

**NORMA 6003/NORMA 6003+/  
NORMA 6004/NORMA 6004+**  
Power Analyzers

Programmers Reference Guide



June 2020

© 2020 Fluke Corporation. All rights reserved. Specifications are subject to change without notice.  
All product names are trademarks of their respective companies.

## **LIMITED WARRANTY AND LIMITATION OF LIABILITY**

Each Fluke product is warranted to be free from defects in material and workmanship under normal use and service. The warranty period is two years and begins on the date of shipment. Parts, product repairs, and services are warranted for 90 days. This warranty extends only to the original buyer or end-user customer of a Fluke authorized reseller, and does not apply to fuses, disposable batteries, or to any product which, in Fluke's opinion, has been misused, altered, neglected, contaminated, or damaged by accident or abnormal conditions of operation or handling. Fluke warrants that software will operate substantially in accordance with its functional specifications for 90 days and that it has been properly recorded on non-defective media. Fluke does not warrant that software will be error free or operate without interruption.

Fluke authorized resellers shall extend this warranty on new and unused products to end-user customers only but have no authority to extend a greater or different warranty on behalf of Fluke. Warranty support is available only if product is purchased through a Fluke authorized sales outlet or Buyer has paid the applicable international price. Fluke reserves the right to invoice Buyer for importation costs of repair/replacement parts when product purchased in one country is submitted for repair in another country.

Fluke's warranty obligation is limited, at Fluke's option, to refund of the purchase price, free of charge repair, or replacement of a defective product which is returned to a Fluke authorized service center within the warranty period.

To obtain warranty service, contact your nearest Fluke authorized service center to obtain return authorization information, then send the product to that service center, with a description of the difficulty, postage and insurance prepaid (FOB Destination). Fluke assumes no risk for damage in transit. Following warranty repair, the product will be returned to Buyer, transportation prepaid (FOB Destination). If Fluke determines that failure was caused by neglect, misuse, contamination, alteration, accident, or abnormal condition of operation or handling, including overvoltage failures caused by use outside the product's specified rating, or normal wear and tear of mechanical components, Fluke will provide an estimate of repair costs and obtain authorization before commencing the work. Following repair, the product will be returned to the Buyer transportation prepaid and the Buyer will be billed for the repair and return transportation charges (FOB Shipping Point).

**THIS WARRANTY IS BUYER'S SOLE AND EXCLUSIVE REMEDY AND IS IN LIEU OF ALL OTHER WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY IMPLIED WARRANTY OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. FLUKE SHALL NOT BE LIABLE FOR ANY SPECIAL, INDIRECT, INCIDENTAL OR CONSEQUENTIAL DAMAGES OR LOSSES, INCLUDING LOSS OF DATA, ARISING FROM ANY CAUSE OR THEORY.**

Since some countries or states do not allow limitation of the term of an implied warranty, or exclusion or limitation of incidental or consequential damages, the limitations and exclusions of this warranty may not apply to every buyer. If any provision of this Warranty is held invalid or unenforceable by a court or other decision-maker of competent jurisdiction, such holding will not affect the validity or enforceability of any other provision.

Fluke Corporation  
P.O. Box 9090  
Everett, WA 98206-9090  
U.S.A.

# Table of Contents

Title	Page
Introduction .....	1
How to Contact Fluke.....	1
Safety Information.....	1
Symbols and Syntax .....	2
Hardware Interfaces.....	6
Interface Specification .....	7
Switch between Remote and Local Modes.....	7
Switch from Local to Remote Mode.....	7
Switch from Remote Mode to Local.....	7
Commands and Responses.....	8
Introduction to SCPI.....	8
Structure of Commands.....	8
Long and Short Form.....	9
RS485 Command's Prefix .....	9
Multiple Commands Per Line .....	9
HDLC Protocol (High-Level Data Link Control) .....	10
Formatted Resolution String Format .....	11
Communication Commands.....	13
FORMula Commands .....	18
[FORMula:]HVF:MOTOr:TYPE?.....	18
[FORMula:]HVF:MOTOr:TYPE {} .....	18
[FORMula:]HVF:NOMinal:TYPE?.....	18
[FORMula:]HVF:NOMinal:TYPE {} .....	18
[FORMula:]HVF:NOMinal:VALue? .....	18
[FORMula:]HVF:NOMinal:VALue <> .....	18
[FORMula:]HCF:NOMinal:TYPE? .....	19
[FORMula:]HCF:NOMinal:TYPE {} .....	19
[FORMula:]HCF:NOMinal:VALue? .....	19
[FORMula:]HCF:NOMinal:VALue <> .....	19
HARMonics Commands .....	19
HARMonics:ORDer? {} .....	19
HARMonics:DATA? {} .....	20
HARMonics:FFt:ITEM? .....	21
HARMonics:FFt:ITEM {} .....	21
HARMonics:FFt:FREQuency? .....	21
HARMonics:FFt:FREQuency {} .....	21
HARMonics:FFt:DATA? .....	21
HOLD Commands .....	22
HOLD:START .....	22

HOLD:STOP .....	22
HOLD:STATus? .....	22
INPut Commands .....	22
INPut#:CURRent:AUTO? .....	22
INPut#:CURRent:AUTO {} .....	22
INPut#:CURRent:RANGe? .....	23
INPut#:CURRent:RANGe .....	23
INPut#:CURRent:RATIo? .....	23
INPut#:CURRent:RATIo<> .....	23
INPut#:CURRent:FILTer:LINE? .....	23
INPut#:CURRent:FILTer:LINE {} .....	23
INPut#:CURRent:EXTErnal[:SENSor]? .....	24
INPut#:CURRent:EXTErnal[:SENSor] {} .....	24
INPut#:CURRent:EXTErnal[:SENSor]:RATIo? {} .....	24
INPut#:VOLTagE:AUTO? .....	24
INPut#:VOLTagE:AUTO {} .....	24
INPut#:VOLTagE:RANGe? .....	25
INPut#:VOLTagE:RANGe {} .....	25
INPut#:VOLTagE:RATIo? .....	25
INPut#:VOLTagE:RATIo<> .....	25
INPut#:VOLTagE:FILTer:LINE? .....	25
INPut#:VOLTagE:FILTer:LINE {} .....	25
INPut:MS:MODE? .....	26
INPut:MS:MODE {} .....	26
INPut:MS:RATIo? .....	26
INPut:MS:RATIo<> .....	26
INPut:MT:RATIo? .....	26
INPut:MT:RATIo<> .....	26
INPut:MS:MODE:RATIo? {} .....	27
INTEgral Commands .....	27
INTEgral:STARt .....	27
INTEgral:STOP .....	27
INTEgral:RESEt .....	27
INTEgral:STATus? .....	28
MEMory Commands .....	28
MEM:DOWNload {},<""> .....	28
MEM:UPLoad {},<""> .....	28
MEM:CATalog? {} .....	29
MEM:CATalog:HDLC? {} .....	29
RATE Commands .....	30
RATE? .....	30
RATE {} .....	30
SCOPE Commands .....	30
SCOPE? .....	30
SCOPE:TRIGger:METHod:TYPE? .....	30
SCOPE:TRIGger:METHod:TYPE {} .....	30
SCOPE:TRIGger:METHod:MODE? .....	31
SCOPE:TRIGger:METHod:MODE {} .....	31
SCOPE:TRIGger:SOURce? .....	31
SCOPE:TRIGger:SOURce {} .....	31
SCOPE:TRIGger:POSition:X? .....	31
SCOPE:TRIGger:POSition:X <> .....	31
SCOPE:TRIGger:POSition:Y? .....	32
SCOPE:TRIGger:POSition:Y <> .....	32

SCOPE:TIME:DIV? .....	32
SCOPE:TIME:DIV {} .....	32
SCOPE:AMPlitude? {} .....	32
SCOPE:AMPlitude {}, {}, {} .....	32
SCOPE:ADD:ITEM {} .....	33
SCOPE:REMOve:ITEM {} .....	33
SCOPE:DATA:ITEM? {} .....	34
SCOPE:HOLD {STOP} .....	34
SENSe Commands .....	34
[SENSe:][FORMatted:]DATA? {""} .....	34
[SENSe:]RAWData? {""} .....	35
SYNC Commands .....	35
SYNC#[:SOURce]? .....	35
SYNC#[:SOURce] {} .....	35
SYNC#[:SOURce]:STATus? {} .....	35
SYSTem Commands .....	35
SYSTem:REMOte {} .....	35
SYSTem:ERRor[:NEXT]? .....	36
SYSTem:ERRor:ALL? .....	36
SYSTem:ERRor:CODE[:NEXT]? .....	36
SYSTem:ERRor:CODE:ALL? .....	37
SYSTem:ERRor:COUNt? .....	37
SYSTem:HOST? .....	37
SYSTem:HOST {} .....	37
SYSTem:COMMunicate:SERial:BAUD? .....	38
SYSTem:COMMunicate:SERial:BAUD {} .....	38
SYSTem:COMMunicate:TYPE? .....	38
SYSTem:COMMunicate:TYPE {} .....	38
SYSTem:DATE < >, < >, < > .....	38
SYSTem:DATE? .....	38
SYSTem:TIME < >, < >, < > .....	39
SYSTem:TIME? .....	39
SYSTem:LANGuage? .....	39
SYSTem:LANGuage {} .....	39
SYSTem:RS485:ADDReSS < > .....	40
SYSTem:RS485:ADDReSS? .....	40
SYSTem:REBOot .....	40
SYSTem[:FACTory]:RESEt .....	40
TREND Commands .....	40
TREND? .....	40
TREND:ADD:ITEM {""} .....	41
TREND:REMOve:ITEM {""} .....	41
TREND:INTERval:TYPE? .....	41
TREND:INTERval:TYPE {} .....	41
TREND:INTERval? .....	42
TREND:INTERval < > .....	42
TREND:DATA:ITEM? {""} .....	42
WIRing Commands .....	42
WIRing:GROUp? .....	42
WIRing:GROUp {}, {}, {} .....	42
WIRing:GROUp:STATus? {}, {}, {} .....	43
ZERO Commands .....	43
ZERO? {""} .....	43
Common Commands .....	44

\*IDN? ..... 44  
\*CLS..... 44  
\*RST ..... 44  
Error Message..... 45

## **Introduction**

This document defines the remote interface commands for the Fluke NORMA 6003 / NORMA 6003+ / NORMA 6004 / NORMA 6004+ Power Analyzers (the Product or Analyzer). These commands may be used by a computer connected through the remote interface port to set settings, read measurement data, and control the operation of the Analyzer. Command syntax and names follow IEEE-488.2 and SCPI standards.

For operational information, see the printed *Getting Started Manual* that ships with the Product and the *Users Manual*, both online at [www.fluke.com](http://www.fluke.com).

## **How to Contact Fluke**

Fluke Corporation operates worldwide. For more contact information, go to our website: [www.fluke.com](http://www.fluke.com).

To register your product, view, print, or download the latest manual or manual supplement, go to [cn.fluke.com](http://cn.fluke.com) (Chinese) or [www.fluke.com](http://www.fluke.com) (English).

Fluke Corporation  
6920 Seaway Blvd  
Everett, WA 98203

+1-425-446-5500

[fluke-info@fluke.com](mailto:fluke-info@fluke.com)

## **Safety Information**

General safety information is located in the printed *Getting Started Manual* that ships with the Product and online at [www.fluke.com](http://www.fluke.com) or [cn.fluke.com](http://cn.fluke.com). More specific safety information is listed where applicable.

## **Symbols and Syntax**

The Analyzer accepts commands to set parameters, execute functions or respond with requested data. These commands are strings of ASCII-encoded characters. See [Introduction to SCPI](#) for standard SCPI command syntax. See [Table 1](#) for the special symbols used in this guide for SCPI commands..

**Table 1. Symbols for SCPI Syntax**

Symbol	Meaning	Command Examples	
		Long form	Actual Entry
{ }	Select one value from the values given in { }	HVF:MOTOr:TYPE {3PHASE NON3PHASE}	HVF:MOTOR:TYPE 3PHASE
{ "" }	Select an available value from the parameters	HARMonics:FFT:ITEM { "" }	HARM:FFT:ITEM "VOLT:3"
< >	Required value	HVF:NOM:VAL < >	HVF:NOM:VAL 3.2
[ ]	Can be omitted	[SENSe:]DATA? "VOLT:RMS:1"	DATA? "VOLT:RMS:1"
#	Suffix for channel or wiring group	INPUT#:CURR:AUTO?	INPUT1:CURR:AUTO?

The Analyzers measures 3 to 4 channels and up to 2 wiring groups. See [Table 2](#) for a list of abbreviations used in the command syntax.

**Table 2. Abbreviations Used in Numeric Definitions**

Abbreviation	Description
<b>1</b>	Channel 1
<b>2</b>	Channel 2
<b>3</b>	Channel 3
<b>4</b>	Channel 4
<b>A1</b>	Phase A in wiring group 1
<b>B1</b>	Phase B in wiring group 1
<b>C1</b>	Phase C in wiring group 1
<b>AB1</b>	Line A to B in wiring group 1
<b>BC1</b>	Line B to C in wiring group 1
<b>CA1</b>	Line C to A in wiring group 1
<b>SGM1</b>	Sum of wiring group 1
<b>N1</b>	Neutral-line in wiring group 1
<b>A2</b>	Phase A in wiring group 2
<b>B2</b>	Phase B in wiring group 2



**Table 2. Abbreviations Used in Numeric Definitions (cont.)**

Abbreviation	Description
<b>C2</b>	Phase C in wiring group 2
<b>AB2</b>	Line A to B in wiring group 2
<b>BC2</b>	Line B to C in wiring group 2
<b>CA2</b>	Line C to A in wiring group 2
<b>SGM2</b>	Sum of wiring group 2
<b>N2</b>	Neutral-line in wiring group 2

The Analyzer uses many symbols for measurements and the measurement results. See [Table 3](#) for the available options for the measurements.

**Table 3. Measurement Options**

Symbol	Measurement Name	Example in Command
$U_{rms}$	RMS voltage	VOLT:RMS[:1 2 3 4 A1 B1 C1 AB1 BC1 CA1 SGM1 A2 B2 C2 AB2 BC2 CA2 SGM2]
$I_{rms}$	RMS current	CURR:RMS[:1 2 3 4 A1 B1 C1 N1 SGM1 A2 B2 C2 N2 SGM2]
$U_{H01}$	Fundamental voltage	VOLT:H01[:1 2 3 4 A1 B1 C1 AB1 BC1 CA1 SGM1 A2 B2 C2 AB2 BC2 CA2 SGM2]
$I_{H01}$	Fundamental current	CURR:H01[:1 2 3 4 A1 B1 C1 N1 SGM1 A2 B2 C2 N2 SGM2]
$U_{DC}$	DC component voltage	VOLT:DC[:1 2 3 4]
$I_{DC}$	DC component current	CURR:DC[:1 2 3 4]
$U_{AC}$	AC component voltage	VOLT:AC[:1 2 3 4]
$I_{AC}$	AC component current	CURR:AC[:1 2 3 4]
$U_{rmc}$	Corrected rectified mean voltage	VOLT:RMCORR[:1 2 3 4]
$I_{rmc}$	Corrected rectified mean current	CURR:RMCORR[:1 2 3 4]
$U_{rm}$	Rectified mean voltage	VOLT:RMEAN[:1 2 3 4]
$I_{rm}$	Rectified mean current	CURR:RMEAN[:1 2 3 4]
$U_{uc}$	Voltage unbalance coefficient	VOLT:UNCOEF[:SGM1 SGM2]
$I_{uc}$	Current unbalance coefficient	CURR:UNCOEF[:SGM1 SGM2]
$P$	Active power	POW[:1 2 3 4 SGM1 SGM2]
$P_{H01}$	Fundamental active power	POW:H01[:1 2 3 4 SGM1 SGM2]

**Table 3. Measurement Options (cont.)**

<b>Symbol</b>	<b>Measurement Name</b>	<b>Example in Command</b>
S	Apparent power	POW:APP[:1 2 3 4 SGM1 SGM2]
S <sub>H01</sub>	Fundamental apparent power	POW:APP:H01[:1 2 3 4]
Q	Reactive power	POW:REACT[:1 2 3 4 SGM1 SGM2]
Q <sub>H01</sub>	Fundamental reactive power	POW:REACT:H01[:1 2 3 4]
$\lambda$	Power factor	POW:FACT[:1 2 3 4 SGM1 SGM2]
$\lambda_{H01}$	Fundamental power factor	POW:FACT:H01[:1 2 3 4]
$\varphi$	Phase	PHAS[:1 2 3 4 SGM1 SGM2]
$\varphi_{UH01}$	Fundamental phase for voltage (relative to the sync source)	PHAS:UH01[:1 2 3 4 A1 B1 C1 AB1 BC1 CA1 A2 B2 C2 AB2 BC2 CA2]
$\varphi_{IH01}$	Fundamental phase for current (relative to the sync source)	PHAS:IH01[:1 2 3 4 A1 B1 C1 N1 A2 B2 C2 N2]
$\Phi_{H01}$	Fundamental phase shift	PHAS:H01[:1 2 3 4 A1 B1 C1 A2 B2 C2]
U <sub>p+</sub>	Peak+ voltage	VOLT:MAX[:1 2 3 4]
I <sub>p+</sub>	Peak+ current	CURR:MAX[:1 2 3 4]
U <sub>p-</sub>	Peak- voltage	VOLT:MIN[:1 2 3 4]
I <sub>p-</sub>	Peak- current	CURR:MIN[:1 2 3 4]
U <sub>pp</sub>	Peak-to-peak voltage	VOLT:PTP[:1 2 3 4]
I <sub>pp</sub>	Peak-to-peak current	CURR:PTP[:1 2 3 4]
U <sub>cf</sub>	Voltage crest factor	VOLT:CFAC[:1 2 3 4]
I <sub>cf</sub>	Current crest factor	CURR:CFAC[:1 2 3 4]
U <sub>rip</sub>	Voltage ripple factor	VOLT:RIP[:1 2 3 4]
I <sub>rip</sub>	Current ripple factor	CURR:RIP[:1 2 3 4]
U <sub>ff</sub>	Voltage form factor	VOLT:FFAC[:1 2 3 4]
I <sub>ff</sub>	Current form factor	CURR:FFAC[:1 2 3 4]
U <sub>THD</sub>	Voltage THD	VOLT:THD[:1 2 3 4 A1 B1 C1 AB1 BC1 CA1 A2 B2 C2 AB2 BC2 CA2]
I <sub>THD</sub>	Current THD	CURR:THD[:1 2 3 4 A1 B1 C1 N1 A2 B2 C2 N2]

**Table 3. Measurement Options (cont.)**

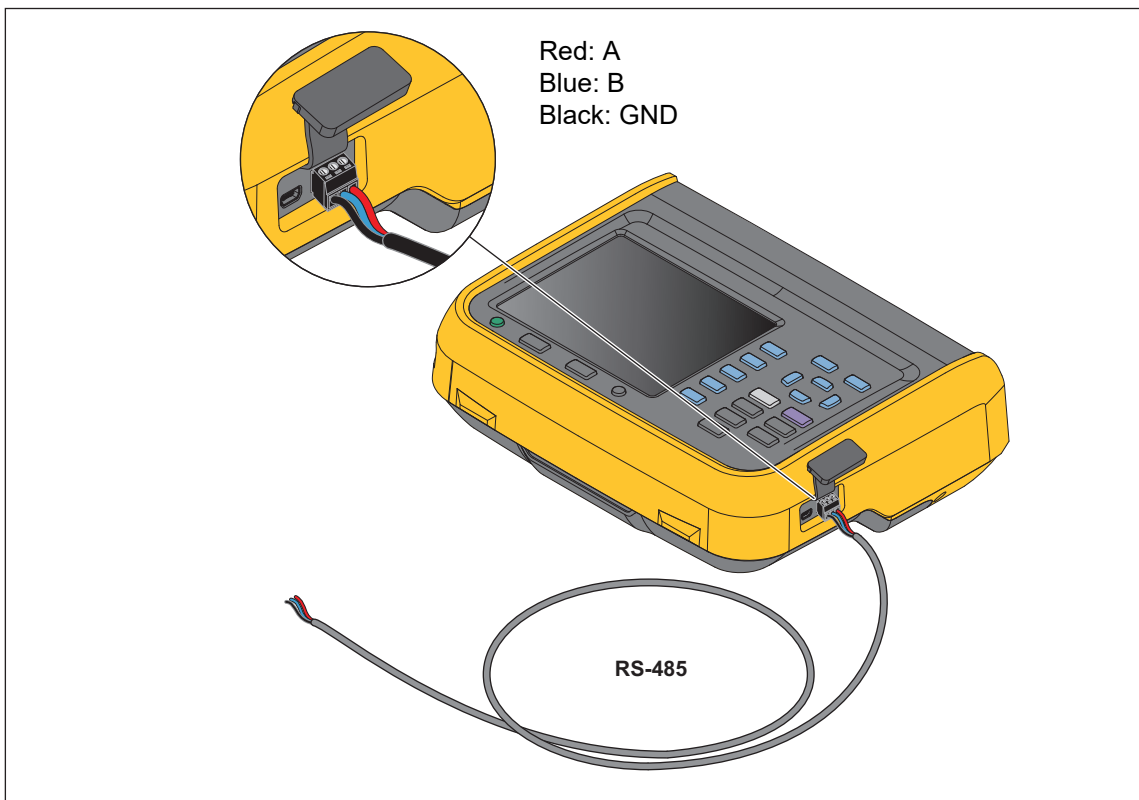
Symbol	Measurement Name	Example in Command
$U_{fc}$	Voltage fundamental content	VOLT:FCONT[:1 2 3 4 A1 B1 C1 AB1 BC1 CA1 A2 B2 C2 AB2 BC2 CA2]
$I_{fc}$	Current fundamental content	CURR:FCONT[:1 2 3 4 A1 B1 C1 N1 A2 B2 C2 N2]
$P_{fc}$	Power fundamental content	POW:FCONT[:1 2 3 4]
$U_{hc}$	Voltage harmonic content	VOLT:HCONT[:1 2 3 4 A1 B1 C1 AB1 BC1 CA1 A2 B2 C2 AB2 BC2 CA2]
$I_{hc}$	Current harmonic content	CURR:HCONT[:1 2 3 4 A1 B1 C1 N1 A2 B2 C2 N2]
hvf	Harmonic voltage factor	VOLT:HF[:1 2 3 4 A1 B1 C1 AB1 BC1 CA1 A2 B2 C2 AB2 BC2 CA2]
hcf	Harmonic current factor	CURR:HF[:1 2 3 4 A1 B1 C1 N1 A2 B2 C2 N2]
$\eta$	Efficiency	POW:EFF
$P_c$	Corrected power	POW:CORR[:1 2 3 4]
Z	Impedance	IMP[:1 2 3 4]
$Z_{H01}$	Fundamental impedance	IMP:H01[:1 2 3 4]
$R_s$	Equivalent serial resistance	RES:SER[:1 2 3 4]
$R_{sH01}$	Fundamental equivalent serial resistance	RES:SER:H01[:1 2 3 4]
$X_s$	Equivalent serial reactance	REACT:SER[:1 2 3 4]
$X_{sH01}$	Fundamental equivalent serial reactance	REACT:SER:H01[:1 2 3 4]
$R_p$	Equivalent parallel resistance	RES:PAR[:1 2 3 4]
$R_{pH01}$	Fundamental equivalent parallel resistance	RES:PAR:H01[:1 2 3 4]
$X_p$	Equivalent parallel reactance	REACT:PAR[:1 2 3 4]
$X_{pH01}$	Fundamental equivalent parallel reactance	REACT:PAR:H01[:1 2 3 4]
W	Electric energy	POW:INT[:1 2 3 4 SGM1 SGM2]
W+	Electric energy +	POW:INT:POS[:1 2 3 4]
W-	Electric energy -	POW:INT:NEG[:1 2 3 4]
q	Electric quantity	CURR:INT[:1 2 3 4]
q+	Electric quantity +	CURR:INT:POS[:1 2 3 4]

**Table 3. Measurement Options (cont.)**

Symbol	Measurement Name	Example in Command
q-	Electric quantity -	CURR:INT:NEG[:1 2 3 4]
f	Fundamental frequency	FREQ[:1 2 3 4]
t	Integral time for both power integral and current integral	INT:TIME
T	Torque	TORQ
n	Rotation speed	SPE
P <sub>m</sub>	Mechanical power	POW:MECH
E <sub>m</sub>	Mechanical energy	POW:MECH:INT

## Hardware Interfaces

The Analyzer has a mini USB port (serial port) and an RS-485 port. Use only one communication port at a time. [Figure 1](#) shows the RS-485 port connections.



**Figure 1. Communication Port**

## Interface Specification

See [Table 4](#) for interface specifications

**Table 4. Interface Specification**

Interface	USB	RS485
<b>Connector</b>	Type B mini	3 wire terminal with 3.81 mm pitch
<b>Number of Ports</b>	1	1
<b>Baud Rate</b>	9600 bps to 921600 bps	9600 bps to 921600 bps
<b>EOL</b>	CR/LF/CRLF	CR/LF/CRLF
<b>Data bits</b>	7 or 8	7 or 8
<b>Parity</b>	None/Odd/Even	None/Odd/Even
<b>Stop bits</b>	1, 1.5, or 2	1, 1.5, or 2
<b>Address</b>	--	1 to 9



### Switch between Remote and Local Modes

Use Local mode for direct operation of the Analyzer. Use Remote mode to program, read data, and send commands from a linked computer.

#### Switch from Local to Remote Mode

To enter Remote mode from Local mode: send the `SYSTEM:REMOte ON` command to the controller. See [SYSTEM:REMOte {}](#).

In Remote mode,



- The Analyzer shows the communication icon:  (USB) or  (RS485)
- The Analyzer disables all keys except **F1** (LOCAL).

#### Switch from Remote Mode to Local

To enter Local mode from Remote mode, choose a method:

- On the Analyzer, push **F1** (LOCAL).
- OR
- From the connected computer, send the `SYSTEM:REMOte OFF` command.

In Local mode,

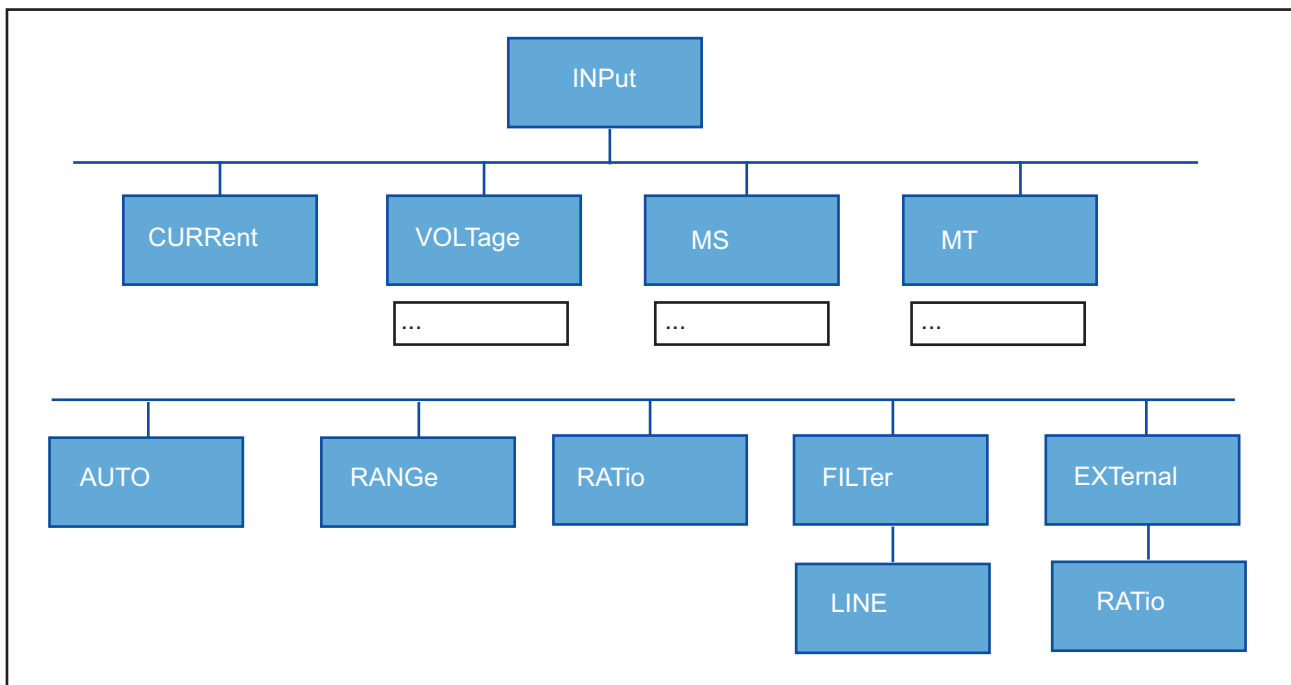
- The Analyzer turns off the communication icon:  (USB) or  (RS485)
- Key operations are enabled.

## Commands and Responses

### Introduction to SCPI

SCPI (Standard Commands for Programmable Instruments) describes a standard command set for programming instruments, irrespective of the type of instrument or manufacturer. The objective of the SCPI consortium is to standardize the device-specific commands to a large extent. The model defines identical functions of a device or of different devices. Command systems are assigned to these functions and it is possible to address identical functions with identical commands. The command systems have a hierarchical structure.

Figure 2 shows an example of the tree structure using the command system INPut, which operates the input channel of the devices. The other examples concerning syntax and structure of the commands are derived from this command system.



**Figure 2. Command Tree Structure**

SCPI is based on standard IEEE 488.2 and uses the same basic syntax elements and common commands defined in this standard. Part of the syntax of the device responses is defined in greater detail than in standard IEEE 488.2 (see [Commands and Responses](#)).

### Structure of Commands

Commands consist of a command header and, if necessary, parameter data.

- The header and the parameters are separated by a “space” (0x20).
- Multiple parameters are separated by a comma (“,” 0x2C).
- Headers may consist of several key words.

- Query commands are commands that request data in response. Query commands have a question mark (“?”, 0x3F) immediately following the command header. Responses to query commands are generated, placed in the output buffer, and transmitted to the PC.
- If a device has several functions or features of the same kind, such as inputs, the desired function can be selected by appending a numerical suffix (“#” is used as a placeholder in a command description) to the Command.
- Commands are not case sensitive.
- Terminate commands with a carriage return (0x0D), new line character (0x0A), or carriage return (0x0D) and new line character (0x0A).

### **Long and Short Form**

Command headers consist of one or more mnemonics. Most mnemonics have a long form that is more readable and a short form consisting of three or four characters that is more efficient. The long form is a complete word. The short form is shown by upper-case letters.

#### *Note*

*This manual uses upper-case to indicate the short form and lower-case letters to indicate the long forms. The commands are not case sensitive.*

Long form example

```
INPut1:CURRent:EXTernal:SENSor?
```

Short form example

```
INP1:CURR:EXT?
```

Both commands query channel 1’s current input using external sensor.

### **RS485 Command’s Prefix**

For the RS485 interface append “:” + “Address” (1 digit code) to the front of the command.

Example:

```
TX: :1*idn?
```

```
RX: :1FLUKE,NORMA_6004+,000-000-00,v4.2.0-1-g1e3151-dev,v4.2.0,V1.2
```

### **Multiple Commands Per Line**

Use a semicolon (;) to separate commands in a command line. If the next command belongs to a different command system, use a semicolon followed by a colon.

USB port examples:

```
TX: *idn?;:input1:current:range?
```

```
RX: FLUKE,NORMA_6004+,000-000-00,v4.2.0-1-g1e3151-dev,v4.2.0,V1.2
```

```
RX: 100mV
```

```
TX: :input1:current:range?;ratio?
```

```
RX: 100mA
```

```
RX: 1
```

RS485 port examples:

TX: :1\*idn?;:1:input1:current:range?

RX: :1FLUKE,NORMA\_6004+,000-000-00,v4.2.0-32-g0b7023-dev,v4.2.0-6-g6621c1,V1.2

RX: :1100Mv

TX: :1:input1:current:range?;:1ratio?

RX: :1100mA

RX: :11

### **HDLC Protocol (High-Level Data Link Control)**

See [Table 5](#) through [Table 8](#) for control information.

**Table 5. HDLC Data Frame**

	<b>STX</b>	<b>Address Byte</b>	<b>Control Byte</b>	<b>Information Bytes</b>	<b>FCS Bytes</b>	<b>ETX</b>
Bytes Number	1 (0x7D)	1 (default is 0x00)	1 (default is 0x00)	Variable, only for iframe	2	1 (0x7E)

**Table 6. Control Byte for iframe**

	<b>N(R)</b>	<b>P/F</b>	<b>N(S)</b>	<b>0</b>
Bits Number	Bit 5 - Bit 7	Bit 4	Bit 1 - Bit 3	Bit 0
Information bytes for iframe: When information bytes are not ASCII code, bytes location follows little-endian.				

**Table 7. Control Byte for sframe**

	<b>N(R)</b>	<b>P/F</b>	<b>Type</b>	<b>0</b>	<b>1</b>
Bits Number	Bit 5 - Bit 7	Bit 4	Bit 2 - Bit 3	Bit 1	Bit 0

**Table 8. Control Byte for uframe**

	<b>N(R)</b>	<b>P/F</b>	<b>Code</b>	<b>1</b>	<b>1</b>
Bits Number	Bit 5~7	Bit 4	Bit 2~3	Bit 1	Bit 0

See [Table 9](#) for escape characters

**Table 9. Escape characters**

HDLC_STX	0x7D
HDLC_ETX	0x7E
HDLC_DEL	0x10



Escape rule:

Insert HDLC\_DEL before the escape character, and then XOR the escape character with 0x20. See [Table 10](#) for an example.

**Table 10. Escape Example**

Byte Index	1	2	3	4	5
Raw Bytes	0x01	0x02	0x10	0x7D	0x05
Converted Bytes	0x7D,0x01	0x02	0x10,0x30	0x10,0x5D	0x05,0x7E

### *Formatted Resolution String Format*

Each formatted resolution string consists of 8 bytes:

**Table 11. Formated String**

Byte Index	1	2	3	4	5	6	7	8
	Raw value (float, little-endian)				Reading status (See <a href="#">Table 12</a> )	Unit (See <a href="#">Table 13</a> )	Magnitude (See <a href="#">Table 14</a> )	Decimal points
Byte 5, byte 6 and byte 7 are enumerated value, byte 8 is an integer value.								

**Table 12. Reading Status**

Enumerated Value	Status
0	Normal
1	Invalid
2	Over upper limit
3	Over lower limit

**Table 13. Unit**

Enumerated Value	Unit
0	None
1	V
2	A
3	$\Omega$
4	$j\Omega$
5	W
6	Hz
7	VA

**Table 13. Unit (cont.)**

<b>Enumerated Value</b>	<b>Unit</b>
8	var
9	degree
10	radian
11	%
12	Wh
13	Ah
14	Nm
15	rpm

**Table 14. Magnitude**

<b>Enumerated Value</b>	<b>Magnitude</b>
0	Normal
1	Tera
2	Giga
3	Mega
4	Kilo
5	Milli
6	Micro
7	Nano
8	Pico

## Communication Commands

This section lists the available commands. See [Table 15](#).

**Table 15. Communication Commands**

Command Groups	Function
<b>FORMula Commands</b>	
<i>[FORMula:]HVF:MOTOr:TYPE?</i> <i>[FORMula:]HVF:MOTOr:TYPE {}</i>	Queries and sets HVF motor type.
<i>[FORMula:]HVF:NOMinal:TYPE?</i> <i>[FORMula:]HVF:NOMinal:TYPE {}</i>	Queries and sets HVF voltage nominal value type.
<i>[FORMula:]HVF:NOMinal:VALue?</i> <i>[FORMula:]HVF:NOMinal:VALue &lt;&gt;</i>	Queries and sets HVF specified nominal value, float value.
<i>[FORMula:]HCF:NOMinal:TYPE?</i> <i>[FORMula:]HVF:NOMinal:TYPE {}</i>	Queries and sets HCF current nominal value type.
<i>[FORMula:]HCF:NOMinal:VALue?</i> <i>[FORMula:]HCF:NOMinal:VALue &lt;&gt;</i>	Queries and sets HCF specified nominal value, float value.
<b>HARMonics Commands</b>	
<i>HARMonics:ORDer? {""}</i>	Queries the maximum output harmonic order of the specified function's harmonic measurement numeric list data.
<i>HARMonics:DATA? {""}</i>	Queries harmonic data for the specified items.
<i>HARMonics:FFT:ITEM?</i> <i>HARMonics:FFT:ITEM {""}</i>	Queries and sets FFT item name.
<i>HARMonics:FFT:FREQuency?</i> <i>HARMonics:FFT:FREQuency {}</i>	Queries and sets the FFT's upper limit frequency for FFT item.
<i>HARMonics:FFT:DATA?</i>	Queries FFT data for the FFT item.
<b>HOLD Commands</b>	
<i>HOLD:START</i>	Starts HOLD for METER/TREND/PHASOR/HARMONIC.
<i>HOLD:STOP</i>	Stops HOLD for METER/TREND/PHASOR/HARMONIC.

**Table 15. Communication Commands (cont.)**

<b>Command Groups</b>	<b>Function</b>
<i>HOLD:STATus?</i>	Queries HOLD status for METER/TREND/PHASOR/HARMONIC.
<b><i>INPut Commands</i></b>	
<i>INPut#:CURRent:AUTO?</i> <i>INPut#:CURRent:AUTO {}</i>	Queries and sets the current auto range for the specified channel.
<i>INPut#:CURRent:RANGe?</i> <i>INPut#:CURRent:RANGe</i>	Queries and sets the current range for the specified channel.
<i>INPut#:CURRent:RATIo?</i> <i>INPut#:CURRent:RATIo&lt;&gt;</i>	Queries and sets the current ratio for the specified channel.
<i>INPut#:CURRent:FILTer:LINE?</i> <i>INPut#:CURRent:FILTer:LINE {}</i>	Queries and sets the current line filter for the specified channel.
<i>INPut#:CURRent:EXTErnal[:SENSor]?</i> <i>INPut#:CURRent:EXTErnal[:SENSor] {}</i>	Queries and sets the status of external sensor for the specified channel.
<i>INPut#:CURRent:EXTErnal[:SENSor]:RATIo? {}</i>	Queries the current ratio for the specified channel when external sensor is enabled/disabled.
<i>INPut#:VOLTag:e:AUTO?</i> <i>INPut#:VOLTag:e:AUTO {}</i>	Queries and sets the voltage auto range for the specified channel.
<i>INPut#:VOLTag:e:RANGe?</i> <i>INPut#:VOLTag:e:RANGe {}</i>	Queries and sets the voltage range for the specified channel.
<i>INPut#:VOLTag:e:RATIo?</i> <i>INPut#:VOLTag:e:RATIo&lt;&gt;</i>	Queries and sets the voltage ratio for the specified channel.
<i>INPut#:VOLTag:e:FILTer:LINE?</i> <i>INPut#:VOLTag:e:FILTer:LINE {}</i>	Queries and sets the voltage line filter for the specified channel.
<i>INPut:MS:MODE?</i> <i>INPut:MS:MODE {}</i>	Queries and sets the motor speed mode.
<i>INPut:MS:RATIo?</i> <i>INPut:MS:RATIo&lt;&gt;</i>	Queries and sets the motor speed ratio.
<i>INPut:MT:RATIo?</i> <i>INPut:MT:RATIo&lt;&gt;</i>	Queries and sets the motor torque ratio.
<i>INPut:MS:MODE:RATIo? {}</i>	Queries the motor speed ratio for the specified mode.

**Table 15. Communication Commands (cont.)**

Command Groups	Function
<b>INTEgral Commands</b>	
<i>INTEgral:START</i>	Starts integral process.
<i>INTEgral:STOP</i>	Stops integral process.
<i>INTEgral:RESEt</i>	Resets integral process.
<i>INTEgral:STATus?</i>	Queries integral status.
<b>MEMory Commands</b>	
<i>MEM:DOWNload {f},&lt;""&gt;</i>	Starts transfer protocol to download a file from host.
<i>MEM:UPLoad {f},&lt;""&gt;</i>	Starts transfer protocol to upload a file to host.
<i>MEM:CATalog? {f}</i>	Lists files in specified folder, response are raw ASCII codes.
<i>MEM:CATalog:HDLC? {f}</i>	Lists files in specified folder, response are encoded as a HDLC iframe.
<b>RATE Commands</b>	
<i>RATE?</i> <i>RATE {f}</i>	Queries and sets data update rate.
<b>SCOPE Commands</b>	
<i>SCOPE?</i>	Queries all scope items' name.
<i>SCOPE:TRIGger:METHod:TYPE?</i> <i>SCOPE:TRIGger:METHod:TYPE {f}</i>	Queries and sets trigger method type.
<i>SCOPE:TRIGger:METHod:MODE?</i> <i>SCOPE:TRIGger:METHod:MODE {f}</i>	Queries and sets trigger method mode.
<i>SCOPE:TRIGger:SOURce?</i> <i>SCOPE:TRIGger:SOURce {f}</i>	Queries and sets trigger source.
<i>SCOPE:TRIGger:POSition:X?</i> <i>SCOPE:TRIGger:POSition:X &lt;&gt;</i>	Queries and sets trigger position (horizontal).
<i>SCOPE:TRIGger:POSition:Y?</i> <i>SCOPE:TRIGger:POSition:Y &lt;&gt;</i>	Queries and sets trigger position (vertical).
<i>SCOPE:TIME:DIV?</i> <i>SCOPE:TIME:DIV {f}</i>	Queries and sets time division.

**Table 15. Communication Commands (cont.)**

<b>Command Groups</b>	<b>Function</b>
<i>SCOPE:AMPLitude? {}</i> <i>SCOPE:AMPLitude {}, {}, {}</i>	Queries and sets the specified item's amplitude_coefficient and amplitude_exponent.
<i>SCOPE:ADD:ITEM {}</i>	Adds a specified scope item.
<i>SCOPE:REMOve:ITEM {}</i>	Removes a specified scope item.
<i>SCOPE:DATA:ITEM? {}</i>	Queries specified item's data.
<i>SCOPE:HOLD {STOP}</i>	Stops hold while in HOLD status.
<b>SENSe Commands</b>	
<i>[SENSe:][FORMatted:]DATA? {""}</i>	Queries the specified numeric item's formatted value with unit/status/resolution.
<i>[SENSe:]RAWData? {""}</i>	Queries the specified numeric item's raw value without unit/status/resolution.
<b>SYNC Commands</b>	
<i>SYNC#[[:SOURce]?</i> <i>SYNC#[[:SOURce] {}</i>	Queries and sets the sync source for specified wiring group.
<i>SYNC#[[:SOURce]:STATus? {}</i>	Sets the sync source for specified wiring group and returns whether setting is OK.
<b>SYSTem Commands</b>	
<i>SYSTem:REMOte {}</i>	Enables/disables remote mode.
<i>SYSTem:ERRor[:NEXT]?</i>	Dequeues one error and returns its error code and error name.
<i>SYSTem:ERRor:ALL?</i>	Dequeues all the unread errors and returns their error code and error name.
<i>SYSTem:ERRor:CODE[:NEXT]?</i>	Dequeues one error and returns only its error code.
<i>SYSTem:ERRor:CODE:ALL?</i>	Dequeues all the unread errors and returns their error code.
<i>SYSTem:ERRor:COUNT?</i>	Queries the unread error count.
<i>SYSTem:HOST?</i> <i>SYSTem:HOST {}</i>	Queries and sets the host type.
<i>SYSTem:COMMunicate:SERial:BAUD?</i> <i>SYSTem:COMMunicate:SERial:BAUD {}</i>	Queries and sets baud rate for the serial port.
<i>SYSTem:COMMunicate:TYPE?</i> <i>SYSTem:COMMunicate:TYPE {}</i>	Queries and sets the type for the serial port.

**Table 15. Communication Commands (cont.)**

<b>Command Groups</b>	<b>Function</b>
<i>SYSTem:DATE</i> <>,<>,<> <i>SYSTem:DATE?</i>	Queries and sets system date.
<i>SYSTem:TIME</i> <>,<>,<> <i>SYSTem:TIME?</i>	Queries and sets system time.
<i>SYSTem:LANGUage?</i> <i>SYSTem:LANGUage</i> { }	Queries and sets system language.
<i>SYSTem:RS485:ADDReSS</i> <> <i>SYSTem:RS485:ADDReSS?</i>	Queries and sets RS485 address.
<i>SYSTem:REBOot</i>	Reboot device.
<i>SYSTem:[:FACTory]:RESEt</i>	Factory reset device's settings.
<b>TREND Commands</b>	
<i>TREND?</i>	Queries all trend items' name.
<i>TREND:ADD:ITEM</i> { "" }	Adds a specified trend item.
<i>TREND:REMOVe:ITEM</i> { "" }	Removes a specified trend item.
<i>TREND:INTErval:TYPE?</i> <i>TREND:INTErval:TYPE</i> { }	Queries and sets the interval type.
<i>TREND:INTErval?</i> <i>TREND:INTErval</i> <>	Queries and sets the interval between 2 data points.
<i>TREND:DATA:ITEM?</i> { "" }	Queries data for specified trend item.
<b>WIRing Commands</b>	
<i>WIRing:GROUp?</i> <i>WIRing:GROUp</i> { {}, {}, {} }	Queries and sets the wiring groups.
<i>WIRing:GROUp:STATus?</i> { {}, {}, {} }	Sets the wiring groups and return whether setting is OK.
<b>ZERO Commands</b>	
<i>ZERO?</i> { "" }	Zero specified ADC channels and return whether zeroing is successful.
<b>Common Commands</b>	
<i>*IDN?</i>	Queries device identification.
<i>*CLS</i>	Clears the standard event register, extended event register, and error queue.
<i>*RST</i>	Reset the standard event register, extended event register, and error queue.

## **FORMula Commands**

*[FORMula:]HVF:MOTOR:TYPE?*

*[FORMula:]HVF:MOTOR:TYPE }*

**Description:** Queries and sets HVF motor type.

**Parameter:** For SET command: 3phase|non3phase

**Response:** For QUERY command: 3phase|non3phase

**Example:**

TX: HVF:MOTOR:TYPE 3phase

TX: HVF:MOTOR:TYPE?

RX: 3phase

*[FORMula:]HVF:NOMinal:TYPE?*

*[FORMula:]HVF:NOMinal:TYPE }*

**Description:** Queries and sets HVF voltage nominal value type.

**Parameter:** For SET command: measured|specified

**Response:** For QUERY command: measured|specified

**Example:**

TX: HVF:NOM:TYPE specified

TX: HVF:NOM:TYPE?

RX: specified

*[FORMula:]HVF:NOMinal:VALue?*

*[FORMula:]HVF:NOMinal:VALue <>*

**Description:** Queries and sets HVF specified nominal value, float value.

**Parameter:** For SET command: Float value, min: 2.4, max: 2.3e+5

**Response:** For QUERY command: Float value, min: 2.4, max: 2.3e+5

**Example:**

TX: HVF:NOM:VAL 3.2

TX: HVF:NOM:VAL?

RX: 3.2



*[FORMula:]HCF:NOMinal:TYPE?*

*[FORMula:]HCF:NOMinal:TYPE {}*

**Description:** Queries and sets HCF current nominal value type.

**Parameter:** For SET command: measured|specified

**Response:** For QUERY command: measured|specified

**Example:**

TX: HCF:NOM:TYPE specified

TX: HCF:NOM:TYPE?

RX: specified

*[FORMula:]HCF:NOMinal:VALue?*

*[FORMula:]HCF:NOMinal:VALue <>*

**Description:** Queries and sets HCF specified nominal value, float value.

**Parameter:** For SET command: Float value, min: 1.0, max: 1.0e+8

**Response:** For QUERY command: Float value, min: 1.0, max: 1.0e+8

**Example:**

TX: HVF:NOM:VAL 3.2

TX: HVF:NOM:VAL?

RX: 3.2

## **HARMonics Commands**

*HARMonics:ORDER? {""}}*

**Description:** Queries the maximum output harmonic order of the specified function's harmonic measurement numeric list data.

Max measurable frequency is 6 kHz, for fundamental frequency < 60Hz, order is 100, otherwise:

$order = 6000 / (fundamental\ frequency)$

**Parameter:** For SET command:

"VOLT:1|2|3|4|A1|B1|C1|AB1|BC1|CA1|A2|B2|C2|AB2|BC2|CA2"

or

"CURR:1|2|3|4|A1|B1|C1|N1|A2|B2|C2|N2"

**Response:** For QUERY command:

"VOLT:1|2|3|4|A1|B1|C1|AB1|BC1|CA1|A2|B2|C2|AB2|BC2|CA2"

or

"CURR:1|2|3|4|A1|B1|C1|N1|A2|B2|C2|N2"

**Example:**

TX: HARM:ORDER? "VOLT:1"

RX: 100

**HARMonics:DATA? {""}**

**Description:** Queries the specified item's harmonic data.

Response data meets HDLC protocol's iframe, and information bytes consist of up to actual order data (HARMonics:ORDer? {""}).

See [Table 16](#) for the order of data information as part of the iframe.

**Table 16. Harmonic Data Information Order: iframe**

Item Name	Type	Length
<b>Actual harmonics orders</b>	Integer	4
<b>Amplitudes for every order</b>	Formatted resolution string format	8 * actual_harmonics_orders
<b>Ratio for every order</b>	Formatted resolution string format	8 * actual_harmonics_orders
<b>Relative phases for every order</b>	Formatted resolution string format	8 * actual_harmonics_orders

**Parameter:** For SET command:

"VOLT:1|2|3|4|A1|B1|C1|AB1|BC1|CA1|A2|B2|C2|AB2|BC2|CA2"

or

"CURR:1|2|3|4|A1|B1|C1|N1|A2|B2|C2|N2"

**Response:** For QUERY command:

"VOLT:1|2|3|4|A1|B1|C1|AB1|BC1|CA1|A2|B2|C2|AB2|BC2|CA2"

or

"CURR:1|2|3|4|A1|B1|C1|N1|A2|B2|C2|N2"

**Example:**

TX: HARM:DATA? "VOLT:1"

RX: ...

### *HARMonics:FFT:ITEM?*

### *HARMonics:FFT:ITEM {""}*

**Description:** Queries and sets FFT item name.

**Parameter:** For SET command:

"VOLT:1|2|3|4|A1|B1|C1|AB1|BC1|CA1|A2|B2|C2|AB2|BC2|CA2"

or

"CURR:1|2|3|4|A1|B1|C1|N1|A2|B2|C2|N2"

**Response:** For QUERY command:

"VOLT:1|2|3|4|A1|B1|C1|AB1|BC1|CA1|A2|B2|C2|AB2|BC2|CA2"

or

"CURR:1|2|3|4|A1|B1|C1|N1|A2|B2|C2|N2"

**Example:**

TX: HARM:FFT:ITEM "VOLT:3"

TX: HARM:FFT:ITEM?

RX: "VOLT:3"

### *HARMonics:FFT:FREQuency?*

### *HARMonics:FFT:FREQuency {}*

**Description:** Queries and sets FFT upper limit frequency for FFT item.

**Parameter:** For SET command: 100kHz|20kHz|4kHz|1kHz

**Response:** For QUERY command: 100kHz|20kHz|4kHz|1kHz

**Example:**

TX: HARM:FFT:FREQ 100kHz

TX: HARM:FFT:FREQ?

RX: 100kHz

### *HARMonics:FFT:DATA?*

**Description:** Queries FFT item's FFT data.

**Parameter:** None

**Response:** Response data meets HDLC protocol's iframe, and information bytes consist of 1024 float value.

**Example:**

TX: HARM:FFT:DATA?

RX: ...

## ***HOLD Commands***

### ***HOLD:START***

**Description:** Starts HOLD for METER/TREND/PHASOR/HARMONIC.

**Parameter:** None

**Response:** None

**Example:**

TX: HOLD:START

### ***HOLD:STOP***

**Description:** Stops HOLD for METER/TREND/PHASOR/HARMONIC.

**Parameter:** None

**Response:** None

**Example:**

TX: HOLD:STOP

### ***HOLD:STATus?***

**Description:** Queries HOLD status for METER/TREND/PHASOR/HARMONIC.

**Parameter:** None

**Response:** Started|Stopped

**Example:**

TX: HOLD:STAT?

RX: Stopped

## ***INPut Commands***

### ***INPut#:CURRent:AUTO?***

### ***INPut#:CURRent:AUTO }***

**Description:** Queries and sets the current auto range for the specified channel.

**Parameter:** For SET command: ON|OFF

**Response:** For QUERY command: ON|OFF

**Example:**

TX:

TX: INPUT1:CURR:AUTO?

RX: ON

*INPut#:CURRent:RANGe?*

*INPut#:CURRent:RANGe*

**Description:** Queries and sets the current range for the specified channel.

**Parameter:** For SET command: 100mA|1A|10A|100mV|1V|10V

**Response:** For QUERY command: 100mA|1A|10A|100mV|1V|10V

**Example:**

TX: INPUT1:CURREN:RANG 1A

TX: INPUT1:CURREN:RANG?

RX: 1A

*INPut#:CURRent:RATIo?*

*INPut#:CURRent:RATIo<>*

**Description:** Queries and sets the current ratio for the specified channel.

**Parameter:** For SET command: max: 1.0e+4, min: 1.0e-5

**Response:** For QUERY command: max: 1.0e+4, min: 1.0e-5

**Example:**

TX: INPUT1:CURREN:RATI 12.34

TX: INPUT1:CURREN:RATI?

RX: 12.34

*INPut#:CURRent:FILTer:LINE?*

*INPut#:CURRent:FILTer:LINE }*

**Description:** Queries and sets the current line filter for the specified channel.

**Parameter:** For SET command: OFF|650Hz|10kHz

**Response:** For QUERY command: OFF|650Hz|10kHz

**Example:**

TX: INPUT1:CURREN:FILT:LINE 650Hz

TX: INPUT1:CURREN:FILT:LINE?

RX: 650Hz

*INPut#:CURRent:EXTernal[:SENSor]?*

*INPut#:CURRent:EXTernal[:SENSor] }*

**Description:** Queries and sets the status of external sensor for the specified channel.

**Parameter:** For SET command: ON|OFF

**Response:** For QUERY command: ON|OFF

**Example:**

TX: INPUT1:CURR:EXT ON

TX: INPUT1:CURR:EXT?

RX: ON

*INPut#:CURRent:EXTernal[:SENSor]:RATIo? }*

**Description:** Queries the current ratio for the specified channel when external sensor is enabled/disabled.

**Parameter:** ON|OFF

**Response:** Max: 1.0e+4, min: 1.0e-5

**Example:**

TX: INPUT1:CURR:EXT:RATI? OFF

RX: 1000

TX: INPUT1:CURR:EXT:RATI? ON

RX: 1

*INPut#:VOLTage:AUTO?*

*INPut#:VOLTage:AUTO }*

**Description:** Queries and sets the voltage auto range for the specified channel.

**Parameter:** For SET command: ON|OFF

**Response:** For QUERY command: ON|OFF

**Example:**

TX: INPUT1:VOLT:AUTO ON

TX: INPUT1:VOLT:AUTO?

RX: ON

*INPut#:VOLTage:RANGe?*

*INPut#:VOLTage:RANGe {}*

**Description:** Queries and sets the voltage range for the specified channel.

**Parameter:** For SET command: 10V|100V|1000V

**Response:** For QUERY command: 10V|100V|1000V

**Example:**

TX: INPUT1:VOLT:RANG 10V

TX: INPUT1:VOLT:RANG?

RX: 10V

*INPut#:VOLTage:RATIo?*

*INPut#:VOLTage:RATIo<>*

**Description:** Queries and sets the voltage ratio for the specified channel.

**Parameter:** For SET command: max: 1.0e+4, min: 1.0e-5

**Response:** For QUERY command: max: 1.0e+4, min: 1.0e-5

**Example:**

TX: INPUT1:VOLT:RATI 12.34

TX: INPUT1:VOLT:RATI?

RX: 12.34

*INPut#:VOLTage:FILTer:LINE?*

*INPut#:VOLTage:FILTer:LINE {}*

**Description:** Queries and sets the voltage line filter for the specified channel.

**Parameter:** For SET command: OFF|650Hz|10kHz

**Response:** For QUERY command: OFF|650Hz|10kHz

**Example:**

TX: INPUT1:VOLT:FILT:LINE 650Hz

TX: INPUT1:VOLT:FILT:LINE?

RX: 650Hz

***INPUT:MS:MODE?***

***INPUT:MS:MODE }***

**Description:** Queries and sets the motor speed mode.

**Parameter:** For SET command: 10V|A|AB|ABZ

**Response:** For QUERY command: 10V|A|AB|ABZ

**Example:**

TX: INPUT:MS:MODE A

TX: INPUT:MS:MODE?

RX: A

***INPut:MS:RATIo?***

***INPut:MS:RATIo<>***

**Description:** Queries and sets the motor speed ratio.

**Parameter:** For SET command:

While MODE is 10V, RATIO is a float value, max: 1.0e+4, min: 1.0e-5;

While MODE is A|AB|ABZ, RATIO is an integer value, max: 10000, min: 1

**Response:** For QUERY command:

While MODE is 10V, RATIO is a float value, max: 1.0e+4, min: 1.0e-5;

While MODE is A|AB|ABZ, RATIO is an integer value, max: 10000, min: 1

**Example:**

TX: INPUT:MS:RATI 1.2

TX: INPUT:MS:RATI?

RX: 1.2

***INPut:MT:RATIo?***

***INPut:MT:RATIo<>***

**Description:** Queries and sets the motor torque ratio.

**Parameter:** For SET command, max: 1.0e+4, min: 1.0e-5

**Response:** For QUERY command, max: 1.0e+4, min: 1.0e-5

**Example:**

TX: INPUT:MT:RATI 1.4

TX: INPUT:MT:RATI?

RX: 1.4



### ***INPut:MS:MODE:RATIo? }***

**Description:** Queries the motor speed ratio for specified mode.

**Parameter:** 10V|A|AB|ABZ

**Response:**

While PARAMETER is 10V, RATIO is a float value, max: 1.0e+4, min: 1.0e-5;

While PARAMETER is A|AB|ABZ, RATIO is an integer value, max: 10000, min: 1

**Example:**

TX: INPUT:MS:MODE:RATI? ABZ

RX: 1

## ***INTEgral Commands***

### ***INTEgral:STARt***

**Description:** Starts integral process.

**Parameter:** None

**Response:** None

**Example:**

TX: INTEgral:STARt

### ***INTEgral:STOP***

**Description:** Stops integral process.

**Parameter:** None

**Response:** None

**Example:**

TX: INTEgral:STOP

### ***INTEgral:RESEt***

**Description:** Resets integral process.

**Parameter:** None

**Response:** None

**Example:**

TX: INTEgral:RESEt

## ***INTEgral:STATus?***

**Description:** Queries INTEGRAL status.

**Parameter:** None

**Response:** Started|Stopped

**Example:**

TX: INTEgral:STAT?

RX: Stopped

## ***MEMory Commands***

### ***MEM:DOWNload {},<">***

**Description:** Starts Kermit transfer protocol to download a file from host to meter. The communication terminal on PC should support Kermit protocol, otherwise file cannot be transferred.

**Parameter:**

1st parameter is meter's folder name: data|screen|settings|log

2nd parameter is file name (MEM:CAT? {})

**Response:** Kermit process data

**Example:**

TX: MEM:DOWNload data,"tt.txt"

RX: ...

### ***MEM:UPLoad {},<">***

**Description:** Starts Kermit transfer protocol to upload a file from meter to host. The communication terminal on PC should support Kermit protocol, otherwise the file cannot be transferred.

**Parameter:**

1st parameter is meter's folder name: data|screen|settings|log

2nd parameter is file name (MEM:CAT? {})

**Response:** Kermit process data

**Example:**

TX: MEM:UPload data,"tt.txt"

RX: ...

### **MEM:CATalog? {}**

**Description:** Lists files in specified folder, response are raw ASCII codes.

**Parameter:** Meter's folder name: data|screen|settings|log

**Response:**

Number of files

1st File name

...

**Example:**

TX: MEM:CAT? screen

RX: 2

Screen\_20190902\_080857.png

Screen\_20190902\_080907.png

### **MEM:CATalog:HDLC? {}**

**Description:** Lists files in specified folder, response are encoded as a HDLC iframe.

**Parameter:** Meter's folder name: data|screen|settings|log

**Response:** Response data meets HDLC protocol's iframe. See [Table 17](#) for data in information part of iframe.

**Table 17. Memory Data Information: iframe**

Item Name	Type
Number of files	String with EOL
1 <sup>st</sup> File name	String with EOL
...	...

**Example:**

TX: MEM:CAT:HDLC? screen

RX: ...

## ***RATE Commands***

### ***RATE?***

### ***RATE {}***

**Description:** Queries and sets data update rate.

**Parameter:** For SET command: 100ms|200ms|500ms|1s

**Response:** For QUERY command: 100ms|200ms|500ms|1s

**Example:**

TX: RATE 500ms

TX: RATE?

RX: 500ms

## ***SCOPE Commands***

### ***SCOPE?***

**Description:** Queries all scope items' name.

**Parameter:** None

**Response:** Items are selected from U1|U2|U3|U4|I1|I2|I3|I4, max number of items is 4.

**Example:**

TX: SCOPE?

RX: U1,U2

### ***SCOPE:TRIGger:METHod:TYPE?***

### ***SCOPE:TRIGger:METHod:TYPE {}***

**Description:** Queries and sets trigger method type.

**Parameter:** For SET command: Positive|Negative|Dual

**Response:** For QUERY command: Positive|Negative|Dual.

**Example:**

TX: SCOPE:TRIGger:METHod:TYPE Negative

TX: SCOPE:TRIGger:METHod:TYPE?

RX: Negative

***SCOPE:TRIGger:METHod:MODE?***

***SCOPE:TRIGger:METHod:MODE }***

**Description:** Queries and sets trigger method mode.

**Parameter:** For SET command: Auto|Normal|Single

**Response:** For QUERY command: Auto|Normal|Single

**Example:**

TX: SCOPE:TRIGger:METHod:MODE Single

TX: SCOPE:TRIGger:METHod:MODE?

RX: Single

***SCOPE:TRIGger:SOURce?***

***SCOPE:TRIGger:SOURce }***

**Description:** Queries and sets trigger source.

**Parameter:** For SET command: U1|U2|U3|U4|I1|I2|I3|I4

**Response:** For QUERY command: U1|U2|U3|U4|I1|I2|I3|I4

**Example:**

TX: SCOPE:TRIGger:SOURce I1

TX: SCOPE:TRIGger:SOURce?

RX: I1

***SCOPE:TRIGger:POSition:X?***

***SCOPE:TRIGger:POSition:X <>***

**Description:** Queries and sets trigger position (horizontal).

**Parameter:** For SET command: min: 0, max: 500

**Response:** For QUERY command: min: 0, max: 500

**Example:**

TX: SCOPE:TRIGger:POSition:X 200

TX: SCOPE:TRIGger:POSition:X?

RX: 200

***SCOPE:TRIGger:POSition:Y?***

***SCOPE:TRIGger:POSition:Y <>***

**Description:** Queries and sets trigger position (vertical).

**Parameter:** For SET command: min: 0, max: 500

**Response:** For QUERY command: min: 0, max: 500

**Example:**

TX: SCOPE:TRIGger:POSition:Y 300

TX: SCOPE:TRIGger:POSition:Y?

RX: 300

***SCOPE:TIME:DIV?***

***SCOPE:TIME:DIV }***

**Description:** Queries and sets time division.

**Parameter:** For SET command: 500us|1ms|2ms|5ms|10ms|20ms|50ms

**Response:** For QUERY command: 500us|1ms|2ms|5ms|10ms|20ms|50ms

**Example:**

TX: SCOPE:TIME:DIV 20ms

TX: SCOPE:TIME:DIV?

RX: 20ms

***SCOPE:AMPlitude? }***

***SCOPE:AMPlitude },},}***

**Description:** Queries and sets specified item's amplitude\_coefficient and amplitude\_exponent.

Amplitude is calculated as  $a \cdot \text{pow}(10, b)$ , a is coefficient, and b is exponent.

**Parameter:**

For QUERY command: U1|U2|U3|U4|I1|I2|I3|I4

For SET command:

1st parameter (item name): U1|U2|U3|U4|I1|I2|I3|I4

2nd parameter (coefficient): 1|2|5

3rd parameter (exponent): -6|-5|-4|...|13|14|15

**Response:**

For QUERY command:

1st returned value:  $a \cdot \text{pow}(10, b)$

2nd returned value: a

3rd returned value: b

**Example:**

TX: SCOPe?

RX: I4

TX: SCOPe:AMP I4,2,-1

TX: SCOPe:AMP? I4

RX: 0.2,2,-1

***SCOPe:ADD:ITEM {***

**Description:** Adds a specified scope item.

**Parameter:** U1|U2|U3|U4|I1|I2|I3|I4

**Response:** None

**Example:**

TX: SCOPe?

RX: U1,U2

TX: SCOPe:ADD:ITEM I1

TX: SCOPe?

RX: U1,U2,I1

***SCOPe:REMOve:ITEM {***

**Description:** Removes a specified scope item.

**Parameter:** U1|U2|U3|U4|I1|I2|I3|I4

**Response:** None

**Example:**

TX: SCOPe?

RX: U1,U2

TX: SCOPe:REMOve:ITEM U1

TX: SCOPe?

RX: U2

**SCOPE:DATA:ITEM? {}**

**Description:** Queries specified item's data.

**Parameter:** U1|U2|U3|U4|I1|I2|I3|I4

**Response:** Response data meets HDLC protocol's iframe. See [Table 18](#).

**Table 18. Scope Data Information Order: iframe**

Item Name					Type	Length
Frequency					float	4
Specified item's max amplitude					float	4
Trigger status					integer	4
Enumerated Value	0	1	2	3		
Status	Wait	Triggered	Finished	Hold		
Number of points is 501, every point include min, max and avg. Replied data are in float mode.					Float	4 * 3 * 501

**Example:**

TX: SCOPE:DATA:ITEM? U1

RX: ...

**SCOPE:HOLD {STOP}**

**Description:** Stops hold while in HOLD status.

**Parameter:** STOP

**Response:** None

**Example:**

TX: SCOPE:HOLD STOP

**SENSe Commands**

**[SENSe:][FORMatted:]DATA? {""}**

**Description:** Queries the specified numeric item's formatted value with unit/status/resolution.

**Parameter:** See [Table 3](#)

**Response:** Formatted value with unit/status/resolution

**Example:**

TX: DATA? "CURR:DC:3"

RX: -0.25mA



### **[SENSe:]RAWData? {""}**

**Description:** Queries the specified numeric item's raw value without unit/status/resolution.

**Parameter:** See [Table 3](#)

**Response:** Raw value with unit/status/resolution

**Example:**

TX: RAWData? "CURR:DC:3"

RX: -0.000256242

## **SYNC Commands**

### **SYNC#[[:SOURce]]?**

### **SYNC#[[:SOURce]] {}**

**Description:** Queries and sets the sync source for specified wiring group. # may be 1|2|3|4.

**Parameter:** For SET command: U1|U2|U3|U4|I1|I2|I3|I4

**Response:** For QUERY command: U1|U2|U3|U4|I1|I2|I3|I4

**Example:**

TX: SYNC1 I1

TX: SYNC1?

RX: I1

### **SYNC#[[:SOURce]]:STATus? {}**

**Description:** Sets the sync source for specified wiring group and returns whether setting is OK. # may be 1|2|3|4.

Some seconds may need to set the wiring group, so if Fail is returned, to retry the setting.

**Parameter:** U1|U2|U3|U4|I1|I2|I3|I4

**Response:** Ok|Fail

**Example:**

TX: SYNC1:STAT? I1

RX: Ok

## **SYSTEM Commands**

### **SYSTEM:REMote {}**

**Description:** Enables/disables remote mode.

**Parameter:** ON|OFF

**Response:** None

**Example:**

TX: SYSTEM:REMote ON

### ***SYSTem:ERRor[:NEXT]?***

**Description:** Dequeues one error and returns its error code and error name.

**Parameter:** None

**Response:**

1st returned: error code

2nd returned: error name

**Example:**

TX: SYST:REMOte

TX: SYST:ERR?

RX: -109,"Missing parameter"

### ***SYSTem:ERRor:ALL?***

**Description:** Dequeues all the unread errors and returns their error code and error name.

**Parameter:** None

**Response:**

1st returned: 1st error's code

2nd returned: 1st error's name

3rd returned: 2nd error's code

4th returned: 2nd error's name

...

**Example:**

TX: SYST:REMOte

TX: SYNC5?

TX: SYST:ERR:ALL?

RX: -109,"Missing parameter",-102,"Syntax error"

### ***SYSTem:ERRor:CODE[:NEXT]?***

**Description:** Dequeues one error and returns its error code and error name.

**Parameter:** None

**Response:**

1st returned: error code

**Example:**

TX: SYST:REMOte

TX: SYST:ERR:CODE?

RX: -109

### ***SYSTem:ERRor:CODE:ALL?***

**Description:** Dequeues all the unread errors and returns their error code and error name.

**Parameter:** None

**Response:**

1st returned: 1st error's code

2nd returned: 2nd error's code

...

**Example:**

TX: SYST:REMOte

TX: SYNC5?

TX: SYST:ERR:CODE:ALL?

RX: -109,-102

### ***SYSTem:ERRor:COUNT?***

**Description:** Queries the unread error count.

**Parameter:** None

**Response:** Errors' count

**Example:**

TX: SYST:REMOte

TX: SYNC5?

TX: SYST:ERR:COUNT?

RX: 2

### ***SYSTem:HOST?***

### ***SYSTem:HOST {}***

**Description:** Queries and sets the host type. When host type is COMPUTER, ECHO mode is enabled; otherwise, ECHO mode is disabled.

**Parameter:** For SET command: TERMINAL|COMPUTER

**Response:** For QUERY command: TERMINAL|COMPUTER

**Example:**

TX: SYSTem:HOST COMPUTER

TX: SYSTem:HOST?

RX: COMPUTER

### ***SYSTem:COMMunicate:SERial:BAUD?***

### ***SYSTem:COMMunicate:SERial:BAUD {***

**Description:** Queries and sets serial port's baud rate. For USB and RS-485 ports, default baud rate is 115200.

**Parameter:** For SET command: 9600|9200|38400|57600|115200|230400|460800|576000|921600

**Response:** For QUERY command: 9600|9200|38400|57600|115200|230400|460800|576000|921600

**Example:**

TX: SYSTem:COMM:SER:BAUD 115200

TX: SYSTem:COMM:SER:BAUD?

RX: 115200

### ***SYSTem:COMMunicate:TYPE?***

### ***SYSTem:COMMunicate:TYPE {***

**Description:** Queries and sets serial port's type. While converting from USB to RS485, baud rate will be changed to be 921600, and all commands need add a prefix ":RS485\_ADDR".

**Parameter:** For SET command: USB|RS485

**Response:** For QUERY command: USB|RS485

**Example:**

TX: SYSTem:COMM:TYPE USB

TX: SYSTem:COMM:TYPE?

RX: USB

If meter's RS485 address is 1, then:

TX: SYSTem:COMM:TYPE RS485

TX: :1SYSTem:COMM:TYPE?   --Note: Use this command when baud rate is 921600

RX: :1RS485

### ***SYSTem:DATE < >,< >,< >***

### ***SYSTem:DATE?***

**Description:** Queries and sets system date.

**Parameter:** For SET command:

1st parameter: year;

2nd parameter: month;

3rd parameter: day

**Response:** For QUERY command:

- 1st returned value: year;
- 2nd returned value: month;
- 3rd returned value: day

**Example:**

TX: SYSTem:DATE 2019,07,19  
TX: SYSTem:DATE?  
RX: 2019,07,19

*SYSTem:TIME < >,< >,< >*

*SYSTem:TIME?*

**Description:** Queries and sets system time.

**Parameter:** For SET command:

- 1st parameter: hour;
- 2nd parameter: minute;
- 3rd parameter: second

**Response:** For QUERY command:

- 1st returned value: hour;
- 2nd returned value: minute;
- 3rd returned value: day

**Example:**

TX: SYSTem:TIME 17,31,30  
TX: SYSTem:TIME?  
RX: 17,31,32

*SYSTem:LANGuage?*

*SYSTem:LANGuage }*

**Description:** Queries and sets system language.

**Parameter:** For SET command: CHN|ENG

**Response:** For QUERY command: CHN|ENG

**Example:**

TX: SYSTem:LANG ENG  
TX: SYSTem:LANG?  
RX: ENG

### ***SYSTem:RS485:ADDRess <>***

#### ***SYSTem:RS485:ADDRess?***

**Description:** Queries and sets RS485 address. Default RS485 address is 1.

**Parameter:** For SET command: 1|2|3|4|5|6|7|8|9

**Response:** For QUERY command: 1|2|3|4|5|6|7|8|9

**Example:**

TX: SYSTem:RS485:ADDR 2

TX: SYSTem:RS485:ADDR?

RX: 2

### ***SYSTem:REBOot***

**Description:** Reboot meter. Rebooting process needs about 80 seconds.

**Parameter:** None

**Response:** None

**Example:**

Example

TX: SYSTem:REBOot

### ***SYSTem:[.:FACTory]:RESET***

**Description:** Factory reset the meter settings.

**Parameter:** None

**Response:** None

**Example:**

TX: SYSTem:RESET

## ***TREND Commands***

### ***TREND?***

**Description:** Queries all trend items' name.

**Parameter:** None

**Response:** Items refer to [Table 3](#), max number of items is 4.

**Example:**

TX: TREND?

RX: "volt:rms:2","volt:rms:4"

### **TREND:ADD:ITEM {""}**

**Description:** Adds a specified trend item.

**Parameter:** Items refer to [Table 3](#)

**Response:** None

**Example:**

```
TX: TREND?
RX: "volt:rms:2","volt:rms:4"
TX: TREND:ADD:ITEM "volt:rms:3"
TX: TREND?
RX: "volt:rms:3","volt:rms:2","volt:rms:4"
```

### **TREND:REMOVe:ITEM {""}**

**Description:** Removes a specified trend item.

**Parameter:** Items refer to [Table 3](#)

**Response:** None

**Example:**

```
TX: TREND?
RX: "volt:rms:3","volt:rms:2","volt:rms:4"
TX: TREND:REMOVe:ITEM "volt:rms:3"
TX: TREND?
RX: "volt:rms:2","volt:rms:4"
```

### **TREND:INTErval:TYPE?**

### **TREND:INTErval:TYPE }**

**Description:** Queries and sets the interval type.

**Parameter:** For SET command: SPECIFIED|RATE

**Response:** For QUERY command: SPECIFIED|RATE

**Example:**

```
TX: TREND:INTErval:TYPE RATE
TX: TREND:INTErval:TYPE?
RX: RATE
```

**TREND:INTERval?**

**TREND:INTERval <>**

**Description:** Queries and sets the interval between 2 data points. Unit is second.

**Parameter:** For SET command: min: 1, max: 24\*60\*60 - 1

**Response:** For QUERY command: min: 1, max: 24\*60\*60 - 1

**Example:**

TX: TREND:INTERval 3

TX: TREND:INTERval?

RX: 3

**TREND:DATA:ITEM? {""}**

**Description:** Queries specified item's data.

**Parameter:** Items refer to [Table 3](#)

**Response:** Response data meets HDLC protocol's iframe. See [Table 19](#).

**Table 19. Trend Data Information Order: iframe**

Item Name	Type	Length
Final point's sample timestamp since 1970-01-01T00:00:00.000	Long integer	8
Number of points	Integer	4
Number of points is 2002, replied data are in float mode.	Float	4 * 2002

**Example:**

TX: TREND:DATA:ITEM? "volt:rms:4"

RX: ...

**WIRing Commands**

**WIRing:GROUp?**

**WIRing:GROUp {{,}},{{,}},{{,}}**

**Description:** Queries and sets the wiring groups. If one channel is not grouped, then 1P2W1M is the default value.

**Parameter:** For SET command: 1P2W1M|3P3W2M|3P4W3M|3P3W3M,

1st parameter: wiring group 1's name,

2nd parameter: wiring group 2's name,

3rd parameter: wiring group 3's name,

4th parameter: wiring group 4's name



**Response:** For QUERY command: 1P2W1M|3P3W2M|3P4W3M|3P3W3M,

1st returned value: wiring group 1's name,

2nd returned value: wiring group 2's name,

3rd returned value: wiring group 3's name,

4th returned value: wiring group 4's name  
Data in information part of iframe are defined as following order:

**Example:**

TX: WIRing:GROUp 1P2W1M,3P4W3M

TX: WIRing:GROUp?

RX: 1P2W1M,3P4W3M

**WIRing:GROUp:STATus? {{{,}}}**

**Description:** Queries and sets the wiring groups. The wiring group takes several seconds to set up, so if Fail is returned, retry the setting.

**Parameter:** For SET command: 1P2W1M|3P3W2M|3P4W3M|3P3W3M,

1st parameter: wiring group 1's name,

2nd parameter: wiring group 2's name,

3rd parameter: wiring group 3's name,

4th parameter: wiring group 4's name

**Response:** Ok|Fail

**Example:**

TX: WIRing:GROUp:STATus? 1P2W1M,3P4W3M

RX: Ok

## **ZERO Commands**

**ZERO? {""}**

**Description:** Zero the specified ADC channels and return whether or not zeroing is successful.

For N6003&N6003+, U4&I4 are not supported.

For N6004&N6003, MT&MS are not supported.

### *Note*

1. For MS, zero is allowed when speed mode is 10V.
2. ZERO command cannot complete when a zeroing is already in process.
3. ZERO is done for every range (each ADC channel has 3 ranges), each range requires about 4 seconds and has nothing to do with the ADC channel number.

**Parameter:** U1|U2|U3|U4|I1|I2|I3|I4|MT|MS

**Response:** None

**Example:**

TX: ZERO? "U1,U2"

RX: Ok,Fail

## **Common Commands**

### **\*IDN?**

**Description:** Queries device identification.

**Parameter:** None

**Response:**

1<sup>st</sup> returned value: company name,

2<sup>nd</sup> returned value: model name,

3<sup>rd</sup> returned value: serial number,

4<sup>th</sup> returned value: ARM firmware version,

5<sup>th</sup> returned value: DSP firmware version,

6<sup>th</sup> returned value: FPGA firmware version

**Example:**

TX: \*IDN?

RX: FLUKE,NORMA\_6004+,12345678WS,v4.2.0,v4.2.0,V1.5

### **\*CLS**

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

**Parameter:** None

**Response:** None

**Example:**

TX: \*CLS

### **\*RST**

**Description:** Reset the standard event register, extended event register, and error queue.

**Parameter:** None

**Response:** None

**Example:**

TX: \*RST

## **Error Message**

See [Table 20](#) for a list of error messages.

**Table 20. Error Messages**

<b>Error Code</b>	<b>Error Name</b>
0	No error
-100	Command error
-101	Invalid character
-102	Syntax error
-103	Invalid separator
-104	Data type error
-105	GET not allowed
-108	Parameter not allowed
-109	Missing parameter
-110	Command header error
-111	Header separator error
-112	Program mnemonic too long
-113	Undefined header
-114	Header suffix out of range
-115	Unexpected number of parameters
-120	Numeric data error
-121	Invalid character in number
-123	Exponent too large
-124	Too many digits
-128	Numeric data not allowed
-130	Suffix error
-131	Invalid suffix
-134	Suffix too long
-138	Suffix not allowed
-140	Character data error
-141	Invalid character data
-144	Character data too long
-148	Character data not allowed
-150	String data error
-151	Invalid string data

**Table 20. Error Messages (cont.)**

<b>Error Code</b>	<b>Error Name</b>
-158	String data not allowed
-160	Block data error
-161	Invalid block data
-168	Block data not allowed
-170	Expression error
-171	Invalid expression
-178	Expression data not allowed
-200	Execution error
-201	Invalid while in local
-202	Settings lost due to rtl
-203	Command protected
-210	Trigger error
-211	Trigger ignored
-212	Arm ignored
-213	Init ignored
-214	Trigger deadlock
-215	Arm deadlock
-220	Parameter error
-221	Settings conflict
-222	Data out of range
-223	Too much data
-224	Illegal parameter value
-225	Out of memory
-226	Lists not same length
-230	Data corrupt or stale
-231	Data questionable
-232	Invalid format
-233	Invalid version
-240	Hardware error
-241	Hardware missing
-250	Mass storage error
-251	Missing mass storage

**Table 20. Error Messages (cont.)**

<b>Error Code</b>	<b>Error Name</b>
-252	Missing media
-253	Corrupt media
-254	Media full
-255	Directory full
-256	File name not found
-257	File name error
-258	Media protected
-260	Expression error
-261	Math error in expression
-280	Program error
-281	Cannot create program
-282	Illegal program name
-283	Illegal variable name
-284	Program currently running
-285	Program syntax error
-286	Program runtime error
-290	Memory use error
-300	Device-specific error
-310	System error
-311	Memory error
-313	Calibration memory lost
-315	Configuration memory lost
-320	Storage fault
-330	Self-test failed
-350	Queue overflow
-360	Communication error
-363	Input buffer overrun
-365	Time out error

