read the response to :STATus:EESR?)
(Return to LOOP)
```

For a description of "`STATus:FILTer1 FALL`" and "`STATus:EESR?`" see the previous section regarding the extended event register.
The "`COMMunicate:WAIT 1`" command indicates that the program will wait for bit 0 of the extended event register to be set to "`1`."
The "`:NUMeric[:NORMal]:VALue?`" command is not executed until bit 0 of the extended event register is set to "`1`."

5.1 A List of Commands

Command	Function	Page
COMMunicate Group		
:COMMunicate?	Queries all communication settings.	5-9
:COMMunicate:HEADer	Sets or queries whether or not a header is added to the response to a query.	5-9
:COMMunicate:LOCKout	Sets or clears local lockout.	5-9
:COMMunicate:OPSE	Sets or queries the overlap command that is used by the *OPC, *OPC?, and *WAI commands.	5-9
:COMMunicate:OPSR?	Queries the operation pending status register.	5-9
:COMMunicate:OVERlap	Sets or queries the commands that operate as overlap commands.	5-9
:COMMunicate:REMote	Sets remote or local.	5-9
:COMMunicate:VERBose	Sets or queries whether the response message format is full or abbreviated.	5-10
:COMMunicate:WAIT	Waits for a specified extended event.	5-10
:COMMunicate:WAIT?	Creates the response that is returned when a specified extended event occurs.	5-10
CURSor Group		
:CURSor?	Queries all cursor measurement settings.	5-11
:CURSor:BAR?	Queries all bar graph display cursor measurement settings.	5-11
:CURSor:BAR:LINKage	Sets or queries the on/off status of the cursor position linkage in the bar graph display.	5-11
:CURSor:BAR:POSition<x>	Sets or queries the position (harmonic order) of the specified cursor in the bar graph display.	5-11
:CURSor:BAR[:STATe]	Sets or queries the on/off status of the cursor display in the bar graph display.	5-11
:CURSor:BAR:{Y<x>|DY}?	Queries the measured value of the specified cursor in the bar graph display.	5-11
:CURSor:TRENd?	Queries all trend display cursor measurement settings.	5-11
:CURSor:TRENd:LINKage	Sets or queries the on/off status of the cursor position linkage in the trend display.	5-11
:CURSor:TRENd:POSition<x>	Sets or queries the position of the specified cursor in the trend display.	5-11
:CURSor:TRENd[:STATe]	Sets or queries the on/off status of the cursor display in the trend display.	5-12
:CURSor:TRENd:TRACe<x>	Sets or queries the target of the specified cursor in the trend display.	5-12
:CURSor:TRENd:{X<x>|Y<x>|DY}?	Queries the measured value of the specified cursor in the trend display.	5-12
:CURSor:WAVE?	Queries all waveform display cursor measurement settings.	5-12
:CURSor:WAVE:LINKage	Sets or queries the on/off status of the cursor position linkage in the waveform display.	5-12
:CURSor:WAVE:PATH	Sets or queries the cursor path in the waveform display.	5-12
:CURSor:WAVE:POSition<x>	Sets or queries the position of the specified cursor in the waveform display.	5-12
:CURSor:WAVE[:STATe]	Sets or queries the on/off status of the cursor display in the waveform display.	5-12
:CURSor:WAVE:TRACe<x>	Sets or queries the target of the specified cursor in the waveform display.	5-12
:CURSor:WAVE:{X<x>|DX|PERDt|Y<x>|DY}?	Queries the measured value of the specified cursor in the waveform display.	5-12
DISPlay Group		
:DISPlay?	Queries all screen display settings.	5-13
:DISPlay:BAR?	Queries all bar graph settings.	5-13
:DISPlay:BAR:FORMat	Sets or queries the bar graph display format.	5-13
:DISPlay:BAR:ITEM<x>	Sets or queries the function and element of the specified bar graph item.	5-13
:DISPlay:BAR:ORDer	Sets or queries the bar graph starting and ending harmonic orders.	5-13
:DISPlay:INFOrmation?	Queries all setup parameter list display settings.	5-13
:DISPlay:INFOrmation[:STATe]	Sets or queries the on/off status of the setup parameter list.	5-13
:DISPlay:MODE	Sets or queries the display mode.	5-14
:DISPlay:NUMeric?	Queries all numeric display settings (the same as :DISPlay:NUMeric:NORMal?).	5-14

5.1 A List of Commands

Command	Function	Page
:DISPlay:NUMeric:NORMal?	Queries all numeric display settings.	5-14
:DISPlay:NUMeric[:NORMal]:ALL?	Queries all numeric display settings (in All display mode).	5-14
:DISPlay:NUMeric[:NORMal]:ALL:CURSor	Sets or queries the cursor position in the numeric display (in All display mode).	5-14
:DISPlay:NUMeric[:NORMal]:ALL:ORDer	Sets or queries the displayed harmonic order on the harmonic measurement function display page of the numeric display (in All display mode).	5-14
:DISPlay:NUMeric[:NORMal]:ALL:PAGE	Sets or queries the page number in the numeric display (in All display mode).	5-14
:DISPlay:NUMeric[:NORMal]:FORMat	Sets or queries the numeric display format.	5-15
:DISPlay:NUMeric[:NORMal]:LIST?	Queries all numeric display settings (in the list display modes).	5-15
:DISPlay:NUMeric[:NORMal]:LIST:CURSor	Sets or queries the cursor position in the numeric display (in the list display modes).	5-15
:DISPlay:NUMeric[:NORMal]:LIST:HEADer	Sets or queries the cursor position in the header section of the numeric display (in the list display modes).	5-15
:DISPlay:NUMeric[:NORMal]:LIST:ITEM<x>	Sets or queries the function and element of the specified item in the numeric display (in the list display modes).	5-15
:DISPlay:NUMeric[:NORMal]:LIST:ORDer	Sets or queries the harmonic order cursor position in the data section of the numeric display (in the list display modes).	5-16
:DISPlay:NUMeric[:NORMal]:{VAL4|VAL8|VAL16|MATRix}?	Queries all settings related to the numeric display ({4-value|8-value|16-value|matrix} display).	5-16
:DISPlay:NUMeric[:NORMal]:{VAL4|VAL8|VAL16|MATRix}:CURSor	Sets or queries the cursor position in the numeric display ({4-value|8-value|16-value|matrix} display).	5-16
:DISPlay:NUMeric[:NORMal]:{VAL4|VAL8|VAL16|MATRix}:ITEM<x>	Sets or queries the function, element, and harmonic order of the specified item in the numeric display (for the 4-value, 8-value, 16-value, and matrix display).	5-16
:DISPlay:NUMeric[:NORMal]:{VAL4|VAL8|VAL16|MATRix}:PAGE	Sets or queries the page number in the numeric display ({4-value|8-value|16-value|matrix} display).	5-17
:DISPlay:NUMeric[:NORMal]:{VAL4|VAL8|VAL16|MATRix}:PRESet	Presets the order of displayed items in the numeric display ({4-value|8-value|16-value|matrix} display).	5-17
:DISPlay:TRENd?	Queries all trend settings.	5-17
:DISPlay:TRENd:ALL	Collectively turns all trends on or off.	5-17
:DISPlay:TRENd:CLEar	Clears all trends.	5-17
:DISPlay:TRENd:FORMat	Sets or queries the display format of all trends.	5-17
:DISPlay:TRENd:ITEM<x>?	Queries all settings for the specified trend.	5-17
:DISPlay:TRENd:ITEM<x>[:FUNCtion]	Sets or queries the function, element, and harmonic order of the specified trend item.	5-17
:DISPlay:TRENd:ITEM<x>:SCALing?	Queries all scaling settings for the specified trend.	5-17
:DISPlay:TRENd:ITEM<x>:SCALing:MODE	Sets or queries the scaling mode of the specified trend.	5-18
:DISPlay:TRENd:ITEM<x>:SCALing:VALue	Sets or queries the upper and lower manual scaling limits of the specified trend.	5-18
:DISPlay:TRENd:TDIV	Sets or queries the trend horizontal axis (T/div).	5-18
:DISPlay:TRENd:T<x>	Sets or queries the on/off status of the specified trend.	5-18
:DISPlay:VECTor?	Queries all vector display settings.	5-18
:DISPlay:VECTor:NUMeric	Sets or queries the on/off status of the numeric data display in the vector display.	5-18
:DISPlay:VECTor:OBJect	Sets or queries the wiring unit to be displayed in the vector display.	5-18
:DISPlay:VECTor:{UMAG|IMAG}	Sets or queries the zoom factor for the vector display.	5-18
:DISPlay:WAVE?	Queries all waveform display settings.	5-18
:DISPlay:WAVE:ALL	Collectively turns all waveform displays on or off.	5-19
:DISPlay:WAVE:FORMat	Sets or queries the waveform display format.	5-19
:DISPlay:WAVE:GRATicule	Sets or queries the graticule (grid) type.	5-19
:DISPlay:WAVE:INTerpolate	Sets or queries the waveform interpolation method.	5-19
:DISPlay:WAVE:MAPPing?	Queries all split screen waveform mapping settings.	5-19
:DISPlay:WAVE:MAPPing[:MODE]	Sets or queries the split screen waveform mapping mode.	5-19
:DISPlay:WAVE:MAPPing:{U<x>|I<x>}	Sets or queries the split screen waveform mapping for the specified waveform.	5-19
:DISPlay:WAVE:POSition?	Queries all waveform vertical position settings (center position levels).	5-19
:DISPlay:WAVE:POSition:{UALL|IALL}	Collectively sets the vertical positions (center position levels) of the voltage or current waveforms of all elements.	5-19
:DISPlay:WAVE:POSition:{U<x>|I<x>}	Sets or queries the vertical position (center position level) of the specified voltage or current waveform.	5-19

Command	Function	Page
:DISPlay:WAVE:SVALue	Sets or queries the on/off status of the scale value display.	5-19
:DISPlay:WAVE:TDIV	Sets or queries the waveform Time/div value.	5-20
:DISPlay:WAVE:TLABel	Sets or queries the on/off status of the waveform labels.	5-20
:DISPlay:WAVE:TRIGger?	Queries all trigger settings.	5-20
:DISPlay:WAVE:TRIGger:LEVel	Sets or queries the trigger level.	5-20
:DISPlay:WAVE:TRIGger:MODE	Sets or queries the trigger mode.	5-20
:DISPlay:WAVE:TRIGger:SLOPe	Sets or queries the trigger slope.	5-20
:DISPlay:WAVE:TRIGger:SOURce	Sets or queries the trigger source.	5-20
:DISPlay:WAVE:{U<x>|I<x>}	Sets or queries the on/off status of the waveform display.	5-20
:DISPlay:WAVE:VZoom?	Queries all waveform vertical zoom factor settings.	5-20
:DISPlay:WAVE:VZoom:{UALL|IALL}	Collectively sets the vertical zoom factor for the voltage or current waveforms of all elements.	5-20
:DISPlay:WAVE:VZoom:{U<x>|I<x>}	Sets or queries the vertical zoom factor for the voltage or current waveform of the specified element.	5-21

FILE Group

:FILE?	Queries all file operation settings.	5-24
:FILE:CDIRectory	Changes the current directory.	5-24
:FILE:DELete:IMAGe:{TIFF|BMP|PSCRipt|PNG|JPEG}	Deletes a screen image data file.	5-24
:FILE:DELete:NUMeric:ASCii	Deletes a numeric data file.	5-24
:FILE:DELete:SETup	Deletes a setup parameter file.	5-24
:FILE:DELete:STORe:{DATA|HEADer}	Deletes a stored numeric data file.	5-24
:FILE:DELete:WAVE:ASCii	Deletes a waveform display data file.	5-24
:FILE:DRIVe	Sets the current drive.	5-24
:FILE:FILTer	Sets or queries the file list filter.	5-24
:FILE:FREE?	Queries the free space (in bytes) on the current drive.	5-24
:FILE:LOAD:ABORt	Aborts file loading.	5-24
:FILE:LOAD:SETup	Loads a setup parameter file.	5-24
:FILE:PATH?	Queries the absolute path of the current directory.	5-24
:FILE:SAVE?	Queries all file save settings.	5-25
:FILE:SAVE:ABORt	Aborts file saving.	5-25
:FILE:SAVE:ANAMing	Sets or queries whether or not files are automatically named when they are saved.	5-25
:FILE:SAVE:COMMent	Sets or queries the comment that is added to files when they are saved.	5-25
:FILE:SAVE:NAME	Sets or queries the name of the file that will be saved.	5-25
:FILE:SAVE:NUMeric[:EXECute]	Saves numeric data to a file.	5-25
:FILE:SAVE:NUMeric:NORMal?	Queries all numeric data file save settings.	5-25
:FILE:SAVE:NUMeric:NORMal:ALL	Collectively turns on or off the output of all element functions when saving numeric data to a file.	5-25
:FILE:SAVE:NUMeric:NORMal:{ELEMent<x>|SIGMA}	Sets or queries the on/off status of the output of each element or of wiring unit Σ when saving numeric data to a file.	5-25
:FILE:SAVE:NUMeric:NORMal:PRESet<x>	Presets the output on/off pattern of the element functions that are used when saving numeric data to a file.	5-25
:FILE:SAVE:NUMeric:NORMal:<Function>	Sets or queries the on/off status of the output of the specified function when saving numeric data to a file.	5-26
:FILE:SAVE:SETup[:EXECute]	Saves setup parameters to a file.	5-26
:FILE:SAVE:WAVE[:EXECute]	Saves waveform display data to a file.	5-26

HARMonics Group

:HARMonics?	Queries all harmonic measurement settings.	5-27
:HARMonics:ORDer	Sets or queries the maximum and minimum harmonic orders to be measured.	5-27
:HARMonics:PLLSource	Sets or queries the PLL source.	5-27
:HARMonics:THD	Sets or queries the equation used to compute the THD (total harmonic distortion).	5-27

HOLD Group

:HOLD	Sets or queries the on/off status of the output hold feature for display, communication, and other types of data.	5-28

Command	Function	Page
IMAGe Group		
:IMAGe?	Queries all screen image data save settings.	5-29
:IMAGe:ABORt	Aborts the saving of screen image data.	5-29
:IMAGe:COLor	Sets or queries the color tone of the screen image data that will be saved.	5-29
:IMAGe:COMMent	Sets or queries the comment displayed at the bottom of the screen.	5-29
:IMAGe:COMPression	Sets or queries the data compression when the screen image data is in BMP format.	5-29
:IMAGe:EXECute	Saves the screen image data.	5-29
:IMAGe:FORMat	Sets or queries the output format of the screen image data that will be saved.	5-29
:IMAGe:SAVE?	Queries all screen image data save settings.	5-29
:IMAGe:SAVE:ANAMing	Sets or queries whether or not screen image data files are automatically named when they are saved.	5-29
:IMAGe:SAVE:CDIRectory	Changes the directory that screen image data is saved to.	5-29
:IMAGe:SAVE:DRIVe	Sets the drive that screen image data is saved to.	5-29
:IMAGe:SAVE:FREE?	Queries the free space (in bytes) on the drive that the screen image data is saved to.	5-29
:IMAGe:SAVE:NAME	Sets or queries the name of the screen image data file that will be saved.	5-30
:IMAGe:SAVE:PATH?	Queries the absolute path of the directory that the screen image data is saved to.	5-30
:IMAGe:SEND?	Queries the screen image data.	5-30
INPut Group		
:INPut?	Queries all input element settings.	5-31
[:INPut]:CFACtor	Sets or queries the crest factor.	5-31
[:INPut]:CURRent?	Queries all electric current measurement settings.	5-31
[:INPut]:CURRent:AUTO?	Queries the electric current auto range on/off status of all elements.	5-31
[:INPut]:CURRent:AUTO[:ALL]	Collectively turns the electric current auto range of all elements on or off.	5-31
[:INPut]:CURRent:AUTO:ELEMent<x>	Sets or queries the electric current auto range on/off status of the specified element.	5-31
[:INPut]:CURRent:AUTO:SIGMA	Collectively turns the electric current auto range of all elements belonging to wiring unit Σ on or off.	5-31
[:INPut]:CURRent:RANGe?	Queries the electric current range of all elements.	5-31
[:INPut]:CURRent:RANGe[:ALL]	Collectively sets the electric current range of all elements.	5-31
[:INPut]:CURRent:RANGe:ELEMent<x>	Sets or queries the electric current range of the specified element.	5-32
[:INPut]:CURRent:RANGe:SIGMA	Collectively sets the electric current range of all elements belonging to wiring unit Σ.	5-32
[:INPut]:CURRent:SRATio?	Queries the external electric current sensor scaling ratios of all elements.	5-32
[:INPut]:CURRent:SRATio[:ALL]	Collectively sets the external electric current sensor scaling ratios of all elements.	5-32
[:INPut]:CURRent:SRATio:ELEMent<x>	Sets or queries the external electric current sensor scaling ratio of the specified element.	5-32
[:INPut]:ESELect	Sets or queries the element whose measurement range will be set.	5-32
[:INPut]:FILTer?	Queries all input filter settings.	5-32
[:INPut]:FILTer:FREQuency?	Queries the frequency filter on/off status of all elements.	5-32
[:INPut]:FILTer:FREQuency[:ALL]	Collectively turns the frequency filter of all elements on or off.	5-33
[:INPut]:FILTer:FREQuency:ELEMent<x>	Sets or queries the frequency filter on/off status of the specified element.	5-33
[:INPut]:FILTer:LINE?	Queries the frequency filter settings of all elements.	5-33
[:INPut]:FILTer[:LINE][:ALL]	Collectively sets the line filters of all elements.	5-33
[:INPut]:FILTer[:LINE]:ELEMent<x>	Sets or queries the line filter of the specified element.	5-33
[:INPut]:INDependent	Sets or queries the on/off status of independent input element configuration.	5-33
[:INPut]:MODUle?	Queries the input element type.	5-33
[:INPut]:NULL	Sets or queries the on/off status of the NULL feature.	5-33
[:INPut]:POVer?	Queries the peak over information.	5-33
[:INPut]:SCALing?	Queries all scaling settings.	5-33
[:INPut]:SCALing:STATe?	Queries the on/off status of the scaling of all elements.	5-34
[:INPut]:SCALing[:STATe][:ALL]	Collectively turns the scaling of all elements on or off.	5-34
[:INPut]:SCALing[:STATe]:ELEMent<x>	Sets or queries the on/off status of the scaling of the specified element.	5-34
[:INPut]:SCALing:{VT|CT|SFACtor}?	Queries the scaling constant of all elements.	5-34

Command	Function	Page
[:INPut]:SCALing:{VT|CT|SFACtor}[:ALL]	Collectively sets the scaling constant of all elements.	5-34
[:INPut]:SCALing:{VT|CT|SFACtor}:ELEMent<x>	Sets or queries the scaling constant of the specified element.	5-34
[:INPut]:SYNChronize?	Queries the synchronization source of all elements.	5-34
[:INPut]:SYNChronize[:ALL]	Collectively sets the synchronization source of all elements.	5-34
[:INPut]:SYNChronize:ELEMent<x>	Sets or queries the synchronization source of the specified element.	5-34
[:INPut]:SYNChronize:SIGMA	Collectively sets the synchronization source of all elements belonging to wiring unit Σ.	5-34
[:INPut]:VOLTage?	Queries all voltage measurement settings.	5-34
[:INPut]:VOLTage:AUTO?	Queries the voltage auto range on/off status of all elements.	5-34
[:INPut]:VOLTage:AUTO[:ALL]	Collectively turns the voltage auto range of all elements on or off.	5-35
[:INPut]:VOLTage:AUTO:ELEMent<x>	Sets or queries the voltage auto range on/off status of the specified element.	5-35
[:INPut]:VOLTage:AUTO:SIGMA	Collectively turns the voltage auto range of all elements belonging to wiring unit Σ on or off.	5-35
[:INPut]:VOLTage:RANGe?	Queries the voltage range of all elements.	5-35
[:INPut]:VOLTage:RANGe[:ALL]	Collectively sets the voltage range of all elements.	5-35
[:INPut]:VOLTage:RANGe:ELEMent<x>	Sets the voltage range of the specified element.	5-35
[:INPut]:VOLTage:RANGe:SIGMA	Collectively sets the voltage range of all elements belonging to wiring unit Σ.	5-35
[:INPut]:WIRing	Sets or queries the wiring system.	5-36

INTEGrate Group

:INTEGrate?	Queries all integration settings.	5-37
:INTEGrate:ACAL	Sets or queries the on/off status of auto calibration.	5-37
:INTEGrate:MODE	Sets or queries the integration mode.	5-37
:INTEGrate:QMODe	Sets or queries the electric current integration mode.	5-37
:INTEGrate:RESet	Resets the integrated value.	5-37
:INTEGrate:RTIMe?	Queries the integration start and end times for real-time integration mode.	5-37
:INTEGrate:RTIMe:{STARt|END}	Sets or queries the integration start or end time for real-time integration mode.	5-37
:INTEGrate:STARt	Starts integration.	5-37
:INTEGrate:STATe?	Queries the integration condition.	5-37
:INTEGrate:STOP	Stops integration.	5-37
:INTEGrate:TIMer	Sets or queries the integration timer value.	5-38
:INTEGrate:WPTYpe	Sets or queries the power level integration method for each polarity (WP+ and WP−).	5-38

MEASure Group

:MEASure?	Queries all computation settings.	5-39
:MEASure:AVERaging?	Queries all averaging settings.	5-39
:MEASure:AVERaging:COUNt	Sets or queries the averaging coefficient.	5-39
:MEASure:AVERaging[:STATe]	Sets or queries the on/off status of averaging.	5-39
:MEASure:AVERaging:TYPE	Sets or queries the averaging type.	5-39
:MEASure:DMeasure?	Queries all delta computation settings.	5-39
:MEASure:DMeasure:MODE	Sets or queries the voltage or current mode that is used in delta computation.	5-39
:MEASure:DMeasure[:SIGMA]	Sets or queries the delta computation mode for wiring unit Σ.	5-40
:MEASure:EFFiciency?	Queries all efficiency computation settings.	5-40
:MEASure:EFFiciency:ETA<x>	Sets or queries the efficiency equation.	5-40
:MEASure:EFFiciency:UDEF<x>	Sets or queries the user-defined parameters used in the efficiency equation.	5-40
:MEASure:FREQuency?	Queries all frequency measurement settings.	5-40
:MEASure:FREQuency:ITEM<x>	Sets or queries the element whose frequency will be measured.	5-40
:MEASure:FUNCtion<x>?	Queries all of the settings of the specified user-defined function.	5-40
:MEASure:FUNCtion<x>:EXPRession	Sets or queries the equation of the specified user-defined function.	5-41
:MEASure:FUNCtion<x>[:STATe]	Sets or queries the on/off status of the specified user-defined function.	5-41
:MEASure:FUNCtion<x>:UNIT	Sets or queries the unit to be added to the computation result of the specified user-defined function.	5-41

Command	Function	Page
:MEASure:MHOLd	Sets or queries the on/off status of the MAX HOLD feature used in user-defined functions.	5-41
:MEASure:PHASe	Sets or queries the display format of the phase difference.	5-41
:MEASure:SFORmula	Sets or queries the equation used to compute S (apparent power).	5-41
:MEASure:SQFormula	Sets or queries the equation used to compute S (apparent power) and Q (reactive power).	5-42
:MEASure:SYNChronize	Sets or queries the synchronized measurement mode.	5-42

NUMeric Group

:NUMeric?	Queries all numeric data output settings.	5-43
:NUMeric:FORMat	Sets or queries the numeric data format.	5-43
:NUMeric:HOLD	Sets or queries whether all numeric data are held (on) or released (off).	5-43
:NUMeric:LIST?	Queries all harmonic measurement numeric list output settings.	5-44
:NUMeric:LIST:CLEar	Clears harmonic measurement numeric list output items.	5-44
:NUMeric:LIST:DELete	Deletes harmonic measurement numeric list output items.	5-44
:NUMeric:LIST:ITEM<x>	Sets or queries the function and element of the specified harmonic measurement numeric list item.	5-44
:NUMeric:LIST:NUMber	Sets or queries the number of numeric list items that are transmitted by :NUMeric:LIST:VALue?.	5-44
:NUMeric:LIST:ORDer	Sets or queries the maximum output harmonic order of the harmonic measurement numeric list data.	5-44
:NUMeric:LIST:PRESet	Presets the harmonic measurement numeric list output item pattern.	5-45
:NUMeric:LIST:SELect	Sets or queries the output component of the harmonic measurement numeric list data.	5-45
:NUMeric:LIST:VALue?	Queries the harmonic measurement numeric list data.	5-45
:NUMeric:NORMal?	Queries all numeric data output settings.	5-46
:NUMeric[:NORMal]:CLEar	Clears all numeric data output items.	5-46
:NUMeric[:NORMal]:DELete	Deletes all numeric data output items.	5-46
:NUMeric[:NORMal]:ITEM<x>	Sets or queries the function, element, and harmonic order of the specified numeric data output item.	5-46
:NUMeric[:NORMal]:NUMber	Sets or queries the number of numeric data items that are transmitted by the :NUMeric[:NORMal]:VALue? command.	5-46
:NUMeric[:NORMal]:PRESet	Presets the numeric data output item pattern.	5-46
:NUMeric[:NORMal]:VALue?	Queries the numeric data.	5-47

RATE Group

:RATE	Sets or queries the data update rate.	5-52

STATus Group

:STATus?	Queries all communication status settings.	5-53
:STATus:CONDition?	Queries the contents of the condition register.	5-53
:STATus:EESE	Sets or queries the extended event enable register.	5-53
:STATus:EESR?	Queries the content of the extended event register and clears the register.	5-53
:STATus:ERRor?	Queries the error code and message information (top of the error queue).	5-53
:STATus:FILTer<x>	Sets or queries the transition filter.	5-53
:STATus:QENable	Sets or queries whether or not messages other than errors will be stored to the error queue (on/off).	5-53
:STATus:QMESsage	Sets or queries whether or not message information will be attached to the response to the STATus:ERRor? query (on/off).	5-53
:STATus:SPOLl?	Executes serial polling.	5-54

STORe Group

:STORe?	Queries all store and recall settings.	5-55
:STORe:COUNt	Sets or queries the store count.	5-55
:STORe:FILE?	Queries all settings related to the saving of the data stored in the WT500 to files.	5-55
:STORe:FILE:ANAMing	Sets or queries whether or not files that the stored data will be saved to will be automatically named.	5-55
:STORe:FILE:CDIRectory	Changes the directory that stored numeric data is saved to.	5-55
:STORe:FILE:CONVert?	Queries all settings related to the conversion of stored numeric data files into CSV format.	5-55

Command	Function	Page
:STORe:FILE:CONVert:ABORt	Stops the conversion of a numeric data file to CSV format.	5-55
:STORe:FILE:CONVert:EXECute	Converts a stored numeric data file to CSV format.	5-55
:STORe:FILE:CONVert:MODE	Sets or queries the conversion mode used to convert stored numeric data files to CSV format.	5-55
:STORe:FILE:DRIVe	Changes the drive that stored numeric data is saved to.	5-55
:STORe:FILE:FREE?	Queries the free space (in bytes) on the drive that the stored numeric data is saved to.	5-55
:STORe:FILE:NAME	Sets or queries the name of the file to which the stored numeric data will be saved.	5-56
:STORe:FILE:PATH?	Queries the absolute path of the directory that the stored numeric data is saved to.	5-56
:STORe:INTerval	Sets or queries the storage interval.	5-56
:STORe:NUMeric?	Queries all numeric data storage settings.	5-56
:STORe:NUMeric:NORMal?	Queries all numeric data storage item settings.	5-56
:STORe:NUMeric[:NORMal]:ALL	Collectively turns on or off the output of all element functions when storing numeric data.	5-56
:STORe:NUMeric[:NORMal]:{ELEMent<x>|SIGMA}	Sets or queries the on/off status of the output of each element or of wiring unit Σ when storing numeric data.	5-56
:STORe:NUMeric[:NORMal]:PRESet<x>	Presets the output on/off pattern of the element functions that are used when storing numeric data.	5-56
:STORe:NUMeric[:NORMal]:<Function>	Sets or queries the on/off status of function output when storing numeric data.	5-56
:STORe:RESet	Resets the numeric data storage feature.	5-56
:STORe:RTIMe?	Queries the storage reservation time for real-time storage mode.	5-57
:STORe:RTIMe:{STARt|END}	Sets or queries the storage start or end time for real-time storage mode.	5-57
:STORe:SMODe	Sets or queries the storage mode.	5-57
:STORe:STARt	Begins the storing of numeric data.	5-57
:STORe:STATe?	Sets or queries the storage state.	5-57
:STORe:STOP	Stops the storing of numeric data.	5-57

SYSTem Group

:SYSTem?	Queries all system settings.	5-58
:SYSTem:CLOCk?	Queries all date/time settings.	5-58
:SYSTem:CLOCk:DISPlay	Sets or queries the on/off status of the date/time display.	5-58
:SYSTem:DATE	Sets or queries the date.	5-58
:SYSTem:ECLear	Clears error messages displayed on the screen.	5-58
:SYSTem:FONT	Sets or queries the size of the screen display font.	5-58
:SYSTem:KLOCk	Sets or queries the on/off status of the key lock.	5-58
:SYSTem:LANGuage?	Queries all display language settings.	5-58
:SYSTem:LANGuage:MENU	Sets or queries the menu language.	5-58
:SYSTem:LANGuage:MESSage	Sets or queries the message language.	5-58
:SYSTem:MODel?	Queries the model.	5-58
:SYSTem:SERial?	Queries the serial number.	5-58
:SYSTem:SLOCk	Sets or queries whether or not the SHIFT-key-on state will be sustained.	5-58
:SYSTem:SUFFix?	Queries the suffix code.	5-59
:SYSTem:TIME	Sets or queries the time.	5-59
:SYSTem:USBKeyboard	Sets or queries the USB keyboard type.	5-59

WAVeform Group

:WAVeform?	Queries all waveform display data output settings.	5-60
:WAVeform:BYTeorder	Sets or queries the output byte order of the waveform display data (FLOAT format) that is transmitted by the :WAVeform:SEND? command.	5-60
:WAVeform:END	Sets or queries the output end point of the waveform display data that is transmitted by the :WAVeform:SEND? command.	5-60
:WAVeform:FORMat	Sets or queries the format of the waveform display data that is transmitted by the :WAVeform:SEND? command.	5-60
:WAVeform:HOLD	Sets or queries whether all waveform display data will be held (on) or released (off).	5-60
:WAVeform:LENGth?	Queries the total number of points of the waveform specified by the :WAVeform:TRACe command.	5-60
:WAVeform:SEND?	Queries the waveform display data specified by the :WAVeform:TRACe command.	5-61
:WAVeform:SRATe?	Queries the sample rate of the retrieved waveform.	5-61

Command	Function	Page
:WAVeform:STARt	Sets or queries the output start point of the waveform display data that is transmitted by the :WAVeform:SEND? command.	5-61
:WAVeform:TRACe	Sets or queries the target waveform for the commands in the WAVeform group.	5-61
:WAVeform:TRIGger?	Queries the trigger position of the retrieved waveform.	5-61

Common Command Group

*CAL?	Executes zero calibration (zero-level compensation, the same operation as pressing CAL (SHIFT+SET) and queries the result.	5-62
*CLS	Clears the standard event register, extended event register, and error queue.	5-62
*ESE	Sets or queries the standard event enable register.	5-62
*ESR?	Queries the standard event register value.	5-62
*IDN?	Queries the instrument model.	5-62
*OPC	Sets bit 0 (the OPC bit) of the standard event register to 1 upon the completion of the specified overlap command.	5-62
*OPC?	Returns ASCII code 1 when the specified overlap command is completed.	5-63
*OPT?	Queries the installed options.	5-63
*RST	Initializes the settings.	5-63
*SRE	Sets or queries the service request enable register value.	5-63
*STB?	Queries the status byte register value.	5-63
*TRG	Executes single measurement (the same operation as when SINGLE (SHIFT+HOLD) is pressed).	5-63
*TST?	Performs a self-test and queries the result.	5-64
*WAI	Holds the subsequent command until the completion of the specified overlap operation.	5-64

5.2 COMMunicate Group

The commands in this group deal with communication. There are no front panel keys that correspond to the commands in this group.

:COMMunicate?

Function Queries all communication settings.
Syntax :COMMunicate?
Example :COMMUNICATE? -> :COMMUNICATE:HEADER 1;OPSE 64;OVERLAP 64;VERBOSE 1

:COMMunicate:HEADer

Function Sets or queries whether or not a header is added to the response to a query. (Example with header: ":DISPLAY:MODE NUMERIC." Example without header: "NUMERIC.")
Syntax :COMMunicate:HEADer {<Boolean>}
:COMMunicate:HEADer?
Example :COMMUNICATE:HEADER ON
:COMMUNICATE:HEADER? -> :COMMUNICATE:HEADER 1

:COMMunicate:LOCKout

Function Sets or clears local lockout.
Syntax :COMMunicate:LOCKout {<Boolean>}
:COMMunicate:LOCKout?
Example :COMMUNICATE:LOCKOUT ON
:COMMUNICATE:LOCKOUT?
-> :COMMUNICATE:LOCKOUT 1

:COMMunicate:OPSE (Operation Pending Status Enable register)

Function Sets or queries the overlap command that is used by the *OPC, *OPC?, and *WAI commands.
Syntax :COMMunicate:OPSE <Register>
:COMMunicate:OPSE?
<Register> = 0 to 65535; for more information, see the figure under :COMMunicate:OPSR?
Example :COMMUNICATE:OPSE 65535
:COMMUNICATE:OPSE? -> :COMMUNICATE:OPSE 64
Description In the above example, all bits are set to 1 to make all overlap commands applicable. However, bits fixed to 0 are not set to 1, so the response to the query only indicates 1 for bit 6.

:COMMunicate:OPSR? (Operation Pending Status Register)

Function Queries the operation pending status register.
Syntax :COMMunicate:OPSR?
Example :COMMUNICATE:OPSR? -> 0

Operation pending status register

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	ACS	0	0	0	0	0	0

When bit 6 (ACS) = 1: Media access is incomplete.

:COMMunicate:OVERlap

Function Sets or queries the commands that operate as overlap commands.
Syntax :COMMunicate:OVERlap <Register>
:COMMunicate:OVERlap?
<Register> = 0 to 65535; for more information, see the figure under :COMMunicate:OPSR?
Example :COMMUNICATE:OVERLAP 65535
:COMMUNICATE:OVERLAP?
-> :COMMUNICATE:OVERLAP 64
Description
- In the above example, all bits are set to 1 to make all overlap commands applicable. However, bits fixed to 0 are not set to 1, so the response to the query only indicates 1 for bit 6.
- For information about how to synchronize a program using COMMunicate:OVERlap, see page 4-8.

:COMMunicate:REMote

Function Sets remote or local. On is remote mode.
Syntax :COMMunicate:REMote {<Boolean>}
:COMMunicate:REMote?
Example :COMMUNICATE:REMOTE ON
:COMMUNICATE:REMOTE? -> :COMMUNICATE:REMOTE 1

:COMMunicate:VERBose

Function Sets or queries whether the response to a query is returned fully spelled out (example: ":INPUT: VOLTAGE:RANGE:ELEMENT1 1.000E+03") or using abbreviation (example: ":VOLT:RANG: ELEM1 1.000E+03").

Syntax
```
:COMMunicate:VERBose {<Boolean>}
:COMMunicate:VERBose?
```

Example
```
:COMMUNICATE:VERBOSE ON
:COMMUNICATE:VERBOSE?
-> :COMMUNICATE:VERBOSE 1
```

:COMMunicate:WAIT

Function Waits for a specified extended event to occur.

Syntax
```
:COMMunicate:WAIT <Register>
```
<Register> = 0 to 65535 (Extended event register. For more information, see page 6-7.)

Example
```
:COMMUNICATE:WAIT 1
```

Description For information about how to synchronize a program using COMMunicate:WAIT, see page 4-8.

:COMMunicate:WAIT?

Function Creates the response that is returned when a specified extended event occurs.

Syntax
```
:COMMunicate:WAIT? <Register>
```
<Register> = 0 to 65535 (Extended event register. For more information, see page 6-7.)

Example
```
:COMMUNICATE:WAIT? 65535 -> 1
```

5.3 CURSor Group

The commands in this group deal with cursor measurements. You can make the same settings and queries that you can make by using CURSOR (SHIFT + FORM) on the front panel.

:CURSor?

Function Queries all cursor measurement settings.
Syntax :CURSor?
Example :CURSOR? -> :CURSOR:···

:CURSor:BAR?

Function Queries all bar graph display cursor measurement settings.
Syntax :CURSor:BAR?
Example :CURSOR:BAR? -> :CURSOR:BAR:···
Description This command is only valid on models with the optional harmonic measurement feature.

:CURSor:BAR:LINKage

Function Sets or queries the on/off status of the cursor position linkage in the bar graph display.
Syntax :CURSor:BAR:LINKage {<Boolean>}
:CURSor:BAR:LINKage?
Example :CURSOR:BAR:LINKAGE ON
:CURSOR:BAR:LINKAGE? -> :CURSOR:BAR:LINKAGE 1
Description This command is only valid on models with the optional harmonic measurement feature.

:CURSor:BAR:POSition<x>

Function Sets or queries the position (harmonic order) of the specified cursor in the bar graph display.
Syntax :CURSor:BAR:POSition<x> {<NRf>}
:CURSor:BAR:POSition<x>?
<x> = 1 or 2 (1 = C1 +, 2 = C2 x)
<NRf> = 0 to 50
Example :CURSOR:BAR:POSITION1 1
:CURSOR:BAR:POSITION1? -> :CURSOR:BAR:POSITION1 1
Description This command is only valid on models with the optional harmonic measurement feature.

:CURSor:BAR[:STATe]

Function Sets or queries the on/off status of the cursor display in the bar graph display.
Syntax :CURSor:BAR[:STATe] {<Boolean>}
:CURSor:BAR:STATe?
Example :CURSOR:BAR:STATE ON
:CURSOR:BAR:STATE? -> :CURSOR:BAR:STATE 1
Description This command is only valid on models with the optional harmonic measurement feature.

:CURSor:BAR:{Y<x>|DY}?

Function Queries the measured value of the specified cursor in the bar graph display.
Syntax :CURSor:BAR:{Y<x>|DY}?
Y<x> = Y-axis value of the cursor position (Y1 = Y1+, Y2+, Y3+ Y2 = Y1x, Y2x, Y3x)
DY = Y-axis value between cursors (ΔY1, ΔY2, Δ Y3)
<x> = 1 or 2 (1 = C1 +, 2 = C2 x)
Example :CURSOR:BAR:Y1? -> 78.628E+00
Description
- This command is only valid on models with the optional harmonic measurement feature.
- When multiple bar graphs are displayed, the cursor measured values of each bar graph are returned in order.
- If the bar graph cursor display is not turned on, NAN (Not A Number) is returned.

:CURSor:TRENd?

Function Queries all trend display cursor measurement settings.
Syntax :CURSor:TRENd?
Example :CURSOR:TREND? -> :CURSOR:TREND:···

:CURSor:TRENd:LINKage

Function Sets or queries the on/off status of the cursor position linkage in the trend display.
Syntax :CURSor:TRENd:LINKage {<Boolean>}
:CURSor:TRENd:LINKage?
Example :CURSOR:TREND:LINKAGE ON
:CURSOR:TREND:LINKAGE? -> :CURSOR:TREND:LINKAGE 1

:CURSor:TRENd:POSition<x>

Function Sets or queries the position of the specified cursor in the trend display.
Syntax :CURSor:TRENd:POSition<x> {<NRf>}
:CURSor:TRENd:POSition<x>?
<x> = 1 or 2 (1 = C1 +, 2 = C2 x)
<NRf> = 0 to 1001
Example :CURSOR:TREND:POSITION1 100
:CURSOR:TREND:POSITION1? -> :CURSOR:TREND:POSITION1 100

:CURSor:TRENd[:STATe]

Function Sets or queries the on/off status of the cursor display in the trend display.

Syntax
```
:CURSor:TRENd[:STATe] {<Boolean>}
:CURSor:TRENd:STATe?
```

Example
```
:CURSOR:TREND:STATE ON
:CURSOR:TREND:STATE? -> :CURSOR:TREND:
STATE 1
```

:CURSor:TRENd:TRACe<x>

Function Sets or queries the target of the specified cursor in the trend display.

Syntax
```
:CURSor:TRENd:TRACe<x> {<NRf>}
:CURSor:TRENd:TRACe<x>?
```
<x> = 1 or 2 (1 = C1 +, 2 = C2 x)
<NRf> = 1 to 8 (T1 to T8)

Example
```
:CURSOR:TREND:TRACE1 1
:CURSOR:TREND:TRACE1? -> :CURSOR:
TREND:TRACE1 1
```

:CURSor:TRENd:{X<x>|Y<x>|DY}?

Function Queries the measured value of the specified cursor in the trend display.

Syntax
```
:CURSor:TRENd:{X<x>|Y<x>|DY}?
```
X<x> = Trend time string of the cursor position (X1 = D+, X2 = Dx)
Y<x> = Y-axis value of the cursor position (Y1 = Y+, Y2 = Yx)
DY = Y-axis value between cursors (ΔY)
<x> = 1 or 2 (1 = C1 +, 2 = C2 x)

Example
```
:CURSOR:TREND:X1?
-> "2008/01/01 12:34:56"
:CURSOR:TREND:Y1? -> 78.628E+00
```

Description If the trend cursor display is not turned on, the following results will be returned.
For X<x>: "****/**/** **:**:**" will be returned.
For Y<x> and DY: NAN (Not A Number) will be returned.

:CURSor:WAVE?

Function Queries all waveform display cursor measurement settings.

Syntax
```
:CURSor:WAVE?
```

Example
```
:CURSOR:WAVE? -> :CURSOR:WAVE:···
```

:CURSor:WAVE:LINKage

Function Sets or queries the on/off status of the cursor position linkage in the waveform display.

Syntax
```
:CURSor:WAVE:LINKage {<Boolean>}
:CURSor:WAVE:LINKage?
```

Example
```
:CURSOR:WAVE:LINKAGE ON
:CURSOR:WAVE:LINKAGE? -> :CURSOR:WAVE:
LINKAGE 1
```

:CURSor:WAVE:PATH

Function Sets or queries the cursor path in the waveform display.

Syntax
```
:CURSor:WAVE:PATH {MAX|MIN|MID}
:CURSor:WAVE:PATH?
```

Example
```
:CURSOR:WAVE:PATH MAX
:CURSOR:WAVE:PATH? -> :CURSOR:WAVE:
PATH MAX
```

:CURSor:WAVE:POSition<x>

Function Sets or queries the position of the specified cursor in the waveform display.

Syntax
```
:CURSor:WAVE:POSition<x> {<NRf>}
:CURSor:WAVE:POSition<x>?
```
<x> = 1 or 2 (1 = C1 +, 2 = C2 x)
<NRf> = 0 to 500

Example
```
:CURSOR:WAVE:POSITION1 100
:CURSOR:WAVE:POSITION1? -> :CURSOR:
WAVE:POSITION1 100
```

:CURSor:WAVE[:STATe]

Function Sets or queries the on/off status of the cursor display in the waveform display.

Syntax
```
:CURSor:WAVE[:STATe] {<Boolean>}
:CURSor:WAVE:STATe?
```

Example
```
:CURSOR:WAVE:STATE ON
:CURSOR:WAVE:STATE? -> :CURSOR:WAVE:
STATE 1
```

:CURSor:WAVE:TRACe<x>

Function Sets or queries the target of the specified cursor in the waveform display.

Syntax
```
:CURSor:WAVE:TRACe<x> {U<x>|I<x>}
:CURSor:WAVE:TRACe<x>?
```
The <x> in TRACe<x> = 1 or 2 (1 = C1 +, 2 = C2 x)
The <x> in U<x> and I<x> = 1 to 3

Example
```
:CURSOR:WAVE:TRACE1 U1
:CURSOR:WAVE:TRACE1? -> :CURSOR:WAVE:
TRACE1 U1
```

:CURSor:WAVE:{X<x>|DX|PERDt|Y<x>|DY}?

Function Queries the measured value of the specified cursor in the waveform display.

Syntax
```
:CURSor:WAVE:{X<x>|DX|PERDt|Y<x>|DY}?
```
X<x> = X-axis value of the cursor position (X1 = X+, X2 = Xx)
DX = X-axis value between cursors (ΔX)
PERDt = 1/DT (1/ΔX) value between cursors
Y<x> = Y-axis value of the cursor position (Y1 = Y+, Y2 = Yx)
DY = Y-axis value between cursors (ΔY)
<x> = 1 or 2 (1 = C1 +, 2 = C2 x)

Example
```
:CURSOR:WAVE:Y1? -> 78.628E+00
```

Description If the waveform cursor display is not turned on, NAN (Not A Number) is returned.

5.4 DISPlay Group

The commands in this group deal with the screen display.
You can make the same settings and queries that you can make by pressing INPUT INFO (SHIFT + SETUP) and the keys in the DISPLAY area.

:DISPlay?

Function Queries all screen display settings.
Syntax `:DISPlay?`
Example `:DISPLAY? -> :DISPLAY:···`
Description Returns all settings corresponding to the current display mode (`:DISPlay:MODE`).

:DISPlay:BAR?

Function Queries all bar graph settings.
Syntax `:DISPlay:BAR?`
Example `:DISPLAY:BAR? -> :DISPLAY:BAR:··`
Description This command is only valid on models with the optional harmonic measurement feature.

:DISPlay:BAR:FORMat

Function Sets or queries the bar graph display format.
Syntax `:DISPlay:BAR:FORMat {SINGle|DUAL|TRIad}`
`:DISPlay:BAR:FORMat?`
Example `:DISPLAY:BAR:FORMAT SINGLE`
`:DISPLAY:BAR:FORMAT? -> :DISPLAY:BAR:FORMAT SINGLE`
Description This command is only valid on models with the optional harmonic measurement feature.

:DISPlay:BAR:ITEM<x>

Function Sets or queries the function and element of the specified bar graph item.
Syntax `:DISPlay:BAR:ITEM<x> {<Function>,<Element>}`
`:DISPlay:BAR:ITEM<x>?`
<x> = 1 to 3 (item number)
<Function> = {U|I|P|S|Q|LAMBda|...}
<Element> = 1 to 3
Example `:DISPLAY:BAR:ITEM1 U,1`
`:DISPLAY:BAR:ITEM1? -> :DISPLAY:BAR:ITEM1 U,1`
Description
- This command is only valid on models with the optional harmonic measurement feature.
- For information about the options available for <Function>, see Function Option List 2 on page 5-22.

:DISPlay:BAR:ORDer

Function Sets or queries the bar graph starting and ending harmonic orders.
Syntax `:DISPlay:BAR:ORDer {<NRf>,<NRf>}`
`:DISPlay:BAR:ORDer?`
The first <NRf> = 0 to 40 (starting harmonic order)
The second <NRf> = 10 to 50 (ending harmonic order)
Example `:DISPLAY:BAR:ORDER 1,50`
`:DISPLAY:BAR:ORDER? -> :DISPLAY:BAR:ORDER 1,50`
Description
- This command is only valid on models with the optional harmonic measurement feature.
- Set the starting harmonic order and then the ending harmonic order.
- Set the ending harmonic order to a value greater than or equal to that of the starting harmonic order + 10.

:DISPlay:INFOrmation?

Function Queries all setup parameter list display settings.
Syntax `:DISPlay:INFOrmation?`
Example `:DISPLAY:INFORMATION? -> :DISPLAY:INFORMATION:···`

:DISPlay:INFOrmation[:STATe]

Function Sets or queries the on/off status of the setup parameter list.
Syntax `:DISPlay:INFOrmation[:STATe] {<Boolean>}`
`:DISPlay:INFOrmation:STATe?`
Example `:DISPLAY:INFORMATION:STATE ON`
`:DISPLAY:INFORMATION:STATE?`
`-> :DISPLAY:INFORMATION:STATE 1`

:DISPlay:MODE

Function Sets or queries the display mode.

Syntax `:DISPlay:MODE {NUMeric|WAVE|TRENd|BAR|VECTor}`
`:DISPlay:MODE?`
NUMeric = Only displays numeric values.
WAVE = Only displays waveforms.
TRENd = Trend display.
BAR = Bar graph display.
VECTor = Vector display

Example `:DISPLAY:MODE NUMERIC`
`:DISPLAY:MODE? -> :DISPLAY:MODE NUMERIC`

Description BAR and VECTor can only be selected on models with the optional harmonic measurement feature.

:DISPlay:NUMeric?

Function Queries all numeric display settings.

Syntax `:DISPlay:NUMeric?`

Example `:DISPLAY:NUMERIC? -> :DISPLAY:NUMERIC:···`

:DISPlay:NUMeric:NORMal?

Function Queries all numeric display settings.

Syntax `:DISPlay:NUMeric:NORMal?`

Example `:DISPLAY:NUMERIC:NORMAL?`
`-> :DISPLAY:NUMERIC:NORMAL:···`

Description Returns all settings corresponding to the current numeric display mode (`:DISPlay:NUMeric[:NORMal]:FORMat`).

:DISPlay:NUMeric[:NORMal]:ALL?

Function Queries all numeric display (in All display mode) settings.

Syntax `:DISPlay:NUMeric[:NORMal]:ALL?`

Example `:DISPLAY:NUMERIC:NORMAL:ALL?`
`-> :DISPLAY:NUMERIC:NORMAL:ALL:···`

:DISPlay:NUMeric[:NORMal]:ALL:CURSor

Function Sets or queries the cursor position in the numeric display (in All display mode).

Syntax `:DISPlay:NUMeric[:NORMal]:ALL:CURSor {<Function>}`
`:DISPlay:NUMeric[:NORMal]:ALL:CURSor?`
<Function> = {URMS|IRMS|P|S|Q|...}

Example `:DISPLAY:NUMERIC:NORMAL:ALL:CURSOR P`
`:DISPLAY:NUMERIC:NORMAL:ALL:CURSOR?`
`-> :DISPLAY:NUMERIC:NORMAL:ALL:CURSOR P`

Description
- Specify the cursor position in terms of the function name.
- For information about the options available for <Function>, see Function Option List 1 on page 5-22.

:DISPlay:NUMeric[:NORMal]:ALL:ORDer

Function Sets or queries the displayed harmonic order on the harmonic measurement function display page of the numeric display (in All display mode).

Syntax `:DISPlay:NUMeric[:NORMal]:ALL:ORDer {<Order>}`
`:DISPlay:NUMeric[:NORMal]:ALL:ORDer?`
<Order> = {TOTal|DC|<NRf>}(<NRf> = 1 to 50)

Example `:DISPLAY:NUMERIC:NORMAL:ALL:ORDER 1`
`:DISPLAY:NUMERIC:NORMAL:ALL:ORDER?`
`-> :DISPLAY:NUMERIC:NORMAL:ALL:ORDER 1`

Description
- This command is only valid on models with the optional harmonic measurement feature.
- This command is valid when the displayed page number (`:DISPlay:NUMeric[:NORMal]:ALL:PAGE`) on the numeric display (in All display mode) is 6.

:DISPlay:NUMeric[:NORMal]:ALL:PAGE

Function Sets or queries the page number in the numeric display (in All display mode).

Syntax `:DISPlay:NUMeric[:NORMal]:ALL:PAGE {<NRf>}`
`:DISPlay:NUMeric[:NORMal]:ALL:PAGE?`
<NRf> = 1 to 5 (page number)
<NRf> = 1 to 7 (when the optional harmonic measurement feature is installed)

Example `:DISPLAY:NUMERIC:NORMAL:ALL:PAGE 1`
`:DISPLAY:NUMERIC:NORMAL:ALL:PAGE?`
`-> :DISPLAY:NUMERIC:NORMAL:ALL:PAGE 1`

Description When the page number is set, the cursor position moves to the beginning of the page.

:DISPlay:NUMeric[:NORMal]:FORMat

Function Sets or queries the numeric display format.

Syntax
```
:DISPlay:NUMeric[:NORMal]:
FORMat {VAL4|VAL8|VAL16|MATRix|ALL|
SINGle|DUAL}
:DISPlay:NUMeric[:NORMal]:FORMat?
```

Example
```
:DISPLAY:NUMERIC:NORMAL:FORMAT VAL4
:DISPLAY:NUMERIC:NORMAL:FORMAT?
-> :DISPLAY:NUMERIC:NORMAL:FORMAT VAL4
```

Description
- The numeric data is displayed in the following ways for each format:
  {VAL4|VAL8|VAL16} = Numeric display items are displayed in order by their item numbers. (The numbers in these options indicate the number of items that are displayed on a single screen/page.)
  MATRix = Selected functions are displayed in order by element.
  ALL = All functions are displayed in order by element.
  SINGle = One list display item is listed by separating the data into even and odd harmonic orders.
  DUAL = Two list display items are listed in order by harmonic order.
- SINGle and DUAL can only be selected on models with the optional harmonic measurement feature.

:DISPlay:NUMeric[:NORMal]:LIST?

Function Queries all numeric display settings (in the list display modes).

Syntax
```
:DISPlay:NUMeric[:NORMal]:LIST?
```

Example
```
:DISPLAY:NUMERIC:NORMAL:LIST?
-> :DISPLAY:NUMERIC:NORMAL:LIST:···
```

Description This command is only valid on models with the optional harmonic measurement feature.

:DISPlay:NUMeric[:NORMal]:LIST:CURSor

Function Sets or queries the cursor position in the numeric display (in the list display modes).

Syntax
```
:DISPlay:NUMeric[:NORMal]:LIST:
CURSor {HEADer|ORDer}
:DISPlay:NUMeric[:NORMal]:LIST:CURSor?
```
HEADer = The cursor moves to the header section (data concerning all the harmonics, left side of the screen).
ORDer = The cursor moves to the data section (Numeric data of each harmonic, right side of the screen).

Example
```
:DISPLAY:NUMERIC:NORMAL:LIST:
CURSOR ORDER
:DISPLAY:NUMERIC:NORMAL:LIST:CURSOR?
-> :DISPLAY:NUMERIC:NORMAL:LIST:
CURSOR ORDER
```

Description This command is only valid on models with the optional harmonic measurement feature.

:DISPlay:NUMeric[:NORMal]:LIST:HEADer

Function Sets or queries the cursor position of the header section in the numeric display (in the list display modes).

Syntax
```
:DISPlay:NUMeric[:NORMal]:LIST:
HEADer {<NRf>}
:DISPlay:NUMeric[:NORMal]:LIST:HEADer?
```
<NRf> = 1 to 49

Example
```
:DISPLAY:NUMERIC:NORMAL:LIST:HEADER 1
:DISPLAY:NUMERIC:NORMAL:LIST:HEADER?
-> :DISPLAY:NUMERIC:NORMAL:LIST:
HEADER 1
```

Description
- This command is only valid on models with the optional harmonic measurement feature.
- This command is valid when the cursor position (`:DISPlay:NUMeric[:NORMal]:LIST:CURSor`) on the numeric display is set to HEADer (in the list display modes).

:DISPlay:NUMeric[:NORMal]:LIST:ITEM<x>

Function Sets or queries the function and element of the displayed items in the numeric display (in the list display modes).

Syntax
```
:DISPlay:NUMeric[:NORMal]:LIST:
ITEM<x> {<Function>,<Element>}
:DISPlay:NUMeric[:NORMal]:LIST:
ITEM<x>?
```
<x> = 1 to 2 (item number)
<Function> = {U|I|P|S|Q|LAMBda|PHI|PHIU|PHII}
<Element> = {<NRf>|SIGMa}(<NRf> = 1 to 3)

Example
```
:DISPLAY:NUMERIC:NORMAL:LIST:
ITEM1 U,1
:DISPLAY:NUMERIC:NORMAL:LIST:ITEM1?
-> :DISPLAY:NUMERIC:NORMAL:LIST:
ITEM1 U,1
```

Description
- This command is only valid on models with the optional harmonic measurement feature.
- For information about the options available for <Function>, see Function Option List 2 on page 5-23.

:DISPlay:NUMeric[:NORMal]:LIST:ORDer

Function Sets or queries the harmonic order cursor position of the data section in the numeric display (in the list display modes).

Syntax
```
:DISPlay:NUMeric[:NORMal]:LIST:
ORDer {<NRf>}
:DISPlay:NUMeric[:NORMal]:LIST:ORDer?
```
<NRf> = 1 to 50 (harmonic order)

Example
```
:DISPLAY:NUMERIC:NORMAL:LIST:ORDER 1
:DISPLAY:NUMERIC:NORMAL:LIST:ORDER?
-> :DISPLAY:NUMERIC:NORMAL:LIST:
ORDER 1
```

Description
- This command is only valid on models with the optional harmonic measurement feature.
- This command is valid when the cursor position (`:DISPlay:NUMeric[:NORMal]:LIST:CURSor`) on the numeric display is set to ORDer (in the list display modes).

:DISPlay:NUMeric[:NORMal]:{VAL4|VAL8|VAL16|MATRix}?

Function Queries all settings related to the numeric display (for the 4-value, 8-value, 16-value, or matrix display).

Syntax
```
:DISPlay:NUMeric[:NORMal]:{VAL4|VAL8|V
AL16|MATRix}?
```

Example
```
:DISPLAY:NUMERIC:NORMAL:VAL4?
-> :DISPLAY:NUMERIC:NORMAL:VAL4:···
```

:DISPlay:NUMeric[:NORMal]:{VAL4|VAL8|VAL16|MATRix}:CURSor

Function Sets or queries the cursor position in the numeric display (for the 4-value, 8-value, 16-value, or matrix display).

Syntax
```
:DISPlay:NUMeric[:NORMal]:{VAL4|VAL8|
VAL16|MATRix}:CURSor {<NRf>}
:DISPlay:NUMeric[:NORMal]:{VAL4|VAL8|
VAL16|MATRix}:CURSor?
```
<NRf> = 1 to 36 (item numbers in the VAL4 display)
<NRf> = 1 to 72 (item numbers in the VAL8 display)
<NRf> = 1 to 144 (item numbers in the VAL16 display)
<NRf> = 1 to 81 (item numbers in the MATRix display)

Example
```
:DISPLAY:NUMERIC:NORMAL:VAL4:CURSOR 1
:DISPLAY:NUMERIC:NORMAL:VAL4:CURSOR?
-> :DISPLAY:NUMERIC:NORMAL:VAL4:
CURSOR 1
```

Description Specify the cursor position with an item number.

:DISPlay:NUMeric[:NORMal]:{VAL4|VAL8|VAL16|MATRix}:ITEM<x>

Function Sets or queries the function, element, and harmonic order of the specified numeric display item (for the 4-value, 8-value, 16-value, or matrix display).

Syntax
```
:DISPlay:NUMeric:[:NORMal]:{VAL4|
VAL8|VAL16|MATRix}:ITEM<x> {NONE|
<Function>,<Element>[,<Order>]}
:DISPlay:NUMeric:[:NORMal]:{VAL4|
VAL8|VAL16|MATRix}:ITEM<x>?
```
<x> = 1 to 36 (item numbers in the VAL4 display)
<x> = 1 to 72 (item numbers in the VAL8 display)
<x> = 1 to 144 (item numbers in the VAL16 display)
<x> = 1 to 81 (item numbers in the MATRix display)
NONE = No display item
<Function> = {URMS|IRMS|P|S|Q|...}
<Element> = {<NRf>|SIGMa} (<NRf> = 1 to 3)
<Order> = {TOTal|DC|<NRf>} (<NRf> = 1 to 50)

Example
```
:DISPLAY:NUMERIC:NORMAL:VAL4:
ITEM1 URMS,1
:DISPLAY:NUMERIC:NORMAL:VAL4:ITEM1?
-> :DISPLAY:NUMERIC:NORMAL:VAL4:
ITEM1 URMS,1
:DISPLAY:NUMERIC:NORMAL:VAL4:
ITEM1 UK,1,1
:DISPLAY:NUMERIC:NORMAL:VAL4:ITEM1?
-> :DISPLAY:NUMERIC:NORMAL:VAL4:
ITEM1 UK,1,1
```

Description
- For information about the options available for <Function>, see Function Option List 1 on page 5-22.
- If <Element> is omitted, the element is set to 1.
- If <Order> is omitted, the order is set to TOTal.
- <Element> and <Order> are omitted from responses to functions that do not need them.
- The <Element> setting does not affect `:DISPlay:NUMeric:[:NORMal]:MATRix:ITEM<x>`.

:DISPlay:NUMeric[:NORMal]:{VAL4|VAL8|VAL16|MATRix}:PAGE

Function Sets or queries the page number in the numeric display (for the 4-value, 8-value, 16-value, or matrix display).

Syntax
```
:DISPlay:NUMeric[:NORMal]:{VAL4|VAL8|VAL16|MATRix}:PAGE {<NRf>}
:DISPlay:NUMeric[:NORMal]:{VAL4|VAL8|VAL16|MATRix}:PAGE?
```
<NRf> = 1 to 9 (page number)

Example
```
:DISPLAY:NUMERIC:NORMAL:VAL4:PAGE 1
:DISPLAY:NUMERIC:NORMAL:VAL4:PAGE?
-> :DISPLAY:NUMERIC:NORMAL:VAL4:PAGE 1
```

Description When the page number is set, the cursor position moves to the beginning of the page.

:DISPlay:NUMeric[:NORMal]:{VAL4|VAL8|VAL16|MATRix}:PRESet

Function Presets the display order pattern of displayed items in the numeric display (for the 4-value, 8-value, 16-value, or matrix display).

Syntax
```
:DISPlay:NUMeric[:NORMal]:{VAL4|VAL8|VAL16|MATRix}:PRESet {<NRf>}
```
<NRf> = 1 to 4

Example
```
:DISPLAY:NUMERIC:NORMAL:VAL4:PRESET 1
```

Description No matter which of the values from 1 to 4 <NRf> is set to, the numeric display item display pattern (order) will be the same as the order when the WT500 is reset using the ITEM setting menu that is displayed on the WT500 screen (Reset Items Exec). For information about the display order after the WT500 is reset, see the *User's Manual IM760201-01E*.

:DISPlay:TRENd?

Function Queries all trend settings.

Syntax
```
:DISPlay:TRENd?
```

Example
```
:DISPLAY:TREND? -> :DISPLAY:TREND:···
```

:DISPlay:TRENd:ALL

Function Collectively turns all trends on or off.

Syntax
```
:DISPlay:TRENd:ALL {<Boolean>}
```

Example
```
:DISPLAY:TREND:ALL ON
```

:DISPlay:TRENd:CLEar

Function Clears all trends.

Syntax
```
:DISPlay:TRENd:CLEar
```

Example
```
:DISPLAY:TREND:CLEAR
```

:DISPlay:TRENd:FORMat

Function Sets or queries the display format of all trends.

Syntax
```
:DISPlay:TRENd:FORMat {SINGle|DUAL|TRIad|QUAD}
:DISPlay:TRENd:FORMat?
```

Example
```
:DISPLAY:TREND:FORMAT SINGLE
:DISPLAY:TREND:FORMAT? -> :DISPLAY:TREND:FORMAT SINGLE
```

:DISPlay:TRENd:ITEM<x>?

Function Queries all settings for the specified trend.

Syntax
```
:DISPlay:TRENd:ITEM<x>?
```
<x> = 1 to 8 (item number)

Example
```
:DISPLAY:TREND:ITEM1?
-> :DISPLAY:TREND:ITEM1:···
```

:DISPlay:TRENd:ITEM<x>[:FUNCtion]

Function Sets or queries the function, element, and harmonic order of the specified trend item.

Syntax
```
:DISPlay:TRENd:ITEM<x>[:FUNCtion] {NONE|<Function>,<Element>[,<Order>]}
:DISPlay:TRENd:ITEM<x>:FUNCtion?
```
<x> = 1 to 8 (item number)
<Function> = {URMS|IRMS|P|S|Q|...}
<Element> = {<NRf>|SIGMa} (<NRf> = 1 to 3)
<Order> = {TOTal|DC|<NRf>} (<NRf> = 1 to 50)

Example
```
:DISPLAY:TREND:ITEM1:FUNCTION URMS,1
:DISPLAY:TREND:ITEM1:FUNCTION?
-> :DISPLAY:TREND:ITEM1:FUNCTION URMS,1
:DISPLAY:TREND:ITEM1:FUNCTION UK,1,1
:DISPLAY:TREND:ITEM1:FUNCTION?
-> :DISPLAY:TREND:ITEM1:FUNCTION UK,1,1
```

Description
- For information about the options available for <Function>, see Function Option List 1 on page 5-22.
- If <Element> is omitted, the element is set to 1.
- If <Order> is omitted, the order is set to TOTal.
- <Element> and <Order> are omitted from responses to functions that do not need them.

:DISPlay:TRENd:ITEM<x>:SCALing?

Function Queries all scaling settings for the specified trend.

Syntax
```
:DISPlay:TRENd:ITEM<x>:SCALing?
```
<x> = 1 to 8 (item number)

Example
```
:DISPLAY:TREND:ITEM1:SCALING?
-> :DISPLAY:TREND:ITEM1:SCALING:···
```

:DISPlay:TRENd:ITEM<x>:SCALing:MODE

Function Sets or queries the scaling mode of the specified trend.

Syntax
```
:DISPlay:TRENd:ITEM<x>:SCALing:
MODE {AUTO|MANual}
:DISPlay:TRENd:ITEM<x>:SCALing:MODE?
```
<x> = 1 to 8 (item number)

Example
```
:DISPLAY:TREND:ITEM1:SCALING:MODE AUTO
:DISPLAY:TREND:ITEM1:SCALING:MODE?
-> :DISPLAY:TREND:ITEM1:SCALING:
MODE AUTO
```

:DISPlay:TRENd:ITEM<x>:SCALing:VALue

Function Sets or queries the upper and lower limits of the manual scaling of the specified trend.

Syntax
```
:DISPlay:TRENd:ITEM<x>:SCALing:
VALue {<NRf>,<NRf>}
:DISPlay:TRENd:ITEM<x>:SCALing:VALue?
```
<x> = 1 to 8 (item number)
<NRf> = −9.999E+30 to 9.999E+30

Example
```
:DISPLAY:TREND:ITEM1:SCALING:
VALUE 100,-100
:DISPLAY:TREND:ITEM1:SCALING:VALUE?
-> :DISPLAY:TREND:ITEM1:SCALING:
VALUE 100.0E+00,-100.0E+00
```

Description
- Set the upper limit and then the lower limit.
- This command is valid when the scaling mode of the trend (`:DISPlay:TRENd:ITEM<x>:SCALing:MODE`) is set to MANual.

:DISPlay:TRENd:TDIV

Function Sets or queries the trend horizontal axis (T/div).

Syntax
```
:DISPlay:TRENd:
TDIV {<NRf>,<NRf>,<NRf>}
:DISPlay:TRENd:TDIV?
```
{<NRf>, <NRf>, <NRf>} = 0, 0, 3 to 24, 0, 0
The first <NRf> = 1, 3, 6, 12, or 24 (hour)
The second <NRf> = 1, 3, 6, 10, or 30 (minute)
The third <NRf> = 3, 6, 10, or 30 (second)

Example
```
:DISPLAY:TREND:TDIV 0,0,3
:DISPLAY:TREND:TDIV? -> :DISPLAY:
TREND:TDIV 0,0,3
```

Description Set the three <NRf>'s so that one <NRf> is a non-zero value and the other two are zeroes.

:DISPlay:TRENd:T<x>

Function Sets or queries the on/off status of the specified trend.

Syntax
```
:DISPlay:TRENd:T<x> {<Boolean>}
:DISPlay:TRENd:T<x>?
```
<x> = 1 to 8 (item number)

Example
```
:DISPLAY:TREND:T1 ON
:DISPLAY:TREND:T1? -> :DISPLAY:TREND:
T1 1
```

:DISPlay:VECTor?

Function Queries all vector display settings.

Syntax
```
:DISPlay:VECTor?
```

Example
```
:DISPLAY:VECTOR? -> :DISPLAY:
VECTOR:···
```

Description This command is only valid on models with the optional harmonic measurement feature.

:DISPlay:VECTor:NUMeric

Function Sets or queries the on/off status of the numeric data display in the vector display.

Syntax
```
:DISPlay:VECTor:NUMeric {<Boolean>}
:DISPlay:VECTor:NUMeric?
```

Example
```
:DISPLAY:VECTOR:NUMERIC ON
:DISPLAY:VECTOR:NUMERIC? -> :DISPLAY:
VECTOR:NUMERIC 1
```

Description This command is only valid on models with the optional harmonic measurement feature.

:DISPlay:VECTor:OBJect

Function Sets or queries the wiring unit that is displayed in the vector display.

Syntax
```
:DISPlay:VECTor:OBJect {SIGMA}
:DISPlay:VECTor:OBJect?
```

Example
```
:DISPLAY:VECTOR:OBJECT SIGMA
:DISPLAY:VECTOR:OBJECT? -> :DISPLAY:
VECTOR:OBJECT SIGMA
```

Description This command is only valid on models with the optional harmonic measurement feature.

:DISPlay:VECTor:{UMAG|IMAG}

Function Sets or queries the voltage or current zoom factor for the vector display.

Syntax
```
:DISPlay:VECTor:{UMAG|IMAG} {<NRf>}
:DISPlay:VECTor:{UMAG|IMAG}?
```
<NRf> = 0.100 to 100.000

Example
```
:DISPLAY:VECTOR:UMAG 1
:DISPLAY:VECTOR:UMAG? -> :DISPLAY:
VECTOR:UMAG 1.000
```

Description This command is only valid on models with the optional harmonic measurement feature.

:DISPlay:WAVE?

Function Queries all waveform display settings.

Syntax
```
:DISPlay:WAVE?
```

Example
```
:DISPLAY:WAVE? -> :DISPLAY:WAVE:···
```

:DISPlay:WAVE:ALL

Function Collectively turns all waveform displays on or off.
Syntax :DISPlay:WAVE:ALL {<Boolean>}
Example :DISPLAY:WAVE:ALL ON

:DISPlay:WAVE:FORMat

Function Sets or queries the display format of all waveforms.
Syntax :DISPlay:WAVE:
FORMat {SINGle|DUAL|TRIad|QUAD}
:DISPlay:WAVE:FORMat?
Example :DISPLAY:WAVE:FORMAT SINGLE
:DISPLAY:WAVE:FORMAT? -> :DISPLAY:
WAVE:FORMAT SINGLE

:DISPlay:WAVE:GRATicule

Function Sets or queries the graticule (grid) type.
Syntax :DISPlay:WAVE:GRATicule {GRID|FRAMe|
CROSshair}
:DISPlay:WAVE:GRATicule?
Example :DISPLAY:WAVE:GRATICULE GRID
:DISPLAY:WAVE:GRATICULE? -> :DISPLAY:
WAVE:GRATICULE GRID

:DISPlay:WAVE:INTerpolate

Function Sets or queries the waveform interpolation method.
Syntax :DISPlay:WAVE:INTerpolate {OFF|LINE}
:DISPlay:WAVE:INTerpolate?
Example :DISPLAY:WAVE:INTERPOLATE LINE
:DISPLAY:WAVE:INTERPOLATE?
-> :DISPLAY:WAVE:INTERPOLATE LINE

:DISPlay:WAVE:MAPPing?

Function Queries all split screen waveform mapping settings.
Syntax :DISPlay:WAVE:MAPPing?
Example :DISPLAY:WAVE:MAPPING? -> :DISPLAY:
WAVE:MAPPING:···

:DISPlay:WAVE:MAPPing[:MODE]

Function Sets or queries the split screen waveform mapping mode.
Syntax :DISPlay:WAVE:MAPPing[:MODE] {AUTO|
FIXed|USER}
:DISPlay:WAVE:MAPPing:MODE?
Example :DISPLAY:WAVE:MAPPING:MODE AUTO
:DISPLAY:WAVE:MAPPING:MODE?
-> :DISPLAY:WAVE:MAPPING:MODE AUTO

:DISPlay:WAVE:MAPPing:{U<x>|I<x>}

Function Sets or queries the split screen voltage or current waveform mapping mode.
Syntax :DISPlay:WAVE:MAPPing:
{U<x>|I<x>} {<NRf>}
:DISPlay:WAVE:MAPPing:{U<x>|I<x>}?
The <x> in U<x> and I<x> = 1 to 3 (element)
<NRf> = 0 to 3
Example :DISPLAY:WAVE:MAPPING:U1 0
:DISPLAY:WAVE:MAPPING:U1?
-> :DISPLAY:WAVE:MAPPING:U1 0
Description This command is valid when the waveform mapping method (:DISPlay:WAVE:MAPPing[:MODE]) is set to USER.

:DISPlay:WAVE:POSition?

Function Queries all waveform vertical position (level of the center position) settings.
Syntax :DISPlay:WAVE:POSition?
Example :DISPLAY:WAVE:POSITION? -> :DISPLAY:
WAVE:POSITION:···

:DISPlay:WAVE:POSition:{UALL|IALL}

Function Collectively sets the vertical positions (levels of the center positions) of the voltage or current waveforms of all elements.
Syntax :DISPlay:WAVE:POSition:
{UALL|IALL} {<NRf>}
<NRf> = −130.000 to 130.000 (%)
Example :DISPLAY:WAVE:POSITION:UALL 0

:DISPlay:WAVE:POSition:{U<x>|I<x>}

Function Sets or queries the vertical position (level of the center position) of the specified voltage or current waveform.
Syntax :DISPlay:WAVE:POSition:
{U<x>|I<x>} {<NRf>}
:DISPlay:WAVE:POSition:{U<x>|I<x>}?
<x> = 1 to 3 (element)
<NRf> = −130.000 to 130.000 (%)
Example :DISPLAY:WAVE:POSITION:U1 0
:DISPLAY:WAVE:POSITION:U1?
-> :DISPLAY:WAVE:POSITION:U1 0.000

:DISPlay:WAVE:SVALue (Scale VALue)

Function Sets or queries the on/off status of the scale value display.
Syntax :DISPlay:WAVE:SVALue {<Boolean>}
:DISPlay:WAVE:SVALue?
Example :DISPLAY:WAVE:SVALUE ON
:DISPLAY:WAVE:SVALUE? -> :DISPLAY:
WAVE:SVALUE 1

:DISPlay:WAVE:TDIV

Function Sets or queries the waveform Time/div value.

Syntax
```
:DISPlay:WAVE:TDIV {<Time>}
:DISPlay:WAVE:TDIV?
```
<Time> = 1, 2, 5, 10, 20, 50, 100, 200, or 500 (ms)

Example
```
:DISPLAY:WAVE:TDIV 5MS
:DISPLAY:WAVE:TDIV? -> :DISPLAY:WAVE:
TDIV 5.0E-03
```

Description The specifiable Time/div value is up to 1/10 of the data update rate (`:RATE`).

:DISPlay:WAVE:TLABel (Trace LABel)

Function Sets or queries the on/off status of the waveform labels.

Syntax
```
:DISPlay:WAVE:TLABel {<Boolean>}
:DISPlay:WAVE:TLABel?
```

Example
```
:DISPLAY:WAVE:TLABEL OFF
:DISPLAY:WAVE:TLABEL? -> :DISPLAY:
WAVE:TLABEL 0
```

:DISPlay:WAVE:TRIGger?

Function Queries all trigger settings.

Syntax
```
:DISPlay:WAVE:TRIGger?
```

Example
```
:DISPLAY:WAVE:TRIGGER? -> :DISPLAY:
WAVE:TRIGGER:···
```

:DISPlay:WAVE:TRIGger:LEVel

Function Sets or queries the trigger level.

Syntax
```
:DISPlay:WAVE:TRIGger:LEVel {<NRf>}
:DISPlay:WAVE:TRIGger:LEVel?
```
<NRf> = −100.0 to 100.0 (%)

Example
```
:DISPLAY:WAVE:TRIGGER:LEVEL 0
:DISPLAY:WAVE:TRIGGER:LEVEL?
-> :DISPLAY:WAVE:TRIGGER:LEVEL 0.0
```

Description Set the value in terms of a percentage of the full scale value displayed on the screen.

:DISPlay:WAVE:TRIGger:MODE

Function Sets or queries the trigger mode.

Syntax
```
:DISPlay:WAVE:TRIGger:
MODE {AUTO|NORMal}
:DISPlay:WAVE:TRIGger:MODE?
```

Example
```
:DISPLAY:WAVE:TRIGGER:MODE AUTO
:DISPLAY:WAVE:TRIGGER:MODE?
-> :DISPLAY:WAVE:TRIGGER:MODE AUTO
```

:DISPlay:WAVE:TRIGger:SLOPe

Function Sets or queries the trigger slope.

Syntax
```
:DISPlay:WAVE:TRIGger:
SLOPe {RISE|FALL|BOTH}
:DISPlay:WAVE:TRIGger:SLOPe?
```

Example
```
:DISPLAY:WAVE:TRIGGER:SLOPE RISE
:DISPLAY:WAVE:TRIGGER:SLOPE?
-> :DISPLAY:WAVE:TRIGGER:SLOPE RISE
```

:DISPlay:WAVE:TRIGger:SOURce

Function Sets or queries the trigger source.

Syntax
```
:DISPlay:WAVE:TRIGger:SOURce
{U<x>|I<x>|EXTernal}
:DISPlay:WAVE:TRIGger:SOURce?
```
<x> = 1 to 3 (element)
EXTernal = External trigger input (Ext Clk)

Example
```
:DISPLAY:WAVE:TRIGGER:SOURCE U1
:DISPLAY:WAVE:TRIGGER:SOURCE?
-> :DISPLAY:WAVE:TRIGGER:SOURCE U1
```

:DISPlay:WAVE:{U<x>|I<x>}

Function Sets or queries the on/off status of the voltage or current waveform display.

Syntax
```
:DISPlay:WAVE:{U<x>|I<x>} {<Boolean>}
:DISPlay:WAVE:{U<x>|I<x>}?
```
The <x> in U<x> and I<x> = 1 to 3 (element)

Example
```
:DISPLAY:WAVE:U1 ON
:DISPLAY:WAVE:U1? -> :DISPLAY:WAVE:
U1 1
```

:DISPlay:WAVE:VZoom?

Function Queries all waveform vertical zoom factor settings.

Syntax
```
:DISPlay:WAVE:VZoom?
```

Example
```
:DISPLAY:WAVE:VZOOM? -> :DISPLAY:WAVE:
VZOOM:···
```

:DISPlay:WAVE:VZoom:{UALL|IALL}

Function Collectively sets the vertical zoom factor for the voltage or current waveforms of all elements.

Syntax
```
:DISPlay:WAVE:VZoom:
{UALL|IALL} {<NRf>}
```
<NRf> = 0.1 to 100

Example
```
:DISPLAY:WAVE:VZOOM:UALL 1
```

Description For information about the zoom factor, see the *User's Manual IM760201-01E*.

:DISPlay:WAVE:VZoom:{U<x>|I<x>}

Function Sets or queries the vertical zoom factor (level of the center position) of the specified voltage or current waveform.

Syntax
```
:DISPlay:WAVE:VZoom:
{U<x>|I<x>} {<NRf>}
:DISPlay:WAVE:VZoom:{U<x>|I<x>}?
```
<x> = 1 to 3 (element)
<NRf> = 0.1 to 100

Example
```
:DISPLAY:WAVE:VZOOM:U1 1
:DISPLAY:WAVE:VZOOM:U1? -> :DISPLAY:
WAVE:VZOOM:U1 1.00
```

Description For information about the zoom factor, see the *User's Manual IM760201-01E*.

*Function Option List (Settings That Can Be Used for <Function>)

(1)Numeric data functions

Applicable commands

:DISPlay:NUMeric[:NORMal]:{VAL4|VAL8|VAL16|MATRix}:
ITEM<x> {NONE|**<Function>**,<Element>[,<Order>]}
:DISPlay:NUMeric[:NORMal]:ALL:CURSor {**<Function>**}
:DISPlay:TRENd:ITEM<x>[:FUNCtion] {**<Function>**,<Element>[,<Order
>]}
:NUMeric[:NORMal]:ITEM<x> {NONE|**<Function>**,<Element>[,<Order>]}
:FILE:SAVE:NUMeric:NORMal:**<Function>** {<Boolean>}
:STORe:NUMeric[:NORMal]:**<Function>** {<Boolean>}

<Function>	Function Name Used on the Menu (Numeric Display Header Name)	<Element>	<Order>
URMS	Urms	Required	Not required
UMN	Umn	Required	Not required
UDC	Udc	Required	Not required
URMN	Urmn	Required	Not required
UAC	Uac	Required	Not required
IRMS	Irms	Required	Not required
IMN	Imn	Required	Not required
IDC	Idc	Required	Not required
IRMN	Irmn	Required	Not required
IAC	Iac	Required	Not required
P	P	Required	Not required
S	S	Required	Not required
Q	Q	Required	Not required
LAMBda	λ	Required	Not required
PHI	φ	Required	Not required
FU	FreqU (fU)	Required	Not required
FI	FreqI (fI)	Required	Not required
UPPeak	U+peak (U+pk)	Required	Not required
UMPeak	U-peak (U-pk)	Required	Not required
IPPeak	I+peak (I+pk)	Required	Not required
IMPeak	I-peak (I-pk)	Required	Not required
CFU	CfU	Required	Not required
CFI	CfI	Required	Not required
TIME	Time	Required	Not required
WH	WP	Required	Not required
WHP	WP+	Required	Not required
WHM	WP-	Required	Not required
AH	q	Required	Not required
AHP	q+	Required	Not required
AHM	q-	Required	Not required
WS	WS	Required	Not required
WQ	WQ	Required	Not required
ETA1	η1	Not required	Not required
ETA2	η2	Not required	Not required
F1	F1	Not required	Not required
F2	F2	Not required	Not required
F3	F3	Not required	Not required
F4	F4	Not required	Not required
F5	F5	Not required	Not required
F6	F6	Not required	Not required
F7	F7	Not required	Not required
F8	F8	Not required	Not required

Functions that require the optional harmonic measurement feature.			
UK	U(k)	Required	Required
IK	I(k)	Required	Required
PK	P(k)	Required	Required
SK	S(k)	Required	Required
QK	Q(k)	Required	Required
LAMBDAK	λ(k)	Required	Required
PHIK	φ(k)	Required	Required
PHIUk	φU(k)	Required	Required
PHIIk	φI(k)	Required	Required
UHDFk	Uhdf(k)	Required	Required
IHDFk	Ihdf(k)	Required	Required
PHDFk	Phdf(k)	Required	Required
UTHD	Uthd	Required	Not required
ITHD	Ithd	Required	Not required
PTHD	Pthd	Required	Not required
PHI_U1U2	φUi-Uj	Not required	Not required
PHI_U1U3	φUi-Uk	Not required	Not required
PHI_U1I1	φUi-Ii	Not required	Not required
PHI_U1I2	φUi-Ij	Not required	Not required
PHI_U1I3	φUi-Ik	Not required	Not required
Functions that require the optional delta computation feature.			
DELTA1	ΔF1	Not required	Not required
DELTA2	ΔF2	Not required	Not required
DELTA3	ΔF3	Not required	Not required
DELTA4	ΔF4	Not required	Not required

Note

- For functions in the list above that do not require the element to be specified but that have a parameter for specifying the element (<Element>), omit the parameter or set it to 1.
- Likewise, for functions in the list above that do not require the harmonic order to be specified but that have a parameter for specifying the harmonic order (<Order>), omit the parameter or set it to TOTal.

(2) Numeric list data functions (These functions require the optional harmonic measurement feature.)

Applicable commands

`:DISPlay:NUMeric[:NORMal]:LIST:ITEM<x> {<Function>,<Element>}`

`:DISPlay:BAR:ITEM<x> {<Function>,<Element>}`

`:NUMeric:LIST:ITEM<x> {NONE|<Function>,<Element>}`

<Function>	Function Name Used on the Menu
U	U
I	I
P	P
S	S
Q	Q
LAMBda	λ
PHI	φ
PHIU	φU
PHII	φI
The function options listed below only work with `:NUMeric:LIST:ITEM<x>`.	
UHDF	Uhdf
IHDF	Ihdf
PHDF	Phdf

5.5 FILE Group

The commands in this group deal with file operations.
You can make the same settings and queries that you can make by using FILE on the front panel.

:FILE?

Function Queries all file operation settings.
Syntax :FILE?
Example :FILE? -> :FILE:···

:FILE:CDIRectory

Function Changes the current directory.
Syntax :FILE:CDIRectory {<String>}
<String> = Directory name
Example :FILE:CDIRECTORY "IMAGE"
Description Specify ".." to move up to the parent directory.

:FILE:DELete:IMAGe:{TIFF|BMP|PSCRipt|PNG|JPEG}

Function Deletes a screen image data file.
Syntax :FILE:DELete:IMAGe:{TIFF|BMP|PSCRipt|PNG|JPEG} {<String>}
<String> = File name
Example :FILE:DELETE:IMAGE:TIFF "IMAGE1"
Description Specify the file name without an extension.

:FILE:DELete:NUMeric:ASCii

Function Deletes a numeric data file.
Syntax :FILE:DELete:NUMeric:ASCii {<String>}
<String> = File name
Example :FILE:DELETE:NUMERIC:ASCII "NUM1"
Description Specify the file name without an extension.

:FILE:DELete:SETup

Function Deletes a setup parameter file.
Syntax :FILE:DELete:SETup {<String>}
<String> = File name
Example :FILE:DELETE:SETUP "SETUP1"
Description Specify the file name without an extension.

:FILE:DELete:STORe:{DATA|HEADer}

Function Deletes a stored numeric data file.
Syntax :FILE:DELete:STORe:{DATA|HEADer} {<String>}
<String> = File name
Example :FILE:DELETE:STORE:DATA "STR1"
Description Specify the file name without an extension.

:FILE:DELete:WAVE:ASCii

Function Deletes a waveform display data file.
Syntax :FILE:DELete:WAVE:ASCii {<String>}
<String> = File name
Example :FILE:DELETE:WAVE:ASCII "WAVE1"
Description Specify the file name without an extension.

:FILE:DRIVe

Function Sets the current drive.
Syntax :FILE:DRIVe {RAM|USB[,<NRf>]}
RAM = Internal RAM drive
USB = USB memory drive
<NRf> = Drive number (0 or 1)
Example :FILE:DRIVE RAM
Description If <NRf> is omitted, the drive will be set to 0 (USB 0-0).

:FILE:FILTer

Function Sets or queries the file list filter.
Syntax :FILE:FILTer {ALL|ITEM}
:FILE:FILTer?
Example :FILE:FILTER ALL
:FILE:FILTER? -> :FILE:FILTER ALL

:FILE:FREE?

Function Queries the free space (in bytes) on the current drive.
Syntax :FILE:FREE?
Example :FILE:FREE? -> 20912128

:FILE:LOAD:ABORt

Function Aborts file loading.
Syntax :FILE:LOAD:ABORt
Example :FILE:LOAD:ABORT

:FILE:LOAD:SETup

Function Loads a setup parameter file.
Syntax :FILE:LOAD:SETup {<String>}
<String> = File name
Example :FILE:LOAD:SETUP "SETUP1"
Description
- Specify the file name without an extension.
- This command is an overlap command.

:FILE:PATH?

Function Queries the absolute path of the current directory.
Syntax :FILE:PATH?
Example :FILE:PATH? -> "USB0-0/TEST"

:FILE:SAVE?

Function Queries all file save settings.
Syntax :FILE:SAVE?
Example :FILE:SAVE? -> :FILE:SAVE:···

:FILE:SAVE:ABORt

Function Aborts file saving.
Syntax :FILE:SAVE:ABORt
Example :FILE:SAVE:ABORT

:FILE:SAVE:ANAMing

Function Sets or queries whether or not files are automatically named when they are saved.
Syntax :FILE:SAVE:ANAMing {<Boolean>}
:FILE:SAVE:ANAMing?
Example :FILE:SAVE:ANAMING ON
:FILE:SAVE:ANAMING? -> :FILE:SAVE:ANAMING 1

:FILE:SAVE:COMMent

Function Sets or queries the comment that will be added to files that are saved.
Syntax :FILE:SAVE:COMMent {<String>}
:FILE:SAVE:COMMent?
<String> = Up to 25 characters
Example :FILE:SAVE:COMMENT "CASE1"
:FILE:SAVE:COMMENT? -> :FILE:SAVE:COMMENT "CASE1"

:FILE:SAVE:NAME

Function Sets or queries the name of the file that will be saved.
Syntax :FILE:SAVE:NAME {<String>}
:FILE:SAVE:NAME?
<String> = File name
Example :FILE:SAVE:NAME "WT500"
:FILE:SAVE:NAME? -> :FILE:SAVE:NAME "WT500"

:FILE:SAVE:NUMeric[:EXECute]

Function Saves numeric data to a file.
Syntax :FILE:SAVE:NUMeric[:EXECute] {<String>}
<String> = File name
Example :FILE:SAVE:NUMERIC:EXECUTE "NUM1"
Description
- Specify the file name without an extension.
- This command is an overlap command.

:FILE:SAVE:NUMeric:NORMal?

Function Queries all numeric data storage item settings.
Syntax :FILE:SAVE:NUMeric:NORMal?
Example :FILE:SAVE:NUMERIC:NORMAL?
-> :FILE:SAVE:NUMERIC:NORMAL:···

:FILE:SAVE:NUMeric:NORMal:ALL

Function Collectively turns on or off the output of all element functions when saving numeric data to a file.
Syntax :FILE:SAVE:NUMeric:NORMal:ALL {<Boolean>}
Example :FILE:SAVE:NUMERIC:NORMAL:ALL ON

:FILE:SAVE:NUMeric:NORMal:{ELEMent<x>|SIGMA}

Function Sets or queries the on/off status of the output of each element or of wiring unit Σ when saving numeric data to a file.
Syntax :FILE:SAVE:NUMeric:NORMal:{ELEMent<x>|SIGMA} {<Boolean>}
:FILE:SAVE:NUMeric:NORMal:{ELEMent<x>|SIGMA}?
<x> = 1 to 3
Example :FILE:SAVE:NUMERIC:NORMAL:ELEMENT1 ON
:FILE:SAVE:NUMERIC:NORMAL:ELEMENT1?
-> :FILE:SAVE:NUMERIC:NORMAL:ELEMENT1 1
Description :FILE:SAVE:NUMeric:NORMal:SIGMA is valid on models with two or more elements. To turn the output ON, you must create wiring unit Σ beforehand by setting the wiring system using the [:INPut]WIRing command.

:FILE:SAVE:NUMeric:NORMal:PRESet<x>

Function Presets the output on/off pattern of the element functions that are used when saving numeric data to a file.
Syntax :FILE:SAVE:NUMeric:NORMal:PRESet<x>
<x> = 1 to 2 (preset pattern number)
Example :FILE:SAVE:NUMERIC:NORMAL:PRESET1
Description For information about the preset output patterns, see the *User's Manual IM760201-01E*.

:FILE:SAVE:NUMeric:NORMal:<Function>

Function Sets or queries the on/off status of function output when saving numeric data to a file.

Syntax
```
:FILE:SAVE:NUMeric:NORMal:
<Function> {<Boolean>}
:FILE:SAVE:NUMeric:NORMal:<Function>?
```
<Function> = {URMS|IRMS|P|S|Q|...}

Example
```
:FILE:SAVE:NUMERIC:NORMAL:URMS ON
:FILE:SAVE:NUMERIC:NORMAL:URMS?
-> :FILE:SAVE:NUMERIC:NORMAL:URMS 1
```

Description For information about the options available for <Function>, Function Option List 1 in the DISPlay Group section on page 5-22.

:FILE:SAVE:SETup[:EXECute]

Function Saves setup parameters to a file.

Syntax
```
:FILE:SAVE:
SETup[:EXECute] {<String>}
```
<String> = File name

Example
```
:FILE:SAVE:SETUP:EXECUTE "SETUP1"
```

Description
- Specify the file name without an extension.
- This command is an overlap command.

:FILE:SAVE:WAVE[:EXECute]

Function Saves waveform display data to a file.

Syntax
```
:FILE:SAVE:WAVE[:EXECute] {<String>}
```
<String> = File name

Example
```
:FILE:SAVE:WAVE:EXECUTE "WAVE1"
```

Description
- Specify the file name without an extension.
- This command is an overlap command.

5.6 HARMonics Group

The commands in this group deal with harmonic measurement.
You can make the same settings and queries that you can make by accessing Harmonics in the Setup Menu that appears when you press SETUP on the front panel.
The commands in this group are only valid on models with the optional harmonic measurement feature.

:HARMonics?

Function Queries all harmonic measurement settings.
Syntax `:HARMonics?`
Example `:HARMONICS? -> :HARMONICS:···`
Description This command is only valid on models with the optional harmonic measurement feature.

:HARMonics:ORDer

Function Sets or queries the maximum and minimum harmonic orders that are measured.
Syntax `:HARMonics:ORDer {<NRf>,<NRf>}`
`:HARMonics:ORDer?`
The first <NRf> = 0 or 1 (minimum harmonic order that is measured)
The second <NRf> = 1 to 50 (maximum harmonic order that is measured)
Example `:HARMONICS:ORDER 1,50`
`:HARMONICS:ORDER? -> :HARMONICS:ORDER 1,50`
Description This command is only valid on models with the optional harmonic measurement feature.

:HARMonics:PLLSource

Function Sets or queries the PLL source.
Syntax `:HARMonics:PLLSource {U<x>|I<x>|EXTernal}`
`:HARMonics:PLLSource?`
<x> = 1 to 3 (element)
EXTernal = External clock input (Ext Clk)
Example `:HARMONICS:PLLSOURCE U1`
`:HARMONICS:PLLSOURCE? -> :HARMONICS:PLLSOURCE U1`
Description This command is only valid on models with the optional harmonic measurement feature.

:HARMonics:THD

Function Sets or queries the equation used to compute the THD (total harmonic distortion).
Syntax `:HARMonics:THD {TOTal|FUNDamental}`
`:HARMonics:THD?`
Example `:HARMONICS:THD TOTAL`
`:HARMONICS:THD? -> :HARMONICS:THD TOTAL`
Description This command is only valid on models with the optional harmonic measurement feature.

5.7 HOLD Group

The command in this group deals with the output data hold feature.
You can make the same settings and queries that you can make by using HOLD on the front panel.

:HOLD

Function Sets or queries the on/off status of the output hold feature for display, communication, and other types of data.

Syntax
```
:HOLD {<Boolean>}
:HOLD?
```

Example
```
:HOLD OFF
:HOLD? -> :HOLD 0
```

5.8 IMAGe Group

The commands in this group deal with saving screen image data.
You can make the same settings and queries that you can make by using IMAGE, MENU (SHIFT+IMAGE) on the front panel.

:IMAGe?

Function Queries all screen image data save settings.
Syntax :IMAGe?
Example :IMAGE? -> :IMAGE:···

:IMAGe:ABORt

Function Aborts the saving of screen image data.
Syntax :IMAGe:ABORt
Example :IMAGE:ABORT

:IMAGe:COLor

Function Sets or queries the color tone of the screen image data that will be saved.
Syntax :IMAGe:COLor {OFF|COLor|REVerse|GRAY}
:IMAGe:COLor?
Example :IMAGE:COLOR OFF
:IMAGE:COLOR? -> :IMAGE:COLOR OFF
Description This command is valid when the format (:IMAGe:FORMat) is not PSCRipt.

:IMAGe:COMMent

Function Sets or queries the comment displayed at the bottom of the screen.
Syntax :IMAGe:COMMent {<String>}
:IMAGe:COMMent?
<String> = Up to 30 characters
Example :IMAGE:COMMENT "THIS IS TEST."
:IMAGE:COMMENT? -> :IMAGE:COMMENT "THIS IS TEST."

:IMAGe:COMPression

Function Sets or queries whether or not BMP format screen image data is compressed.
Syntax :IMAGe:COMPression {<Boolean>}
:IMAGe:COMPression?
Example :IMAGE:COMPRESSION ON
:IMAGE:COMPRESSION? -> :IMAGE:COMPRESSION 1
Description This command is valid when the format (:IMAGe:FORMat) is BMP and the color tone (:IMAGe:COLor) is {COLor|REVerse|GRAY}.

:IMAGe:EXECute

Function Saves the screen image data.
Syntax :IMAGe:EXECute
Example :IMAGE:EXECUTE
Description This command is an overlap command.

:IMAGe:FORMat

Function Sets or queries the format of the screen image data that will be saved.
Syntax :IMAGe:FORMat {TIFF|BMP|PSCRipt|PNG|JPEG}
:IMAGe:FORMat?
Example :IMAGE:FORMAT TIFF
:IMAGE:FORMAT? -> :IMAGE:FORMAT TIFF

:IMAGe:SAVE?

Function Queries all screen image data save settings.
Syntax :IMAGe:SAVE?
Example :IMAGE:SAVE? -> :IMAGE:SAVE:···

:IMAGe:SAVE:ANAMing

Function Sets or queries whether or not screen image data files are automatically named when they are saved.
Syntax :IMAGe:SAVE:ANAMing {<Boolean>}
:IMAGe:SAVE:ANAMing?
Example :IMAGE:SAVE:ANAMING ON
:IMAGE:SAVE:ANAMING? -> :IMAGE:SAVE:ANAMING 1

:IMAGe:SAVE:CDIRectory

Function Changes the directory that screen image data is saved to.
Syntax :IMAGe:CDIRectory {<String>}
<String> = Directory name
Example :IMAGE:CDIRECTORY "IMAGE"
Description Specify ".." to move up to the parent directory.

:IMAGe:SAVE:DRIVe

Function Sets the drive that screen image data is saved to.
Syntax :IMAGe:SAVE:DRIVe {RAM|USB[,<NRf>]}
RAM = Internal RAM drive
USB = USB memory drive
<NRf> = Drive number (0 or 1)
Example :IMAGE:SAVE:DRIVE RAM
Description If <NRf> is omitted, the drive will be set to 0 (USB 0-0).

:IMAGe:SAVE:FREE?

Function Queries the free space (in bytes) on the drive that the screen image data is saved to.
Syntax :IMAGe:SAVE:FREE?
Example :IMAGE:SAVE:FREE? -> 20912128

:IMAGe:SAVE:NAME

Function Sets or queries the name of the screen image data file that will be saved.

Syntax `:IMAGe:SAVE:NAME {<String>}`
`:IMAGe:SAVE:NAME?`
<String> = File name

Example `:IMAGE:SAVE:NAME "IMAGE1"`
`:IMAGE:SAVE:NAME? -> :IMAGE:SAVE:NAME "IMAGE1"`

Description
- Set the save destination drive with the `:IMAGe:SAVE:DRIVe` command and the directory with the `:IMAGe:SAVE:CDIRectory` command.
- You can query the path that screen image data is saved to by using the `:IMAGe:SAVE:PATH?` command.

:IMAGe:SAVE:PATH?

Function Queries the absolute path of the directory that the screen image data is saved to.

Syntax `:IMAGe:SAVE:PATH?`

Example `:IMAGE:SAVE:PATH? -> "USB0-0/IMAGE"`

:IMAGe:SEND?

Function Queries the screen image data.

Syntax `:IMAGe:SEND?`

Example `:IMAGE:SEND? -> #N` (N-digit byte number; data byte sequence)

Description The number of digits N in the data byte number varies depending on the output data size.

5.9 INPut Group

The commands in this group deal with the measurement conditions of the input elements.
You can make the same settings and queries that you can make by pressing the keys on the front panel's measurement condition setup area (the area enclosed in light blue) and by using NULL (SHIFT + MISC) or by accessing Wiring, Range, Scaling, Sync Source, and Filters in the Setup Menu that appears when you press SETUP.

:INPut?

Function Queries all input element settings.
Syntax `:INPut?`
Example `:INPUT? -> :INPUT:···`

[:INPut]:CFACtor

Function Sets or queries the crest factor.
Syntax `[:INPut]:CFACtor {<NRf>}`
`[:INPut]:CFACtor?`
<NRf> = 3 or 6
Example `:INPUT:CFACTOR 3`
`:INPUT:CFACTOR? -> :INPUT:CFACTOR 3`

[:INPut]:CURRent?

Function Queries all electric current measurement settings.
Syntax `[:INPut]:CURRent?`
Example `:INPUT:CURRENT? -> :INPUT:CURRENT:···`

[:INPut]:CURRent:AUTO?

Function Queries the electric current auto range on/off status of all elements.
Syntax `[:INPut]:CURRent:AUTO?`
Example `:INPUT:CURRENT:AUTO? -> :INPUT:CURRENT:AUTO:···`

[:INPut]:CURRent:AUTO[:ALL]

Function Collectively turns the electric current auto range of all elements on or off.
Syntax `[:INPut]:CURRent:AUTO[:ALL] {<Boolean>}`
Example `:INPUT:CURRENT:AUTO:ALL ON`

[:INPut]:CURRent:AUTO:ELEMent<x>

Function Sets or queries the electric current auto range on/off status of the specified element.
Syntax `[:INPut]:CURRent:AUTO:ELEMent<x> {<Boolean>}`
`[:INPut]:CURRent:AUTO:ELEMent<x>?`
<x> = 1 to 3 (element)
Example `:INPUT:CURRENT:AUTO:ELEMENT1 ON`
`:INPUT:CURRENT:AUTO:ELEMENT1?`
`-> :INPUT:CURRENT:AUTO:ELEMENT1 1`

[:INPut]:CURRent:AUTO:SIGMA

Function Collectively turns the electric current auto range of all elements belonging to wiring unit Σ on or off.
Syntax `[:INPut]:CURRent:AUTO:SIGMA {<Boolean>}`
Example `:INPUT:CURRENT:AUTO:SIGMA ON`
Description
- Only valid on models with two or more elements.
- This command is invalid if the wiring system setting (`[:INPut]:WIRing`) is made in such a way that wiring unit Σ does not exist.

[:INPut]:CURRent:RANGe?

Function Queries the electric current range of all elements.
Syntax `[:INPut]:CURRent:RANGe?`
Example `:INPUT:CURRENT:RANGE? -> :INPUT:CURRENT:RANGE:···`

[:INPut]:CURRent:RANGe[:ALL]

Function Collectively sets the electric current range of all elements.
Syntax `[:INPut]:CURRent:RANGe[:ALL] {<Current>|(EXTernal,<Voltage>)}`
- When the crest factor is set to 3:
  <Current> = 500 (mA), 1, 2, 5, 10, 20, or 40 (A) (for direct current input)
  <Voltage> = 50, 100, 200, 500 (mV), 1, 2, 5, or 10 (V) (for external current sensor input)
- When the crest factor is set to 6:
  <Current> = 250, 500 (mA), 1, 2.5, 5, 10, or 20 (A) (for direct current input)
  <Voltage> = 25, 50, 100, 250, 500 (mV), 1, 2.5, or 5 (V) (for external current sensor input)

Example `:INPUT:CURRENT:RANGE:ALL 40A`
`:INPUT:CURRENT:RANGE:ALL EXTERNAL,10V`
Description The EXTernal and <Voltage> options are only available on models with the optional external electric current sensor input.

[:INPut]:CURRent:RANGe:ELEMent<x>

Function Sets or queries the electric current range of the specified element.

Syntax
```
[:INPut]:CURRent:RANGe:ELEMent<x>
{<Current>|(EXTernal,<Voltage>)}
[:INPut]:CURRent:RANGe:ELEMent<x>?
```
<x> = 1 to 3 (element)
- When the crest factor is set to 3:
  <Current> = 500 (mA), 1, 2, 5, 10, 20, or 40 (A) (for direct current input)
  <Voltage> = 50, 100, 200, 500 (mV), 1, 2, 5, or 10 (V) (for external sensor input)
- When the crest factor is set to 6:
  <Current> = 250, 500 (mA), 1, 2.5, 5, 10, or 20 (A) (for direct current input)
  <Voltage> = 25, 50, 100, 250, 500 (mV), 1, 2.5, or 5 (V) (for external current sensor input)

Example
```
:INPUT:CURRENT:RANGE:ELEMENT1 40A
:INPUT:CURRENT:RANGE:ELEMENT1?
-> :INPUT:CURRENT:RANGE:
ELEMENT1 40.0E+00

:INPUT:CURRENT:RANGE:
ELEMENT1 EXTERNAL,10V
:INPUT:CURRENT:RANGE:ELEMENT1?
-> :INPUT:CURRENT:RANGE:
ELEMENT1 EXTERNAL,10.00E+00
```

Description The EXTernal and <Voltage> options are only available on models with the optional external electric current sensor input.

[:INPut]:CURRent:RANGe:SIGMA

Function Collectively sets the electric current range of all elements belonging to wiring unit Σ.

Syntax
```
[:INPut]:CURRent:RANGe:
SIGMA {<Current>|(EXTernal,<Voltage>)}
```
- When the crest factor is set to 3:
  <Current> = 500 (mA), 1, 2, 5, 10, 20, or 40 (A) (for direct current input)
  <Voltage> = 50, 100, 200, 500 (mV), 1, 2, 5, or 10 (V) (for external sensor input)
- When the crest factor is set to 6:
  <Current> = 250, 500 (mA), 1, 2.5, 5, 10, or 20 (A) (for direct current input)
  <Voltage> = 25, 50, 100, 250, 500 (mV), 1, 2.5, or 5 (V) (for external current sensor input)

Example
```
:INPUT:CURRENT:RANGE:SIGMA 40A
:INPUT:CURRENT:RANGE:
SIGMA EXTERNAL,10V
```

Description
- Only valid on models with two or more elements.
- This command is invalid if the wiring system setting ([:INPut]:WIRing) is made in such a way that wiring unit Σ does not exist.
- The EXTernal and <Voltage> options are only available on models with the optional external electric current sensor input.

[:INPut]:CURRent:SRATio?

Function Queries the external electric current sensor scaling ratios of all elements.

Syntax
```
[:INPut]:CURRent:SRATio?
```

Example
```
:INPUT:CURRENT:SRATIO? -> :INPUT:
VCURRENT:SRATIO:···
```

Description This command is only valid on models with the optional external electric current sensor input.

[:INPut]:CURRent:SRATio[:ALL]

Function Collectively sets the external electric current sensor scaling ratios of all elements.

Syntax
```
[:INPut]:CURRent:SRATio[:ALL] {<NRf>}
```
<NRf> = 0.0001 to 99999.9999

Example
```
:INPUT:CURRENT:SRATIO:ALL 10
```

Description This command is only valid on models with the optional external electric current sensor input.

[:INPut]:CURRent:SRATio:ELEMent<x>

Function Sets or queries the external electric current sensor scaling ratio of the specified element.

Syntax
```
[:INPut]:CURRent:SRATio:
ELEMent<x> {<NRf>}
[:INPut]:CURRent:SRATio:ELEMent<x>?
```
<x> = 1 to 3 (element)
<NRf> = 0.0001 to 99999.9999

Example
```
:INPUT:CURRENT:SRATIO:ELEMENT1 10
:INPUT:CURRENT:SRATIO:ELEMENT1?
-> :INPUT:CURRENT:SRATIO:
ELEMENT1 10.0000
```

Description This command is only valid on models with the optional external electric current sensor input.

[:INPut]:ESELect

Function Sets or queries the element whose measurement range will be set.

Syntax
```
[:INPut]:ESELect {<NRf>|ALL}
[:INPut]:ESELect?
```
<NRf> = 1 to 3 (element)

Example
```
:INPUT:ESELECT 1
:INPUT:ESELECT? -> :INPUT:ESELECT 1
```

[:INPut]:FILTer?

Function Queries all input filter settings.

Syntax
```
[:INPut]:FILTer?
```

Example
```
:INPUT:FILTER? -> :INPUT:FILTER:···
```

[:INPut]:FILTer:FREQuency?

Function Queries the frequency filter on/off status of all elements.

Syntax
```
[:INPut]:FILTer:FREQuency?
```

Example
```
:INPUT:FILTER:FREQUENCY? -> :INPUT:
FILTER:···
```

[:INPut]:FILTer:FREQuency[:ALL]

Function Collectively sets the frequency filter of all elements.
Syntax [:INPut]:FILTer:FREQuency[:ALL] {<Boolean>}
Example :INPUT:FILTER:FREQUENCY:ALL OFF

[:INPut]:FILTer:FREQuency:ELEMent<x>

Function Sets or queries the frequency filter of the specified element.
Syntax [:INPut]:FILTer:FREQuency:ELEMent<x> {<Boolean>}
[:INPut]:FILTer:FREQuency:ELEMent<x>?
<x> = 1 to 3 (element)
Example :INPUT:FILTER:FREQUENCY:ELEMENT1 ON
:INPUT:FILTER:FREQUENCY:ELEMENT1?
-> :INPUT:FILTER:FREQUENCY:ELEMENT1 1

[:INPut]:FILTer:LINE?

Function Queries the line filter settings of all elements.
Syntax [:INPut]:FILTer:LINE?
Example :INPUT:FILTER:LINE? -> :INPUT:FILTER:LINE:···

[:INPut]:FILTer[:LINE][:ALL]

Function Collectively sets the line filter of all elements.
Syntax [:INPut]:FILTer[:LINE][:ALL] {OFF|<Frequency>}
OFF = Line filter OFF
<Frequency> = 500 Hz or 5.5 kHz (line filter on; cutoff frequency)
Example :INPUT:FILTER:LINE:ALL OFF
Description The frequency filter ([:INPut]:FILTer:FREQuency) is turned on automatically when 500 Hz is selected.

[:INPut]:FILTer[:LINE]:ELEMent<x>

Function Sets or queries the line filter of the specified element.
Syntax [:INPut]:FILTer[:LINE]:ELEMent<x> {OFF|<Frequency>}
[:INPut]:FILTer[:LINE]:ELEMent<x>?
<x> = 1 to 3 (element)
OFF = Line filter OFF
<Frequency> = 500 Hz or 5.5 kHz (line filter on; cutoff frequency)
Example :INPUT:FILTER:LINE:ELEMENT1 OFF
:INPUT:FILTER:LINE:ELEMENT1?
-> :INPUT:FILTER:LINE:ELEMENT1 OFF
Description The frequency filter ([:INPut]:FILTer:FREQuency) is turned on automatically when 500 Hz is selected.

[:INPut]:INDependent

Function Sets or queries the on/off status of independent input element configuration.
Syntax [:INPut]:INDependent {<Boolean>}
[:INPut]:INDependent?
Example :INPUT:INDEPENDENT OFF
:INPUT:INDEPENDENT? -> :INPUT:INDEPENDENT 0
Description Only valid on models with two or more elements.

[:INPut]:MODUle?

Function Queries the input element type.
Syntax [:INPut]:MODUle? {<NRf>}
[:INPut]:MODUle?
<NRf> = 1 to 3 (element)
Example :INPUT:MODULE? 1 -> 40
:INPUT:MODULE? -> 40,40,40
Description
- The response information is as follows:
  40 = (Standard) power element (max. current range = 40 A)
  0 = No input element
- If the parameter is omitted, the input element types of all elements are output in order, starting with element 1.

[:INPut]:NULL

Function Sets or queries the on/off status of the NULL feature.
Syntax [:INPut]:NULL {<Boolean>}
[:INPut]:NULL?
Example :INPUT:NULL ON
:INPUT:NULL? -> :INPUT:NULL 1

[:INPut]:POVer?

Function Queries the peak over information.
Syntax [:INPut]:POVer?
Example :INPUT:POVER? -> 0
Description
- The peak over information of each element is mapped as shown below. For the response, the sum of the values of each bit is returned in decimal format.
- For example, a response of 16 indicates that a peak over is occurring at U3.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	I3	U3	I2	U2	I1	U1

[:INPut]:SCALing?

Function Queries all scaling settings.
Syntax [:INPut]:SCALing?
Example :INPUT:SCALING? -> :INPUT:SCALING:···

[:INPut]:SCALing:STATe?

Function Queries the on/off status of the scaling of all elements.

Syntax `[:INPut]:SCALing:STATe?`

Example
```
:INPUT:SCALING:STATE? -> :INPUT:
SCALING:STATE:···
```

[:INPut]:SCALing[:STATe][:ALL]

Function Collectively turns the scaling of all elements on or off.

Syntax
```
[:INPut]:
SCALing[:STATe][:ALL] {<Boolean>}
```

Example
```
:INPUT:SCALING:STATE:ALL OFF
```

[:INPut]:SCALing[:STATe]:ELEMent<x>

Function Sets or queries the on/off status of the scaling of the specified element.

Syntax
```
[:INPut]:SCALing[:STATe]:
ELEMent<x> {<Boolean>}
[:INPut]:SCALing[:STATe]:ELEMent<x>?
```
<x> = 1 to 3 (element)

Example
```
:INPUT:SCALING:STATE:ELEMENT1 OFF
:INPUT:SCALING:STATE:ELEMENT1?
-> :INPUT:SCALING:STATE:ELEMENT1 0
```

[:INPut]:SCALing:{VT|CT|SFACtor}?

Function Queries the VT ratio, CT ratio, or power factor of all elements.

Syntax `[:INPut]:SCALing:{VT|CT|SFACtor}?`

Example
```
:INPUT:SCALING:VT? -> :INPUT:SCALING:
VT:···
```

[:INPut]:SCALing:{VT|CT|SFACtor}[:ALL]

Function Collectively sets the VT ratio, CT ratio, or power factor of all elements.

Syntax
```
[:INPut]:SCALing:{VT|CT|SFACtor}
[:ALL] {<NRf>}
```
<NRf> = 0.0001 to 99999.9999

Example
```
:INPUT:SCALING:VT:ALL 1
```

[:INPut]:SCALing:{VT|CT|SFACtor}:ELEMent<x>

Function Sets or queries the VT ratio, CT ratio, or power factor of the specified element.

Syntax
```
[:INPut]:SCALing:{VT|CT|SFACtor}:
ELEMent<x> {<NRf>}
[:INPut]:SCALing:{VT|CT|SFACtor}:
ELEMent<x>?
```
<x> = 1 to 3 (element)
<NRf> = 0.0001 to 99999.9999

Example
```
:INPUT:SCALING:VT:ELEMENT1 1
:INPUT:SCALING:VT:ELEMENT1?
-> :INPUT:SCALING:VT:ELEMENT1 1.0000
```

[:INPut]:SYNChronize?

Function Queries the synchronization source of all elements.

Syntax `[:INPut]:SYNChronize?`

Example
```
INPUT:SYNCHRONIZE? -> :INPUT:
SYNCHRONIZE:···
```

[:INPut]:SYNChronize[:ALL]

Function Collectively sets the synchronization source of all elements.

Syntax
```
[:INPut]:SYNChronize[:ALL] {U<x>|
I<x>|EXTernal|NONE}
```
<x> = 1 to 3 (element)
EXTernal = External clock input (Ext Clk)
NONE = No synchronization source

Example
```
:INPUT:SYNCHRONIZE:ALL I1
```

[:INPut]:SYNChronize:ELEMent<x>

Function Sets or queries the synchronization source of the specified element.

Syntax
```
[:INPut]:SYNChronize:
ELEMent<x> {U<x>|I<x>|EXTernal|NONE}
[:INPut]:SYNChronize:ELEMent<x>?
```
<x> = 1 to 3 (element)
EXTernal = External clock input (Ext Clk)
NONE = No synchronization source

Example
```
:INPUT:SYNCHRONIZE:ELEMENT1 I1
:INPUT:SYNCHRONIZE:ELEMENT1?
-> :INPUT:SYNCHRONIZE:ELEMENT1 I1
```

[:INPut]:SYNChronize:SIGMA

Function Collectively sets the synchronization source of all elements belonging to wiring unit Σ.

Syntax
```
[:INPut]:SYNChronize:SIGMA {U<x>|
I<x>|EXTernal|NONE}
```

Example
```
:INPUT:SYNCHRONIZE:SIGMA I1
```

Description
- Only valid on models with two or more elements.
- This command is invalid if the wiring system setting (`[:INPut]:WIRing`) is made in such a way that wiring unit Σ does not exist.

[:INPut]:VOLTage?

Function Queries all voltage measurement settings.

Syntax `[:INPut]:VOLTage?`

Example
```
:INPUT:VOLTAGE? -> :INPUT:VOLTAGE:···
```

[:INPut]:VOLTage:AUTO?

Function Queries the voltage auto range on/off status of all elements.

Syntax `[:INPut]:VOLTage:AUTO?`

Example
```
:INPUT:VOLTAGE:AUTO? -> :INPUT:
VOLTAGE:AUTO:···
```

[:INPut]:VOLTage:AUTO[:ALL]

Function Collectively turns the voltage auto range of all elements on or off.

Syntax
```
[:INPut]:VOLTage:
AUTO[:ALL] {<Boolean>}
```

Example
```
:INPUT:VOLTAGE:AUTO:ALL ON
```

[:INPut]:VOLTage:AUTO:ELEMent<x>

Function Sets or queries the voltage auto range on/off status of the specified element.

Syntax
```
[:INPut]:VOLTage:AUTO:
ELEMent<x> {<Boolean>}
[:INPut]:VOLTage:AUTO:ELEMent<x>?
```
<x> = 1 to 3 (element)

Example
```
:INPUT:VOLTAGE:AUTO:ELEMENT1 ON
:INPUT:VOLTAGE:AUTO:ELEMENT1?
-> :INPUT:VOLTAGE:AUTO:ELEMENT1 1
```

[:INPut]:VOLTage:AUTO:SIGMA

Function Collectively turns the voltage auto range of all elements belonging to wiring unit Σ on or off.

Syntax
```
[:INPut]:VOLTage:AUTO:
SIGMA {<Boolean>}
```

Example
```
:INPUT:VOLTAGE:AUTO:SIGMA ON
```

Description
- Only valid on models with two or more elements.
- This command is invalid if the wiring system setting ([:INPut]:WIRing) is made in such a way that wiring unit Σ does not exist.

[:INPut]:VOLTage:RANGe?

Function Queries the voltage range of all elements.

Syntax
```
[:INPut]:VOLTage:RANGe?
```

Example
```
:INPUT:VOLTAGE:RANGE? -> :INPUT:
VOLTAGE:RANGE:···
```

[:INPut]:VOLTage:RANGe[:ALL]

Function Collectively sets the voltage range of all elements.

Syntax
```
[:INPut]:VOLTage:RANGe[:ALL]
{<Voltage>}
```
- When the crest factor is set to 3: <Voltage> = 15, 30, 60, 100, 150, 300, 600, or 1000 (V)
- When the crest factor is set to 6: <Voltage> = 7.5, 15, 30, 50, 75, 150, 300, or 500 (V)

Example
```
:INPUT:VOLTAGE:RANGE:ALL 1000V
```

[:INPut]:VOLTage:RANGe:ELEMent<x>

Function Sets or queries the voltage range of the specified element.

Syntax
```
[:INPut]:VOLTage:RANGe:
ELEMent<x> {<Voltage>}
[:INPut]:VOLTage:RANGe:ELEMent<x>?
```
<x> = 1 to 3 (element)
- When the crest factor is set to 3: <Voltage> = 15, 30, 60, 100, 150, 300, 600, or 1000 (V)
- When the crest factor is set to 6: <Voltage> = 7.5, 15, 30, 50, 75, 150, 300, 500 (V)

Example
```
:INPUT:VOLTAGE:RANGE:ELEMENT1 1000V
:INPUT:VOLTAGE:RANGE:ELEMENT1?
-> :INPUT:VOLTAGE:RANGE:
ELEMENT1 1.000E+03
```

[:INPut]:VOLTage:RANGe:SIGMA

Function Collectively sets the voltage range of all elements belonging to wiring unit Σ.

Syntax
```
[:INPut]:VOLTage:RANGe:
SIGMA {<Voltage>}
```
- When the crest factor is set to 3: <Voltage> = 15, 30, 60, 100, 150, 300, 600, or 1000 (V)
- When the crest factor is set to 6: <Voltage> = 7.5, 15, 30, 50, 75, 150, 300, 500 (V)

Example
```
:INPUT:VOLTAGE:RANGE:SIGMA 1000V
```

Description
- Only valid on models with two or more elements.
- This command is invalid if the wiring system setting ([:INPut]:WIRing) is made in such a way that wiring unit Σ does not exist.

[:INPut]:WIRing

Function Sets or queries the wiring system.

Syntax
```
[:INPut]:WIRing {(P1W2|P1W3|P3W3|
P3W4|V3A3)[,(P1W2|P1W3|P3W3|NONE)]
[,(P1W2|NONE)]}
[:INPut]:WIRing?
```
P1W2 = Single-phase, two-wire system [1P2W]
P1W3 = Single-phase, three-wire system [1P3W]
P3W3 = Three-phase, three-wire system [3P3W]
P3W4 = Three-phase, four-wire system [3P4W]
V3A3 = Three-phase, three-wire (three-voltage, three-current) system [3P3W(3V3A)]
NONE = No wiring

Example
- Example for a 3-element model
```
:INPUT:WIRING P1W2,P1W2,P1W2
:INPUT:WIRING? -> :INPUT:
WIRING P1W2,P1W2,P1W2
:INPUT:WIRING P1W3,P1W2

:INPUT:WIRING? -> :INPUT:
WIRING P1W3,P1W2
:INPUT:WIRING P3W4
:INPUT:WIRING? -> :INPUT:WIRING P3W4
```
- Example for a 2-element model
```
:INPUT:WIRING P3W3
:INPUT:WIRING? -> :INPUT:WIRING P3W3
```

Description
- Set the wiring system pattern in order starting from the element with the smallest number.
- Some wiring system patterns cannot be selected on certain model types. For information about wiring system patterns, see the *User's Manual IM760201-01E*.
- The pattern is fixed to P1W2 on 1-element models. No other setting is allowed.

5.10 INTEGrate Group

The commands in this group deal with integration.
You can make the same settings and queries that you can make by accessing Integrator in the Setup Menu that appears when you press SETUP on the front panel or by pressing INTEGRATOR (START/STOP) or RESET (SHIFT + START/STOP).

:INTEGrate?

Function Queries all integration settings.
Syntax :INTEGrate?
Example :INTEGRATE? -> :INTEGRATE:···

:INTEGrate:ACAL

Function Sets or queries the on/off status of auto calibration.
Syntax :INTEGrate:ACAL {<Boolean>}
:INTEGrate:ACAL?
Example :INTEGRATE:ACAL OFF
:INTEGRATE:ACAL? -> :INTEGRATE:ACAL 0

:INTEGrate:MODE

Function Sets or queries the integration mode.
Syntax :INTEGrate:MODE {NORMal|CONTinuous|RNORmal|RCONtinuous}
:INTEGrate:MODE?
NORMal = Normal integration mode
CONTinuous = Continuous integration mode
RNORmal = Real-time normal integration mode
RCONtinuous = Real-time continuous integration mode
Example :INTEGRATE:MODE NORMAL
:INTEGRATE:MODE? -> :INTEGRATE:MODE NORMAL

:INTEGrate:QMODe

Function Sets or queries the electric current integration mode.
Syntax :INTEGrate:QMODe {RMS|MEAN|DC|RMEAN|AC}
:INTEGrate:QMODe?
Example :INTEGRATE:QMODE DC
:INTEGRATE:QMODE? -> :INTEGRATE:QMODE DC

:INTEGrate:RESet

Function Resets the integrated value.
Syntax :INTEGrate:RESet
Example :INTEGRATE:RESET

:INTEGrate:RTIMe?

Function Queries the integration start and end times for real-time integration mode.
Syntax :INTEGrate:RTIMe?
Example :INTEGRATE:RTIME? -> :INTEGRATE:RTIME:···

:INTEGrate:RTIMe:{STARt|END}

Function Sets or queries the integration start or end time for real-time integration mode.
Syntax :INTEGrate:RTIMe:{STARt|END} {<NRf>,<NRf>,<NRf>,<NRf>,<NRf>,<NRf>}
:INTEGrate:RTIMe:{STARt|END}?
{<NRf>, <NRf>, <NRf>, <NRf>, <NRf>, <NRf>} = 2001, 1, 1, 0, 0, 0 to 2099, 12, 31, 23, 59, 59
The first <NRf> = 2001 to 2099 (year)
The second <NRf> = 1 to 12 (month)
The third <NRf> = 1 to 31 (day)
The fourth <NRf> = 0 to 23 (hour)
The fifth <NRf> = 0 to 59 (minute)
The sixth <NRf> = 0 to 59 (second)
Example :INTEGRATE:RTIME:START 2008,1,1,0,0,0
:INTEGRATE:RTIME:START?
-> :INTEGRATE:RTIME:START 2008,1,1,0,0,0

:INTEGrate:STARt

Function Starts integration.
Syntax :INTEGrate:STARt
Example :INTEGRATE:START

:INTEGrate:STATe?

Function Queries the integration condition.
Syntax :INTEGrate:STATe?
Example :INTEGRATE:STATE? -> RESET
Description The response information is as follows:
RESet = Integration reset
READy = Waiting (real-time integration mode)
STARt = Integration in progress
STOP = Integration stop
ERRor = Abnormal integration termination (integration overflow, power failure)
TIMeup = Integration stop due to integration timeout

:INTEGrate:STOP

Function Stops integration.
Syntax :INTEGrate:STOP
Example :INTEGRATE:STOP

:INTEGrate:TIMer

Function Sets or queries the integration timer value.

Syntax `:INTEGrate:TIMer {<NRf>,<NRf>,<NRf>}`
`:INTEGrate:TIMer?`
{<NRf>, <NRf>, <NRf>} = 0, 0, 0 to 10000, 0, 0
The first <NRf> = 0 to 10000 (hour)
The second <NRf> = 0 to 59 (minute)
The third <NRf> = 0 to 59 (second)

Example
```
:INTEGRATE:TIMER 1,0,0
:INTEGRATE:TIMER? -> :INTEGRATE:
TIMER 1,0,0
```

:INTEGrate:WPTYpe

Function Sets or queries the power level integration method for each polarity (WP+ and WP−).

Syntax `:INTEGrate:WPTYpe {CHARge|SOLD}`
`:INTEGrate:WPTYpe?`
CHARge = Charge-discharge
SOLD = Commercial power

Example
```
:INTEGRATE:WPTYPE CHARGE
:INTEGRATE:WPTYPE? -> :INTEGRATE:
WPTYPE CHARGE
```

5.11 MEASure Group

The commands in this group deal with computation.
You can make the same settings and queries that you can make by accessing Wiring(Efficiency), Averaging, Measure, User Function, Freq Items, and Delta Measure in the Setup Menu that appears when you press SETUP on the front panel.

:MEASure?

Function Queries all computation settings.
Syntax :MEASure?
Example :MEASURE? -> :MEASURE:···

:MEASure:AVERaging?

Function Queries all averaging settings.
Syntax :MEASure:AVERaging?
Example :MEASURE:AVERAGING? -> :MEASURE:AVERAGING:···

:MEASure:AVERaging:COUNt

Function Sets or queries the averaging coefficient.
Syntax :MEASure:AVERaging:COUNt {<NRf>}
:MEASure:AVERaging:COUNt?
<NRf> = 2, 4, 8, 16, 32, or 64 (attenuation constant when TYPE = EXPonent)
<NRf> = 8, 16, 32, or 64 (moving average count when TYPE = LINear)
Example :MEASURE:AVERAGING:COUNT 2
:MEASURE:AVERAGING:COUNT?
-> :MEASURE:AVERAGING:COUNT 2
Description The averaging of harmonic measurement (option) functions is only valid when TYPE is set to EXPonent (attenuation constant). For details, see the *User's Manual IM760201-01E*.

:MEASure:AVERaging[:STATe]

Function Sets or queries the on/off status of averaging.
Syntax :MEASure:AVERaging[:STATe] {<Boolean>}
:MEASure:AVERaging:STATe?
Example :MEASURE:AVERAGING:STATE ON
:MEASURE:AVERAGING:STATE?
-> :MEASURE:AVERAGING:STATE 1

:MEASure:AVERaging:TYPE

Function Sets or queries the averaging type.
Syntax :MEASure:AVERaging:
TYPE {EXPonent|LINear}
:MEASure:AVERaging:TYPE?
Example :MEASURE:AVERAGING:TYPE EXPONENT
:MEASURE:AVERAGING:TYPE? -> :MEASURE:AVERAGING:TYPE EXPONENT
Description The averaging of harmonic measurement (option) functions is only valid when the type is set to EXPonent. For details, see the *User's Manual IM760201-01E*.

:MEASure:DMeasure?

Function Queries all delta computation settings.
Syntax :MEASure:DMeasure?
Example :MEASURE:DMEASURE? -> :MEASURE:DMEASURE:···
Description This command is only valid on models with the optional delta computation feature.

:MEASure:DMeasure:MODE

Function Sets or queries the voltage or current mode that is used in delta computation.
Syntax :MEASure:DMeasure:
MODE {RMS|MEAN|DC|RMEAN|AC}
:MEASure:DMeasure:MODE?
Example :MEASURE:DMEASURE:MODE RMS
:MEASURE:DMEASURE:MODE? -> :MEASURE:DMEASURE:MODE RMS
Description This command is only valid on models with the optional delta computation feature.

:MEASure:DMeasure[:SIGMA]

Function Sets or queries the delta computation mode for wiring unit Σ.

Syntax
```
:MEASure:DMeasure[:SIGMA] {OFF|DIFFerence|P3W3_V3A3|ST_DT|DT_ST}
:MEASure:DMeasure:SIGMA?
```

Example
```
:MEASURE:DMEASURE:SIGMA OFF
:MEASURE:DMEASURE:SIGMA?
-> :MEASURE:DMEASURE:SIGMA OFF
```

Description
- This command is only valid on models with the optional delta computation feature.
- The different modes are explained below: The modes that can be selected vary depending on the wiring system of wiring unit Σ.
  OFF = No delta computation (only selectable with a single-phase, two-wire system, 1P2W)
  DIFFerence = Differential voltage, differential current (only selectable with a single-phase, three-wire system, 1P3W, or a three-phase, three-wire system, 3P3W)
  P3W3_V3A3 = 3P3W-to-3V3A conversion (only selectable with a single-phase, three-wire system, 1P3W, or a three-phase, three-wire system, 3P3W)
  ST_DT = Star-to-delta conversion (only selectable with a three-phase, four-wire system, 3P4W)
  DT_ST = Delta-to-star conversion (only selectable with a three-phase, three-wire/three-voltage, three current system, 3P3W/3V3A)

:MEASure:EFFiciency?

Function Queries all efficiency computation settings.

Syntax
```
:MEASure:EFFiciency?
```

Example
```
:MEASURE:EFFICIENCY? -> :MEASURE:EFFICIENCY:···
```

:MEASure:EFFiciency:ETA<x>

Function Sets or queries the efficiency equation.

Syntax
```
:MEASure:EFFiciency:ETA<x> {(OFF|P<x>|PA|UDEF<x>)[,(P<x>|PA|UDEF<x>)]}
:MEASure:EFFiciency:ETA<x>?
```
The <x> in ETA<x> = 1 or 2 (η1 to η2)
OFF = No computation (the denominator is ignored)
The <x> in P<x> = 1 to 3 (element)
PA = PΣ (only on models with 2 or 3 elements)
The <x> in UDEF<x> = 1 or 2 (Udef1 to Udef2)

Example
```
:MEASURE:EFFICIENCY:ETA1 P3,PA
:MEASURE:EFFICIENCY:ETA1?
-> :MEASURE:EFFICIENCY:ETA1 P3,PA
```

Description
- Set the numerator and then the denominator.
- The denominator can be omitted. The denominator is set to 1 when it is omitted.
- The denominator is omitted from the response to a query when it is 1.

:MEASure:EFFiciency:UDEF<x>

Function Sets or queries the user-defined parameters used in the efficiency equation.

Syntax
```
:MEASure:EFFiciency:UDEF<x> {(NONE|P<x>|PA)[,(NONE|P<x>|PA)][,(NONE|P<x>|PA)][,(NONE|P<x>|PA)]}
:MEASure:EFFiciency:UDEF<x>?
```
The <x> in UDEF<x> = 1 or 2 (Udef1 to Udef2)
NONE = No parameters
The <x> in P<x> = 1 to 3 (element)
PA = PΣ (only on models with 2 or 3 elements)

Example
```
:MEASURE:EFFICIENCY:UDEF1 P1,P2,P3
:MEASURE:EFFICIENCY:UDEF1?
-> :MEASURE:EFFICIENCY:UDEF1 P1,P2,P3
```

Description
- Set the parameters in ascending order.
- Parameters 2 to 4 can be omitted. Omitted parameters are set to NONE.
- If any parameter from 2 to 4 and all of its subsequent parameters are NONE, those parameters are omitted.

:MEASure:FREQuency?

Function Queries all frequency measurement settings.

Syntax
```
:MEASure:FREQuency?
```

Example
```
:MEASURE:FREQUENCY? -> :MEASURE:FREQUENCY:···
```

Description This command is invalid on models with the optional frequency measurement add-on, because the frequency can be measured simultaneously on all input elements.

:MEASure:FREQuency:ITEM<x>

Function Sets or queries the element whose frequency will be measured.

Syntax
```
:MEASure:FREQuency:ITEM<x> {U<x>|I<x>}
:MEASure:FREQuency:ITEM<x>?
```
The <x> in ITEM<x> = 1 or 2 (Freq.1 to Freq.2)
The <x> in U<x> and I<x> = 1 to 3 (element)

Example
```
:MEASURE:FREQUENCY:ITEM1 U1
:MEASURE:FREQUENCY:ITEM1?
-> :MEASURE:FREQUENCY:ITEM1 U1
```

Description This command is invalid on models with the optional frequency measurement add-on, because the frequency can be measured simultaneously on all input elements.

:MEASure:FUNCtion<x>?

Function Queries all of the settings of the specified user-defined function.

Syntax
```
:MEASure:FUNCtion<x>?
```
<x> = 1 to 8 (F1 to F8)

Example
```
:MEASURE:FUNCTION1? -> :MEASURE:FUNCTION1:···
```

:MEASure:FUNCtion<x>:EXPRession

Function Sets or queries the equation of the specified user-defined function.

Syntax
```
:MEASure:FUNCtion<x>:
EXPRession {<String>}
:MEASure:FUNCtion<x>:EXPRession?
```
<x> = 1 to 8 (F1 to F8)
<String> = Up to 50 characters

Example
```
:MEASURE:FUNCTION1:
EXPRESSION "WH(E1)/TI(E1)*3600"
:MEASURE:FUNCTION1:EXPRESSION?
-> :MEASURE:FUNCTION1:
EXPRESSION "WH(E1)/TI(E1)*3600"
```

:MEASure:FUNCtion<x>[:STATe]

Function Sets or queries the on/off status of the specified user-defined function.

Syntax
```
:MEASure:FUNCtion<x>[:STATe]
{<Boolean>}
:MEASure:FUNCtion<x>:STATe?
```
<x> = 1 to 8 (F1 to F8)

Example
```
:MEASURE:FUNCTION1:STATE ON
:MEASURE:FUNCTION1:STATE?
-> :MEASURE:FUNCTION1:STATE 1
```

:MEASure:FUNCtion<x>:UNIT

Function Sets or queries the unit that is added to the computation result of the specified user-defined function.

Syntax
```
:MEASure:FUNCtion<x>:UNIT {<String>}
:MEASure:FUNCtion<x>:UNIT?
```
<x> = 1 to 8 (F1 to F8)
<String> = Up to 8 characters

Example
```
:MEASURE:FUNCTION1:UNIT "W"
:MEASURE:FUNCTION1:UNIT? -> :MEASURE:
FUNCTION1:UNIT "W"
```

Description This command does not affect the computation result.

:MEASure:MHOLd

Function Sets or queries the on/off status of the MAX HOLD feature used in user-defined functions.

Syntax
```
:MEASure:MHOLd {<Boolean>}
:MEASure:MHOLd?
```

Example
```
:MEASURE:MHOLD ON
:MEASURE:MHOLD? -> :MEASURE:MHOLD 1
```

Description
- The MAX HOLD operation starts when the MAX HOLD feature is specified by a user-defined function and `:MEASure:MHOLd` is set to on.
- When `:MEASure:MHOLd` is set to off, the MAX HOLD operation terminates, and the MAX HOLD value becomes “no data.”
- If `:MEASure:MHOLd` is set to on after having already been set to on before, the MAX HOLD value is reset once, and the MAX HOLD operation starts again.
- For information about specifying the MAX HOLD feature, see the *User’s Manual IM760201-01E*.

:MEASure:PHASe

Function Sets or queries the display format of the phase difference.

Syntax
```
:MEASure:PHASe {<NRf>}
:MEASure:PHASe?
```
<NRf> = 180 or 360

Example
```
:MEASURE:PHASE 180
:MEASURE:PHASE? -> :MEASURE:PHASE 180
```

Description When 180 is selected, the phase is displayed using 0 to ±180° (Lead/Lag). When 360 is selected, the phase is displayed using 0 to 360°.

:MEASure:SFORmula

Function Sets or queries the equation used to compute S (apparent power).

Syntax
```
:MEASure:SFORmula
{RMS|MEAN|DC|MRMS|RMEAN}
:MEASure:SFORmula?
```

Example
```
:MEASURE:SFORMULA RMS
:MEASURE:SFORMULA? -> :MEASURE:
SFORMULA RMS
```

Description The equations that correspond to each option are as follows:

RMS:	S = Urms*Irms
MEAN:	S = Umean*Imean
DC:	S = Udc*Idc
MRMS:	S = Umean*Irms
RMEAN:	S = Urmean*Irmean

:MEASure:SQFormula

Function Sets or queries the equation used to compute S (apparent power) and Q (reactive power).

Syntax
```
:MEASure:SQFormula {TYPE1|TYPE2|TYPE3}
:MEASure:SQFormula?
```

Example
```
:MEASURE:SQFORMULA TYPE1
:MEASURE:SQFORMULA? -> :MEASURE:
SQFORMULA TYPE1
```

Description
- For information about the equations that correspond to TYPE1, 2, and 3, see the *User's Manual IM760201-01E*.
- TYPE3 can only be selected on models with the optional harmonic measurement feature.

:MEASure:SYNChronize

Function Sets or queries the synchronized measurement mode.

Syntax
```
:MEASure:SYNChronize {MASTer|SLAVe}
:MEASure:SYNChronize?
```

Example
```
:MEASURE:SYNCHRONIZE MASTER
:MEASURE:SYNCHRONIZE? -> :MEASURE:
SYNCHRONIZE MASTER
```

5.12 NUMeric Group

The command in this group deal with numeric data output.
There are no front panel keys that correspond to the commands in this group. The NUMERIC key on the front panel is used to make the same settings and queries as the commands in the DISPlay group.

:NUMeric?

Function Queries all numeric data output settings.
Syntax :NUMeric?
Example :NUMERIC? -> :NUMERIC:···

:NUMeric:FORMat

Function Sets or queries the numeric data format.
Syntax :NUMeric:FORMat {ASCii|FLOat}
:NUMeric:FORMat?
Example :NUMERIC:FORMAT ASCII
:NUMERIC:FORMAT? -> :NUMERIC:FORMAT ASCII
Description
- The format of the numeric data that is output varies depending on how this command is set. The different formats are explained below.
  (1) When the format is set to ASCii:
  _Physical values are output in <NR3> format. (However, the elapsed time of integration, TIME, is output in <NR1> format.)
  Each item of data is separated by a comma.
  (2) When the format is set to FLOat:
  A header (for example, "#260" or "#3208") is added in front of each numeric data block. A physical value in IEEE single-precision floating point (4-byte) format follows the header. The byte order of the data of each item is MSB First.
- For the formats of each individual numeric data item, see "Numeric Data Formats" at the end of this group of commands (page 5-48).

:NUMeric:HOLD

Function Sets or queries whether or not all numeric data are held (on) or released (off).
Syntax :NUMeric:HOLD {<Boolean>}
:NUMeric:HOLD?
Example :NUMERIC:HOLD ON
:NUMERIC:HOLD? -> :NUMERIC:HOLD 1
Description
- If :NUMeric:HOLD is turned on before :NUMeric[:NORMal]:VALue? or :NUMeric:LIST:VALue? is executed, all the numeric data at that point in time can be held internally.
- As long as :NUMeric:HOLD is on, numeric data is held even when the numeric data on the screen is updated.
- For example, if you wish to retrieve various types of numeric data from each element at the same point in time, use these commands:
  :NUMeric:HOLD ON
  :NUMeric[:NORMal]:ITEM1 URMS,1;ITEM2 IRMS,1;... (set the numeric data items of element 1)
  :NUMeric[:NORMal]:VALue?
  (receive the numeric data of element 1)
  :NUMeric[:NORMal]:ITEM1 URMS,2;ITEM2 IRMS,2;... (set the numeric data items of element 2)
  :NUMeric[:NORMal]:VALue?
  (receive the numeric data of element 2)
  :NUMeric[:NORMal]:ITEM1 URMS,3;ITEM2 IRMS,3;... (set the numeric data items of element 3)
  :NUMeric[:NORMal]:VALue?
  (receive the numeric data of element 3)
  :NUMeric:HOLD OFF
- If :NUMeric:HOLD is set to on after having already been set to on before, the numeric data is cleared once, and the most recent numeric data is held internally. When retrieving numeric data continuously, this method can be used to circumvent the need to repeatedly set :NUMeric:HOLD to off.

:NUMeric:LIST?

Function Queries all harmonic measurement numeric list output settings.

Syntax `:NUMeric:LIST?`

Example `:NUMERIC:LIST? -> :NUMERIC:LIST:···`

Description
- This command is only valid on models with the optional harmonic measurement feature.
- The number of numeric list items output by `:NUMeric:LIST:ITEM<x>` is determined by `:NUMeric:LIST:NUMber`.

:NUMeric:LIST:CLEar

Function Clears harmonic measurement numeric list output items (sets the items to NONE).

Syntax `:NUMeric:LIST:CLEar {ALL|<NRf>[,<NRf>]}`
ALL = Clear all items
The first <NRf> = 1 to 64 (Item number at which to start clearing)
The second <NRf> = 1 to 64 (Item number at which to stop clearing)

Example `:NUMERIC:LIST:CLEAR ALL`

Description
- This command is only valid on models with the optional harmonic measurement feature.
- If the 2nd <NRf> is omitted, all output items including and after the first <NRf> (up to 64) are cleared.

:NUMeric:LIST:DELete

Function Deletes harmonic measurement numeric list output items.

Syntax `:NUMeric:LIST:DELete {<NRf>[,<NRf>]}`
The first <NRf> = 1 to 64 (Item number at which to start deleting)
The second <NRf> = 1 to 64 (Item number at which to stop deleting)

Example `:NUMERIC:LIST:DELETE 1` (Deletes ITEM1 and shifts ITEM2 and subsequent items forward)
`:NUMERIC:LIST:DELETE 1, 3` (Deletes ITEM1 to ITEM3 and shifts ITEM4 and subsequent items forward)

Description
- This command is only valid on models with the optional harmonic measurement feature.
- The positions of deleted output items are filled by the items that follow them, and empty sections at the end are set to NONE.
- If the 2nd <NRf> is omitted, all output items including and after the first <NRf> are deleted.

:NUMeric:LIST:ITEM<x>

Function Sets or queries the function and element of the specified harmonic measurement numeric list item.

Syntax `:NUMeric:LIST:ITEM<x> {NONE|<Function>,<Element>}`
`:NUMeric:LIST:ITEM<x>?`
<x> = 1 to 64 (item number)
NONE = No output item
<Function> = {U|I|P|S|Q|LAMBda|PHI|PHIU|PHII|UHDF|IHDF|PHDF}
<Element> = {<NRf>|SIGMa} (<NRf> = 1 to 3)

Example `:NUMERIC:LIST:ITEM1 U,1`
`:NUMERIC:LIST:ITEM1? -> :NUMERIC:LIST:ITEM1 U,1`

Description This command is only valid on models with the optional harmonic measurement feature.

:NUMeric:LIST:NUMber

Function Sets or queries the number of numeric list items that are transmitted by :NUMeric:LIST:VALue?.

Syntax `:NUMeric:LIST:NUMber {<NRf>|ALL}`
`:NUMeric:LIST:NUMber?`
<NRf> = 1 to 64 (ALL)

Example `:NUMERIC:LIST:NUMBER 5`
`:NUMERIC:LIST:NUMBER -> :NUMERIC:LIST:NUMBER 5`

Description
- This command is only valid on models with the optional harmonic measurement feature.
- If the parameter is omitted from the `:NUMeric:LIST:VALue?` command, the numeric list data items from 1 to the specified value are output in order.
- By default, the number of numeric data items is set to 1.

:NUMeric:LIST:ORDer

Function Sets or queries the maximum output harmonic order of the harmonic measurement numeric list data.

Syntax `:NUMeric:LIST:ORDer {<NRf>|ALL}`
`:NUMeric:LIST:ORDer?`
<NRf> = 1 to 50 (ALL)

Example `:NUMERIC:LIST:ORDER 50`
`:NUMERIC:LIST:ORDER? -> :NUMERIC:LIST:ORDER 50`

Description This command is only valid on models with the optional harmonic measurement feature.

:NUMeric:LIST:PRESet

Function Presets the harmonic measurement numeric list output item pattern.

Syntax `:NUMeric:LIST:PRESet {<NRf>}`
<NRf> = 1 to 4

Example `:NUMERIC:LIST:PRESET 1`

Description
- This command is only valid on models with the optional harmonic measurement feature.
- For information about the output items that are preset, see "(2) Preset Patterns for Harmonic Measurement Numeric List Output Items" on page 5-51.
- By default, the output items of Pattern 2 are selected.

:NUMeric:LIST:SELect

Function Sets or queries the output component of the harmonic measurement numeric list data.

Syntax `:NUMeric:LIST:SELect {EVEN|ODD|ALL}`
`:NUMeric:LIST:SELect?`

Example `:NUMERIC:LIST:SELECT ALL`
`:NUMERIC:LIST:SELECT? -> :NUMERIC:LIST:SELECT ALL`

Description
- This command is only valid on models with the optional harmonic measurement feature.
- The selectable output component options are explained below:
  EVEN = Outputs the components of TOTal, DC, and even-order harmonics
  ODD = Outputs the components of TOTal, DC, and odd-order harmonics
  ALL = Outputs all components

:NUMeric:LIST:VALue?

Function Queries the harmonic measurement numeric list data.

Syntax `:NUMeric:LIST:VALue? {<NRf>}`
<NRf> = 1 to 64 (item number)

Example
- When <NRf> is specified:
  `:NUMERIC:LIST:VALUE? 1`
  `-> 103.58E+00,0.00E+00,103.53E+00,0.09E+00,2.07E+00,0.04E+00,..(omitted)..,0.01E+00,0.01E+00` (up to 52 items of data)
- When <NRf> is omitted (when `:NUMeric:LIST:NUMber` is set to 5):
  `:NUMERIC:LIST:VALUE? -> 103.58E+00,0.00E+00,103.53E+00,0.09E+00,2.07E+00,0.04E+00,..(omitted)..,0.00E+00,0.00E+00` (up to 52*5 = 260 items of data)
- When `:NUMeric:FORMat` is set to FLOat:
  `:NUMERIC:LIST:VALUE? -> #N` (N-digit byte number; data byte sequence)

Description
- This command is only valid on models with the harmonic measurement feature (/G5 option).
- A single numeric list data item consists of up to 52 items of numeric data in the following order: TOTal, DC, 1st order, ..., `:NUMeric:LIST:ORDer`.
- If <NRf> is specified, only the numeric list data of the item number is output (up to 52 items of data).
- If <NRf> is omitted, the numeric list data of item numbers from 1 to the number specified by the `:NUMeric:LIST:NUMber` command is output in order (the number of data items is up to 52 times the number specified by `:NUMeric:LIST:NUMber`).
- For the formats of each individual numeric data item, see "Numeric Data Formats" at the end of this group of commands, on page 5-48.

:NUMeric:NORMal?

Function Queries all numeric data output settings.
Syntax :NUMeric:NORMal?
Example :NUMERIC:NORMAL? -> :NUMERIC:NORMAL:···
Description The number of numeric data items output by :NUMeric[:NORMal]:ITEM<x> is determined by :NUMeric[:NORMal]NUMber.

:NUMeric[:NORMal]:CLEar

Function Clears numeric data output items (sets the items to NONE).
Syntax :NUMeric[:NORMal]:CLEar {ALL|<NRf>[,<NRf>]}
ALL = Clear all items
The first <NRf> = 1 to 255 (Item number at which to start clearing)
The second <NRf> = 1 to 255 (Item number at which to stop clearing)
Example :NUMERIC:NORMAL:CLEAR ALL
Description If the 2nd <NRf> is omitted, all output items including and after the first <NRf> (up to 255) are cleared.

:NUMeric[:NORMal]:DELete

Function Deletes all numeric data output items.
Syntax :NUMeric[:NORMal]:DELete {<NRf>[,<NRf>]}
The first <NRf> = 1 to 255 (Item number at which to start deleting)
The second <NRf> = 1 to 255 (Item number at which to stop deleting)
Example :NUMERIC:NORMAL:DELETE 1 (Deletes ITEM1 and shifts ITEM2 and subsequent items forward)
:NUMERIC:NORMAL:DELETE 1, 3 (Deletes ITEM1 to ITEM3 and shifts ITEM4 and subsequent items forward)
Description
- The positions of deleted output items are filled by the items that follow them, and empty sections at the end are set to NONE.
- If the 2nd <NRf> is omitted, all output items including and after the first <NRf> are deleted.

:NUMeric[:NORMal]:ITEM<x>

Function Sets or queries the function, element, and harmonic order of the specified numeric data output item.
Syntax :NUMeric[:NORMal]:ITEM<x> {NONE|<Function>,<Element>[,<Order>]}
:NUMeric[:NORMal]:ITEM<x>?
<x> = 1 to 255 (item number)
NONE = No output item
<Function> = {URMS|IRMS|P|S|Q|...}
<Element> = {<NRf>|SIGMa} (<NRf> = 1 to 3)
<Order> = {TOTal|DC|<NRf>} (<NRf> = 1 to 50)
Example :NUMERIC:NORMAL:ITEM1 URMS,1
:NUMERIC:NORMAL:ITEM1? -> :NUMERIC:NORMAL:ITEM1 URMS,1
:NUMERIC:NORMAL:ITEM1 UK,1,1
:NUMERIC:NORMAL:ITEM1? -> :NUMERIC:NORMAL:ITEM1 UK,1,1
Description
- For information about the options available for <Function>, see Function Option List 1 in the DISPlay Group section on page 5-22.
- If <Element> is omitted, the element is set to 1.
- If <Order> is omitted, the order is set to TOTal.
- <Element> and <Order> are omitted from responses to functions that do not need them.

:NUMeric[:NORMal]:NUMber

Function Sets or queries the number of numeric data items that are transmitted by the :NUMeric[:NORMal]:VALue? command.
Syntax :NUMeric[:NORMal]:NUMber {<NRf>|ALL}
:NUMeric[:NORMal]:NUMber?
<NRf> = 1 to 255 (ALL)
Example :NUMERIC:NORMAL:NUMBER 15
:NUMERIC:NORMAL:NUMBER -> :NUMERIC:NORMAL:NUMBER 15
Description
- If the parameter is omitted from the :NUMeric[:NORMal]:VALue? command, the numeric data items from 1 to the specified value are output in order.
- By default, the number of numeric data items is set to 15.

:NUMeric[:NORMal]:PRESet

Function Presets the numeric data output item pattern.
Syntax :NUMeric[:NORMal]:PRESet {<NRf>}
<NRf> = 1 to 4
Example :NUMERIC:NORMAL:PRESET 1
Description
- For information about the output items that are preset, see “(1) Preset Patterns for Numeric Data Items” on page 5-49.
- By default, the output items of Pattern 2 are selected.

:NUMeric[:NORMal]:VALue?

Function Queries the numeric data.

Syntax `:NUMeric[:NORMal]:VALue? {<NRf>}`
<NRf> = 1 to 255 (item number)

Example
- When <NRf> is specified:
  `:NUMERIC:NORMAL:VALUE? 1`
  `-> 103.79E+00`
- When <NRf> is omitted:
  `:NUMERIC:NORMAL:VALUE?`
  `-> 103.79E+00,1.0143E+00,`
  `105.27E+00,..(omitted)..,1.428E+00`
- When :NUMeric:FORMat is set to FLOat:
  `:NUMERIC:NORMAL:VALUE? -> #N` (N-digit byte number; data byte sequence)

Description
- If <NRf> is specified, only the numeric data for the specified item is output.
- If <NRf> is omitted, the numeric data items from 1 to the number specified by the `:NUMeric[:NORMal]:NUMber` command are output in order.
- For the formats of each individual numeric data item, see “Numeric Data Formats” at the end of this group of commands, on page 5-48.

*Numeric Data Formats

(1)Normal Data

- The Σ of electric power values P, S, and Q
- Integrated values WH, WHP, WHM, AH, AHP, AHM, WS, and WQ
- Efficiency values ETA1 and ETA2; harmonic distortion factor values UHDFk, IHDFk, and PHDFk; and distortion factor values UTHD, ITHD, and PTHD.
  ASCII: <NR3> format (mantissa: up to 6 digits, exponent: 2 digits. Example: [-]123.456E+00)
  FLOAT: IEEE single-precision floating point (4-byte) format
- Elapsed time of integration (TIME)
  ASCII: <NR1> format in units of seconds. Example: 3600 for 1 hour (1:00:00).
  FLOAT: IEEE single-precision floating point (4-byte) format in units of seconds. Example: 0x45610000 for 1 hour (1:00:00).
- No items (NONE)
  ASCII: NAN (Not A Number)
  FLOAT: 0x7E951BEE (9.91E+37)
- Other than above
  ASCII: <NR3> format (mantissa: up to 5 digits, exponent: 2 digits. Example: [-]123.45.456E+00)
  FLOAT: IEEE single-precision floating point (4-byte) format

(2)Error Data

- Data does not exist (the display shows “---------”)
  ASCII: NAN (Not A Number)
  FLOAT: 0x7E951BEE (9.91E+37)
- Overrange (the display shows “---O L---”)
- Overflow (the display shows “---O F---”)
- Data over (the display shows “ Error ”)
  ASCII: INF (INFinity)
  FLOAT: 0x7E94F56A (9.9E+37)

Note

- In 180° (Lead/Lag) display, the phase differences φ (PHI) of elements 1 to 3 are output in the range between −180.00 to 180.00 with lead (D) and lag (G) set to negative and positive values, respectively.
- The number of digits in the mantissa of the Σ of power values P, S, and Q may be equal to 6 depending on the combination of the voltage range and current range (e.g. the power range). For more information, see the list of power ranges in the *User’s Manual IM760201-01E*.
- These values always have three digits after the decimal point: efficiency values ETA1 and ETA2; harmonic distortion factor values UHDFk, IHDFk, and PHDFk; and distortion factor values UTHD, ITHD, and PTHD. If the values exceed 100 %, the mantissa will have 6 digits.

*Preset Patterns for Numeric Data Items

The Function Option List in the DISPlay Group section contains a list of the function names used in commands (where the command syntax contains <Function>) and the function names in the WT500 display menu that correspond to them.

Note

This list indicates the measurement function and element that are assigned to each item number (ITEM<x>). Items that are not set to be measured are displayed or output in the same fashion as when the data does not exist. For example, if frequency FI of the current of element 2 is not set to be measured, the output of ITEM19 in pattern 1 is the same as the output when the data does not exist (NAN if the data format is ASCII).

(1)Preset Patterns for Numeric Data Items

These patterns apply to the `:NUMeric[:NORMal]:PRESet` command.

Pattern 1

ITEM<x>	<Function>	<Element>
1	URMS	1
2	IRMS	1
3	P	1
4	S	1
5	Q	1
6	LAMBda	1
7	PHI	1
8	FU	1
9	FI	1
10	NONE	
11 to 19	URMS to FI	2
20	NONE	
21 to 29	URMS to FI	3
30	NONE	
31 to 39	URMS to FI	SIGMA
40	NONE	
41 to 255	NONE	

Pattern 2

ITEM<x>	<Function>	<Element>
1	URMS	1
2	IRMS	1
3	P	1
4	S	1
5	Q	1
6	LAMBda	1
7	PHI	1
8	FU	1
9	FI	1
10	UPPeak	1
11	UMPeak	1
12	IPPeak	1
13	IMPeak	1
14	CFU	1
15	CFI	1
16 to 30	URMS to CFI	2
31 to 45	URMS to CFI	3
46 to 60	URMS to CFI	SIGMA
61 to 255	NONE	

Pattern 3

ITEM<x>	<Function>	<Element>
1	URMS	1
2	IRMS	1
3	P	1
4	S	1
5	Q	1
6	TIME	1
7	WH	1
8	WHP	1
9	WHM	1
10	AH	1
11	AHP	1
12	AHM	1
13	WS	1
14	WQ	1
15	NONE	
16 to 29	URMS to WQ	2
30	NONE	
31 to 44	URMS to WQ	3
45	NONE	
46 to 59	URMS to WQ	SIGMA
60	NONE	
61 to 255	NONE	

Pattern 4

ITEM<x>	<Function>	<Element>
1	URMS	1
2	UMN	1
3	UDC	1
4	URMN	1
5	UAC	1
6	IRMS	1
7	IMN	1
8	IDC	1
9	IRMN	1
10	IAC	1
11	P	1
12	S	1
13	Q	1
14	LAMBda	1
15	PHI	1
16	FU	1
17	FI	1
18	UPPeak	1
19	UMPeak	1
20	IPPeak	1
21	IMPeak	1
22	CFU	1
23	CFI	1
24	TIME	1
25	WH	1
26	WHP	1
27	WHM	1
28	AH	1
29	AHP	1
30	AHM	1
31	WS	1
32	WQ	1
33 to 64	URMS to WQ	2
65 to 96	URMS to WQ	3

97 to 128	URMS to WQ	SIGMA
129 to 255	NONE	

(2) Preset Patterns for Harmonic Measurement Numeric List Output Items

These patterns apply to the :NUMeric:LIST:PRESet command.

Pattern 1

ITEM<x>	<Function>	<Element>
1	U	1
2	I	1
3	P	1
4 to 6	U to P	2
7 to 9	U to P	3
10 to 64	NONE	

Pattern 2

ITEM<x>	<Function>	<Element>
1	U	1
2	I	1
3	P	1
4	PHIU	1
5	PHII	1
6 to 10	U to PHII	2
11 to 15	U to PHII	3
16 to 64	NONE	

Pattern 3

ITEM<x>	<Function>	<Element>
1	U	1
2	I	1
3	P	1
4	S	1
5	Q	1
6	LAMBda	1
7	PHI	1
8	PHIU	1
9	PHII	1
10 to 18	U to PHII	2
19 to 27	U to PHII	3
28 to 64	NONE	

Pattern 4

ITEM<x>	<Function>	<Element>
1	U	1
2	I	1
3	P	1
4	S	1
5	Q	1
6	LAMBda	1
7	PHI	1
8	PHIU	1
9	PHII	1
10	UHDF	1
11	IHDF	1
12	PHDF	1
13 to 24	U to PHDF	2
25 to 36	U to PHDF	3
37 to 64	NONE	

5.13 RATE Group

The command in this group deals with the data update rate.
You can make the same settings and queries that you can make by accessing Update Rate in the Setup Menu that appears when you press SETUP on the front panel.

:RATE

Function Sets or queries the data update rate.

```
Syntax   :RATE {<Time>}
         :RATE?
         <Time> = 100, 200, 500 (ms), 1, 2, or 5 (s)
Example  :RATE 500MS
         :RATE? -> :RATE 500.0E-03
```

5.14 STATus Group

The commands in this group are used to make settings and inquiries related to the status report. There are no front panel keys that correspond to the commands in this group. For information about the status report, see chapter 6.

:STATus?

Function Queries all communication status settings.

Syntax :STATus?

Example :STATUS? -> :STATUS:EESE 0;
FILTER1 NEVER;FILTER2 NEVER;
FILTER3 NEVER;FILTER4 NEVER;
FILTER5 NEVER;FILTER6 NEVER;
FILTER7 NEVER;FILTER8 NEVER;
FILTER9 NEVER;FILTER10 NEVER;
FILTER11 NEVER;FILTER12 NEVER;
FILTER13 NEVER;FILTER14 NEVER;
FILTER15 NEVER;FILTER16 NEVER;
QENABLE 1;QMESSAGE 1

:STATus:CONDition?

Function Queries the contents of the condition register.

Syntax :STATus:CONDition?

Example :STATUS:CONDITION? -> 16

Description For information about the condition register, see chapter 6, “Status Reports.”

:STATus:EESE

(Extended Event Status Enable register)

Function Sets or queries the extended event enable register.

Syntax :STATus:EESE <Register>
:STATus:EESE?
<Register> = 0 to 65535

Example :STATUS:EESE #B0000000000000000
:STATUS:EESE? -> :STATUS:EESE 0

Description For information about the extended event enable register, see chapter 6, “Status Reports.”

:STATus:EESR?

(Extended Event Status Register)

Function Queries the content of the extended event register and clears the register.

Syntax :STATus:EESR?

Example :STATUS:EESR? -> 0

Description For information about the extended event register, see chapter 6, “Status Reports.”

:STATus:ERRor?

Function Queries the error code and message information (top of the error queue).

Syntax :STATus:ERRor?

Example :STATUS:ERROR?
-> 113, "Undefined Header"

Description
- 0, No error is returned when there is no error.
- The message cannot be returned in Japanese.
- You can specify whether or not to add the message using the STATus:QMESsage command.

:STATus:FILTer<x>

Function Sets or queries the transition filter.

Syntax :STATus:FILTer<x> {RISE|FALL|BOTH|NEVer}
:STATus:FILTer<x>?
<x> = 1 to 16

Example :STATUS:FILTER2 RISE
:STATUS:FILTER2? -> :STATUS:FILTER2 RISE

Description
- Set how each bit in the condition register must change to trigger the setting of an event. If a bit is set to RISE, an event is set when the bit changes from 0 to 1.
- For information about the transition filter, see chapter 6, “Status Reports.”

:STATus:QENable

Function Sets or queries whether or not messages other than errors will be stored to the error queue (on/off).

Syntax :STATus:QENable {<Boolean>}
:STATus:QENable?

Example :STATUS:QENABLE ON
:STATUS:QENABLE? -> :STATUS:QENABLE 1

:STATus:QMESsage

Function Sets or queries whether or not message information will be attached to the response to the STATus:ERRor? query (on/off).

Syntax :STATus:QMESsage {<Boolean>}
:STATus:QMESsage?

Example :STATUS:QMESSAGE ON
:STATUS:QMESSAGE? -> :STATUS:QMESSAGE 1

:STATus:SPOLl? (Serial Poll)

Function Executes serial polling.

Syntax :STATus:SPOLl?

Example :STATUS:SPOLL? -> :STATUS:SPOLL 0

5.15 STORe Group

The commands in this group deal with storage.
You can make the same settings and queries that you can make by using STORE and STORE SET (SHIFT+STORE) on the front panel.

:STORe?

Function Queries all numeric data storage settings.
Syntax :STORe?
Example :STORE? -> STORE:···

:STORe:COUNt

Function Sets or queries the store count.
Syntax :STORe:COUNt {<NRf>}
:STORe:COUNt?
<NRf> = 1 to 9999999
Example :STORE:COUNT 100
:STORE:COUNT? -> :STORE:COUNT 100

:STORe:FILE?

Function Queries all settings related to the saving of the data stored in the WT500 to files.
Syntax :STORe:FILE?
Example :STORE:FILE? -> :STORE:FILE:···

:STORe:FILE:ANAMing

Function Sets or queries whether or not files that the stored data will be saved to will be automatically named.
Syntax :STORe:FILE:ANAMing {<Boolean>}
:STORe:FILE:ANAMing?
Example :STORE:FILE:ANAMING ON
:STORE:FILE:ANAMING? -> :STORE:FILE:ANAMING 1

:STORe:FILE:CDIRectory

Function Changes the directory that stored numeric data is saved to.
Syntax :STORe:FILE:CDIRectory {<String>}
<String> = Directory name
Example :STORE:FILE:CDIRECTORY "STORE"
Description Specify ".." to move up to the parent directory.

:STORe:FILE:CONVert?

Function Queries all settings related to the conversion of stored numeric data files into CSV format.
Syntax :STORe:FILE:CONVert?
Example :STORE:FILE:CONVERT? -> :STORE:FILE:CONVERT:···

:STORe:FILE:CONVert:ABORt

Function Stops the conversion of a numeric data file to CSV format.
Syntax :STORe:FILE:CONVert:ABORt
Example :STORE:FILE:CONVERT:ABORT

:STORe:FILE:CONVert:EXECute

Function Converts a stored numeric data file to CSV format.
Syntax :STORe:FILE:CONVert:EXECute {<String>}
<String> = File name
Example :STORE:FILE:CONVERT:EXECUTE "STORE1"
Description • Specify the file name without an extension.
• This command is an overlap command.

:STORe:FILE:CONVert:MODE

Function Sets or queries the conversion mode used to convert stored numeric data files to CSV format.
Syntax :STORe:FILE:CONVert:MODE {AUTO|MANual}
:STORe:FILE:CONVert:MODE?
Example :STORE:FILE:CONVERT:MODE AUTO
:STORE:FILE:CONVERT:MODE? -> :STORE:FILE:CONVERT:MODE AUTO

:STORe:FILE:DRIVe

Function Changes the drive that stored numeric data is saved to.
Syntax :STORe:FILE:DRIVe {RAM|USB[,<NRf>]}
RAM = Internal RAM drive
USB = USB memory drive
<NRf> = Drive number (0 or 1)
Example :STORE:FILE:DRIVE RAM
Description If <NRf> is omitted, the drive will be set to 0 (USB 0-0).

:STORe:FILE:FREE?

Function Queries the free space (in bytes) on the drive that the stored numeric data is saved to.
Syntax :STORe:FILE:FREE?
Example :STORE:FILE:FREE? -> 20912128

:STORe:FILE:NAME

Function Sets or queries the name of the file to which the stored numeric data will be saved.

Syntax
```
:STORe:FILE:NAME {<Filename>}
:STORe:FILE:NAME?
```

Example
```
:STORE:FILE:NAME "STORE1"
:STORE:FILE:NAME? -> :STORE:FILE:
NAME "STORE1"
```

Description
- Set the save destination drive with the `:STORe:FILE:DRIVe` command and the directory with the `:STORe:FILE:CDIRectory` command.
- You can query the path to which stored numeric data is saved to by using the `:STORE:PATH?` command.
- Specify the file name without an extension.

:STORe:FILE:PATH?

Function Queries the absolute path of the directory that the stored numeric data is saved to.

Syntax
```
:STORe:FILE:PATH?
```

Example
```
:STORE:FILE:PATH? -> "USB0-0/STORE"
```

:STORe:INTerval

Function Sets or queries the storage interval.

Syntax
```
:STORe:INTerval {<NRf>,<NRf>,<NRf>}
:STORe:INTerval?
```
The first <NRf> = 0 to 99 (hour)
The second <NRf> = 0 to 59 (minute)
The third <NRf> = 1 to 59 (second)

Example
```
:STORE:INTERVAL 0,0,0
:STORE:INTERVAL? -> :STORE:
INTERVAL 0,0,0
```

:STORe:NUMeric?

Function Queries all numeric data storage settings.

Syntax
```
:STORe:NUMeric?
```

Example
```
:STORE:NUMERIC? -> :STORE:NUMERIC:···
```

:STORe:NUMeric:NORMal?

Function Queries all numeric data storage item settings.

Syntax
```
:STORe:NUMeric:NORMal?
```

Example
```
:STORE:NUMERIC:NORMAL? -> :STORE:
NUMERIC:NORMAL:···
```

:STORe:NUMeric[:NORMal]:ALL

Function Collectively turns on or off the output of all element functions when storing numeric data.

Syntax
```
:STORe:NUMeric[:NORMal]:
ALL {<Boolean>}
```

Example
```
:STORE:NUMERIC[:NORMAL]:ALL ON
```

:STORe:NUMeric[:NORMal]:{ELEMent<x>|SIGMA}

Function Sets or queries the on/off status of the output of each element or of wiring unit Σ when storing numeric data.

Syntax
```
:STORe:NUMeric[:NORMal]:{ELEMent<x>|
SIGMA} {<Boolean>}
:STORe:NUMeric[:NORMal]:{ELEMent<x>|
SIGMA}?
```
<x> = 1 to 3

Example
```
:STORE:NUMERIC:NORMAL:ELEMENT1 ON
:STORE:NUMERIC:NORMAL:ELEMENT1?
-> :STORE:NUMERIC:NORMAL:ELEMENT1 1
```

Description `:STORe:NUMeric[:NORMal]:SIGMA` is valid on models with two or more elements. To turn the output ON, you must create wiring unit Σ beforehand by setting the wiring system using the `[:INPut]WIRing` command.

:STORe:NUMeric[:NORMal]:PRESet<x>

Function Presets the output on/off pattern of the element functions that are used when storing numeric data.

Syntax
```
:STORe:NUMeric[:NORMal]:PRESet<x>
```
<x> = 1 or 2 (preset pattern number)

Example
```
:STORE:NUMERIC:NORMAL:PRESET1
```

Description For information about the preset storage patterns, see the *User's Manual IM760201-01E*.

:STORe:NUMeric[:NORMal]:<Function>

Function Sets or queries the on/off status of function output when storing numeric data.

Syntax
```
:STORe:NUMeric[:NORMal]:
<Function> {<Boolean>}
:STORe:NUMeric[:NORMal]:<Function>?
```
<Function> = {URMS|IRMS|P|S|Q|...}

Example
```
:STORE:NUMERIC:NORMAL:URMS ON
:STORE:NUMERIC:NORMAL:URMS?
-> :STORE:NUMERIC:NORMAL:URMS 1
```

Description For information about the options available for <Function>, see Function Option List 1, in the DISPlay Group section on page 5-22.

:STORe:RESet

Function Resets the numeric data storage feature.

Syntax
```
:STORe:RESet
```

Example
```
:STORE:RESET
```

:STORe:RTIMe?

Function Queries the storage reservation time for real-time storage mode.

Syntax `:STORe:RTIMe?`

Example `:STORE:RTIME? -> :STORE:RTIME:···`

:STORe:RTIMe:{STARt|END}

Function Sets or queries the storage start or end time for real-time storage mode.

Syntax `:STORe:RTIMe:{STARt|END} {<NRf>,<NRf>,<NRf>,<NRf>,<NRf>,<NRf>}`
`:STORe:RTIMe:{STARt|END}?`
{<NRf>, <NRf>, <NRf>, <NRf>, <NRf>, <NRf>} = 2001, 1, 1, 0, 0, 0 to 2099, 12, 31, 23, 59, 59
The first <NRf> = 2001 to 2099 (year)
The second <NRf> = 1 to 12 (month)
The third <NRf> = 1 to 31 (day)
The fourth <NRf> = 0 to 23 (hour)
The fifth <NRf> = 0 to 59 (minute)
The sixth <NRf> = 0 to 59 (second)

Example `:STORE:RTIME:START 2008,1,1,0,0,0`
`:STORE:RTIME:START? -> :STORE:RTIME:START 2008,1,1,0,0,0`

Description The settings made by this command are valid when `:STORe:SMODe` is used to set the mode to real-time storage mode (RTIMe).

:STORe:SMODe

Function Sets or queries the storage mode.

Syntax `:STORe:SMODe {MANual|RTIMe|INTEGrate}`
`:STORe:SMODe?`
MANual = Manual store mode
RTIMe = Real-time store mode
INTEGrate = Integration synchronization store mode

Example `:STORE:SMODE MANUAL`
`:STORE:SMODE? -> :STORE:SMODE MANUAL`

:STORe:STARt

Function Begins the storing of numeric data.

Syntax `:STORe:STARt`

Example `:STORE:START`

Description The storage operation is executed when `:STORe:SMODe` is set to MANual. When `:STORe:SMODe` is set to RTIMe or INTEGrate, the WT500 enters into a storage wait state.

:STORe:STATe?

Function Sets or queries the storage state.

Syntax `:STORe:STATe?`

Example `:STORE:STATE? -> RESET`

Description The response information is as follows:
RESet = Storage reset
READy = Waiting (real-time storage mode)
STARt = Currently storing
STOP = Storage stopped
CONVert = Converting stored data to CSV format

:STORe:STOP

Function Stops the storing of numeric data.

Syntax `:STORe:STOP`

Example `:STORE:STOP`

5.16 SYSTem Group

The commands in this group deal with the system.
You can make the same settings and queries that you can make by using MISC on the front panel.

:SYSTem?

Function Queries all system settings.
Syntax :SYSTem?
Example :SYSTEM? -> :SYSTEM:···

:SYSTem:CLOCk?

Function Queries all date/time settings.
Syntax :SYSTem:CLOCk?
Example :SYSTEM:CLOCK? -> :SYSTEM:CLOCK:DISPLAY 1

:SYSTem:CLOCk:DISPlay

Function Sets or queries the on/off status of the date/time display.
Syntax :SYSTem:CLOCk:DISPlay {<Boolean>}
:SYSTem:CLOCk:DISPlay?
Example :SYSTEM:CLOCK:DISPLAY ON
:SYSTEM:CLOCK:DISPLAY? ->
:SYSTEM:CLOCK:DISPLAY 1

:SYSTem:DATE

Function Sets or queries the date.
Syntax :SYSTem:DATE {<String>}
:SYSTem:DATE?
<String> = “YY/MM/DD” (YY = year, MM = month, DD = day)
Example :SYSTEM:DATE "08/04/01"
:SYSTEM:DATE? -> "08/04/01"
Description For year, enter in the last two digits of the year according to the Gregorian calendar.

:SYSTem:ECLear

Function Clears error messages displayed on the screen.
Syntax :SYSTem:ECLear
Example :SYSTEM:ECLEAR

:SYSTem:FONT

Function Sets or queries the size of the screen display font.
Syntax :SYSTem:FONT [STANdard|LARGe]
:SYSTem:FONT?
Example :SYSTEM:FONT STANDARD
:SYSTEM:KLOCK? -> :SYSTEM:FONT STANDARD
Description This command can be used on WT500s with firmware version 1.02 or later.

:SYSTem:KLOCk

Function Sets or queries the on/off status of the key lock.
Syntax :SYSTem:KLOCk {<Boolean>}
:SYSTem:KLOCk?
Example :SYSTEM:KLOCK OFF
:SYSTEM:KLOCK? -> :SYSTEM:KLOCK 0

:SYSTem:LANGuage?

Function Queries all display language settings.
Syntax :SYSTem:LANGuage?
Example :SYSTEM:LANGUAGE? ->
:SYSTEM:LANGUAGE:···

:SYSTem:LANGuage:MENU

Function Sets or queries the menu language.
Syntax :SYSTem:LANGuage:
MENU {JAPANese|ENGLish}
:SYSTem:LANGuage:MENU?
Example :SYSTEM:LANGUAGE:MENU ENGLISH
:SYSTEM:LANGUAGE:MENU? ->
:SYSTEM:LANGUAGE:MENU ENGLISH

:SYSTem:LANGuage:MESSage

Function Sets or queries the message language.
Syntax :SYSTem:LANGuage:
MESSage {JAPANese|ENGLish}
:SYSTem:LANGuage:MESSage?
Example :SYSTEM:LANGUAGE:MESSAGE ENGLISH
:SYSTEM:LANGUAGE:MESSAGE? ->
:SYSTEM:LANGUAGE:MESSAGE ENGLISH

:SYSTem:MODel?

Function Queries the model.
Syntax :SYSTem:MODel?
Example :SYSTEM:MODEL? -> :SYSTEM:
MODEL "760203"
Description Returns the text that appears in parentheses next to Model in the System Overview display, which can be accessed by pressing MISC.

:SYSTem:SERial?

Function Queries the serial number.
Syntax :SYSTem:SERial?
Example :SYSTEM:SERIAL? -> :SYSTEM:
SERIAL "123456789"
Description Returns the text that appears next to No. in the System Overview display, which can be accessed by pressing MISC.

:SYSTem:SLOCk

Function Sets or queries whether or not the SHIFT-key-on state will be sustained.
Syntax :SYSTem:SLOCk {<Boolean>}
:SYSTem:SLOCk?
Example :SYSTEM:SLOCK OFF
:SYSTEM:SLOCK? -> :SYSTEM:SLOCK 0

:SYSTem:SUFFix?

Function Queries the suffix code.

Syntax :SYSTem:SUFFix?

Example :SYSTEM:SUFFIX? -> :SYSTEM:
SUFFIX "-M/C1/EX3/G5/FQ/7N"

Description Returns the text that appears next to Suffix in the System Overview display, which can be accessed by pressing MISC.

:SYSTem:TIME

Function Sets or queries the time.

Syntax :SYSTem:TIME {<String>}
:SYSTem:TIME?
<String> = “HH:MM:SS” (HH = hour, MM = minute, SS = second)

Example :SYSTEM:TIME "14:30:00"
:SYSTEM:TIME? -> "14:30:00"

:SYSTem:USBKeyboard

Function Sets or queries the USB keyboard type (language).

Syntax :SYSTem:USBKeyboard {JAPANese|ENGLish}
:SYSTem:USBKeyboard?

Example :SYSTEM:USBKEYBOARD JAPANESE
:SYSTEM:USBKEYBOARD? ->
:SYSTEM:USBKEYBOARD JAPANESE

5.17 WAVeform Group

The commands in this group deal with the output of the retrieved waveform display data.
There are no front panel keys that correspond to the commands in this group.

:WAVeform?

Function Queries all waveform display data output settings.
Syntax :WAVeform?
Example :WAVEFORM? -> :WAVEFORM:···

:WAVeform:BYTeorder

Function Sets or queries the output byte order of the waveform display data (FLOAT format) that is transmitted by the :WAVeform:SEND? command.
Syntax :WAVeform:BYTeorder {LSBFirst|MSBFirst}
:WAVeform:BYTeorder?
Example :WAVEFORM:BYTEORDER MSBFIRST
:WAVEFORM:BYTEORDER? -> :WAVEFORM:BYTEORDER MSBFIRST
Description Settings made using this command are valid when :WAVeform:FORMat is set to FLOat.

:WAVeform:END

Function Sets or queries the output end point of the waveform display data that is transmitted by the :WAVeform:SEND? command.
Syntax :WAVeform:END {<NRf>}
:WAVeform:END?
<NRf> = 0 to 1001
Example :WAVEFORM:END 1001
:WAVEFORM:END? -> :WAVEFORM:END 1001

:WAVeform:FORMat

Function Sets or queries the format of the waveform display data that is transmitted by the :WAVeform:SEND? command.
Syntax :WAVeform:FORMat {ASCii|FLOat}
:WAVeform:FORMat?
Example :WAVEFORM:FORMAT FLOAT
:WAVEFORM:FORMAT? -> :WAVEFORM:FORMAT FLOAT
Description For information about the differences in waveform display data output between formats, see the description of the :WAVeform:SEND? command.

:WAVeform:HOLD

Function Sets or queries whether all waveform display data will be held (on) or released (off).
Syntax :WAVeform:HOLD {<Boolean>}
:WAVeform:HOLD?
Example :WAVEFORM:HOLD ON
:WAVEFORM:HOLD? -> :WAVEFORM:HOLD 1
Description
- If :WAVeform:HOLD is turned on before :WAVeform:SEND? is executed, all the waveform data at that point can be held internally.
- As long as :WAVeform:HOLD is on, waveform data is held even when the waveform display on the screen is updated.
- For example, if you wish to retrieve U1 and I1 waveform display data from the same point in time, use these commands:
  :WAVeform:HOLD ON
  :WAVeform:TRACe U1
  :WAVeform:SEND?
  (Receive the waveform display data of U1)
  :WAVeform:TRACe I1
  :WAVeform:SEND?
  (Receive the waveform display data of I1)
  :WAVeform:HOLD OFF
- If :WAVeform:HOLD is set to on after having already been set to on before, the waveform display data is cleared once, and the most recent waveform data is held internally. When retrieving waveform display data continuously, this method can be used to circumvent the need to repeatedly set :WAVeform:HOLD to off.

:WAVeform:LENGth?

Function Queries the total number of points of the waveform specified by the :WAVeform:TRACe command.
Syntax :WAVeform:LENGth?
Example :WAVEFORM:LENGTH? -> 1002
Description The number of data points is fixed. This command always returns 1002.

:WAVeform:SEND?

Function Queries the waveform display data specified by the :WAVeform:TRACe command.

Syntax :WAVeform:SEND?

Example
- When :WAVeform:FORMat is set to ASCii:
  :WAVEFORM:SEND? -> <NR3>,<NR3>,...
- When :WAVeform:FORMat is set to FLOat:
  :WAVEFORM:SEND? -> #N (N-digit byte number; data byte sequence)

Description
- The format of the waveform display data that is output varies depending on how the :WAVeform:FORMat command is set. The different formats are explained below.
  (1) When the format is set to ASCii:
  Physical values are output in <NR3> format. Each item of data is separated by a comma.
  (2) When the format is set to FLOat
  Physical values are output in IEEE single-precision floating point (4-byte) format. The output byte order of the data of each point follows the order that is set by using the :WAVeform:BYTeorder command.

:WAVeform:SRATe?

Function Queries the sample rate of the retrieved waveform.

Syntax :WAVeform:SRATe?

Example :WAVEFORM:SRATE? -> 20.000E+03

:WAVeform:STARt

Function Sets or queries the output start point of the waveform display data that is transmitted by the :WAVeform:SEND? command.

Syntax :WAVeform:STARt {<NRf>}
:WAVeform:STARt?
<NRf> = 0 to 1001

Example :WAVEFORM:START 0
:WAVEFORM:START? -> :WAVEFORM:START 0

:WAVeform:TRACe

Function Sets or queries the target waveform for the :WAVeform:SEND? command.

Syntax :WAVeform:TRACe {U<x>|I<x>}
:WAVeform:TRACe?
<x> = 1 to 3 (element)

Example :WAVEFORM:TRACE U1
:WAVEFORM:TRACE? -> :WAVEFORM:TRACE U1

:WAVeform:TRIGger?

Function Queries the trigger position of the retrieved waveform.

Syntax :WAVeform:TRIGger?

Example :WAVEFORM:TRIGGER? -> 0

Description Because the trigger position is always at the beginning of the waveform display data, 0 is returned.

5.18 Common Command Group

The commands in this group are defined in IEEE488.2-1992 and are independent from the instrument's individual functions. There are no front panel keys that correspond to the commands in this group.

*CAL? (CALibrate)

Function Executes zero calibration (zero-level compensation, the same operation as pressing CAL (SHIFT+SET)) and queries the result.

Syntax `*CAL?`

Example `*CAL? -> 0`

Description If the calibration terminates normally, 0 is returned. If an error is detected, 1 is returned.

*CLS (CLear Status)

Function Clears the standard event register, extended event register, and error queue.

Syntax `*CLS`

Example `*CLS`

Description
- If the `*CLS` command is located immediately after the program message terminator, the output queue is also cleared.
- For information about each register and queue, see chapter 6.

*ESE (standard Event Status Enable register)

Function Sets or queries the standard event enable register.

Syntax `*ESE {<NRf>}`
`*ESE?`
<NRf> = 0 to 255

Example `*ESE 251`
`*ESE? -> 251`

Description
- Specify the value as a decimal format sum of the values of each bit.
- For example, specifying `*ESE 251` will cause the standard enable register to be set to 11111011. In this case, bit 2 of the standard event register is disabled. This means that bit 5 (ESB) of the status byte register is not set to 1, even if a query error occurs.
- The default value is `*ESE 0` (all bits disabled).
- A query using *ESE? will not clear the contents of the standard event enable register.
- For information about the standard event enable register, see page 6-5.

*ESR? (standard Event Status Register)

Function Queries and clears the standard event register.

Syntax `*ESR?`

Example `*ESR? -> 32`

Description
- A sum of the values of each bit is returned in decimal format.
- When an SRQ is generated, you can check what types of events have occurred.
- For example, if a value of 32 is returned, this indicates that the standard event register is set to 00100000. This means that the SRQ occurred due to a command syntax error.
- A query using `*ESR?` will clear the contents of the standard event register.
- For information about the standard event register, see page 6-5.

*IDN? (IDeNtify)

Function Queries the instrument model.

Syntax `*IDN?`

Example `*IDN? -> YOKOGAWA,760203,123456789, F1.01`

Description The information is returned in this form: <Manufacturer>, <Model>, <Serial no.>, <Firmware version>.

*OPC (OPeration Complete)

Function Sets bit 0 (the OPC bit) of the standard event register to 1 upon the completion of the specified overlap command.

Syntax `*OPC`

Example `*OPC`

Description
- For information about how to synchronize a program using `*OPC`, see page 4-8.
- The `COMMunicate:OPSE` command is used to specify the overlap command.
- If `*OPC` is not the last command of the message, its operation is not guaranteed.

*OPC? (OPeration Complete)

Function Returns ASCII code 1 when the specified overlap command is completed.

Syntax `*OPC?`

Example `*OPC? -> 1`

Description
- For information about how to synchronize a program using `*OPC?`, see page 5-7.
- The `COMMunicate:OPSE` command is used to specify the overlap command.
- If `*OPC?` is not the last command of the message, its operation is not guaranteed.

*OPT? (OPTion)

Function Queries the installed options.

Syntax `*OPT?`

Example `*OPT? -> C1,C7,EX3,G5,DT,FQ,V1`

Description
- The command indicates whether the following options are present or absent: GP-IB communication (C1), Ethernet (C7), external sensor input (EX1/EX2/EX3), harmonic measurement (G5), delta computation (DT), frequency measurement (FQ), and VGA output (V1).
- If none of the options is installed, an ASCII-code 0 is returned.
- The `*OPT?` query must be the last query of a program message. An error occurs if there is a query after the `*OPT` query.

*PSC (Power-on Status Clear)

Function Sets or queries whether or not the registers below are cleared when the power is turned on. The register is cleared when its value rounded to an integer is a non-zero value.
- Standard event enable register
- Extended event enable register
- Transition filter

Syntax `*PSC {<NRf>}`
`*PSC?`
<NRf> = 0 (do not clear), non-zero (clear)

Example `*PSC 1`
`*PSC? -> 1`

Description For information about each register, see chapter 6.

*RST (ReSeT)

Function Initializes the settings.

Syntax `*RST`

Example `*RST`

Description
- Also clears `*OPC` and `*OPC?` commands that have been sent.
- All settings except communication settings are reset to their factory default values.

*SRE (Service Request Enable register)

Function Sets or queries the service request enable register value.

Syntax `*SRE {<NRf>}`
`*SRE?`
<NRf> = 0 to 255

Example `*SRE 239`
`*SRE? -> 175`(since the bit 6, MSS, setting is ignored)

Description
- Specify the value as a decimal format sum of the values of each bit.
- For example, specifying `*SRE 239` will cause the standard enable register to be set to 11101111. In this case, bit 4 of the service request enable register is disabled. This means that bit 4 (MAV) of the status byte register is not set to 1, even if the output queue is not empty.
- Bit 6 (MSS) of the status byte register is the MSS bit itself and is therefore ignored.
- The default value is `*SRE 0` (all bits disabled).
- A query using `*SRE?` will not clear the contents of the service request enable register.
- For information about the service request enable register, see page 6-3.

*STB? (STatus Byte)

Function Queries the status byte register value.

Syntax `*STB?`

Example `*STB? -> 4`

Description
- A sum of the values of each bit is returned as a decimal value.
- Because the register is read without executing serial polling, bit 6 is an MSS bit, not an RQS bit.
- For example, if a value of 4 is returned, this indicates that the status byte register is set to 00000100. This means that the error queue is not empty (in other words, an error occurred).
- A query using `*STB?` will not clear the contents of the status byte register.
- For information about the status byte register, see page 6-3.

*TRG (TRiGger)

Function Executes single measurement (the same operation as when SINGLE (SHIFT+HOLD) is pressed).

Syntax `*TRG`

Example `*TRG`

*TST? (TeST)

Function Performs a self-test and queries the result.

Syntax `*TST?`

Example `*TST? -> 0`

Description
- The self-test consists of tests of each kind of internal memory.
- The command returns 0 if the self-test is successful and 1 if it is not. If the condition of the WT500 prevents the self-test from being executed (this happens for example if the WT500 is integrating or storing), the command will return an appropriate error code.
- It takes approximately 30 s for the test to complete. When receiving a response from the WT500, set the timeout to a relatively large value.

*WAI (WAIt)

Function Holds the subsequent command until the completion of the specified overlap operation.

Syntax `*WAI`

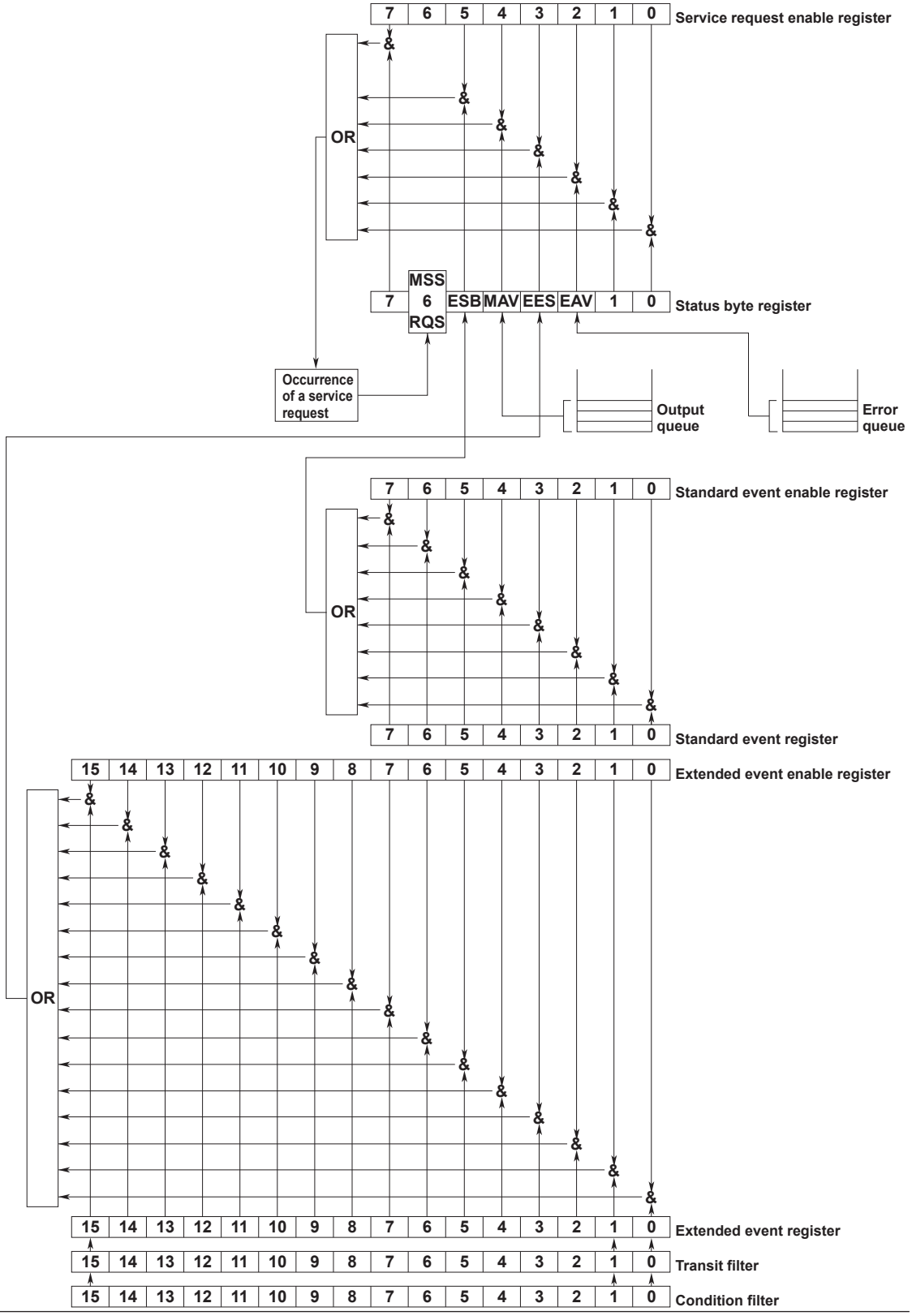
Example `*WAI`

Description
- For information about how to synchronize a program using `*WAI`, see page 4-8.
- The `COMMunicate:OPSE` command is used to specify the overlap command.

6.1 Status Reports

Status Reports

The figure below shows the status report that is read by serial polling. This status report is an extended version of the status report defined in IEEE 488.2-1992.



Overview of the Registers and Queues

Name (Function)	Writing	Reading
Status byte	–	Serial polling (RQS) *STB?(MSS)
Service request enable register (Status byte mask)	*SRE	*SRE?
Standard event register (Changes in device status)	–	*ESR?
Standard event enable register (Standard event register mask)	*ESE	*ESE?
Extended event register (Changes in device status)	–	:STATus:EESR?
Extended event enable register (Extended event register mask)	:STATus:EESE	:STATus:EESE?
Condition register (Current device status)	–	:STATus:CONDition?
Transition filter (Conditions that change the extended event register)	:STATus:FILTer<x>	:STATus:FILTer<x>?
Output queue (Stores a response message to a query)	All query commands	–
Error queue (Stores the error No. and message)	–	:STATus:ERRor?

Registers and Queues That Affect the Status Byte

Registers that affect the bits of the status byte are shown below.

Standard Event Register

Sets bit 5 (ESB) of the status byte to 1 or 0.

Output Queue

Sets bit 4 (MAV) of the status byte to 1 or 0.

Extended Event Register

Sets bit 3 (EES) of the status byte to 1 or 0.

Error Queue

Sets bit 2 (EAV) of the status byte to 1 or 0.

Enable Registers

Registers that are used to mask a bit so that the bit will not affect the status byte even when it is set to 1, are shown below.

Status Byte

Mask the bits using the service request enable register.

Standard Event Register

Mask the bits using the standard event enable register.

Extended Event Register

Mask the bits using the extended event enable register.

Writing/Reading from Registers

The *ESE command is used to set the bits in the standard event enable register to 1's or 0's. The *ESE? command is used to query whether the bits in the standard event enable register are 1's or 0's. For details regarding these commands, see chapter 5.

6.2 Status Byte

Status Byte

Bits 0, 1, and 7
Not used (always 0)

Bit 2 EAV (Error Available)
Set to 1 when the error queue is not empty. In other words, this bit is set to 1 when an error occurs. See the page 6-8.

Bit 3 EES (Extend Event Summary Bit)
Set to 0 when the logical product of the extended event register and the corresponding enable register is 1. In other words, this bit is set to 1 when an event takes place inside the instrument. See the page 6-7.

Bit 4 MAV (Message Available)
Set to 1 when the output queue is not empty. In other words, this bit is set to 1 when there are data to be transmitted. See the page 6-9.

Bit 5 ESB (Event Summary Bit)
Set to 0 when the logical product of the standard event register and the corresponding enable register is 1. In other words, this bit is set to 1 when an event takes place inside the instrument. See the page 6-5.

Bit 6 RQS (Request Service)/MSS (Master Status Summary)
Set to 1 when the logical AND of the status byte excluding Bit 6 and the service request enable register is not 0. In other words, this bit is set to 1 when the instrument is requesting service from the controller.
RQS is set to 1 when the MSS bit changes from 0 to 1, and cleared when serial polling is carried out or when the MSS bit changes to 0.

Bit Masking

If you wish to mask a certain bit of the status byte so that it does not cause a SRQ, set the corresponding bit of the service request enable register to 0. For example, to mask bit 2 (EAV) so that service is not requested when an error occurs, set bit 2 of the service request enable register to 0. This can be done using the `*SRE` command. To query whether each bit of the service request enable register is 1 or 0, use `*SRE?`. For details on the `*SRE` command, see chapter 5.

Operation of the Status Byte

A service request is issued when bit 6 of the status byte becomes 1. Bit 6 is set to 1 when any of the other bits becomes a 1 (when the corresponding bit of the service request enable register is also set to 1).For example, if an event occurs and any of the bits of the logical AND of the standard event register and the corresponding enable register becomes a 1, then bit 5 (ESB) is set to 1. In this case, if bit 5 of the service request enable register is 1, bit 6 (MSS) is set to 1, thus requesting service from the controller. It is also possible to check what type of event has occurred by reading the contents of the status byte.

Reading from the Status Byte

The following two methods are provided for reading the status byte.

Inquiry Using the *STB? Query

Making an inquiry using the `*STB?` query sets bit 6 to MSS. This causes the MSS to be read. After completion of the read-out, none of the bits in the status byte will be cleared.

Serial Polling

Execution of a serial polling changes bit 6 to RQS. This causes RQS to be read. After completion of the read-out, only RQS is cleared. It is not possible to read MSS using serial polling.

Clearing the Status Byte

No method is provided for forcibly clearing all the bits in the status byte. The bits that are cleared for each operation are shown below.

When a Query Is Made Using the *STB? Command

No bits are cleared.

When a Serial Polling Is Executed

Only the RQS bit is cleared.

When a *CLS Command Is Received

When the `*CLS` command is received, the status byte itself is not cleared, but the contents of the standard event register (which affects the bits in the status byte) are cleared. As a result, the corresponding bits in the status byte are cleared, except bit 4 (MAV), since the output queue cannot be emptied by the `*CLS` command. However, the output queue is also cleared if the `*CLS` command is received just after a program message terminator.

6.3 Standard Event Register

Standard Event Register

7	6	5	4	3	2	1	0
PON	URQ	CME	EXE	DDE	QYE	RQC	OPC

Bit 7 PON (Power ON)
Set to 1 when the power is turned ON.

Bit 6 URQ (User Request)
Not used (always 0)

Bit 5 CME (Command Error)
Set to 1 when the command syntax is incorrect.

Example Received a command name with a spelling error or character data not in the selection.

Bit 4 EXE (Execution Error)
Set to 1 when the command syntax is correct but the command cannot be executed in the current state.

Example Received a command with a parameter outside the range or a command dealing with an unsupported option.

Bit 3 DDE (Device Error)
Set to 1 when execution of the command is not possible due to an internal problem in the instrument that is not a command error or an execution error.

Bit 2 QYE (Query Error)
Set to 1 if the output queue is empty or if the data is missing even after a query has been sent.

Example No response data; data is lost due to an overflow in the output queue.

Bit 1 RQC (Request Control)
Not used (always 0)

Bit 0 OPC (Operation Complete)
Set to 1 when the operation designated by the `*OPC` command (see chapter 5) has been completed.

Bit Masking

If you wish to mask a certain bit of the standard event register so that it does not cause bit 5 (ESB) of the status byte to change, set the corresponding bit of the standard event enable register to 0. For example, to mask bit 2 (QYE) so that ESB is not set to 1, even if a query error occurs, set bit 2 of the standard event enable register to 0. This can be done using the `*ESE` command. To query whether each bit of the standard event enable register is 1 or 0, use the `*ESE?`. For details on the `*ESE` command, see chapter 5.

Operation of the Standard Event Register

The standard event register is provided for eight different kinds of event which can occur inside the instrument. Bit 5 (ESB) of the status byte is set to 1 when any of the bits in this register becomes 1 (or when the corresponding bit of the standard event enable register becomes 1).

Example

1. A query error occurs.
2. Bit 2 (QYE) is set to 1.
3. Bit 5 (ESB) of the status byte is set to 1 if bit 2 of the standard event enable register is 1.

It is also possible to check what type of event has occurred inside the instrument by reading the contents of the standard event register.

Reading from the Standard Event Register

The contents of the standard event register can be read by the `*ESR?` command. After the register is read, it is cleared.

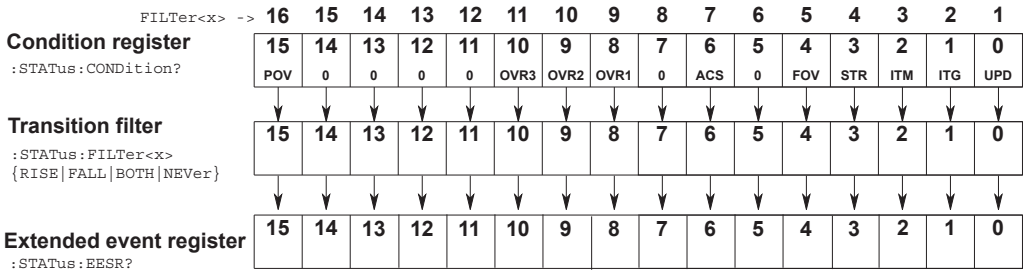
Clearing the Standard Event Register

The standard event register is cleared in the following three cases.

- When the contents of the standard event register are read using the `*ESR?` command.
- When a `*CLS` Command Is Received
- When the instrument is power cycled.

6.4 Extended Event Register

Reading the extended event register tells you whether changes in the condition register (reflecting internal conditions) have occurred. A transition filter can be applied which allows you to decide which events are reported to the extended event register.



The meaning of each bit of the condition register is as follows:

Bit 0 UPD (Updating)
Set to 1 when the measured data is being updated. The falling edge of UPD (1 -> 0) signifies the end of the updating.

Bit 1 ITG (Integrate Busy)
Set to 1 while integration is in progress.

Bit 2 ITM (Integrate Timer Busy)
Set to 1 while the integration timer is running.

Bit 3 STR (Store/Busy)
Set to 1 while storing data.

Bit 4 FOV (Frequency Over)
Set to 1 when the frequency is in error.

Bit 6 ACS (Accessing)
Set to 1 when the internal RAM disk, or USB storage medium drive is being accessed.

Bit 8 OVR1(Element1 Measured Data Over)
Set to 1 when the voltage or current of element 1 is overrange.

Bit 9 OVR2(Element2 Measured Data Over)
Set to 1 when the voltage or current of element 2 is overrange.

Bit 10 OVR3 (Element3 Measured Data Over)
Set to 1 when the voltage or current of element 3 is overrange.

Bit 15 POV (ElementX Input Peak Over)
Set to 1 when peak over (input exceeding the peak) is detected in any of the elements.

The transition filter parameters detect changes in the specified bit (numerical suffix, 1 to 16) of the condition register in the following manner and overwrite the extended event register.

RISE	The specified bit of the extended event register is set to 1 when the bit of the condition register changes from 0 to 1.
FALL	The specified bit of the extended event register is set to 1 when the bit of the condition register changes from 1 to 0.
BOTH	The specified bit of the extended event register is set to 1 when the bit of the condition register changes from 0 to 1 or 1 to 0.
NEVer	Always 0.

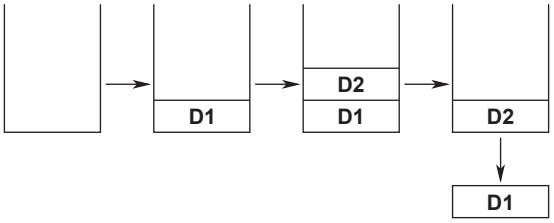
6.5 Output Queue and Error Queue

Output Queue

The output queue is provided to store response messages to queries. For example, if you send the `WAVeform:SEND?` command, which requests the output of acquired data, the data is stored in the output queue until it is read. As shown below, data are stored in order and read from the oldest ones first. The output queue is emptied in the following cases (in addition to when read-out is performed).

- When a new message is received from the controller.
- When a deadlock occurs (see page 4-2).
- When a device clear command (`DCL` or `SDC`) is received.
- When the instrument is power cycled.

The output queue cannot be emptied using the `*CLS` command. To see whether the output queue is empty or not, check bit 4 (MAV) of the status byte.



Error Queue

The error queue stores the error No. and message when an error occurs. For example, if the controller sends an incorrect program message, the error number and message "`113, "Undefined header"`" are stored in the error queue when the error is displayed. The contents of the error queue can be read using the `STATus:ERRor?` query. As with the output queue, the messages are read from the oldest ones first. When the error queue overflows, the last message is replaced by the message "`350, "Queue overflow"`." The error queue is emptied in the following cases (in addition to when read-out is performed).

- When a `*CLS` command is received
- When the instrument is power cycled.

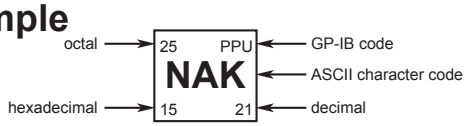
To see whether the error queue is empty or not, check bit 2 (EAV) of the status byte.

Appendix 1 ASCII Character Codes

The following table shows the ASCII character codes.

	0	1	2	3	4	5	6	7
0	0 NUL 0 0	20 DEL 10 16	40 0 SP 20 32	60 16 0 30 48	100 0 @ 40 64	120 16 P 50 80	140 0 ‘ 60 96	160 16 p 70 112
1	1 GTL SOH 1 1	21 LLO DC1 11 17	41 1 ! 21 33	61 17 1 31 49	101 1 A 41 65	121 17 Q 51 81	141 1 a 61 97	161 17 q 71 113
2	2 STX 2 2	22 DC2 12 18	42 2 ” 22 34	62 18 2 32 50	102 2 B 42 66	122 18 R 52 82	142 2 b 62 98	162 18 r 72 114
3	3 ETX 3 3	23 DC3 13 19	43 3 # 23 35	63 19 3 33 51	103 3 C 43 67	123 19 S 53 83	143 3 c 63 99	163 19 s 73 115
4	4 SDC EOT 4 4	24 DCL DC4 14 20	44 4 $ 24 36	64 20 4 34 52	104 4 D 44 68	124 20 T 54 84	144 4 d 64 100	164 20 t 74 116
5	5 PPC ENQ 5 5	25 PPU NAK 15 21	45 5 % 25 37	65 21 5 35 53	105 5 E 45 69	125 21 U 55 85	145 5 e 65 101	165 21 u 75 117
6	6 ACK 6 6	26 SYN 16 22	46 6 & 26 38	66 22 6 36 54	106 6 F 46 70	126 22 V 56 86	146 6 f 66 102	166 22 v 76 118
7	7 BEL 7 7	27 ETB 17 23	47 7 ’ 27 39	67 23 7 37 55	107 7 G 47 71	127 23 W 57 87	147 7 g 67 103	167 23 w 77 119
8	10 GET BS 8 8	30 SPE CAN 18 24	50 8 (28 40	70 24 8 38 56	110 8 H 48 72	130 24 X 58 88	150 8 h 68 104	170 24 x 78 120
9	11 TCT HT 9 9	31 SPD EM 19 25	51 9) 29 41	71 25 9 39 57	111 9 I 49 73	131 25 Y 59 89	151 9 i 69 105	171 25 y 79 121
A	12 LF A 10	32 SUB 1A 26	52 10 * 2A 42	72 26 : 3A 58	112 10 J 4A 74	132 26 Z 5A 90	152 10 j 6A 106	172 26 z 7A 122
B	13 VT B 11	33 ESC 1B 27	53 11 + 2B 43	73 27 ; 3B 59	113 11 K 4B 75	133 27 [5B 91	153 11 k 6B 107	173 27 { 7B 123
C	14 FF C 12	34 FS 1C 28	54 12 , 2C 44	74 28 < 3C 60	114 12 L 4C 76	134 28 \ 5C 92	154 12 l 6C 108	174 28 | 7C 124
D	15 CR D 13	35 GS 1D 29	55 13 - 2D 45	75 29 = 3D 61	115 13 M 4D 77	135 29] 5D 93	155 13 m 6D 109	175 29 } 7D 125
E	16 SO E 14	36 RS 1E 30	56 14 . 2E 46	76 30 > 3E 62	116 14 N 4E 78	136 30 ^ 5E 94	156 14 n 6E 110	176 30 ~ 7E 126
F	17 SI F 15	37 US 1F 31	57 15 / 2F 47	77 UNL ? 3F 63	117 15 O 4F 79	137 UNT _ 5F 95	157 15 o 6F 111	177 DEL (RUBOUT) 7F 127
	Address Command	**Universal Command**	**Listener Address**		**Talker Address**		**Secondary Command**	

Example



Appendix 2 Error Messages

This section explains the communication error messages.

- The messages can be displayed in English or Japanese on the WT500. However, if a messages is queried with the `:STATus:ERRor?` command and displayed on a PC, the message is displayed in English.
- If servicing is required, contact your nearest YOKOGAWA dealer for repairs.
- Only error messages related to communications are listed here. For other error messages, see *User's Manual IM760201-01E*.

Error in Communication Command (100-199)

Code	Messages	Corrective Action	Page
102	Syntax error	Invalid syntax.	Chapter 4, 5
103	Invalid separator	Use a comma to separate the data.	4-1
104	Data type error	See pages 5-5 and 5-6 and write using the correct data form.	4-5 and 4-6
108	Parameter not allowed	Check the number of data points.	4-5, chapter 5
109	Missing parameter	Enter the required data.	4-5, chapter 5
111	Header separator error	Use a space to separate the header and data.	4-1
112	Program mnemonic too long	Check the mnemonic (alphanumerical character string).	Chapter 5
113	Undefined header	Check the header.	Chapter 5
114	Header suffix out of range	Check the header.	Chapter 5
120	Numeric data error	A number is required in the <NRf> form.	4-5
123	Exponent too large	Use a smaller exponent for <NR3> format.	4-5, chapter 5
124	Too many digits	The value must be less than equal to 255 digits.	4-5, chapter 5
128	Numeric data not allowed	Enter in a format other than <NRf> format.	4-5, chapter 5
131	Invalid suffix	Check the unit of the <Voltage>, <Current>, <Time>, and <Frequency>.	4-5
134	Suffix too long	Check the unit of the <Voltage>, <Current>, <Time>, and <Frequency>.	4-5
138	Suffix not allowed	No units are allowed other than <Voltage>, <Current>, <Time>, and <Frequency>.	4-5
141	Invalid character data	Select character data from the selections available in {...|...|...}.	Chapter 5
144	Character data too long	Check the spelling of the character strings in {...|...|...}.	Chapter 5
148	Character data not allowed	Write in a data form other than {...|...|...}.	Chapter 5
150	String data error	Enclose <String> in double quotation or single quotation marks.	4-6
151	Invalid string data	<String> is too long or contains characters which cannot be used.	Chapter 5
158	String data not allowed	Enter in a data format other than <String>.	Chapter 5
161	Invalid block data	<Block data> is not allowed.	4-6, chapter 5
168	Block data not allowed	<Block data> is not allowed.	4-6, chapter 5
171	Invalid expression	Equations cannot be used.	Chapter 5
178	Expression data not allowed	Equations cannot be used.	Chapter 5
181	Invalid outside macro definition	Macro functions defined in IEEE488.2 are not supported.	–

Error in Communication Execution (200 to 299)

Code	Messages	Corrective Action	Page
221	Setting conflict	Check the relevant settings.	Chapter 5
222	Data out of range	Check the range.	Chapter 5
223	Too much data	Check the length of the data.	Chapter 5
224	Illegal parameter value	Check the range.	Chapter 5
241	Hardware missing	Check the installed options.	–
260	Expression error	Equations cannot be used.	–
270	Macro error	Macro functions defined in IEEE488.2 are not supported.	–
272	Macro execution error	Macro functions defined in IEEE488.2 are not supported.	–
273	Illegal macro label	Macro functions defined in IEEE488.2 are not supported.	–
275	Macro definition too long	Macro functions defined in IEEE488.2 are not supported.	–
276	Macro recursion error	Macro functions defined in IEEE488.2 are not supported.	–
277	Macro redefinition not allowed	Macro functions defined in IEEE488.2 are not supported.	–
278	Macro header not found	Macro functions defined in IEEE488.2 are not supported.	–

Error in Communication Query (400 to 499)

Code	Messages	Corrective Action	Page
410	Query INTERRUPTED	Check transmission/reception order.	4-2
420	Query UNTERMINATED	Check transmission/reception order.	4-2
430	Query DEADLOCKED	Limit the length of the program message including <PMT> to 1024 bytes or less.	4-2
440	Query UNTERMINATED after indefinite response	Do not specify a query after the *IDN? or *OPT? command.	–

Error in System Operation (399)

Code	Messages	Corrective Action	Page
399	Fatal error in the communication driver	Servicing required.	–

Warning (50)

Code	Messages	Corrective Action	Page
50	*OPC/? exists in message	Place the *OPC or *OPC? command at the end of the program message.	–

Other Errors (350)

Code	Messages	Corrective Action	Page
350	Queue overflow	Read the error queue.	6-8

Note

Code 350 indicates overflow of error queue. This code is returned as a response to the “STATus: ERRor?” query; it does not appear on the screen.

Appendix 3 IEEE 488.2-1992

The GP-IB interface of the instrument conforms to the IEEE 488.2-1992 Standard. This standard specifies that the following 23 points be stated in the document. This section describes these points.

(1)Of the IEEE 488.1 interface functions, the subsets that are supported
See section 2.3, “GP-IB Interface Specifications.”

(2)The operation of the device when it is assigned an address outside the 0 to 30 range
The address of this instrument cannot be set to an address outside the 0 to 30 range.

(3)Reaction of the device when the user changes the address
The address change occurs when the address is specified using the MISC key menu. The new address is valid until the next time it is changed.

(4)Device settings at power-up. The commands that can be used at power-up.
Basically, the previous settings (i.e. the settings which were valid when power was turned OFF) are valid.
All commands can be used at power-up.

(5)Message exchange options
a) Input buffer size
1024 bytes

b) Queries that return multiple response units
See the example of the commands given in chapter 5.

c) Queries that create response data when the command syntax is being analyzed
All queries create response data when the command syntax is analyzed.

d) Queries that create response data during reception
There are no queries of which the response data are created upon receiving a send request from the controller.

e) Commands that have parameters the restrict one another
See the example of the commands given in chapter 5.

(6)Items that are included in the functional or composite header elements constituting a command
See chapter 4 and 5.

(7)Buffer sizes that affect block data transmission
During block data transmission, the output queue is expanded according to the size.

(8)A list of program data elements that can be used in equations and their nesting limitations
No equations can be used.

(9)Syntax of the responses to queries
See the example of the commands given in chapter 5.

(10) Communication between devices that do not follow the response syntax
None.

(11) Size of the response data block
1 to 308922 bytes

(12) A list of supported common commands
See section 5.18, “Common Command Group.”

(13) Device condition after a successful calibration
Measurement execution condition.

(14) The maximum length of block data that can be used for the *DDT trigger macro definition
Not supported.

(15) The maximum length of the macro label for defining macros, the maximum length of block data that can be used for the macro definition, and the process when recursion is used in macro definitions
Macro functions are not supported.

(16) Reply to the IDN? query
See section 5.18, “Common Command Group.”

(17) The size of the storage area for protected user data for *PUD and *PUD? commands
*PUD and *PUD? are not supported.

(18) The length of the *RDT and *RDT? resource names
*RDT and *RDT? are not supported.

(19) The change in the status due to *RST, *LRN?, *RCL, and *SAV
*RST
See section 5.18, “Common Command Group.”

*LRN?, *RCL, and *SAV
These common commands are not supported.

(20) The extent of the self-test using the *TST? command
Executes all the MEMORY tests (each internal memory) of the Self Test menu of the MISC key.

(21) The structure of the extended return status
See chapter 6.

(22) Whether each command is processed in an overlap fashion or sequentially
See section 4.5, “Synchronization with the Controller” and chapter 5.

(23) The description of the execution of each command
See the functions of each command in chapter 5, *User’s Manual IM760201-01E*.

Index

Symbols

Numeric

A

B

C

D

E

F

G

H

I

K

L

M

N

O

P

Q

R

S

T

U

V

W

Z