UNI-T®

Instruments.uni-trend.com



Programming Manual

UTE300 Series Three-Phase Digital Power Meter

REV 1.01

30th March, 2026

Warranty and Statement

Copyright

Copyright © 2025 Uni-Trend Technology (China) Co., Ltd.

Trademark

UNI-T is the registered trademark of Uni-Trend Technology (China) Co., Ltd.

File No.

UTE30020260330

Statement

- UNI-T products are protected by patents (including obtained and pending) in China and other countries and regions.
- UNI-T reserves the right to change specifications and prices.
- The information provided in this manual supersedes all previous publications.
- Information provided in this manual is subject to change without prior notice.
- UNI-T shall not be liable for any errors that may be contained in this manual, or for any incidental or consequential damage arising out of the use of the information and functions provided in this manual.
- Without the written permission of UNI-T, this manual cannot be photocopied, reproduced, or adapted.

Product Certification

UNI-T has certified that the product conforms to China national product standards and industry product standards, as well as ISO9001:2008 and ISO14001:2004 standards. UNI-T will further certify the product to meet the standards of other members of the International Organization for Standardization.

1. Communication Interface

1.1 USB Interface

The user can send commands to the power meter via the USB interface to perform functions corresponding to the front-panel keys.

After receiving a command, the power meter returns the measured and calculated data, configuration parameters, status byte, and error codes.

For USB interface settings, refer to Section 8.5 *USB Interface*, in the user manual for more details.

Note: The user does not need to configure the USB interface parameters on the power meter. Do not use other communication interfaces to control the power meter while using the USB interface.

Table 1.1 USB Interface Specifications

Port Number	1
Connector	Type-B connector (plug)
Electrical and Mechanical Specifications	USB 2.0
Transmission Mode	Full speed
Protocol	USBTMC-USB488
PC Requirements	Windows 10, Windows 8, Windows 7 (32-bit/64-bit), Windows Vista (32-bit), Windows XP (SP2 or later, 32-bit)

1.2 GPIB Interface

The user can send commands to the power meter via the GPIB interface to perform functions corresponding to the front-panel keys.

After receiving a command, the power meter returns the measured and calculated data, configuration parameters, status byte, and error codes.

Table 1.2 GPIB Interface Specifications

Applicable Equipment	National Instruments PCI-GPIB or PCI-GPIB+; PCIe-GPIB or PCIe-GPIB+ PCMCIA-GPIB or PCMCIA-GPIB+ (not supported on Windows Vista or later) GPIB-USB-HS requires the NI-488.2M driver, version 2.8.1 or later.
Electrical and Mechanical Specifications	IEEE Std 488-1978 (JIS C 1901-1987)

1.2.1 GPIB Setting

Each GPIB device has a unique address used to identify it on the bus. Therefore, before using the GPIB interface of the power meter, the user must first set the GPIB address.

For GPIB address settings, refer to Section 7.5 *GPIB Interface*. For connection settings, refer to 8.3 *GPIB Setting*, in the user manual for more details.

1.3 RS-232 Interface

The user can send commands to the power meter via the RS-232 interface to perform functions corresponding to the front-panel keys.

After receiving a command, the power meter returns the measured and calculated data, configuration parameters, status byte, and error codes.

Table 1.3 RS-232 Interface Specifications

Port Number	D-Sub 9-pin (plug)
Electrical and Mechanical Specifications	EIA-574 (RS-232, 9-pin)
Baud Rate	1200, 2400, 4800, 9600, 19200, 57600, 115200
Data Bit	7-bit or 8-bit
Parity Bit	None, odd, even
Stop Bit	1-bit or 2-bit

The RS-232 interface supports SCPI, Modbus RTU, and Modbus ASCII communication protocols. When

communicating via the RS-232 interface, ensure that parameters such as the communication protocol and baud rate are correctly configured.

For RS-232 settings, refer to Section 7.4 *Serial Port*. For connection settings, refer to 8.2 *RS-232 Interface*, in the user manual for more details.

1.4 Ethernet Interface

The user can send commands to the power meter via the Ethernet interface to perform functions corresponding to the front-panel keys.

After receiving a command, the power meter returns the measured and calculated data, configuration parameters, status byte, and error codes.

Table 1.4 Ethernet Interface Specifications

Port Number	1
Connector Type	RJ-45 connector
Electrical and Mechanical Specifications	IEEE802.3
Transmission System	Ethernet (100BASE-TX, 10BASE-T)
Transmission Rate	Maximum 100 Mbps
Communication Protocol	TCP/IP (Raw Socket, VXI-11,Modbus/TCP)
Supported Service	DHCP, remote control

The Ethernet interface supports SCPI and Modbus TCP commands. The control port for Raw Socket is fixed at 5025, and this port supports SCPI commands. And the control port for Modbus TCP commands is fixed at 502.When communicating via the Ethernet interface, ensure that the IP address, subnet mask, and gateway are correctly configured.

For Ethernet settings, refer to Section 7.3 *Network Settings*. For connection settings, refer to 8.4 *Ethernet Interface*, in the user manual for more details.

2. Programming Overview

2.1 Message

Messages facilitate communication between the controller and the power meter.

Messages sent by the controller are command messages, and messages sent by the power meter are response messages.

Upon receiving an inquiry command, the power meter immediately returns a response message. Each response message corresponds to a single command message.

2.1.1 Command Message

The format of a command message is shown in Figure 2.1.

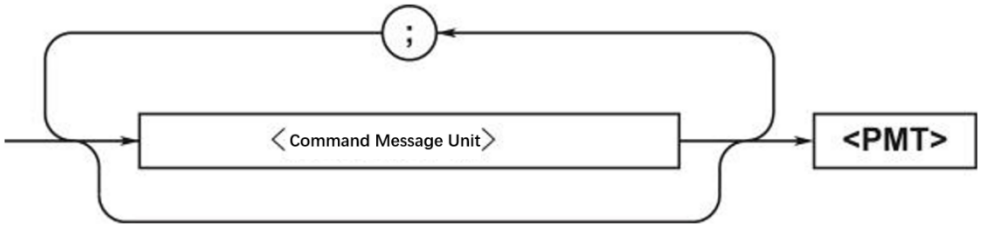


Figure 2.1 Command Format

- Command Message Unit

  A command message consists of one or more command message units. Each unit corresponds to a single command. Command message units are separated by a semicolon (";"), and the power meter executes the first command received before processing subsequent units.

  The syntax of command message units is shown in Figure 2.2, and an example is shown in Figure 2.3.

  The command header specifies the command type, while the command data contains the command parameters. The command data is separated from the command header by a space. For commands with multiple data parameters, a comma separates each parameter. Examples of command headers and command data are shown in Figure 2.4.

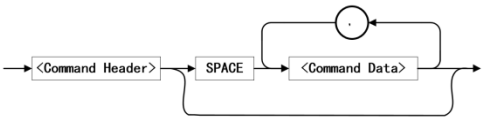


Figure 2.2 Command Unit Syntax



Figure 2.3 Command Unit Syntax Example

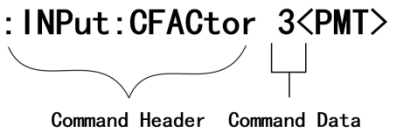


Figure 2.4 Example of Command Header and Command Data

- PMT

  PMT indicates the end of a command message. There are three types of terminators:

  NL (New Line): Same as LF (line feed).ASCII code "0AH".

  ^END: The END message as defined by IEEE 488.1.The data byte that is sent with the END message is the last data byte of the program message.

  NL^END: NL with an END message attached.

2.1.2 Response Command

The syntax of a response message is shown in Figure 2.5.

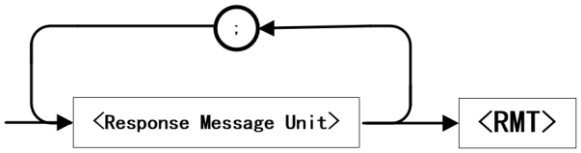


Figure 2.5 Response Command Syntax

- Response Message Unit

  A response message consists of one or more response message units, each separated by a semicolon, as shown in Figure 2.6.

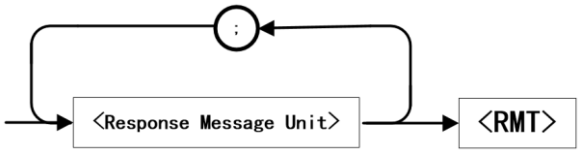


Figure 2.6 Response Command Unit Example

The syntax of response message units is shown in Figure 2.7.

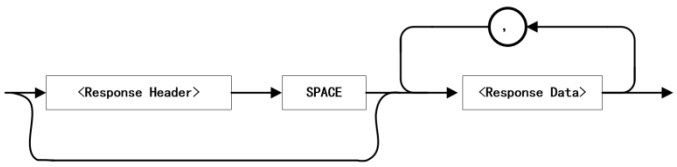


Figure 2.7 Syntax of Response Command Unit

The response header precedes the response data and is separated from it by a space. The response data contains the content of the response, and multiple sets of response data are separated by commas (,).

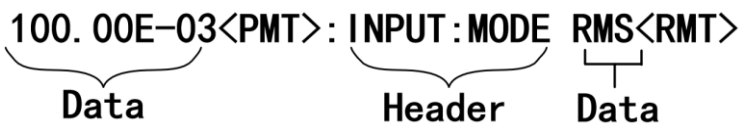


Figure 2.8 Example of Response Header and Response Data

If a command message contains multiple query requests, the power meter responds only to the first query. In most cases, a single query generates a single response. However, some queries may return multiple responses. In such cases, the first response corresponds to the first query, but the Nth response does not necessarily correspond to the Nth query. Therefore, to ensure that every response is received, a command message should contain only one query request.

- RMT

  RMT is the terminator of a response message and is represented by NL^END.

2.1.3 Notes

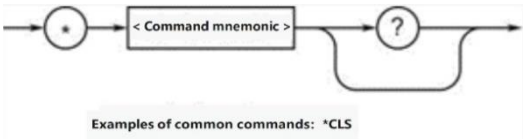
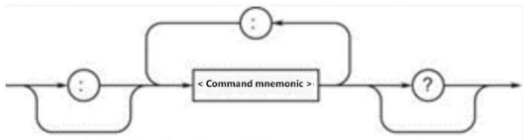
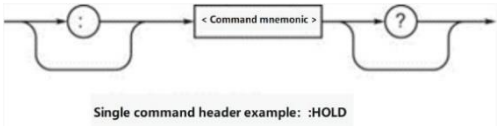
1. If the controller sends a command message without a query, another command message can be sent at any time afterward.
2. If the controller sends a command message containing a query, the next command message can only be sent after all response messages have been received; otherwise, an error will occur.
3. If the controller receives a response that does not exist, an error will occur.
4. If the controller receives a response when no command message has been sent, an error will occur.
5. If the controller sends an incomplete command message, an error will occur.

2.2 Command

2.2.1 Type

According to the format of the command header, commands sent by the controller to the power meter can be classified into three types, as listed in Table 2.1.

Table 2.1 Command Header Format

Command Header Format	Description
	Commands defined by the standard IEEE 488.2-1992 are common commands. The format of a common command is shown in the figure on the left, and an asterisk (*) must precede the common command.
	Some special commands are organized hierarchically according to their function. Different levels are separated by a colon (:). The format and examples are shown in the figure on the left.
	Other commands are functionally independent, and their command headers use a simple, non-hierarchical format, as shown in the figure on the left.

2.2.2 Command Connection

1. Connecting Different Command Groups

   When two different command groups are used in a command message, they must be separated by a colon (:). The colon cannot be omitted.

   For example, :INTEGrate:MODE NORMal;:INPut:MODE RMS<PMT>

2. Single Command Following Another Command

   If a single command follows another command, a colon (:) must precede the single command.

   For example, :INTEGrate:MODE NORMal;:HOLD ON<PMT>

3. Connecting Common Commands

   Common commands are defined in IEEE 488.2-1992. When using a common command, a colon (:) is not required before the command.

   For example, :INTEGrate:MODE NORMal;*CLS;:INTEGrate:TIMer 1,0,0<PMT>

4. Using <PMT> to Separate Commands

   When <PMT> is used to separate two commands, two command messages are sent. In this case, the command header cannot be omitted, even if both commands are in the same command group and share the same command header.

2.2.3 Upper-Level Request

Adding a question mark (?) after the highest-level command in a command group allows the controller to query information provided by all lower-level query commands within that group.

For example, :INTEGrate?<PMT>->:INTEGRATE:MODE NORMAL;TIMER 0,0,0; ; STATE START<RMT>

The response from the upper-level query can also be returned to the controller as a command message, enabling the previously returned settings to be re-validated.

Note: Not all information in a command group is necessarily returned. For example, in some upper-level query requests, setting information that is not currently in use will not be included in the response.

2.2.4 Writing of Command Header

1. Command mnemonics are not case-sensitive. For example, INPut can be written as input or INPUT.
2. Lowercase letters in the command mnemonic can be omitted. For example, INPut can be written as INPu or INP.
3. A question mark (?) at the end of a command header indicates a query. The question mark cannot be omitted. For example, INPut? can be abbreviated as INP?.
4. If a numerical value following the command mnemonic is omitted, it is assumed to be 1. For example, ELEMent written as ELEM is interpreted as ELEMent1.
5. Commands or parameters enclosed in brackets [ ] can be omitted. For example,

[:INPut]:SCALing[:STATe] ON can be written as SCAL ON. However, for upper-level commands, the command or parameter in the last bracket cannot be omitted. For example, "SCALing?" is different from "SCALing:STATe?".

2.3 Response

The response message returned by the power meter can take two forms:

1. Response message with a command header

   This type of response consists of a header and data. Because it includes the command header, it can be used directly as a command message.

   For example, :INTEGrate:MODE?<PMT> -> :INTEGRATE:MODE NORMAL<RMT>

2. Response message with data only

   This type contains data only and does not include a command header. Therefore, it cannot be used directly as a command message.

   For example, INTEGrate:STATe?<PMT> -> RESET<RMT>

2.3.1 Command Header Requirement

If the response message returned by the power meter does not need to include a command header, the user can configure this behavior using the COMMunicate:HEADer command.

2.3.2 Abbreviated Form of Responses

The user can enable or disable the abbreviated form of responses using the COMMunicate:VERBose command. This setting controls whether the returned command headers and data include lowercase segments.

Portions of the command header enclosed in square brackets [ ] are also omitted in the abbreviated form.

2.4 Data Format

The data follows the command header and is separated from the header by a space. The data contains the relevant conditions and values, as listed in Table 2.2.

Table 2.2 Data Format

Data Type	Description	Example
<Decimal>	Value expressed in decimal format	VT ratio setting ->[:INPut]:SCALing:VT 100
<Voltage><Current><Time>	Physical quantity values	Voltage range setting ->[:INPut]:VOLTage:RANge 150V
<Register>	Register values, which can be expressed in binary, octal, decimal, or hexadecimal	Extended event register ->:STATUS:EESE #HFE
<Character data>	Predefined character string	Measurement mode ->[:INPut]:MODE {RMS|VMEan|DC}
<Boolean>	Indicates ON/OFF status. Usually ON, OFF, or a numeric value	Set hold mode ->:HOLD ON
<String data>	Arbitrary character string	Return model information ->:SYSTEM:MODEL "UTE310"
<Block data>	Multiple 8-bit data values	Measured data in response message -> #40012ABCDEFGHIJKL

2.4.1 Multiplier and Symbols

The user can apply multipliers and symbols as listed in Table 2.3.

Multiplier symbols are not case-sensitive.

Table 2.3 Multiplier and Symbols

Symbol	Multiplier
EX	10^{18}
PE	10^{15}

T	10^{12}
G	10^{9}
MA	10^{6}
K	10^{3}
M	10^{-3}
U	10^{-6}
N	10^{-9}
P	10^{-12}
F	10^{-15}

2.4.3 Units

The units used by the power meter are listed in Table 2.4.

Table 2.4 Units

Symbol	Unit	Description
V	Volt	Voltage
A	Ampere	Current
S	Second	Time

If both the multiplier and unit are omitted, the basic unit is used by default (V, A, S).

Response messages returned by the power meter typically use the format <NR3> and do not include multipliers or units, instead using the basic unit format.

2.4.4 <Decimal>

<Decimal> indicates that the data is expressed in decimal format.

In ANSI X3.42-1975, the prefix NR is used to denote different decimal formats, as listed in Table 2.5.

Table 2.5 Decimal Value

Symbol	Description	Example

<NR1>	Integer	125, -1, +100
<NR2>	Fixed-point	125.0, -.90, +001.
<NR3>	Floating-point	125.0E+0, -9E-1, +.1E4
<NRf>	Any of <NR1> to <NR3>	

Notes:

1. The power meter can receive decimal values sent by the controller in the formats <NR1> to <NR3>.
2. The format of <Decimal> data returned by the power meter is the same as the format used in the command request.
3. In the <NR3> format, the plus sign + can be omitted, but the minus sign - must be included.
4. If the input data exceeds the allowed range, the value will be recognized to the nearest valid range limit.
5. If the input contains more valid digits than supported, the excess digits will be discarded.

2.4.5 Register

<Register> indicates that the data is an integer and can be expressed in binary, decimal, octal, or hexadecimal format.

When each bit of an integer has a specific meaning, <Register> can be used to represent the data.

Table 2.6 Register Data

Format	Example
<NRf>	1
#H<Hexadecimal format, using digits 0-9 and A-F>	#H0F
#Q<Octal format, using digits 0-7>	#Q777
#B<Binary format, using digits 0-1>	#B001100

The <Register> format is not case-sensitive, and the numerical value of <Register> in a response message typically uses the <NR1> (integer) format.

2.4.6 <Character Data>

<Character Data> is a character string with a specific meaning, typically used to represent an operation or a

selectable function parameter.

The writing format for <Character Data> is described in Section 2.2.4 *Writing of Command Header*.

2.4.7 <Boolean>

<Boolean> data indicates the ON/OFF state. It can be ON, OFF, or an integer.

If <Boolean Data> uses the integer format, a value rounded to 0 is considered OFF, and any other value is considered ON.

Response messages typically return 1 for ON and 0 for OFF.

2.4.8 <String Data>

<String Data> differs from <Character Data> in that it has no predefined meaning and can contain any character string.

<String Data> must be enclosed in single or double quotes, as listed in Table 2.7.

Table 2.7 <String Data>

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

2.4.9 <Block Data>

<Block Data> consists of 8-bit data and is used only in response messages.

The syntax of <Block Data> is listed in Table 2.8.

Table 2.8 Syntax

Format	Example
#N<Data byte count><Data byte sequence>	#800000010ABCDEFGHIJ

- #N indicates the length of the data bytes in <Block Data>.

  For example, in #800000010ABCDEFGHIJ, the 8 in #8 represents the length of the byte count field (00000010). 00000010 specifies the number of data bytes, which is 10 in decimal.

  The following byte sequence ABCDEFGHIJ contains the actual data bytes.

- Byte count specifies the number of data bytes, expressed in decimal format.

- Byte sequence contains the actual data values, e.g., ABCDEFGHIJ.

2.5 Synchronization Between the Instrument and the Controller

The controller refers to a device that can communicate with and control the instrument, typically a PC.

There are two types of commands: overlapping commands and sequential commands.

Overlapping commands: Execution of the next command may begin before the previous command has finished.

Sequential commands: Execution of the next command will not start until the previous command has completed, even if multiple commands are sent consecutively.

Note: All commands for this instrument are sequential commands.

Even when only sequential commands are available, synchronization may be required to correctly retrieve measured data.

For example, if the user wishes to retrieve the latest measured value each time the data is updated, this can be attempted by sending the NUMeric[:NORMal]:VALue? command at fixed time intervals. However, the instrument will return only the currently measured data, regardless of whether it has been updated since the last query. This approach may return results identical to previous data.

If this occurs, the user must use a method to synchronize with the end of the measured data update.

Querying Using STATus:CONDition?

The STATus:CONDition? command is used to query the contents of the condition register(Refer to Section 7.4).

The user can determine whether measurement data is being updated by reading Bit 0 of the condition register:

Bit 0 = 1 → Measurement data is currently being updated.

Bit 0 = 0 → Data update has completed, and the measurement data can be safely queried.

Note: For this instrument, using STATus:CONDition? to determine whether measurement data is being updated is not practical, because the duration during which Bit 0 remains 1 is extremely short.

Using the Extended Event Register and COMMunicate:WAIT

Changes in the status register are reflected in the extended event register.

The COMMunicate:WAIT command is used to wait for a specific event to occur in the extended event register.

Examples

:STATus:FILTer1 FALL; :STATus:EESR?<PMT> *(Reads the response to :STATus:EESR?)*

Loop

:COMMunicate:WAIT 1<PMT>

:STATus:EESR?<PMT> *(Reads the response to :STATus:EESR?)*

:NUMeric[:NORMal]:VALue?<PMT> *(Reads the response to :NUMeric[:NORMal]:VALue?)*

Go to Loop

The command :STATus:FILTer1 FALL configures the transition filter so that Bit 0 of the extended event register is set to 1 when Bit 0 of the condition register changes from 1 to 0 (i.e., when the measurement data update is completed).

The command :STATus:EESR? is used to read and clear the extended event register.

The command :COMMunicate:WAIT 1 command instructs the program to wait until Bit 0 of the extended event register becomes 1.

The command :NUMeric[:NORMal]:VALue? will not be executed until Bit 0 of the extended event register is set to 1.

3. SCPI Set

3.1 D/A Output Command

This command is used to set the D/A output.

3.1.1 AOUTput?

Function: Queries all D/A output settings.

Syntax:

:AOUTput?

3.1.2 AOUTput[:NORMal]:PRESet

Function: Restores the D/A output settings to their default values.

Syntax:

:AOUTput[:NORMal]:PRESet {NORMal|INTEGrate}

Examples:

:AOUTPUT:NORMAL:PRESET NORMAL

3.1.3 AOUTput[:NORMal]:CHANnel<x>

Function: Sets or queries the D/A output function parameters.

Syntax:

:AOUTput[:NORMal]:CHANnel<x> {NONE|<Function>[,<Element>]}

:AOUTput[:NORMal]:CHANnel<x>?

<x> = 1 to 12 (Output channel)

NONE = no output item

<Function>

={U|I|P|S|Q|LAMBda|PHI|FU|FI|WH|WHP|WHM|AH|AHP|AHM|MATH|UPeak|IPeak|CFU|CFI|AVP|EFFiciency}

<Element> = {<NRf>|SIGMa} (<NRf> = 1 to 3)

Examples:

:AOUTPUT:NORMAL:CHANNEL1 U,1

:AOUTPUT:NORMAL:CHANNEL1? -> :AOUTPUT:NORMAL:CHANNEL1 U,1

3.1.4 AOUTput[:NORMal]:IRTime

Function: Sets or queries the integral time of the D/A output.

Syntax:

:AOUTput[:NORMal]:IRTime {<NRf>,<NRf>,<NRf>}

:AOUTput[:NORMal]:IRTime?

{<NRf>,<NRf>,<NRf>} = 0,0,0 to 10000,0,0

First <NRf> = 0 to 10000 (Hours)

Second <NRf> = 0 to 59 (Minutes)

Third <NRf> = 0 to 59 (Seconds)

Examples:

:AOUTPUT:NORMAL:IRTIME 1,0,0

:AOUTPUT:NORMAL:IRTIME? -> :AOUTPUT:IRTIME 1,0,0

3.1.5 AOUTput[:NORMal]:MODE<x>

Function: Sets or queries the mode of the D/A output.

Syntax:

:AOUTput[:NORMal]:MODE<x> {FIXed|MANual|COMPare}

:AOUTput[:NORMal]:MODE<x>?

<x> = 1 to 12 (Output channel)

Examples:

:AOUTPUT:NORMAL:MODE1 FIXED

:AOUTPUT:NORMAL:MODE1? -> :AOUTPUT:NORMAL:MODE1 FIXED

- FIXed (Fixed Range, Default)

  When the measured value reaches the rated value of the selected measurement function, the D/A output is set to +5 V.

- MANual (Manual Mode)

  In manual mode, the relationship between the displayed measurement value and the D/A output voltage (–5 V to +5 V) can be freely configured.

  Allows user-defined scaling between measurement values and output voltage. Supports both zoom-in and zoom-out adjustment of the D/A output range for each channel.

- COMPare (Compare Mode)

  The instrument compares the measured value with predefined comparator limits and outputs a corresponding analog voltage: +5 V, 0 V, or -5 V.

  The output analog voltage can be used to drive external devices, such as relays.

3.1.6 AOUTput[:NORMal]:RATE<x>

Function: Sets or queries the maximum and minimum value of the D/A output in manual mode.

Sets or queries the upper and lower limits of the D/A output in comparator mode.

Syntax:

:AOUTput[:NORMal]:RATE<x> {<NRf>,<NRf>}

:AOUTput[:NORMal]:RATE<x>?

<x> = 1 to 12 (Output channel)

<NRf> = -9.999E+12 to 9.999E+12

Examples:

:AOUTPUT:NORMAL:RATE1 100,-100

:AOUTPUT:NORMAL:RATE1? -> :AOUTPUT:NORMAL:RATE1 100.0E+00,-100.0E+00

- Manual Mode: When the D/A output is set to the manual mode (:AOUTput[:NORMal]:MODE<x> MANual): The rated maximum output is set to +5V. The rated minimum output to -5V.
- Comparator Mode: When the D/A output is set to the comparator mode (:AOUTput[:NORMal]:MODE<x> COMPare): Sets the upper and lower output limits.
- Fixed Range Mode: When the D/A output is set to the fixed range mode (:AOUTput[:NORMal]:MODE<x> FIXed): No rated value needs to be set. These values do not affect the output.

3.2 Communication Command

This command is used to set the communication settings.

This command corresponds to a function that cannot be executed from the power meter's front panel.

3.2.1 :COMMunicate?

Function: Queries all communication settings.

Syntax:

:COMMunicate?

3.2.2 :COMMunicate:HEADer

Function: Sets or queries whether the query response includes the command header.

Syntax:

:COMMunicate:HEADer {<Boolean>}

:COMMunicate:HEADer?

Examples:

:COMMUNICATE:HEADER ON

:COMMUNICATE:HEADER? -> :COMMUNICATE:HEADER 1

Response message with command header: :INPUT:VOLTAGE:RANGE 150.0E+00

Response message without command header: 150.0E+00

3.2.3 :COMMunicate:REMote

Function: Sets the communication mode of the power meter to either remote or local.

Syntax:

:COMMunicate:REMote {<Boolean>}

:COMMunicate:REMote?

Examples:

:COMMUNICATE:REMOTE ON

:COMMUNICATE:REMOTE? -> :COMMUNICATE:REMOTE 1

3.2.4 :COMMunicate:STATus?

Function: Queries the communication status of the RS-232 interface. When a communication event occurs, the corresponding status bit is set. The bit is cleared (set to 0) after being read.

Syntax:

:COMMunicate:STATus?

Examples:

:COMMUNICATE:STATUS? -> 0

The meanings of the status bits are as follows:

Bit	RS232
0	Parity error
1	Frame error
2	Incorrect characters detected
3 or higher	Always 0

3.2.5 COMMunicate:VERBose

Function: Sets or queries the response command format, either abbreviated or complete.

Syntax:

:COMMunicate:VERBose {<Boolean>}

:COMMunicate:VERBose?

Examples:

:COMMUNICATE:VERBOSE ON

:COMMUNICATE:VERBOSE? -> :COMMUNICATE:VERBOSE 1

Complete response command format: :INPUT:VOLTAGE:RANGE 150.0E+00

Abbreviated response command format: :VOLT:RANG 150.0E+00

3.2.6 :COMMunicate:WAIT

Function: Waits for the specified event to occur.

Syntax:

:COMMunicate:WAIT <Register>

<Register> = 0 to 65535 (Extended event register)

Examples:

:COMMUNICATE:WAIT 1

3.2.7 :COMMunicate:WAIT?

Function: The query returns a response when the specified event occurs.

Syntax:

:COMMunicate:WAIT? <Register>

<Register> = 0 to 65535 (Extended event register)

Examples:

:COMMUNICATE:WAIT? 65535 -> 1

3.3 Harmonic Command

This command is used to set the harmonic measurement settings.

3.3.1 :HARMonics?

Function: Queries all harmonic measurement settings.

Syntax:

:HARMonics?

3.3.2 :HARMonics:MODE

Function: Sets the mode of harmonic measurement to Normal or IEC.

Syntax:

:HARMonics:MODE {NORMal|IEC}

:HARMonics:MODE?

Examples:

:HARMonics:MODE NORMal

:HARMonics:MODE? -> :HARMONICS:MODE NORMAL

NORMal: Normal mode

IEC: IEC mode

3.3.3 :HARMonics:PLLSource

Function: Sets or queries the PLL source.

Syntax:

:HARMonics:PLLSource {U|I}

:HARMonics:PLLSource?

Examples:

:HARMONICS:PLLSOURCE U1

:HARMONICS:PLLSOURCE? -> :HARMONICS:PLLSOURCE U1

3.3.4 :HARMonics:ORDer

Function: Sets or queries the maximum and minimum harmonic orders.

Syntax:

:HARMonics:ORDer {<NRf>,<NRf>}

:HARMonics:ORDer?

First <NRf> = 1 (Minimum harmonic order, fixed at 1)

Second <NRf> = 1 to 50 (Maximum order)

Examples:

:HARMONICS:ORDER 1,50

:HARMONICS:ORDER? -> :HARMONICS:ORDER 1,50

3.3.5 :HARMonics:THD

Function: Sets or queries the THD (Total Harmonic Distortion) formula.

Syntax:

:HARMonics:THD {TOTal|FUNDamental}

:HARMonics:THD?

Examples:

:HARMONICS:THD FUNDAMENTAL

:HARMONICS:THD? -> :HARMONICS:THD FUNDAMENTAL

3.4 Hold Command

This command is used to perform the data hold function.

3.4.1 :HOLD

Function: Sets or queries the state of data hold.

Syntax:

:HOLD {<Boolean>}

:HOLD?

Examples:

:HOLD OFF

:HOLD? -> :HOLD 0

3.5 Input Command

This command is used to execute functions related to the input measurement conditions.

3.5.1 :INPut?

Function: Queries all input settings.

Syntax:

:INPut?

3.5.2 [:INPut]:CFACtor

Function: Sets or queries the crest factor.

Syntax:

[:INPut]:CFACtor {<NRf>|A6}

[:INPut]:CFACtor?

<NRf> = 3,6

Examples:

:INPUT:CFACTOR 3

:INPUT:CFACTOR? -> :INPUT:CFACTOR 3

3.5.3 [:INPut]:WIRing

Function: Sets or queries the wiring configuration.

Syntax:

[:INPut]:WIRing {P1W3|P3W3|P3W4|V3A3}

[:INPut]:WIRing?

P1W3 = Single-phase, three-wire

P3W3 = Three-phase, three-wire

P3W4 = Three-phase, four-wire

V3A3 = Three-voltage, three-current

Examples:

:INPUT:WIRING P3W3

:INPUT:WIRING? -> :INPUT:WIRING P3W3

3.5.4 [:INPut]:MODE

Function: Sets or queries the voltage and current measurement mode.

Syntax:

[:INPut]:MODE {RMS|VMEan|DC}

[:INPut]:MODE?

Examples:

:INPUT:MODE RMS

:INPUT:MODE? -> :INPUT:MODE RMS

3.5.5 [:INPut]:VOLTage?

Function: Queries all voltage settings.

Syntax:

[:INPut]:VOLTage?

3.5.6 [:INPut]:VOLTage:RANGe

Function: Sets or queries the voltage range of all input units.

Syntax:

[:INPut]:VOLTage:RANGe {<Voltage>}

[:INPut]:VOLTage:RANGe?

When the peak factor is set to **3**:

<Voltage> =15, 30, 60,150, 300, 600V

When the peak factor is set to **6** or **6A**:

<Voltage> = 7.5, 15,30, 75, 150, 300V

Examples:

:INPUT:VOLTAGE:RANGE 600V

:INPUT:VOLTAGE:RANGE? -> :INPUT:VOLTAGE:RANGE 600.0E+00

3.5.7 [:INPut]:VOLTage:AUTO

Function: Sets or queries the state of automatic voltage range.

Syntax:

[:INPut]:VOLTage:AUTO {<Boolean>}

[:INPut]:VOLTage:AUTO?

Examples:

:INPUT:VOLTAGE:AUTO ON

:INPUT:VOLTAGE:AUTO? -> :INPUT:VOLTAGE:AUTO 1

3.5.8 [:INPut]:VOLTage:CONFig

Function: Sets or queries the allowable step range under automatic voltage range.

Syntax:

[:INPut]:VOLTage:CONFig {ALL|<Voltage>[,<Voltage>][,<Voltage>]...}

[:INPut]:VOLTage:CONFig?

ALL = All ranges are allowable

<Voltage> range, refer to the command :INPut:VOLTage:RANGe

Examples:

:INPUT:VOLTAGE:CONFIG ALL

:INPUT:VOLTAGE:CONFIG? -> :INPUT:VOLTAGE:CONFIG ALL

:INPUT:VOLTAGE:CONFIG 600,150,15

:INPUT:VOLTAGE:CONFIG? ->:INPUT:VOLTAGE:CONFIG 600.0E+00,150.0E+00,15.0E+00

Select the enabled voltage range using the parameter. To allow all ranges, set the parameter to "ALL".

3.5.9 [:INPut]:VOLTage:POJump

Function: Sets or queries the target voltage range to which the instrument switches when the voltage peak exceeds the current range.

Syntax:

[:INPut]:VOLTage:POJump {OFF|<Voltage>}

[:INPut]:VOLTage:POJump?

OFF = Automatic jump to the target voltage range is disabled

<Voltage> range, refer to the command :INPut:VOLTage:RANGe

Examples:

:INPUT:VOLTAGE:POJUMP 600V

:INPUT:VOLTAGE:POJUMP? -> :INPUT:VOLTAGE:POJUMP 600.0E+00

3.5.10 [:INPut]:CURRent?

Function: Queries all electrical parameters of current measurement.

Syntax:

[:INPut]:CURRent?

3.5.11 [:INPut]:CURRent:RANGe

Function: Sets or queries the current range.

Syntax:

[:INPut]:CURRent:RANGe {<Current>|(EXTernal,<Voltage>)}

[:INPut]:CURRent:RANGe?

- For direct current input,

  When the peak factor is set to 3:

  <Current> = 0.5, 1, 2, 5, 10, 20(A)

  When the peak factor is set to 6 or 6A:

  <Current> = 0.25, 0.5, 1, 2.5, 5, 10(A)

- For external current sensor input,

  When the peak factor is set to 3:

  < Voltage > = 2.5, 5, 10(V)(/EX1)

  < Voltage > = 50, 100, 200, 500(mV), 1, 2(V)(/EX2)

  When the peak factor is set to 6 or 6A:

  <Voltage> = 1.25, 2.5, 5(V)(/EX1)

  <Voltage> = 25, 50, 100, 250(mV), 0.5, 1(V)(/EX2)

Examples:

:INPUT:CURRENT:RANGE 20A

:INPUT:CURRENT:RANGE? -> :INPUT:CURRENT:RANGE 20.0E+00

:INPUT:CURRENT:RANGE EXTERNAL,10V

:INPUT:CURRENT:RANGE? ->:INPUT:CURRENT:RANGE EXTERNAL,10.0E+00

3.5.12 [:INPut]:CURRent:AUTO

Function: Sets or queries the state of automatic current range.

Syntax:

[:INPut]:CURRent:AUTO {<Boolean>}

[:INPut]:CURRent:AUTO?

Examples:

:INPUT:CURRENT:AUTO ON

:INPUT:CURRENT:AUTO? -> :INPUT:CURRENT:AUTO 1

3.5.13 [:INPut]:CURRent:CONFig

Function: Sets or queries the allowable step range under automatic current range.

Syntax:

[:INPut]:CURRent:CONFig {ALL|<Current>[,<Current>][,<Current>]...}

[:INPut]:CURRent:CONFig?

ALL = All ranges are allowable

<Current> range, refer to the command :INPut:CURRent:RANGe

Examples:

:INPUT:CURRENT:CONFIG ALL

:INPUT:CURRENT:CONFIG? -> :INPUT:CURRENT:CONFIG ALL

:INPUT:CURRENT:CONFIG 20,5,1

:INPUT:CURRENT:CONFIG? -> :INPUT:CURRENT:CONFIG 20.0E+00,5.0E+00,1.0E+00

Select the enabled current range using the parameter. To allow all ranges, set the parameter to "ALL".

3.5.14 [:INPut]:CURRent:POJump

Function: Sets or queries the target current range to which the instrument switches when the current peak exceeds the current range.

Syntax:

[:INPut]:CURRent:POJump {OFF|<Current>}

[:INPut]:CURRent:POJump?

OFF = Automatic jump to the target current range is disabled

<Current> range, refer to the command :INPut:CURRent:RANGe

Examples:

:INPUT:CURRENT:POJUMP 20A

:INPUT:CURRENT:POJUMP? -> :INPUT:CURRENT:POJUMP 20.0E+00

3.5.15 [:INPut]:CURRent:EXTSensor<x>:CONFig

Function: Sets or queries the allowable jump range under the automatic range for the external current sensor.

Syntax:

[:INPut]:CURRent:EXTSensor<x>:CONFig {ALL|<Voltage>[,<Voltage>][,<Voltage>]...}

[:INPut]:CURRent:EXTSensor<x>:CONFig?

<x> = 1,2 (EXT1,EXT2)

ALL = All ranges are allowable

<Voltage> range, refer to the command :INPut:CURRent:RANGe

Examples:

:INPUT:CURRENT:EXTSENSOR1:CONFIG ALL

:INPUT:CURRENT:EXTSENSOR1:CONFIG? -> :INPUT:CURRENT:EXTSENSOR1:CONFIG ALL

:INPUT:CURRENT:EXTSENSOR1:CONFIG 2.5,5

:INPUT:CURRENT:EXTSENSOR1:CONFIG? -> 5.0E+00,2.5E+00

3.5.16 [:INPut]:CURRent:EXTSensor<x>:POJump

Function: Sets or queries the target current range to which the instrument switches when the current peak of the external current sensor exceeds the current range.

Syntax:

[:INPut]:CURRent:EXTSensor<x>:POJump {OFF|<Voltage>}

[:INPut]:CURRent:EXTSensor<x>:POJump?

<x> = 1,2 (EXT1,EXT2)

OFF = Range jump is disabled

<Voltage> range, refer to the command :INPut:CURRent:RANGe

Examples:

:INPUT:CURRENT:EXTSENSOR2:POJUMP 2V

:INPUT:CURRENT:EXTSENSOR2:POJUMP? -> :INPUT:CURRENT:EXTSENSOR2:POJUMP 2.00E+00

3.5.17 [:INPut]:CURRent:SRATio?

Function: Queries all conversion ratios of the external current sensor for all input units.

Syntax:

[:INPut]:CURRent:SRATio?

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

Function: Sets all conversion ratios of the external current sensor for all input units.

Syntax:

[:INPut]:CURRent:SRATio[:ALL]{<NRf>}

<NRf> = 0.001 to 9999.999

Examples:

:INPUT:CURRENT:SRATIO:ALL 10

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

Function: Sets or queries conversion ratios of the external current sensor for the specified input unit.

Syntax:

[:INPut]:CURRent:SRATio:ELEMent<x>{<NRf>}

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

<x> = 1 to 3 (Input unit)

<NRf> = 0.001 to 9999

Examples:

:INPUT:CURRENT:SRATIO:ELEMENT1 10

:INPUT:CURRENT:SRATIO:ELEMENT1? -> :INPUT:CURRENT:SRATIO:ELEMENT1 10.00

3.5.20 [:INPut]:RCONfig

Function: Sets or queries the state of range configuration.

Syntax:

[:INPut]:RCONfig{<Boolean>}

[:INPut]:RCONfig?

Examples:

:INPUT:RCONFIG OFF

:INPUT:RCONFIG? -> :INPUT:RCONFIG 0

The following commands are available when this command is set to ON:

:INPut:VOLTage:CONFig;

:INPut:VOLTage:POJump;

:INPut:CURRent:CONFig;

:INPut:CURRent:POJump;

:INPut:CURRent:EXTSensor<x>:CONFig;

:INPut:CURRent:EXTSensor<x>:POJump

3.5.21 [:INPut]:SCALing?

Function: Queries all scale settings.

Syntax:

[:INPut]:SCALing?

3.5.22 [:INPut]:SCALing[:STATe]

Function: Sets or queries the state of conversion scale.

Syntax:

[:INPut]:SCALing[:STATe]{<Boolean>}

[:INPut]:SCALing:STATe?

Examples:

:INPUT:SCALING:STATE OFF

:INPUT:SCALING:STATE? -> :INPUT:SCALING:STATE 0

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

Function:

Queries the VT conversion scale, CT conversion scale, and power factor for all input units.

Syntax:

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

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

Function: Sets the VT conversion scale, CT conversion scale, and power factor for all input units globally.

Syntax:

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

<NRf> = 0.001 to 9999.

Examples:

:INPUT:SCALING:VT:ALL 1

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

Function: Sets the VT conversion scale, CT conversion scale, and power factor for the specified input unit.

Syntax:

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

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

<x> = 1 to 3 (Number of input unit)

<NRf> = 0.001 to 9999

Examples:

:INPUT:SCALING:VT:ELEMENT11

:INPUT:SCALING:VT:ELEMENT1? -> :INPUT:SCALING:VT:ELEMENT11.000

3.5.26 [:INPut]:SYNChronize

Function: Sets or queries the synchronization source.

Syntax:

[:INPut]:SYNChronize {VOLTage|CURRent|OFF}

Examples:

:INPUT:SYNCHRONIZE VOLTAGE

:INPUT:SYNCHRONIZE? -> :INPUT:SYNCHRONIZE VOLTAGE

3.5.27 [:INPut]:FILTer?

Function: Queries all input filter settings.

Syntax:

[:INPut]:FILTer?

3.5.28 [:INPut]:FILTer:LINE

Function: Sets or queries the line filter.

Syntax:

[:INPut]:FILTer:LINE {<Boolean>}

[:INPut]:FILTer:LINE?

Examples:

:INPUT:FILTER:LINE OFF

:INPUT:FILTER:LINE? -> :INPUT:FILTER:LINE 0

The command [:INPut]:FILTer:LINE:BWidth is available when it is set to ON.

3.5.29 [:INPut]:FILTer:LINE:BWidth

Function: Sets or queries the bandwidth of the line filter.

Syntax:

[:INPut]:FILTer:LINE:BWidth {<Bandwidth>}

[:INPut]:FILTer:LINE:BWidth?

<Bandwidth> = 500,1k,2k,3k,4k,5k,5.5k,6k,7k,8k,9k,10k,20k,30k,40k,50k,60k,70k,80k,100k(Hz)

Examples:

:INPut:FILTer:LINE:BWidth 5500

:INPut:FILTer:LINE:BWidth? -> :INPut:FILTer:LINE:BWidth 5.5E+03

When the <Bandwidth> = 500, the line filter uses the analog filter.

When the <Bandwidth> = 1k to 100k, the line filter uses the digital filter.

3.5.30 [:INPut]:FILTer:FREQuency

Function: Sets or queries the frequency filter.

Syntax:

[:INPut]:FILTer:FREQuency {<Boolean>}

[:INPut]:FILTer:FREQuency?

Examples:

:INPUT:FILTER:FREQUENCY OFF

:INPUT:FILTER:FREQUENCY? -> :INPUT:FILTER:FREQUENCY 0

3.5.31 [:INPut]:POVer?

Function: Queries the peak overrange status.

Syntax:

[:INPut]:POVer?

Examples:

:INPUT:POVER? -> 0

The peak overrange condition for each input unit is listed in the table below.

The response message returns the decimal number corresponding to the sum of the bit values. For example, a value of 16 in the response message indicates that a peak overrange has occurred in U3.

Bit	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
Status			I3	U3	I2	U2	I1	U1

3.5.32 [:INPut]:CRANge?

Function: Queries the current overrange status.

Syntax:

[:INPut]:CRANge?

Examples:

:INPUT:CRANGE? -> 0

The current overrange status is listed in the table below.

The response message returns the decimal number corresponding to the sum of the bit values. For example, a value of 64 in the response message indicates that a current overrange has occurred.

Bit	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
Status	AP	AO	AH	AL	VP	VO	VH	VL

VL:Voltage at automatic decrease range condition

VH: Voltage at automatic increase range condition

VO: Voltage overrange

VP: Voltage peak overrange

AL: Current at automatic decrease range condition

AH: Current at automatic increase range condition

AO: Current overrange

AP: Current peak overrange

3.6 Integration Command

3.6.1 :INTEGrate?

Function: Queries all integration settings.

Syntax:

:INTEGrate?

3.6.2 :INTEGrate:MODE

Function: Sets or queries the integration mode.

Syntax:

:INTEGrate:MODE {NORMal|CONTinuous}

:INTEGrate:MODE?

NORMal = Standard integration mode

CONTinuous = Continuous integration mode

Examples:

:INTEGRATE:MODE NORMAL

:INTEGRATE:MODE? -> :INTEGRATE:MODE NORMAL

3.6.3 :INTEGrate:TIMer

Function: Sets or queries the integration timer.

Syntax:

:INTEGrate:TIMer {<NRf>,<NRf>,<NRf>}

:INTEGrate:TIMer?

{<NRf>,<NRf>,<NRf>} = 0,0,0 to 10000,0,0

First <NRf> = 0 to 10000 (Hours)

Second <NRf> = 0 to 59 (Minutes)

Third <NRf> = 0 to 59 (Seconds)

Examples:

:INTEGRATE:TIMER 1,0,0

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

3.6.4 :INTEGrate:STARt

Function: Starts the integration.

Syntax:

:INTEGrate:STARt

Examples:

:INTEGRATE:START

3.6.5 :INTEGrate:STOP

Function: Stops the integration.

Syntax:

:INTEGrate:STOP

Examples:

:INTEGRATE:STOP

3.6.6 :INTEGrate:RESet

Function: Resets the integration.

Syntax:

:INTEGrate:RESet

Examples:

:INTEGRATE:RESET

3.6.7 :INTEGrate:STATe?

Function: Queries the integration state.

Syntax:

:INTEGrate:STATe?

Examples:

:INTEGRATE:STATE? -> RESET

RESet = Resets the integration

STARt = Starts the integration

STOP = Stops the integration

ERRor = Integration errors (e.g., data overflows or power interruption)

TIMeup = Integration stopped due to timeout

3.7 Math Operation Command

3.7.1 :MATH

Function: Sets or queries the mathematical operation expression.

Syntax:

:MATH {ADD|SUB|MUL|DIV|DIVA|DIVB}

Examples:

:MATH ADD

:MATH? -> :MATH ADD

Mathematical operation expression:

ADD: A＋B

SUB: A－B

MUL: A×B

DIV: A/B

DIVA: A/B^2

DIVB: A^2/B

A and B are determined by the commands :MATH:ITEMA and :MATH:ITEMB

3.7.2 :MATH:ITEMA

Function: Sets or queries the A parameter.

Syntax:

:MATH:ITEMA {<Function>,<Element>}

:MATH:ITEMA?

<Function> = {U|I|P|S|Q|PF}

<Element> = {<NRf>|SIGMa}　(<NRf> = 1 to 3)

Examples:

:MATH:ITEMA U,1

:MATH:ITEMA? -> :MATH:ITEMA U,1

3.7.3 :MATH:ITEMB

Function: Sets or queries the B parameter.

Syntax:

:MATH:ITEMB {<Function>,<Element>}

:MATH:ITEMB?

<Function> = {U|I|P|S|Q|PF}

<Element> = {<NRf>|SIGMa} (<NRf> = 1 to 3)

Examples:

:MATH:ITEMB P,1

:MATH:ITEMB? -> :MATH:ITEMB P,1

3.8 Measurement Command

3.8.1 :MEASure?

Function: Queries all output settings for measurement and calculated data.

Syntax:

:MEASure?

3.8.2 :MEASure:AVERaging?

Function: Queries all averaging settings.

Syntax:

:MEASure:AVERaging?

3.8.3 :MEASure:AVERaging[:STATe]

Function: Sets or queries the state of the averaging function.

Syntax:

:MEASure:AVERaging[:STATe]{<Boolean>}

:MEASure:AVERaging:STATe?

Examples:

:MEASURE:AVERAGING:STATE ON

:MEASURE:AVERAGING:STATE? -> :MEASURE:AVERAGING:STATE 1

3.8.4 :MEASure:AVERaging:TYPE

Function: Sets or queries the averaging type.

Syntax:

:MEASure:AVERaging:TYPE {LINear|EXPonent}

:MEASure:AVERaging:TYPE?

Examples:

:MEASURE:AVERAGING:TYPE LINEAR

:MEASURE:AVERAGING:TYPE? -> :MEASURE:AVERAGING:TYPE LINEAR

The average function can be enabled in harmonic measurement, but it is only available when the averaging type is set to exponential average (EXPonent).

3.8.5 :MEASure:AVERaging:COUNt

Function: Sets or queries the average factor.

Syntax:

:MEASure:AVERaging:COUNt {<NRf>}

:MEASure:AVERaging:COUNt?

<NRf> = 8, 16, 32, 64 (moving average constant and exponential average)

Examples:

:MEASure:AVERAGING:COUNT 8

:MEASure:AVERAGING:COUNT? -> :MEASure:AVERAGING:COUNT 8

The average function can be enabled in harmonic measurement, but it is only available when the averaging type is set to exponential average (EXPonent).

3.8.6 :MEASure:MHOLd

Function: Sets the state of maximum hold.

Syntax:

:MEASure:MHOLd {<Boolean>}

:MEASure:MHOLd?

Examples:

:MEASURE:MHOLD ON

:MEASURE:MHOLD? -> :MEASURE:MHOLD 1

3.9 Numeric Command

This command is used to set output settings for numerical values.

3.9.1 :NUMeric?

Function: Queries all output settings for numerical values.

Syntax:

:NUMeric?

3.9.2 :NUMeric:FORMat

Function: Sets or queries the output format for numerical values. Refer to *Numerical Data Format* for more details.

ASCII: All physical quantity values are output in <NR3> format, except for the integration time parameter TIME, which is output in <NR1> format. Data items are separated by commas.

FLOat: A header may precede each numeric data block, such as "#240" or "#3208". The header is followed by physical quantities in IEEE standard single-precision floating-point (4-byte) format. The byte order within the data places the MSB (Most Significant Byte) before the LSB (Least Significant Byte).

Syntax:

:NUMeric:FORMat {ASCii|FLOat}

:NUMeric:FORMat?

Examples:

:NUMERIC:FORMAT ASCII

:NUMERIC:FORMAT? -> :NUMERIC:FORMAT ASCII

3.9.3 :NUMeric:NORMal?

Function: Queries all output settings for numerical data.

Syntax:

:NUMeric:NORMal?

The number of output data items for the command :NUMeric:ITEM<x> is determined by the command :NUMeric:NUMber.

3.9.4 :NUMeric[:NORMal]:VALue?

Function: Queries the numerical data. For the format of each numerical data, refer to *Numerical Data Format* for more details.

Syntax:

:NUMeric[:NORMal]:VALue? {<NRf>}

<NRf> = 1 to 255

If a <NRf> value is specified, only the specified numerical data is output. If <NRf> is omitted, the instrument outputs numerical data from 1 to N, where N is determined by the command :NUMeric[:NORMal]:NUMber.

Examples:

When the **<NRf>** is specified:

:NUMERIC:NORMAL:VALUE? 1 -> 103.79E+00

When the **<NRF>** is omitted:

:NUMERIC:NORMAL:VALUE? -> 103.79E+00,1.0143E+00,105.27E+00,...,50.001E+00

When **:NUMeric:FORMat** is set to **{FLOat}**:

:NUMERIC:NORMAL:VALUE? -> #N(N-byte count)(data byte sequence)

3.9.5 :NUMeric[:NORMal]:NUMber

Function: Sets or queries the number of output data items for the command :NUMeric[:NORMal]:VALue.

Syntax:

:NUMeric[:NORMal]:NUMber {<NRf>|ALL}

:NUMeric[:NORMal]:NUMber?

<NRf> = 1 to 255(ALL)

Examples:

:NUMERIC:NORMAL:NUMBER 10

:NUMERIC:NORMAL:NUMBER? -> :NUMERIC:NORMAL:NUMBER 10

3.9.6 :NUMeric[:NORMal]:ITEM<x>

Function: Sets or queries the output of a specified numerical data item.

Syntax:

:NUMeric[:NORMal]:ITEM<x> {NONE|<Function>[,<Element>][,<Order>]}

:NUMeric[:NORMal]:ITEM<x>?

<x> = 1 to 255 (Item number)

NONE = No output

<Function> = {U|I|P|S|Q|...}

<Element> = {<NRf>|SIGMa})

(<NRf> = 1 to 3

<Order> = {TOTal|DC|<NRf>}

(<NRf> = 1 to 50)

Examples:

:NUMERIC:NORMAL:ITEM1 U,1

:NUMERIC:NORMAL:ITEM1? -> :NUMERIC:NORMAL:ITEM1 U,1

:NUMERIC:NORMAL:ITEM1 UK,1,1

:NUMERIC:NORMAL:ITEM1? -> :NUMERIC:NORMAL:ITEM1 UK,1,1

If the parameter <Element> is omitted, it defaults to 1.

If <Order> is omitted, it defaults to TOTal.

Refer to the *Function Parameter Table* for the <Function> parameter.

For functions that do not require <Element> or <Order>, the query response will not include these parameters.

The instrument does not execute <Order> = DC measurement.

3.9.7 :NUMeric[:NORMal]:PRESet

Function: Presets the output format of preset data.

Syntax:

:NUMeric[:NORMal]:PRESet {<NRf>}

<NRf> = 1 to 4 (Default: 2)

Examples:

:NUMERIC:NORMAL:PRESET 1

For the output format of :NUMeric[:NORMal]:PRESet, refer to Table 3.1 Format 1, Table 3.2 Format 2, Table 3.3 Format 3, and Table 3.4 Format 4.

Table 3.1 Format 1

ITEM<x>	<Function>	<Element>
1	U	1
2	I	1
3	P	1
4-6	U-P	2

7-9	U-P	3
10-12	U-P	SIGMA
13-255	NONE	

Table 3.2 Format 2

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	FU	1
9	FI	1
10	NONE	
11-19	U-FI	2
20	NONE	
21-29	U-FI	3
30	NONE	
31-39	U-FI	SIGMA
40-255	NONE	

Table 3.3 Format 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	FU	1
9	FI	1
10	UPPeak	1
11	UMPeak	1
12	IPPeak	1
13	IMPeak	1
14	PPPeak	1
15	PMPeak	1
16-30	U-PMPeak	2
31-45	U-PMPeak	3
46-60	U-PMPeak	SIGMA
61-255	NONE	

Table 3.4 Format 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	FU	1
9	FI	1

10	UPPeak	1
11	UMPeak	1
12	IPPeak	1
13	IMPeak	1
14	TIME	1
15	WH	1
16	WHP	1
17	WHM	1
18	AH	1
19	AHP	1
20	AHM	1
21-40	U-AHM	2
41-60	U-AHM	3
61-80	U-AHM	SIGMA
81-255	NONE	

3.9.8 :NUMeric[:NORMal]:CLEar

Function: Clears the output of numerical data.

Syntax:

:NUMeric[:NORMal]:CLEar {ALL|<NRf>[,<NRf>]}

ALL = Clears all output numerical data items

First <NRf> = 1 to 255 (Specifies the first output numerical data item to be cleared.)

Second <NRf> = 1 to 255 (Specifies the last output numerical data item to be cleared. If omitted, clearing continues to the last numeric data item.)

Examples:

:NUMERIC:NORMAL:CLEAR ALL

3.9.9 :NUMeric[:NORMal]:DELete

Function: Deletes the output of numerical data.

Syntax:

:NUMeric[:NORMal]:DELete {<NRf>[,<NRf>]}

First <NRf> = 1 to 255 (Specifies the first numerical data item to be deleted)

Second <NRf> = 1 to 255 (Specifies the last numeric data item to be deleted. If this parameter is omitted, only the numeric data item specified by the first <NRf> is deleted.)

Examples:

:NUMERIC:NORMAL:DELETE 1 (Deletes Item 1 and shifts Item 2 and subsequent numeric data items forward.)

:NUMERIC:NORMAL:DELETE 1,3 (Deletes Items 1 through 3 and shifts Item 4 and subsequent numeric data items forward.)

After numeric data items are deleted, subsequent items shift forward to fill the vacated positions. Any empty positions created at the end of the queue are set to NONE.

3.9.10 :NUMeric[:NORMal]:HEADer?

Function: Queries the header of the numerical data.

Syntax:

:NUMeric[:NORMal]:HEADer? {<NRf>}

<NRf> = 1 to 255 (Item number)

If a <NRf> value is specified, only the data name (i.e., data header) of the specified numerical data is output.

If <NRf> is omitted, data names are output from the first numerical data item up to the item determined by the command :NUMeric:NUMber.

Examples:

When the **<NRF>** is specified:

:NUMERIC:NORMAL:HEADER? 1 -> U-E1

When the **<NRF>** is omitted:

:NUMERIC:NORMAL:HEADER? -> U-E1,I-E1,P-E1

3.9.11 :NUMeric:LIST?

Function: Queries all output settings of the harmonic measurement data table.

Syntax:

:NUMeric:LIST?

The number of output items for :NUMeric:LIST:ITEM<x> is determined by the

command :NUMeric:LIST:NUMber.

3.9.12 :NUMeric:LIST:VALue?

Function: Queries all numerical data of the harmonic measurements.

Syntax:

:NUMeric:LIST:VALue? {<NRf>}

<NRf> = 1 to 32 (Number)

Examples:

When the **<NRF>** is specified:

:NUMERIC:LIST:VALUE? 1

-> 103.58E+00,NAN,103.53E+00,0.09E+00,2.07E+00,0.04E+00,..,0.01E+00,0.01E+00 (Up to 52 data items)

When the **<NRF>** is omitted, **(:NUMeric:LIST:NUMber** is set to **5)**

:NUMERIC:LIST:VALUE?

-> 103.58E+00,NAN,103.53E+00,0.09E+00,2.07E+00,0.04E+00,..,0.00E+00,0.00E+00 (Up to 52*5 = 260 data items)

When **:NUMeric:FORMat** is set to **{FLOat}**

:NUMERIC:LIST:VALUE? -> #N (N-byte count)(data byte sequence)

A single numeric list data item may consist of up to 52 numeric values, in the following order: TOTal, DC, 1st harmonic, ..., as specified by :NUMeric:LIST:ORDer.

If a <NRf> value is specified, only the selected numeric list data item is output.

If <NRf> is omitted, data items from 1 to the value specified by :NUMeric:LIST:NUMber are output sequentially.

This instrument does not perform DC component measurement for harmonic tests; the DC result is always NAN.

3.9.13 :NUMeric:LIST:NUMber

Function: Sets or queries the number of data returned by the command :NUMeric:LIST:VALue?.

Syntax:

:NUMeric:LIST:NUMber {<NRf>|ALL}

:NUMeric:LIST:NUMber?

<NRf> = 1 to 32 (ALL)

By default, the <NRF> value is set to 1.

Examples:

:NUMERIC:LIST:NUMBER 5

:NUMERIC:LIST:NUMBER? -> :NUMERIC:LIST:NUMBER 5

If the parameter is omitted in the command :NUMeric:LIST:VALue?, data items from 1 up to the value specified by this command are output sequentially.

3.9.14 :NUMeric:LIST:ORDer

Function: Sets or queries the maximum harmonic order of the harmonic measurement data table.

Syntax:

:NUMeric:LIST:ORDer {<NRf>|ALL}

:NUMeric:LIST:ORDer?

<NRf> = 1 to 50 (ALL)

Examples:

:NUMERIC:LIST:ORDER 50

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

3.9.15 :NUMeric:LIST:SELect

Function: Sets or queries the output settings of the harmonic measurement list.

Syntax:

:NUMeric:LIST:SELect {EVEN|ODD|ALL}

:NUMeric:LIST:SELect?

Examples:

:NUMERIC:LIST:SELECT ALL

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

EVEN = Outputs TOTal, DC, EVEN

ODD = Outputs TOTal, DC, ODD

ALL = Outputs ALL

3.9.16 :NUMeric:LIST:ITEM<x>

Function: Queries or sets the output items of the harmonic measurement list.

Syntax:

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

:NUMeric:LIST:ITEM<x>?

<x> = 1 to 32 (Number)

NONE = No output

<Function> = {U|I|P|PHIU|PHII|UHDF|IHDF|PHDF}

<Element> = {<NRf>}(<NRf> = 1 to 3)

Examples:

:NUMERIC:LIST:ITEM1 U,1

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

Refer to the *Function Parameter Table* for the <Function> parameter.

3.9.17 :NUMeric:LIST:PRESet

Function: Presets the output format for the harmonic measurement list.

Syntax:

:NUMeric:LIST:PRESet {<NRf>}

<NRf> = 1 to 4 (Default: 2)

Examples:

:NUMERIC:LIST:PRESET 1

For the output format of :NUMeric:LIST:PRESet, refer to Table 3.5 Format 1, Table 3.6 Format 2, Table 3.7 Format 3, and Table 3.8 Format 4.

Table 3.5 Format 1

ITEM<x>	<Function>	<Element>
1	U	1
2	I	1
3	P	1
4 to 6	U-P	2

7 to 9	U-P	3
10 to 32	NONE	

Table 3.6 Format 2

ITEM<x>	<Function>	<Element>
1	U	1
2	I	1
3	P	1
4	PHIU	1
5	PHII	1
6 to 10	U-PHII	2
11 to 15	U-PHII	3
16 to 32	NONE	

Table 3.7 Format 3

ITEM<x>	<Function>	<Element>
1	U	1
2	I	1
3	P	1
4	UHDF	1
5	IHDF	1
6	PHDF	1
7 to 12	U-PHDF	2
13 to 18	U-PHDF	3
19 to 32	NONE	

Table 3.8 Format 4

ITEM<x>	<Function>	<Element>

1	U	1
2	I	1
3	P	1
4	PHIU	1
5	PHII	1
6	UHDF	1
7	IHDF	1
8	PHDF	1
9 to 16	U-PHDF	2
17 to 24	U-PHDF	3
25 to 32	NONE	

3.9.18 :NUMeric:LIST:CLEar

Function: Clears the output item of the harmonic measurement list.

Syntax:

:NUMeric:LIST:CLEar {ALL|<NRf>[,<NRf>]}

ALL = Clears all numerical data items

First <NRf> = 1 to 32 (Specifies the first output numerical data item to be cleared.)

Second <NRf> = 1 to 32 (Specifies the last output numerical data item to be cleared.)

Note: If the second <NRf> parameter is omitted, all output data items from the first specified item to the end will be cleared.

Examples:

:NUMERIC:LIST:CLEAR ALL

3.9.19 :NUMeric:LIST:DELete

Function: Deletes the output item of the harmonic measurement list.

Syntax:

:NUMeric:LIST:DELete {<NRf>[,<NRf>]}

First <NRf> = 1 to 32 (Specifies the first output item to be deleted.)

Second <NRf> = 1 to 32 (Specifies the last output item to be deleted.)

Note: If the second <NRf> parameter is omitted, only the first specified data item is deleted.

When an output item is deleted, subsequent items shift forward to fill the vacated positions. Any empty positions created at the end are set to NONE.

Examples:

:NUMERIC:LIST:DELETE 1 (Deletes Item 1 and shifts Item 2 and subsequent numeric data items forward.)

:NUMERIC:LIST:DELETE 1,3 (Deletes Items 1 through 3 and shifts Item 4 and subsequent numeric data items forward.)

3.9.20 :NUMeric:HOLD

Function: Sets or queries the state of data hold.

Syntax:

:NUMeric:HOLD {<Boolean>}

:NUMeric:HOLD?

Examples:

:NUMERIC:HOLD ON

:NUMERIC:HOLD? -> :NUMERIC:HOLD 1

- Data Hold

  If :NUMeric:HOLD is set to ON before executing the command :NUMeric:VALue? or :NUMeric:LIST:VALue?, all numerical data is held internally.

  When :NUMeric:HOLD is set to ON, the held data remains unchanged even if the displayed values on the screen are updated.

- Acquiring Data for Input Units at a Specific Time Point

  To obtain numerical data for each input unit captured at the same time point, use the following commands:

  :NUMeric:HOLD ON

  :NUMeric[:NORMal]:ITEM1 U,1;ITEM2 I,1;...

  (Sets the numeric data items for input unit 1)

  :NUMeric[:NORMal]:VALue?

  (Receives the numeric data items of input unit 1)

:NUMeric[:NORMal]:ITEM1 U,2;ITEM2 I,2;

(Sets the numeric data items for input unit 2)

:NUMeric[:NORMal]:VALue?

(Receives the numeric data items of input unit 2)

:NUMeric[:NORMal]:ITEM1 U,3;ITEM2 I,3;...

(Sets the numeric data items for input unit 3)

:NUMeric[:NORMal]:VALue?

(Receives the numeric data items of input unit 3)

:NUMeric:HOLD OFF

- Re-Hold

  After executing the command :NUMeric:HOLD to set HOLD to ON, the user can send the same command :NUMeric:HOLD again to re-enable HOLD. This clears the existing numeric data and holds the latest numeric values instead.

  This operation eliminates the need to repeatedly execute the command :NUMeric:HOLD OFF, enabling continuous retrieval of updated numeric data.

3.9.21 Function Parameter Table

1. Numeric Data Function (NORMAL)

Applicable Command:

:NUMeric[:NORMal]:ITEM<x> {NONE|**<Function>**[,<Element>][,<Order>]}

:AOUTput[:NORMal]:CHANnel<x> {NONE|**<Function>**[,<Element>]}

<Function>	Description	Value Display	<Element>	<Order>
U	Voltage (U)	U	YES	NO
I	Current (I)	I	YES	NO
P	Active power (P)	P	YES	NO
S	Apparent power (S)	S	YES	NO
Q	Reactive power (Q)	Q	YES	NO
LAMBda	Power factor (λ)	PF	YES	NO

PHI	Phase difference (Φ)	PHI	YES	NO
FU	Voltage frequency (fU)	fU	YES	NO
FI	Current frequency (fI)	fI	YES	NO
UPPeak	Positive voltage peak (U+PK)	Upk+	YES	NO
UMPeak	Negative voltage peak (U-PK)	Upk-	YES	NO
IPPeak	Positive current peak (I+PK)	Ipk+	YES	NO
IMPeak	Negative current peak (I-PK)	Ipk-	YES	NO
PPPeak	Positive power peak (P+pk)	Ppk+	YES	NO
PMPeak	Negative power peak (P-pk)	Ppk-	YES	NO
TIME	Integration time	Test time	NO	NO
WH	Watt-hour (WP)	WP	YES	NO
WHP	Positive Watt-hour (WP+)	WP+	YES	NO
WHM	Negative Watt-hour (WP)	WP-	YES	NO
AH	Ampere-hour (q)	q	YES	NO
AHP	Positive Ampere-hour (q+)	q+	YES	NO
AHM	Negative Ampere-hour (q-)	q-	YES	NO
MATH	Calculated value	MATH	NO	NO
URANge	Voltage range		NO	NO
IRANge	Current range		NO	NO
URMS	Voltage RMS (Urms)	Urms	YES	NO
UMN	Rectified average voltage calibrated to RMS (Umn)	Umn	YES	NO
UDC	Simple average voltage (Udc)	Udc	YES	NO
URMN	Rectified average voltage (Urmn)	Urmn	YES	NO

UAC	AC component of voltage (Uac)	Uac	YES	NO
IRMS	Current RMS	Irms	YES	NO
IMN	Rectified average current calibrated to RMS (Imn)	Imn	YES	NO
IDC	Simple average current (Idc)	Idc	YES	NO
IRMN	Rectified average current (Irmn)	Irmn	YES	NO
IAC	AC component of current (Iac)	Iac	YES	NO
CFU	Measured voltage crest factor (CFU)	CFU	YES	NO
CFI	Measured current crest factor (CFI)	CFI	YES	NO
AVP	Average integrated power	AVP	YES	NO
EFFiciency	Efficiency	EFFi	NO	NO
UPeak(1)	Voltage peak (Upk)	Upk+ or Upk-	YES	NO
IPeak(1)	Current peak (Ipk)	Ipk+ or Ipk-	YES	NO
UK	Voltage RMS of k-th harmonic U(k)	U(k)[V]	YES	YES
IK	Current RMS of k-th harmonic I(k)	I(k)[A]	YES	YES
PK	Active power of k-th harmonic	P(k)[W]	YES	YES
LAMBDAK	Power factor of k-th harmonic	PF	YES	YES (only k=1)
PHIK	Phase difference between k-th harmonic voltage and current	U(k)[°] or I(k)[°]	YES	YES (only k=1)

PHIUk	Phase difference φU(k) between harmonic voltage U(k) and fundamental voltage U(1)	U(k)[°]	YES	YES (k> =2)
PHIIk	Phase difference φI(k) between harmonic current I(k) and fundamental current I(1)	I(k)[°]	YES	YES (k> =2)
UHDFk	Uhdf(k): Voltage harmonic distortion factor	U(k)[%]	YES	YES
IHDFk	Ihdf(k): Current harmonic distortion factor	I(k)[%]	YES	YES
PHDFk	Phdf(k): Active power harmonic distortion factor	P(k)[%]	YES	YES
UTHD	Total harmonic distortion of voltage	THDU	YES	NO
ITHD	Total harmonic distortion of current	THDI	YES	NO
FPLL	PLLsource frequency fPLL	fU or fI	NO	NO

YES: Required NO: Not required

• UPeak and IPeak are only applicable to the command:

:AOUTput[:NORMal]:CHANnel<x> {NONE|**<Function>**[,<Element>]}

2. Numeric Data List Function (LIST)

Applicable Command:

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

<Function>	Description
U	Voltage (U)

I	Current (I)
P	Active power (P)
PHIU	Phase difference φU(k) between harmonic voltage U(k) and fundamental voltage U(1)
PHII	Phase difference φI(k) between harmonic current I(k) and fundamental current I(1)
UHDF	Voltage harmonic distortion factor (Uhdf)
IHDF	Current harmonic distortion factor (Ihdf)
PHDF	Active power harmonic distortion factor (Phdf)

3.9.21 Numerical Data Format

1) Normal Data

- Integrated values: WH, WHP, WHM, AH, AHP, and AHM

  ASCII: <NR3> format (up to 6 mantissa digits and 2 exponent digits). For example, [-]123.456E+00

  FLOAT: IEEE single-precision floating-point format (4 bytes)

- Integration time (TIME)

  ASCII: <NR1> format in seconds. For example, 3600 seconds equals one hour (1:00:00).

  FLOAT: IEEE single-precision floating-point format, with the unit in seconds (s). For example, 0x45610000 represents one hour (1:00:00).

- NONE

  ASCII: NAN (Not a Number)

  FLOAT: 0x7E951BEE (9.91E+37)

- Others

  ASCII: <NR3> format (up to 5 digits for the mantissa and 2 digits for the exponent). For example, 1 [-]45.456E+00

  FLOAT: IEEE single-precision floating-point format (4 bytes)

2) Error Data

- Data not available (displayed as "-----")

  ASCII: NAN (Not a Number)

  FLOAT: 0x7E951BEE (9.91E+37)

- Out of range (measured value shown as "--OL--" on the screen)
- Calculation overflow (measured value shown as "--OF--" on the screen)
- Data overflow (measured value shown as "error" on the screen)

  ASCII: INF (Infinity)

  FLOAT: 0x7E94F56A (9.9E+37)

Note: In 180° (Lead/Lag) display mode, the phase difference Φ (PHI) of input units 1-3 is within the range –180.0 to 180.0. The value takes a negative or positive sign depending on whether it is Lead (D) or Lag (G).

3.10 Display Update Command

3.10.1 :RATE

Function: Sets or queries the data update interval.

Syntax:

:RATE {<Time>|AUTO}

:RATE?

<Time> = 100, 250, 500(ms), 1, 2, 5, 10, 20(s)

Examples:

:RATE 250MS

:RATE? -> :RATE 250.0E-03

3.10.2 :RATE:AUTO?

Function: Queries the data update interval in automatic mode.

Syntax:

:RATE:AUTO?

3.10.3 :RATE:AUTO:TIMeout

Function: Sets or queries the automatic timeout for the data update interval.

Syntax:

:RATE:AUTO:TIMeout {<NRf>}

:RATE:AUTO:TIMeout?

<NRf> = 1, 5, 10, 20(s)

Examples:

:RATE:AUTO:TIMEOUT 1

:RATE:AUTO:TIMEOUT? -> :RATE:AUTO:TIMEOUT 1

3.10.4 :RATE:AUTO:SYNChronize

Function: Sets or queries the synchronization source for the data update interval in automatic mode.

Syntax:

:RATE:AUTO:SYNChronize {U<x>|I<x>}

:RATE:AUTO:SYNChronize?

<x> = 1 to 3 (element)

Examples:

:RATE:AUTO:SYNCHRONIZE U1

:RATE:AUTO:SYNCHRONIZE? -> :RATE:AUTO:SYNCHRONIZE U1

3.11 Status Command

3.11.1 :STATus?

Function: Queries the communication status settings.

Syntax:

:STATus?

3.11.2 :STATus:CONDition?

Function: Queries the value of the status register.

Syntax:

:STATus:CONDition?

Examples:

:STATUS:CONDITION? -> 16

3.11.3 :STATus:EESE

Function: Sets or queries the extended event enable register.

Syntax:

:STATus:EESE <Register>

:STATus:EESE?

<Register> = 0 to 65535

Examples:

:STATUS:EESE #B0000000000000000

:STATus:EESE? -> :STATUS:EESE 0

3.11.4 :STATus:EESR?

Function: Queries and clears the extended event enable register.

Syntax:

:STATus:EESR?

Examples:

:STATUS:EESR? -> 0

3.11.5 :STATus:ERRor?

Function: Queries the code and details of the last error.

Syntax:

:STATus:ERRor?

Examples:

:STATUS:ERROR? -> 113,"Underfined Header"

If no error occurs, the query returns "No error."

3.11.6 :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

Examples:

:STATUS:FILTER1 RISE

:STATus:FILTER1? -> :STATUS:FILTER1 RISE

Sets the correspondence between bit changes in the status register and event trigger settings. For example, if a bit is set to RISE, an event is triggered when the bit transitions from 0 to 1.

Note: When using the :STATus:FILTer<x> command, <x> = 1 to 16 corresponds to Bit 0 to Bit 15 of the transition filter.

3.12 Storage Command

3.12.1 :STORe?

Function: Queries all storage settings.

Syntax:

:STORe?

3.12.2 :STORe[:STATe]

Function: Sets or queries the storage state.

Syntax:

:STORe[:STATe]{<Boolean>}

:STORe:STATe?

Examples:

:STORE:STATE ON

:STORE:STATE? -> :STORE:STATE 1

3.12.3 :STORe:INTerval

Function: Sets or queries the storage time interval.

Syntax:

:STORe:INTerval {<NRf>,<NRf>,<NRf>}

:STORe:INTerval?

{<NRf>,<NRf>,<NRf>} = 0,0,1 to 99,59,59

First <NRf> = 0 to 99 (Hours)

Second <NRf> = 0 to 59 (Minutes)

Third <NRf> = 1 to 59 (Seconds)

Examples:

:STORE:INTERVAL 0,0,1

:STORE:INTERVAL? -> :STORE:INTERVAL 0,0,1

3.13 System Command

3.13.1 :SYSTem?

Function: Queries all system settings.

Syntax:

:SYSTem?

3.13.2 :SYSTem:BRIGhtness

Function: Queries and sets the LCD backlight brightness.

Syntax:

:SYSTem:BRIGhness {<NRf>}

:SYSTem:BRIGhness?

<NRf> = 10, 20,30,40,50,60,70,80,90,100

Examples:

:SYSTEM:BRIGHTNESS 100

:SYSTEM:BRIGHTNESS? -> :SYSTEM:BRIGHTNESS 100

3.13.3 :SYSTem:KEY:BEEPer

Function: Sets or returns the beeper state.

Syntax:

:SYSTem:KEY:BEEPer {<Boolean>|OFF|ON}

:SYSTem:KEY:BEEPer?

Examples:

:SYSTEM:KEY:BEEPER OFF

:SYSTEM:KEY:BEEPER? -> :SYSTEM:KEY:BEEPER 0

3.13.4 :SYSTem:KLOCK

Function: Sets or returns the key lock state.

Syntax:

:SYSTem:KLOCk {<Boolean>|OFF|ON}

:SYSTem:KLOCk?

Examples:

:SYSTEM:KLOCK OFF

:SYSTEM:KLOCK? -> :SYSTEM:KLOCK 0

3.13.5 :SYSTem:SERial?

Function: Queries the serial number.

Syntax:

:SYSTem:SERial?

Examples:

:SYSTEM:SERIAL? -> :SYSTEM:SERIAL "APA1234567890"

3.13.6 :SYSTem:VERsion[:FIRMware]?

Function: Queries the firmware version.

Syntax:

:SYSTem:VERsion[:FIRMware]?

Examples:

:SYSTEM:VERSION:FIRMWARE? -> "V1.01.0003,V1.01.0002,V1.01.0003"

The returned data consists of the version numbers of the MCU, DSP, and FPGA, separated by commas.

3.13.7 :SYSTem:SUFFix?

Function: Queries the standard accessory code for the power meter.

Syntax:

:SYSTem:SUFFix?

Examples:

:SYSTEM:SUFFIX? -> :SYSTEM:SUFFIX "-C1-H/C7/EX1/EX2/G5/DA12"

3.13.8 :SYSTem:MODel?

Function: Queries the model number.

Syntax:

:SYSTem:MODel?

Examples:

:SYSTEM:MODEL? -> :SYSTEM:MODEL "UTE323"

3.13.9 :SYSTem:TIMer

Function: Sets the system time.

Syntax:

:SYSTEM:TIMer {<NRf>,<NRf>,<NRf>}

{<NRf>,<NRf>,<NRf>} = 0,0,0 to 23,59,59

First <NRf> = 0 to 23 (Hours)

Second <NRf> = 0 to 59 (Minutes)

Third <NRf> = 0 to 59 (Second)

Examples:

:SYSTEM:TIMer 17,28,52

3.13.10 :SYSTem:DATe

Function: Sets the system date.

Syntax:

:SYSTEM:DATe {<NRf>,<NRf>,<NRf>}

{<NRf>,<NRf>,<NRf>} = 0,0,0 to 2999,0,0

First <NRf> = 0 to 2999 (Years)

Second <NRf> = 1 to 12 (Months)

Third <NRf> = 1 to 31 (Days)

Examples:

:SYSTEM:DATe 2023,8,15

3.14 Waveform Command

3.14.1 :WAVeform?

Function: Queries all output settings for waveform data.

Syntax:

:WAVeform?

3.14.2 :WAVeform:BYTeorder

Function: Sets or queries the output byte order of waveform display data (floating-point format) transmitted by the command :WAVeform:SEND?.

Syntax:

:WAVeform:BYTeorder {LSBFirst|MSBFirst}

:WAVeform:BYTeorder?

Examples:

:WAVEFORM:BYTEORDER LSBFIRST

:WAVEFORM:BYTEORDER? -> :WAVEFORM:BYTEORDER LSBFIRST

This command is only available :WAVeform:FORMat when is set to FLOat.

3.14.3 :WAVeform:END

Function: Sets or queries the output end point of waveform display data transmitted by the command :WAVeform:SEND?.

Syntax:

:WAVeform:END {<NRf>}

:WAVeform:END?

<NRf> = 0 to 1199

Examples:

:WAVEFORM:END 1199

:WAVEFORM:END? -> :WAVEFORM:END 1199

3.14.4 :WAVeform:FORMat

Function: Sets or queries the format of waveform display data transmitted by the command :WAVeform:SEND?.

Syntax:

:WAVeform:FORMat {ASCii|FLOat}

:WAVeform:FORMat?

Examples:

:WAVEFORM:FORMAT FLOAT

:WAVEFORM:FORMAT? -> :WAVEFORM:FORMAT FLOAT

For details on differences in waveform display data output between formats, refer to the description of the command :WAVeform:SEND?.

3.14.5 :WAVeform:HOLD

Function: Sets or queries the data hold state of the waveform display data.

Syntax:

:WAVeform:HOLD {<Boolean>}

:WAVeform:HOLD?

Examples:

:WAVeform:HOLD ON

:WAVeform:HOLD? -> :WAVeform:HOLD 1

If :WAVeform:HOLD is set to ON before executing the command :WAVeform:SEND?, all waveform data is held internally.

When :WAVeform:HOLD is set to ON, the held command data remains unchanged even if the displayed values on the screen are updated.

For example, to obtain waveform data for U1 and I1 captured at the same time point, use the following commands:

:WAVeform:HOLD ON

:WAVeform:TRACe U1

:WAVeform:SEND?

(Receives the waveform display data of U1)

:WAVeform:TRACe I1

:WAVeform:SEND?

(Receives the waveform display data of I1)

:WAVeform:HOLD OFF

After executing the command :WAVeform:HOLD to set HOLD to ON, the user can send the same

command :WAVeform:HOLD again to re-enable HOLD. This clears the existing waveform data and holds the latest waveform data instead.

This operation eliminates the need to repeatedly execute the command :WAVeform:HOLD OFF, enabling continuous retrieval of updated waveform data.

3.14.6 :WAVeform:LENGth?

Function: Queries the total number of points in the waveform data.

Syntax:

:WAVeform:LENGth?

Examples:

:WAVeform:LENGth? -> 1200

The number of data points is fixed. The query will always return 1200.

3.14.7 :WAVeform:RATed?

Function: Queries the rated maximum and minimum values of the waveform display data specified by the command :WAVeform:TRACe.

Syntax:

:WAVeform:RATed?

Examples:

:WAVEFORM:TRACE U1

:WAVeform:RATed? -> :WAVeform:RATed 45.000E+00,-45.000E+00

3.14.5 :WAVeform:SEND?

Function: Queries the waveform display data specified by the command :WAVeform:TRACe.

Syntax:

:WAVeform:SEND?

Examples:

- When :WAVeform:FORMat is set to ASCII

  :WAVeform:SEND? -> <NR3>,<NR3>...

- When WAVeform:FORMat is set to FLOat:

:WAVeform:SEND? -> #N(N-bit bytes)(data byte sequence)

The format of waveform data varies depending on the setting of the command :WAVeform:FORMAT.

1) When the format is set to ASCII:

Physical values are output in <NR3> format, separated by commas between data points.

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 for each data point follows the configuration of the command :WAVeform:BYTeorder.

3.14.6 :WAVeform:STARt

Function: Sets or queries the start point of the waveform display data transmitted by the command :WAVeform:SEND?.

Syntax:

:WAVeform:STARt {<NRf>}

:WAVeform:STARt?

<NRf> = 0 to 1199

Examples:

:WAVEFORM:START 0

:WAVEFORM:START? -> :WAVEFORM:START 0

3.14.7 :WAVeform:TDIV

Function: Sets or queries the Time/div of the waveform display data.

Syntax:

:WAVeform:TDIV {<Time>}

:WAVeform:TDIV?

<Time> = 100us,200us,500us,1ms.2ms,5ms,10ms,25ms,50ms,100ms,200ms,500ms,1s,2s

Examples:

:WAVeform:TDIV 10MS

:WAVeform:TDIV? -> :WAVeform:TDIV 10.00E-03

3.14.8 :WAVeform:TRACe

Function: Sets or queries the target waveform to be used for the :WAVeform:SEND? command.

Syntax:

:WAVeform:TRACe {U<x>|I<x>}

:WAVeform:TRACe?

<x> = 1 to 3

Examples:

:WAVEFORM:TRACE U1

:WAVEFORM:TRACE? -> :WAVEFORM:TRACE U1

3.15 Common Command

3.15.1 *CAL?

Function: Executes the zero-clearing.

Syntax:

*CAL?

Examples:

*CAL? -> 0

3.15.2 *CLS

Function: Clears the standard event register, extended event register, and error queue.

Syntax:

*CLS

If the *CLS command immediately follows the command message terminator, the output queue will be cleared as well.

3.15.3 *ESE

Function: Sets or queries the standard event enable register.

Syntax:

*ESE {<NRf>}

*ESE?

<NRf> = 0 to 255 (Default: 0)

Examples:

*ESE 251

*ESE? -> 251

The command uses the sum of decimal values corresponding to individual bits.

For example, *ESE 251 sets the standard event enable register to 11111011. In this case, bit 2 of the standard event register is disabled. This also means that bit 5 (ESB) of the status byte register will not be set to 1, even if a query error occurs.

The *ESE? query does not clear the contents of the standard event enable register.

3.15.4 *ESR

Function: Queries and clears the standard event register.

Syntax:

*ESR?

Examples:

*ESR? -> 32

The query returns the sum of each numerical value in decimal format.

3.15.5 *IDN?

Function: Queries the instrument information.

Syntax:

*IDN?

The query returns <Manufacturer>, <Model Number>, <Serial Number>, <Firmware Version>.

3.15.6 *OPC

Function: When the specified overlap command is executed, the standard event register will be set to 0.

Note: Currently, the power meter does not support this command.

Syntax:

*OPC

Note: Currently, the power meter does not support this command. OPC is always set to 1.

3.15.7 *OPC?

Function: When the specified overlap commands C1, C7, EX1, EX2, G5, DA12 are executed, the query returns

ASCII 1.

Note: Currently, the power meter does not support this command.

Syntax:

*OPC?

Examples:

*OPC? -> 1

Note: Currently, the power meter does not support this command. The query returns always 1.

3.15.8 *OPT?

Function: Queries the interface state.

Syntax:

*OPT?

Examples:

*OPT? -> C1,C7,EX1,EX2,G5,DA12

The query returns the status of the following interfaces: GPIB (C1), RS232 (C2), Ethernet communication interface (C7), external current sensor input 1 (EX1, 2.5 V/5 V/10 V), external current sensor input 2 (EX2, 50 mV/100 mV/200 mV/500 mV/1 V/2 V), harmonic measurement (G5), and 12-channel DA output (DA12).

3.15.9 *RST

Function: Restores the instrument to its default values.

Syntax:

*RST

Examples:

*RST

Except for the communication settings, all other settings are restored to their default values.

Note: When this command is sent, wait 5 seconds before sending any other commands.

3.15.10 *SRE

Function: Sets or queries the service request enable register.

Syntax:

*SRE {<NRf>}

*SRE?

<NRf> = 0 to 255

Examples:

*SRE 239

cc

The parameter value is the decimal sum of the individual bit values. Querying with the *SRE? command does not clear the contents of the Service Request Enable Register.

3.15.11 *STB?

Function: Queries the status byte value.

Syntax:

*STB?

Examples:

*STB? -> 4 (The parameter value is the decimal sum of the individual bit values.)

For example, a return value of 4 indicates that the status byte is set to 00000100, meaning the error queue is not empty — an error has occurred. Executing the *STB? command does not clear the contents of the status byte.

3.15.12 *TRG

Function:

Executes a single measurement (equivalent to pressing the Single key).

Syntax:

*TRG

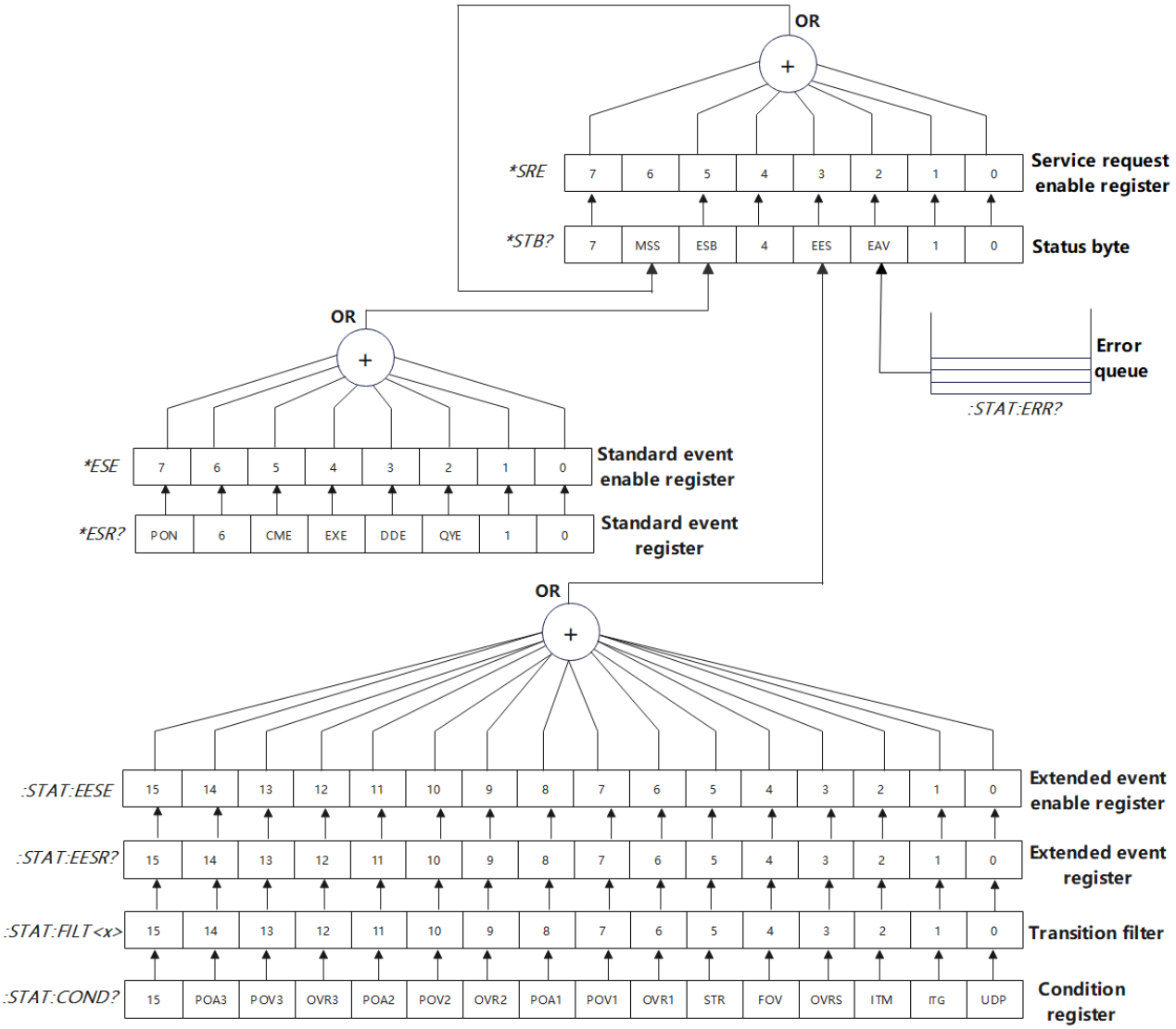
Examples:

*TRG

4. Status System

4.1 Status Report Diagram

The status report diagram illustrates the relationships between components in the status system, as shown in the figure below.



4.2 Status Byte

Bit	7	6	5	4	3	2	1	0
Definition	Not used	MSS or RQS	ESB	MAV	ESS	EAV	Not used	Not used

- Bit0, 1, 7: Not used Not used

- Bit2 EAV (Error Valid)

  Indicates that one or more errors are present in the error queue. Use the command :STATUS:ERROR? to query the error details.

- Bit3 ESS (Extended Summary Status)

  Set to 1 when the logical AND of the extended event register and the extended event enable register is 1. This bit is set to 1 when an extended event occurs.

- Bit4 MAV (Message Valid)

  Set to 1 when data is available in the output queue.

  Note: The output queue function is not currently supported on this instrument, so the MAV bit remains 0.

- Bit5 ESB (Event Summary Bit)

  Set to 1 when the logical AND of the standard event register and the standard event enable register is 1. This bit is set to 1 when a standard event occurs.

- Bit6 MSS (Master Summary Status)/RQS (Request Service)

  MSS: Set to 1 when the logical AND of the status byte (excluding bit 6) and the service request enable register is 1.

  RQS: Set to 1 when MSS transitions from 0 to 1, and cleared when serial polling is performed or MSS becomes 0. When RQS is 1, the instrument sends a service request to the controller.

  Note: Serial polling and RQS (Request Service) functionality are not supported on this instrument.

The *STB? command is used to read the status byte, but it does not clear any bits within the status byte. When the *CLS command is executed, the status byte itself is not cleared. However, the contents of the standard event register, which affect specific bits in the status byte, are cleared, thereby clearing the corresponding bits in the status byte.

4.3 Standard Event Register

Bit	7	6	5	4	3	2	1	0
Definition	PON	URQ	CME	EXE	DDE	QYE	RQC	OPC

- Bit7: PON (Power On)

Set to 1 when the instrument is powered on.

- Bit6: URQ (User Request)

  Not used; always returns 0.

- Bit5: CME (Command Error)

  Set to 1 when a command syntax error occurs.

  Examples: Misspelled command name or invalid character data that does not match the available options.

- Bit4: EXE (Execution Error)

  Set to 1 when the command syntax is valid, but the command cannot be executed due to the current instrument state.

  Examples: Parameter out of range or command intended for an uninstalled option.

- Bit3 DDE (Device Error)

  Set to 1 when the command cannot be executed due to internal issues (excluding command and execution errors).

- Bit2 QYE (Query Error)

  Set to 1 when a query command is issued, but the output queue is empty or data is lost.

  Examples: No response data or data loss due to output queue overflow.

- Bit 1 RQC (Request Control), Bit0 OPC (Operation Complete)

  Note: The RQC and OPC functions are not supported by this instrument. These bits always return 0.

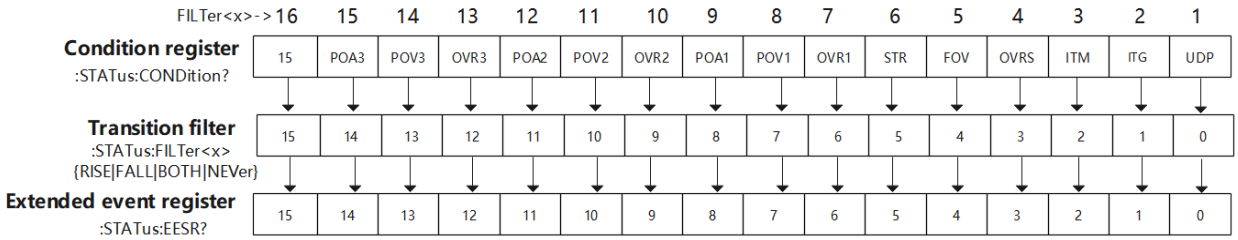
The contents of the standard event register can be read using the *ESR? command. After reading, the register is cleared.

The Standard Event Register can be cleared in the following ways:

1) By reading its contents with the ***ESR?** command.
2) When the ***CLS** command is executed.
3) When the instrument is restarted.

4.4 Extended Event Register

The extended event register receives information about changes in the status register. This information is obtained through edge detection via the transition filter and reflects the instrument's internal status.



Note: When using the command :STATus:FILTer<x>, <x> = 1 to 16 corresponds to Bit 0 to 15 of the transition filter.

Table Definition of Status Register

Bit	Definition	Description
Bit0	UPD (Data Update)	Set to 1 while measurement data is being updated. When UPD changes from 1 to 0, the update is complete.
Bit1	ITG (Integration Busy)	Set to 1 while integration is in progress.
Bit2	ITM (Integration Timer Busy)	Set to 1 while the integration timer is active.
Bit3	OVRS (∑ Result Overflow)	Set to 1 when the summation (∑) calculation result overflows.
Bit4	FOV (Frequency Overflow)	Set to 1 when the frequency exceeds the measurement range.
Bit5	STR (Storage Busy)	Set to 1 while the storage function is active.
Bit6	OVR1 (Element 1 Measurement Data Overflow)	Set to 1 when the voltage or current of Element 1 exceeds its range.
Bit7	POV1 (Element 1 Voltage Peak Overflow)	Set to 1 when the voltage peak of Element 1 exceeds its range.
Bit8	POA1 (Element 1 Current Peak Overflow)	Set to 1 when the current peak of Element 1 exceeds its range.

Bit9	OVR2 (Element 2 Measurement Data Overflow)	Set to 1 when the voltage or current of Element 2 exceeds its range.
Bit10	POV2 (Element2 Voltage Peak Overflow)	Set to 1 when the voltage peak of Element 2 exceeds its range.
Bit11	POA2 (Element2 Current Peak Overflow)	Set to 1 when the current peak of Element 2 exceeds its range.
Bit12	OVR3 (Element3 Measurement Data Overflow)	Set to 1 when the voltage or current of Element 3 exceeds its range.
Bit13	POV3 (Element3 Voltage Peak Overflow)	Set to 1 when the voltage peak of Element 3 exceeds its range.
Bit14	POA3 (Element3 Current Peak Overflow)	Set to 1 when the current peak of Element 3 exceeds its range.

The transition filter detects changes in the specified status register and updates the extended event register according to the mode listed in the table below.

Mode	Description
RISE	Sets the corresponding bit in the extended event register to 1 when a bit in the status register changes from 0 to 1.
FALL	Sets the corresponding bit in the extended event register to 1 when a bit in the status register changes from 1 to 0.
BOTH	Sets the corresponding bit in the extended event register to 1 when a bit in the status register changes, either from 0 to 1 or from 1 to 0.
NEVer	Sets the corresponding bit in the extended event register to 0 at all time, regardless of any bit changes in the status register.

4.5 Error Queue

When an error occurs, the error queue stores the error number and corresponding error message. For example, if the instrument receives an incorrect program message from the controller, the error number (113) and the error message ("Undefined header") will be stored in the error queue when the instrument displays the error information.

The user can read the contents of the error queue using the :STATus:ERRor? query. Messages in the error queue are read in a first-in, first-out (FIFO) order.

If the error queue overflows, the last message is replaced with: 350, "Queue overflow."

5. Modbus Protocol

Modbus is a widely used fieldbus protocol. Multiple slaves can be easily networked with a master via Modbus; the master can be a PC or PLC. Modbus is an application-layer messaging protocol with three message types: ASCII, RTU, and TCP.

For this instrument:

- The RS232 interface supports Modbus-ASCII and Modbus-RTU communication modes.
- The Ethernet interface supports Modbus-TCP communication.
- The port number for Modbus-TCP is fixed at 502.

5.1 Modbus Format

5.1.1 Communication Data Structure

RTU Mode

Slave Address	Communication address: 1-byte
Function	Function code: 1-byte
Data (n-1)	Data content
....	
Data(0)	
CRC	Check code: 2-byte

ASCII Mode

Start	Start character ":"(3AH)
Slave Address	Communication address: 1-byte, comprising 2 ASCII characters
Function	Function code: 1-byte, comprising 2 ASCII characters
Data (n-1)	Data content
....	
Data(0)	
LRC	Check code: 2 ASCII characters
End1	End code: (ODH)(CR)
End0	End code: (ODH)(LF)

TCP Mode

MBAP Header	Transaction identifier	2-byte
	Protocol identifier	2-byte
	Length (indicates the length of the remaining data)	2-byte
	Unit identifier	1-byte
PDU Message	Function	Function code: 1-byte
	Data (n-1)	Data content
	....	
	Data(0)	

5.1.2 Function Code 04H - Read Input Registers

This function reads input registers in a continuous block. A maximum of 123 registers, or a minimum of 1 register, can be read in a single request. The following exception codes may be returned: 01, 02, 03, and 04.

The following example shows the master sending a command to slave address 1 to read 2 consecutive words starting from starting address 100 (64H).

RTU Mode

Master Command Information

Slave Address	01H
Function Code	04H
Starting Address	00H (High Byte)
	64H (Low Byte)
Quantity of Registers	00H (High Byte)
	02H (Low Byte)
CRC (Check Low)	30H (Low Byte)
CRC (Check High)	14H (High Byte)

Slave Response Message

Slave Address	01H
Function Code	03H
Byte Count	04H
Register 1	40H (High Byte)
	E2H (Low Byte)
Register 2	55H (High Byte)
	63H (Low Byte)
CRC (Check Low)	31H (Low Byte)
CRC (Check High)	0BH (High Byte)

Floating-point data in this instrument conforms to the IEEE 754 standard. In the slave response message, the combined content of two consecutive registers is 40E25563H (transmitted as 40E2H, 5563H). Interpreted as a big-endian 32-bit IEEE 754 float, this value is 7.072923183441, consistent with the voltage measurement of Element 1.

ASCII Mode

Master Command Information

Start	':'
Slave Address	'0'
	'1'
Function Code	'0'
	'4'
Starting Address	'0'
	'0'
	'6'
	'4'
Quantity of Registers	'0'
	'0'
	'0'
	'2'
LRC Check code	'9'
	'5'
End1	(0DH)(CR)
End0	(0AH)(LF)

Slave Response Message

Start	':'
Slave Address	'0'
	'1'
Function Code	'0'
	'4'
Byte Count	'0'
	'4'
Register 1	'4'
	'0'
	'E'
	'2'
Register 2	'5'
	'5'
	'6'
	'3'
LRC Check code	'1'
	'D'
End1	(0DH)(CR)
End0	(0AH)(LF)

TCP Mode

Master Command Information

MBAP Header	Transaction Identifier	00H (High Byte)
		00H (Low Byte)
	Protocol identifier	00H (High Byte)
		00H (Low Byte)
	Length	00H (High Byte)
		06H (Low Byte)
	Unit Identifier	00H
PDU Message	Function Code	04H
	Starting Address	00H (High Byte)
		64H (Low Byte)
	Quantity of Registers	00H (High Byte)
		02H (Low Byte)

Slave Response Message

MBAP Header	Transaction Identifier	00H (High Byte)
		00H (Low Byte)
	Protocol identifier	00H (High Byte)
		00H (Low Byte)
	Length	00H (High Byte)
		07H (Low Byte)
	Unit Identifier	00H
PDU Message	Function Code	04H
	Byte Count	04H
	Register 1	40H (High Byte)
		E2H (Low Byte)
	Register 2	55H (High Byte)
		63H (Low Byte)

5.1.3 Function Code 10H – Write Holding Register

Writes data to multiple holding registers. A maximum of 123 registers can be written per request.

Possible exception codes: 01, 02, 03, 04.

The following example shows the master sending a write command to slave address 1, starting from address 1001 (03E9H), writing 2 registers with values 4316H and 0000H. Specifically, 4316H is written to address 1001, and 0000H to address 1002. The slave responds after completing the write operation.

RTU Mode

Master Command Information

Slave Address	01H
Function Code	10H
Starting Address	03H
	E9H
Quantity of Registers	00H (High Byte)
	02H (Low Byte)
Quantity of Registers	04H
Register 1	43H (High Byte)
	16H (Low Byte)
Register 2	00H (High Byte)
	00H (Low Byte)
CRC (Low Byte)	DDH
CRC (High Byte)	3DH

Slave Response Message

Slave Address	01H
Function Code	10H
Starting Address	03H (High Byte)
	E9H (Low Byte)
Quantity of Registers	00H (High Byte)
	02H (Low Byte)
CRC (Low Byte)	90H
CRC (High Byte)	78H

The master command contains two registers with combined value 43160000H (transmitted as 4316H, 0000H in big-endian order). Interpreted as a 32-bit IEEE 754 float, this value is 150.0, which sets the instrument's voltage range to 150 V upon execution.

ASCII Mode

Master Command Information

Start	':'
Slave Address	'0'
	'1'
Function Code	'1'
	'0'
Starting Address	'0'
	'3'
	'E'
	'9'
Quantity of Registers	'0'
	'0'
	'0'
	'2'
Byte Count	'0'
	'4'
Register 1	'4'
	'3'
	'1'
	'6'
Register 2	'0'
	'0'
	'0'
	'0'
LRC Check code	'A'
	'4'
End1	(0DH)(CR)
End0	(0AH)(LF)

Slave Response Message

Start	':'
Slave Address	'0'
	'1'
Function Code	'1'
	'0'
Starting Address	'0'
	'3'
	'E'
	'9'
Quantity of Registers	'0'
	'0'
	'0'
	'2'
LRC Check code	'0'
	'1'
End1	(0DH)(CR)
End0	(0AH)(LF)

TCP Mode

Master Command Information

MBAP Header	Transaction Identifier	00H (High Byte)
		00H (Low Byte)
	Protocol identifier	00H (High Byte)
		00H (Low Byte)
	Length	00H (High Byte)
		0BH (Low Byte)
	Unit Identifier	00H
PDU Message	Function Code	10H
	Starting Address	03H (High Byte)
		E9H (Low Byte)
	Quantity of Registers	00H (High Byte)
		02H (Low Byte)
	Byte Count	04H
	Register 1	43H (High Byte)
		16H (Low Byte)
	Register 2	00H (High Byte)
		00H (Low Byte)

Slave Response Message

MBAP Header	Transaction Identifier	00H (High Byte)
		00H (Low Byte)
	Protocol identifier	00H (High Byte)
		00H (Low Byte)
	Length	00H (High Byte)
		06H (Low Byte)
	Unit Identifier	00H
	Function Code	10H

PDU Message	Starting Address	03H (High Byte)
		E9H (Low Byte)
	Quantity of Registers	00H (High Byte)
		02H (Low Byte)

5.1.4 Function Code 03H – Read Holding Registers

Reads holding registers in a contiguous block. Up to 125 registers can be read by request.

Possible exception codes: 01, 02, 03, 04.

The following example shows the master sending a command to slave address 1 to read 2 consecutive registers starting from address 1017 (03F9H).

RTU Mode

Master Command Information

Slave Address	01H
Function Code	03H
Starting Address	03H (High Byte)
	F9H (Low Byte)
Quantity of Registers	00H (High Byte)
	02H (Low Byte)
CRC (Low Byte)	14H
CRC (High Byte)	7EH

Slave Response Message

Slave Address	01H
Function Code	03H
Byte Count	04H
Register 1	41H (High Byte)
	20H (Low Byte)
Register 2	00H (High Byte)
	00H (Low Byte)
CRC (Low Byte)	EFH
CRC (High Byte)	C5H

The slave response message contains two registers with combined value 41200000H (4120H, 0000H). Interpreted as a 32-bit IEEE 754 float, this value is 10.0, consistent with the external sensor scaling factor of Element 1.

ASCII Mode

Master Command Information

Start	':'
Slave Address	'0'
	'1'
Function Code	'0'
	'3'
Starting Address	'0'
	'3'
	'F'
	'9'
Quantity of Registers	'0'
	'0'
	'0'
	'2'
LRC Check code	'F'
	'E'
End1	(0DH)(CR)
End0	(0AH)(LF)

Slave Response Message

Start	':'
Slave Address	'0'
	'1'
Function Code	'0'
	'3'
Byte Count	'0'
	'4'
Register 1	'4'
	'1'
	'2'
	'0'
Register 2	'0'
	'0'
	'0'
	'0'
LRC Check code	'9'
	'7'
End1	(0DH)(CR)
End0	(0AH)(LF)

TCP Mode

Master Command Information

MBAP Header	Transaction Identifier	00H (High Byte)
		00H (Low Byte)
	Protocol identifier	00H (High Byte)
		00H (Low Byte)
	Length	00H (High Byte)
		06H (Low Byte)
	Unit Identifier	00H
PDU Message	Function Code	03H
	Starting Address	03H (High Byte)
		F9H (Low Byte)
	Quantity of Registers	00H (High Byte)
		02H (Low Byte)

Slave Response Message

MBAP Header	Transaction Identifier	00H (High Byte)
		00H (Low Byte)
	Protocol identifier	00H (High Byte)
		00H (Low Byte)
	Length	00H (High Byte)
		07H (Low Byte)
	Unit Identifier	00H
PDU Message	Function Code	03H
	Byte Count	04H
	Register 1	41H (High Byte)
		20H (Low Byte)
	Register 2	00H (High Byte)
		00H (Low Byte)

5.1.5 Slave Exception Response

If the master command contains an error that prevents normal slave operation, the slave returns an exception response. The following illustrates the RTU exception response format for the Read Holding Registers function.

Slave Address	01H
Error Code	83H
Exception Code	02H
CRC (Low Byte)	C0H
CRC (High Byte)	F1H

The error code equals the received function code plus 80H.

The exception code specifies the cause of the error, see table below for details.

Exception Code	Name	Description
01	Illegal Function	The slave does not support the requested function code.
02	Illegal Data Address	The starting data address received by the slave, or the combination of the starting data address and the transmitted Quantity of Registers, is invalid.
03	Illegal Data Value	The data received by the slave is invalid, such as incorrect length.
04	Slave Device Failure	An unrecoverable error occurred while the slave was attempting to execute the requested operation.

5.1.6 CRC and LRC Transmission Error Checking

In RTU communication mode, CRC (Cyclic Redundancy Check) is used for error detection, whereas ASCII communication mode utilizes LRC (Longitudinal Redundancy Check). The error-checking algorithms for each mode are described below.

CRC (RTU Mode)

The following C language example demonstrates the calculation of the CRC value and the transmission of a data frame.

```
unsigned short crc_check(unsigned char *data,unsigned char length)
{
    int j;
    unsigned short reg_crc = 0xFFFF;
    while(length--)
    {
        reg_crc^=*data++;
        for(j=0;j<8;j++)
        {
            if(reg_crc&0x01)
            {
                reg_crc = (reg_crc>>1)^0xA001;
            }
            else
            {
                reg_crc = (reg_crc>>1);
            }
        }
    }
    return reg_crc;
}

unsigned char SendBuf[30];
void main(void)
{
     unsigned short CRC;
     unsigned char SendLen;
```

```
SendLen = 0;

SendBuf[ SendLen++ ] = 0x01;

SendBuf[ SendLen++ ] = 0x04;

SendBuf[ SendLen++ ] = 0x00;

SendBuf[ SendLen++ ] = 0x64;

SendBuf[ SendLen++ ] = 0x00;

SendBuf[ SendLen++ ] = 0x02;

CRC = crc_check(SendBuf,SendLen); /*Calculate CRC */

SendBuf[ SendLen++ ] = CRC&0xFF; /* CRC Low byte */

SendBuf[ SendLen++ ] = (CRC>>8)&0xFF; /* CRC High byte */

uart_send(SendBuf,SendLen);    /*Send a data frame */
```

LRC (ASCII Mode)

Start	':'
Slave Address	'0'
	'1'
Function Code	'0'
	'3'
Starting Address	'0'
	'3'
	'F'
	'9'
Quantity of Registers	'0'
	'0'
	'0'
	'2'
LRC Check code	'F'
	'E'
End1	(0DH)(CR)
End0	(0AH)(LF)

LRC Algorithm Description

The LRC algorithm is calculated as follows: Sum all bytes, discard any carry, and then compute the 2's complement of the result.

Example:

01H + 03H + 03H + F9H + 00H + 02H = 102H

Discard the carry (1), retaining only 02H.

Compute the 2's complement of 02H → FEH.

5.1.7 Floating-Point and Word Conversion

In the Modbus protocol, each register is 16 bits, equivalent to one word. As noted earlier, a floating-point number occupies two registers, or two words.

After receiving byte data, the user must convert words to floating-point numbers or converting floating-point numbers to words.

The following code examples demonstrate the conversion between floating-point values and words.

```
/* C Program to Convert a Floating-Point Number into Two Words */
void FloatToWord(float Data,unsigned short *Word)
{
    union
    {
        float data;
        unsigned char Byte[4];
    }FloatData;
    FloatData.Data=Data;
    Word[0]=(u16)FloatData.Byte[3]<<8|FloatData.Byte[2];
    Word[1]=(u16)FloatData.Byte[1]<<8|FloatData.Byte[0];
}
/* C Program to Convert Two Words into a Floating-Point Number */
float WordToFloat(const unsigned short *Word)
```

```
{
    union
    {
        float data;
        unsigned char Byte[4];
    }FloatData;
    FloatData.Byte[3]=(Word[0]>>8)&0xFF;
    FloatData.Byte[2]=(Word[0])&0xFF;
    FloatData.Byte[1]=(Word[1]>>8)&0xFF;
    FloatData.Byte[0]=(Word[1])&0xFF;
    return FloatData.Data;
}
```

5.2 Register Allocation Table

	Address	Description
Input Register	0 to 14	Measurement data, status (data unrelated to status, input units, or wiring configurations)
	100 to 193	Input Unit 1: General measurement data & summary harmonic measurement data
	200 to 293	Input Unit 2: General measurement data & summary harmonic summary measurement data
	300 to 393	Input Unit 3: General measurement data & summary harmonic measurement data
	400 to 447	Unit ∑: General measurement data
	2000 to 2509	Measurement data: Mapped to communication output item (:NUMeric[:NORMal]:ITEM<X>)
	4000 to 4799	Input Unit 1: Harmonic list data
	5000 to 5799	Input Unit 2: Harmonic list data

	6000 to 6799	Input Unit 3: Harmonic list data
Holding Register	0 to 10	Control data (:NUMeric:HOLD, integration control)
	100 to 509	Configuration of communication output item (:NUMeric[:NORMal]:ITEM<X>)
	1000 to 1101	Measurement control data

When a PLC (Programmable Logic Controller) or HMI (Human-Machine Interface) needs to control the power meter via the Modbus protocol, the raw register addresses of the power meter cannot be used directly. The register memory addresses in the PLC and HMI must be mapped according to the following relationships:

PLC/HMI Input Register Address = Power Meter Input Register Address + 30001

PLC/HMI Holding Register Address = Power Meter Holding Register Address + 40001

Example: To read the Update Count from the HMI, set the memory address of the control element to 30001.

5.3 Input Register Address Mapping Table

Measurement Data, Status: Data Unrelated to Status, Input Units, or Wiring Configuration (Input Register)

Data Name	Description	Data Format	Unit	Register Start Address	Register Count
Update Count	0 to 65535	uint16		0	1
				1	1
Peak Over Range	Peak overflow status	uint16		2	1
Check Range	Range monitoring status	uint16		3	1
URange	Voltage range	float	V	4	2
IRange	Current range	float	A	6	2
MATH	Calculation results	float		8	2
fPLL	PLL sync source	float	Hz	10	2
EFFiciency	Efficiency	float	%	12	2

Integration Mode Status	0: RESET, 1: STOP, 2: START, 3: ERROR, 4: TIMEUP	uint16		14	1

Input Unit 1 General Measurement Data (Input Register)

Data Name	Description	Data Format	Unit	Register Start Address	Register Count
U1	Voltage	float	V	100	2
I1	Current	float	A	102	2
P1	Active power	float	W	104	2
S1	Apparent power	float	VA	106	2
Q1	Reactive power	float	var	108	2
Lambda1	Power factor (PF)	float		110	2
Phi1	Phase difference	float	°	112	2
fU1	Voltage frequency	float	Hz	114	2
fI1	Current frequency	float	Hz	116	2
U+pk1	Positive voltage peak	float	V	118	2
U-pk1	Negative voltage peak	float	V	120	2
I+pk1	Positive current peak	float	A	122	2
I-pk1	Negative current peak	float	A	124	2
P+pk1	Positive power peak	float	W	126	2
P-pk1	Negative power peak	float	W	128	2
Time1	Integration running time	float	S	130	2
WP1	Power integral value (Watt-hour)	float	Wh	132	2
WP+1	Positive power integral value (Positive Watt-hour)	float	Wh	134	2
WP-1	Negative power integral value (negative Watt-hour)	float	Wh	136	2
q1	Current integral value (Ampere-hour)	float	Ah	138	2

q+1	Positive current integral value (Positive Ampere-hour)	float	Ah	140	2
q-1	Negative current integral value (Negative Ampere-hour)	float	Ah	142	2
Urms1	Voltage RMS	float	V	144	2
Umn1	Rectified average voltage calibrated to RMS	float	V	146	2
Udc1	Simple average voltage	float	V	148	2
Urmn1	Rectified average voltage	float	V	150	2
Uac1	AC RMS	float	V	152	2
Irms1	Current RMS	float	A	154	2
Imn1	Rectified average current calibrated to RMS	float	A	156	2
Idc1	DC current (simple average)	float	A	158	2
Irmn1	Rectified average current	float	A	160	2
Iac1	AC current	float	A	162	2
CfU1	Measured voltage crest factor	float		164	2
CfI1	Measured current crest factor	float		166	2
AVP1	Average integrated power	float	W	168	2

Input Unit 1 Summary Harmonic Measurement Data (Input Register)

Data Name	Description	Data Format	Unit	Register Start Address	Register Count
U1(Total)	Total harmonic voltage	float	V	170	2
U1(1)	Fundamental voltage	float	V	172	2
I1(Total)	Total harmonic current	float	A	174	2
I1(1)	Fundamental current	float	A	176	2
P1(Total)	Total harmonic power	float	W	178	2
P1(1)	Fundamental power	float	W	180	2
Lambda1(1)	Fundamental power factor	float		182	2

Phi1(1)	Phase difference between fundamental voltage and fundamental current	float	°	184	2
PhiU1(3)	Phase difference between fundamental voltage and 3rd harmonic voltage	float	°	186	2
PhiI1(3)	Phase difference between fundamental current and 3rd harmonic voltage	float	°	188	2
UTHD1	Total harmonic distortion of voltage	float	%	190	2
ITHD1	Total harmonic distortion of current	float	%	192	2

Input Unit 2 General Measurement Data (Input Register)

Data Name	Description	Data Format	Unit	Register Start Address	Register Count
U2	Voltage	float	V	200	2
I2	Current	float	A	202	2
P2	Active power	float	W	204	2
S2	Apparent power	float	VA	206	2
Q2	Reactive power	float	var	208	2
Lambda2	Power factor (PF)	float		210	2
Phi2	Phase difference	float	°	212	2
fU2	Voltage frequency	float	Hz	214	2
fI2	Current frequency	float	Hz	216	2
U+pk2	Positive voltage peak	float	V	218	2
U-pk2	Negative voltage peak	float	V	220	2
I+pk2	Positive current peak	float	A	222	2
I-pk2	Negative current peak	float	A	224	2
P+pk2	Positive power peak	float	W	226	2
P-pk2	Negative power peak	float	W	228	2
Time2	Integration running time	float	S	230	2

WP2	Power integral value (Watt-hour)	float	Wh	232	2
WP+2	Positive power integral value (Positive Watt-hour)	float	Wh	234	2
WP-2	Negative power integral value (Negative Watt-hour)	float	Wh	236	2
q2	Current integral value (Ampere-hour)	float	Ah	238	2
q+2	Positive current integral value (Positive Ampere-hour)	float	Ah	240	2
q-2	Negative current integral value (Negative Ampere-hour)	float	Ah	242	2
Urms2	Voltage RMS	float	V	244	2
Umn2	Rectified average voltage calibrated to RMS	float	V	246	2
Udc2	Simple average voltage	float	V	248	2
Urmn2	Rectified average voltage	float	V	250	2
Uac2	AC voltage	float	V	252	2
Irms2	Current RMS	float	A	254	2
Imn2	Rectified average current calibrated to RMS	float	A	256	2
Idc2	DC current (simple average)	float	A	258	2
Irmn2	Rectified average current	float	A	260	2
Iac2	AC current	float	A	262	2
CfU2	Measured voltage crest factor	float		264	2
CfI2	Measured current crest factor	float		266	2
AVP2	Average integrated power	float	W	268	2

Input Unit 2 Summary Harmonic Measurement Data (Input Register)

Data Name	Description	Data Format	Unit	Register Start Address	Register Count
U2(Total)	Total harmonic voltage	float	V	270	2

U2(1)	Fundamental voltage	float	V	272	2
I2(Total)	Total harmonic current	float	A	274	2
I2(1)	Fundamental current	float	A	276	2
P2(Total)	Total harmonic power	float	W	278	2
P2(1)	Fundamental power	float	W	280	2
Lambda2(1)	Fundamental power factor	float		282	2
Phi2(1)	Phase difference between fundamental voltage and fundamental current	float	°	284	2
PhiU2(3)	Phase difference between fundamental voltage and 3rd harmonic voltage	float	°	286	2
PhiI2(3)	Phase difference between fundamental current and 3rd harmonic voltage	float	°	288	2
UTHD2	Total harmonic distortion of voltage	float	%	290	2
ITHD2	Total harmonic distortion of current	float	%	292	2

Input Unit 3 General Measurement Data (Input Register)

Data Name	Description	Data Format	Unit	Register Start Address	Register Count
U3	Voltage	float	V	300	2
I3	Current	float	A	302	2
P3	Active power	float	W	304	2
S3	Apparent power	float	VA	306	2
Q3	Reactive power	float	var	308	2
Lambda3	Power factor (PF)	float		310	2
Phi3	Phase difference	float	°	312	2
fU3	Voltage frequency	float	Hz	314	2
fI3	Current frequency	float	Hz	316	2
U+pk3	Positive voltage peak	float	V	318	2

U-pk3	Negative voltage peak	float	V	320	2
I+pk3	Positive current peak	float	A	322	2
I-pk3	Negative current peak	float	A	324	2
P+pk3	Positive power peak	float	W	326	2
P-pk3	Negative power peak	float	W	328	2
Time3	Integration running time	float	S	330	2
WP3	Power integral value (Watt-hour)	float	Wh	332	2
WP+3	Positive power integral value (Positive Watt-hour)	float	Wh	334	2
WP-3	Negative power integral value (Negative Watt-hour)	float	Wh	336	2
q3	Current integral value (Ampere-hour)	float	Ah	338	2
q+3	Positive current integral value (Positive Ampere-hour)	float	Ah	340	2
q-3	Negative current integral value (Negative Ampere-hour)	float	Ah	342	2
Urms3	Voltage RMS	float	V	344	2
Umn3	Rectified average voltage calibrated to RMS	float	V	346	2
Udc3	Simple average voltage	float	V	348	2
Urmn3	Rectified average voltage	float	V	350	2
Uac3	AC voltage	float	V	352	2
Irms3	Current RMS	float	A	354	2
Imn3	Rectified average current calibrated to RMS	float	A	356	2
Idc3	DC current (simple average)	float	A	358	2
Irmn3	Rectified average current	float	A	360	2
Iac3	AC current	float	A	362	2
CfU3	Measured voltage crest factor	float		364	2
CfI3	Measured current crest factor	float		366	2

AVP3	Average integrated power	float	W	368	2

Input Unit 3 Summary Harmonic Measurement Data (Input Register)

Data Name	Description	Data Format	Unit	Register Start Address	Register Count
U3(Total)	Total harmonic voltage	float	V	370	2
U3(1)	Fundamental voltage	float	V	372	2
I3(Total)	Total harmonic current	float	A	374	2
I3(1)	Fundamental current	float	A	376	2
P3(Total)	Total harmonic power	float	W	378	2
P3(1)	Fundamental power	float	W	380	2
Lambda3(1)	Fundamental power factor	float		382	2
Phi3(1)	Phase difference between fundamental voltage and fundamental current	float	°	384	2
PhiU3(3)	Phase difference between fundamental voltage and 3rd harmonic voltage	float	°	386	2
PhiI3(3)	Phase difference between fundamental current and 3rd harmonic voltage	float	°	388	2
UTHD3	Total harmonic distortion of voltage	float	%	390	2
ITHD3	Total harmonic distortion of current	float	%	392	2

Unit ∑ General Measurement Data (Input Register)

Data Name	Description	Data Format	Unit	Register Start Address	Register Count
U∑	Voltage	float	V	400	2
I∑	Current	float	A	402	2
P∑	Active power	float	W	404	2

S∑	Apparent power	float	VA	406	2
Q∑	Reactive power	float	var	408	2
Lambda∑	Power factor (PF)	float		410	2
Phi∑	Phase difference	float	°	412	2
WP∑	Power integral value (Watt-hour)	float	Wh	414	2
WP+∑	Positive power integral value (Positive Watt-hour)	float	Wh	416	2
WP-∑	Negative power integral value (Negative Watt-hour)	float	Wh	418	2
q∑	Current integral value (Ampere-hour)	float	Ah	420	2
q+∑	Positive current integral value (Positive Ampere-hour)	float	Ah	422	2
q-∑	Negative current integral value (Negative Ampere-hour)	float	Ah	424	2
Urms∑	Voltage RMS	float	V	426	2
Umn∑	Rectified average voltage calibrated to RMS	float	V	428	2
Udc∑	Simple average voltage	float	V	430	2
Urmn∑	Rectified average voltage	float	V	432	2
Uac∑	AC voltage	float	V	434	2
Irms∑	Current RMS	float	A	436	2
Imn∑	Rectified average current calibrated to RMS	float	A	438	2
Idc∑	DC current (simple average)	float	A	440	2
Irmn∑	Rectified average current	float	A	442	2
Iac∑	AC current	float	A	444	2
AVP∑	Average integrated power	float	W	446	2

Measurement data: Mapped to Communication Output Item (:NUMeric[:NORMal]:ITEM<X>)

Data Name	Description	Data Format	Unit	Register Start Address	Register Count
ItemX	Measurement data mapped to ItemX	Register Start Address: 2000+(X-1)*2			
Item1	Measurement data mapped to Item1	float		2000	2
Item2	Measurement data mapped to Item2	float		2002	2
Item3	Measurement data mapped to Item3	float		2004	2
to					
Item255	Measurement data mapped to Item255	float		2508	2

Input Unit 1 Harmonic List Data

Data Name	Description	Data Format	Unit	Register Start Address	Register Count
U1(1) to U1(50)	Voltage measurements for 1st to 50th harmonics	float	V	4000	100
I1(1) to I1(50)	Current measurements for 1st to 50th harmonics	float	A	4100	100
P1(1) to P1(50)	Power measurements for 1st to 50th harmonics	float	W	4200	100
UHDF1(1) to UHDF1(50)	Voltage distortion factors for 1st to 50th Harmonics	float	%	4300	100
IHDF1(1) to IHDF1(50)	Current distortion factors for 1st to 50th Harmonics	float	%	4400	100
PHDF1(1) to PHDF1(50)	Power distortion factors for 1st to 50th Harmonics	float	%	4500	100
PHI1(1), PHIU1(2) to PHIU1(50)	PHI1(1): Phase difference between fundamental voltage and fundamental current	float	°	4600	100

	PHIU1(2) to PHIU1(50): Phase difference the Kth voltage harmonic and fundamental voltage				
PHI1(1), PHII1(2) to PHII1(50)	PHI1(1): Phase difference between fundamental voltage and fundamental current PHII1(2) to PHII1(50): Phase difference the Kth current harmonic and fundamental current	float	°	4700	100

Input Unit 2 Harmonic List Data

Data Name	Description	Data Format	Unit	Register Start Address	Register Count
U2(1) to U2(50)	Voltage measurements for 1st to 50th harmonics	float	V	5000	100
I2(1) to I2(50)	Current measurements for 1st to 50th harmonics	float	A	5100	100
P1(1) to P1(50)	Power measurements for 1st to 50th harmonics	float	W	5200	100
UHDF2(1) to UHDF2(50)	Voltage distortion factors for 1st to 50th Harmonics	float	%	5300	100
IHDF2(1) to IHDF2(50)	Current distortion factors for 1st to 50th Harmonics	float	%	5400	100
PHDF2(1) to PHDF2(50)	Power distortion factors for 1st to 50th Harmonics	float	%	5500	100
PHI2(1), PHIU2(2) to PHIU2(50)	PHI2(1): Phase difference between fundamental voltage and fundamental current PHIU2(2) to PIU2(50): Phase difference the Kth voltage harmonic and fundamental voltage	float	°	5600	100

PHI2(1), PHII2(2) to PHII2(50)	PHI2(1): Phase difference between fundamental voltage and fundamental current PHII2(2) to PHII2(50): Phase difference the Kth current harmonic and fundamental current	float	°	5700	100

Input Unit 3 Harmonic List Data

Data Name	Description	Data Format	Unit	Register Start Address	Register Count
U3(1) to U3(50)	Voltage measurements for 1st to 50th harmonics	float	V	6000	100
I3(1) to I3(50)	Current measurements for 1st to 50th harmonics	float	A	6100	100
P3(1) to P3(50)	Power measurements for 1st to 50th harmonics	float	W	6200	100
UHDF3(1) to UHDF3(50)	Voltage distortion factors for 1st to 50th Harmonics	float	%	6300	100
IHDF3(1) to IHDF3(50)	Current distortion factors for 1st to 50th Harmonics	float	%	6400	100
PHDF3(1) to PHDF3(50)	Power distortion factors for 1st to 50th Harmonics	float	%	6500	100
PHI3(1), PHIU3(2) to PHIU3(50)	PHI3(1): Phase difference between fundamental voltage and fundamental current PHIU3(2) to PHIU3(50): Phase difference the Kth voltage harmonic and fundamental voltage	float	°	6600	100
PHI3(1), PHII3(2) to PHII3(50)	PHI3(1): Phase difference between fundamental voltage and fundamental current	float	°	6700	100

	PHII2(2) to PHII2(50): Phase difference the Kth current harmonic and fundamental current				

Peak Overflow Status (Input register: 0002)

The peak overrange status of each input channel is represented by individual bits. The bit corresponding to an input that has experienced a peak overrange is set to 1.

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

Range Monitoring Status (Input register: 0003)

The range monitoring status displayed on the instrument is represented by individual bits. For details, refer to the description of the command :INPut:CRANge? in Section 3.5.32.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
								AP	AO	AH	AL	VP	VO	VH	VL

Floating-Point Data

IEEE single-precision floating-point format. Data under error conditions is represented as follows:

- Data not available (displayed as "-----"):

  NAN 0x7FC00000

- Out of range (displayed as "--OL-"), Calculation out of range (displayed as "--OF-"), Error (displayed as "error"):

  INF 0x7F800000

5.4 Holding Register Address Mapping Table

Control Data (Holding Register)

Data Name	Description	Data Format	Unit	Register Start Address	Register Count
NUMeric:HOLD	0: Not held	uint16		0	1

	1: Hold				
				1	1
INTEG:START/ STOP	0: Stop 1: Start	uint16		2	1
INTEG:RESET	1: Reset Non-zero values: Invalid	uint16		3	1
				4	1
				5	1
				6	1
				7	1
				8	1
				9	1

Configuration of Communication Output Item (:NUMeric[:NORMal]:ITEM<X>)(Holding Register)

Data Name	Description	Data Format	Unit	Register Start Address	Register Count
ItemX	Configuration for Output ItemX	Register Start Address:100+(X-1)*2			
Item1	Configuration for Output Item1	uint32		100	2
Item2	Configuration for Output Item2	uint32		102	2
Item3	Configuration for Output Item3	uint32		104	2
to					
Item255	Configuration for Output Item255	uint32		608	2

Configuration Method for ItemX

- **bit[7:0]: Set Item Function**

  0: U, 1:I, 2:P, 3:S, 4:Q, 5:LAMB, 6:PHI, 7:FU, 8:FI, 9:UPP, 10:UMP, 11:IPP, 12:IMP, 13:PPP, 14:PMP, 15:WH, 16:WHP, 17:WHM, 18:AH,

19:AHP, 20:AHM, 21:URMS, 22:UMN, 23:UDC, 24:URMN, 25:UAC, 26:IRMS, 27:IMN, 28:IDC, 29:IRMN, 30:IAC, 31:CFU, 32:CFI, 33:AVP, 34:UK, 35:IK, 36:PK, 37:PHIUK, 38:PHIIK, 39:UHDFK, 40:IHDFK, 41:PHDFK, 42:UTHD, 43:ITHD, 44:LAMBDAK, 45:PHIK, 46:FPLL, 47:MATH, 48:URAN, 49:IRAN, 50:EFFI, 51:TIME, 52: NONE

- **bit[15:8]: Set Item Element**

  0: [Element 1], 1: [Element 2], 2: [Element 3], 3: [Sigma]

- **bit[23:16]: Set Harmonic Order**

  0: DC, 1 to 50: Harmonic Order, 51: Total

  Example: To configure Item1 as the fundamental current of Input Unit 2, set the value of Item1 as follows:

  Item1 = 35+(1<<8)+(1<<16) = 65827(10123H)

Measurement Control Data (Holding Register)

Data Name	Description	Data Format	Unit	Register Start Address	Register Count
Peak Factor	0: CF=3, 1: CF=6, 2: CF=6A	uint16		1000	1
Voltage Range	0: Auto	float	V	1001	2
External Current Sensor	0: OFF, 1: EXT1, 2: EXT2	uint16		1003	1
Current Range	0: Auto When the external current sensor is set to 0, indicates direct current input; the unit is A. When the external current sensor is set to 1 or 2, the current is measured using the external current sensor; the unit is V.	float	A or V	1004	2
Data Update Interval	0: [0.1s], 1: [0.25s], 2: [0.5s], 3: [1s], 4: [2s], 5: [5s], 6: [10s], 7: [20s], 8: [Auto]	uint16		1006	1
Data Update Timeout	The data update interval is configurable in auto mode:	uint16		1007	1

	0: [1s], 1: [5s], 2: [10s], 3: [20s]				
Data Update Sync Source	The data update interval is configurable in auto mode: 0: [u1], 1: [i1], 2: [u2], 3: [i2], 4: [u3], 5: [i3]	uint16		1008	1
Sync Source	0: [V], 1: [A], 2: [OFF]	uint16		1009	1
Line Filter	0: [OFF], 1: [500Hz], 2: [1KHz], 3: [2KHz], 4: [3KHz], 5: [4KHz], 6: [5KHz], 7: [5.5KHz], 8: [6KHz], 9: [7KHz], 10: [8KHz], 11: [9KHz], 12: [10KHz], 13: [20KHz], 14: [30KHz], 15: [40KHz], 16: [50KHz], 17: [60KHz], 18: [70KHz], 19: [80KHz], 20: [90KHz], 21: [100KHz]	uint16		1010	1
Frequency Filter	0: OFF, 1: ON	uint16		1011	1
Test Mode	0: RMS, 1: VMean, 2: DC	uint16		1012	1
Wiring Connection	0: 1P3W, 1: 3P3W, 2: 3P4W, 3: 3V3A	uint16		1013	1
Averaging Switch	0: OFF, 1: ON	uint16		1014	1
Averaging Type	0: LIN, 1: EP	uint16		1015	1
Averaging Factor	0:[8], 1:[16], 2:[32], 3:[64]	uint16		1016	1
External Current Sensor Ratio for Element 1	0.001 to 9999.999	float	mV/A	1017	2
External Current Sensor Ratio for Element 2	0.001 to 9999.999	float	mV/A	1019	2
External Current Sensor Ratio for Element 3	0.001 to 9999.999	float	mV/A	1021	2
Ratio Control	0: OFF, 1: ON	uint16		1023	2

Voltage Scale Factor for Element 1	0.001 to 9999.999	float		1024	2
Voltage Scale Factor for Element 2	0.001 to 9999.999	float		1026	2
Voltage Scale Factor for Element 3	0.001 to 9999.999	float		1028	2
Current Scale Factor for Element 1	0.001 to 9999.999	float		1030	2
Current Scale Factor for Element 2	0.001 to 9999.999	float		1032	2
Current Scale Factor for Element 3	0.001 to 9999.999	float		1034	2
Power Scale Factor for Element 1	0.001 to 9999.999	float		1036	2
Power Scale Factor for Element 2	0.001 to 9999.999	float		1038	2
Power Scale Factor for Element 3	0.001 to 9999.999	float		1040	2
Integration Time	0 to 36000000 (10000 hours)	uint32	s	1042	2
Integration Mode	0: Standard, 1: Cycle	uint16		1044	1

Hold	0: OFF, 1: ON	uint16		1045	1
Max. Hold	0: OFF, 1: ON	uint16		1046	1
Single	Set to 1 to execute. Read-back value is always 0.	uint16		1047	1
Math Formula	0: [A+B], 1: [A-B], 2: [A*B], 3: [A/B], 4: [A/(B*B)], 5: [(A*A)/B]	uint16		1048	1
Math Item A	bit[7:0] Function: 0: U, 1: A, 2: W, 3: VA, 4: var 5:PF bit[15:8] Element: 0: [Element 1], 1: [Element 2], 2: [Element 3], 3: [Sigma]	uint16		1049	1
Math Item B	bit[7:0] Function: 0: U; 1: A; 2: W; 3: VA; 4: var 5:PF bit[15:8] Element: 0: [Element 1], 1: [Element 2], 2: [Element 3], 3: [Sigma]	uint16		1050	1
Harmonic Test Mode	0: Normal, 1: IEC	uint16		1051	1
Harmonic PLL SYNC Source	0: [u1], 1: [i1], 2: [u2], 3: [i2], 4: [u3], 5: [i3]	uint16		1052	1
Harmonic Order	1 to 50th	uint16		1053	1
THD Formula	0: IEC, 1: CSA	uint16		1054	1
Range Switch	0: OFF, 1: ON	uint16		1055	1
Voltage Allowable Range	Effective when range configuration enable is set to 1(ON).	uint16		1056	1
Voltage Peak Auto-Range	Effective when range configuration enable is set to 1(ON).	uint16		1057	1
Current Allowable Range	Effective when range configuration enable is set to 1(ON).	uint16		1058	1
Current Peak Auto-Range	Effective when range configuration enable is set to 1(ON).	uint16		1059	1

EXT1 Allowable Range	Effective when range configuration enable is set to 1(ON).	uint16		1060	1
EXT1 Peak Auto-Range	Effective when range configuration enable is set to 1(ON).	uint16		1061	1
EXT2 Allowable Range	Effective when range configuration enable is set to 1(ON).	uint16		1062	1
EXT2 Peak Auto-Range	Effective when range configuration enable is set to 1(ON).	uint16		1063	1
Measurement Setup Initialize	Set to 1 to execute. Read-back value is always 0. Wait 5 seconds after execution before subsequent operations.	uint16		1064	1
Zero Level Compensation	Set to 1 to execute. Read-back value is always 0.	uint16		1065	
Remote Status	0: OFF, 1: ON	uint16		1100	1
Key Lock Status	0: OFF, 1: ON	uint16		1101	1
Preset Communication Output Items	1 to 4. Read-back value is always 0. Reference command :NUMeric[:NORMal]:PRESet	uint16		1102	1

Voltage Allowable Range (Holding Register Address: 1056)

Each bit represents the valid state of a range: bit=0: OFF, bit=1: ON.

For CF=3:

bit0: 15V, bit1: 30V...bit5: 600V,bit6:1000V[1]

For CF=6 or 6A:

bit0: 7.5V, bit1: 15V...bit5: 300V,bit6:500V[1]

Voltage Peak Auto-Range (Holding Register Address: 1057)

For CF=3:

0: [OFF], 1: [15V], 2: [30V]...6: [600V],7: [1000V][1]

For CF=6 or 6A:

0: [OFF], 1: [7.5V], 2: [15V]...6: [300V],7: [500V][1]

Current Allowable Range (Holding Register Address: 1058)

Each bit represents the valid state of a range: bit=0: OFF, bit=1: ON.

For CF=3:

bit0: 0.5A, bit1: 1A...bit5: 20A

For CF=6 or 6A:

bit0: 0.25A, bit1: 0.5A...bit5: 10A

Current Peak Auto-Range (Holding Register Address: 1059)

For CF=3:

0: [OFF], 1: [0.5A], 2: [1A]...6: [20A]

For CF=6 or 6A:

0: [OFF], 1: [0.25A], 2: [0.5A]...6: [10A]

EXT1 Allowable Range (Holding Register Address: 1060)

Each bit represents the valid state of a range: bit=0: OFF, bit=1: ON.

For CF=3:

bit0: 2.5V, bit1: 5V...bit2: 10V

For CF=6 or 6A:

bit0: 2.5V, bit1: 5V...bit2: 10V

EXT1 Peak Auto-Range (Holding Register Address: 1061)

For CF=3:

0: [OFF], 1: [2.5V], 2: [5V],3: [10V]

For CF=6 or 6A:

0: [OFF], 1: [1.25V], 2: [2.5V], 3: [5V]

EXT2 Allowable Range (Holding Register Address: 1062)

Each bit represents the valid state of a range: bit=0: OFF, bit=1: ON.

For CF=3:

bit0: 50mV, bit1: 100mV...bit5: 2V

For CF=6 or 6A:

bit0: 25mV, bit1: 50mV...bit5: 1V

EXT2 Peak Auto-Range (Holding Register Address: 1063)

For CF=3:

0: [OFF], 1: [50mV], 2: [100mV]...6: [2V]

For CF=6 or 6A

0: [OFF], 1: [25mV], 2: [50mV]...6: [1V]

Note: [1] This range is only applicable to UTE333H and UTE333H-2K.