



Programming Manual

UT5470 Series Programmable Withstand Voltage Testers

V1.0

June 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.

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. SCPI

SCPI (Standard Commands for Programmable Instruments) is a standardized instrument programming language that builds on existing standards IEEE 488.1 and IEEE 488.2 and follows the floating-point rules of IEEE 754 standard, ISO 646 message exchange 7-bit encoding notation (equivalent to ASCII programming) and many other standards.

This section introduces the format, symbols, parameters, and abbreviations of the SCPI command.

1.1 Command Parsing

The host computer can send command strings to the instrument. The instrument's command parser begins processing when it detects the message terminator.

Example: Valid Command Format:
AAA:BBB CCC;DDD EEE;:FFF

The command parser interprets and executes all commands. Understanding the command syntax rules is essential before writing your program.

1.1.1 Command Parser Rules

1. The command parser processes ASCII data only.
2. The parser begins processing upon receiving the termination character. This instrument recognizes the following three termination characters:
 - CR (Carriage Return)
 - CR+LF (Carriage Return + Line Feed)
 - LF (Line Feed)
3. The parser terminates processing immediately if an invalid command is detected; the command is then discarded.
4. The parser is case-sensitive.
5. The parser supports command abbreviation; see the following section for formats.
6. In RS485 mode, prefix the SCPI command with ADDR□Local Address::□, where the local address ranges from 1 to 255.
7. This enables communication with multiple devices via the SCPI protocol.
For example, ADDR□1::□IDN? (□ represents a space character)
8. The instrument terminates responses with 0x0D0A (CR+LF) by default.
9. Multiple commands may be sent on a single line, separated by semicolons (;).

1.1.2 Notation and Definitions

This section describes symbols used to clarify command syntax. These symbols are not part of the command tree itself.

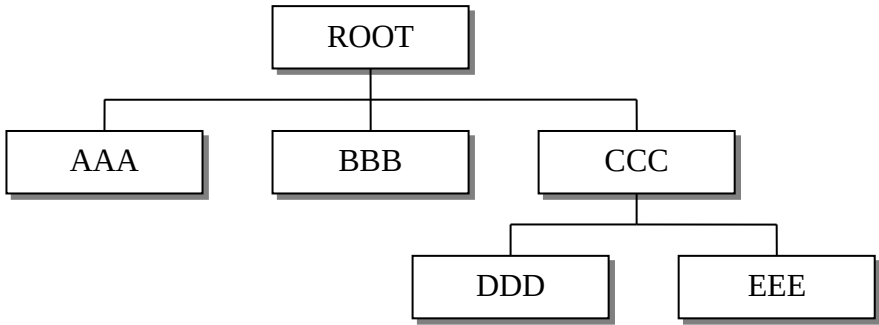
Symbol	Description
<.....>	Angle brackets indicate a command parameter. For example, <float> represents a floating-point parameter <integer> represents an integer parameter
[.....]	Square brackets indicate optional commands or parameters.
{.....}	Braces enclose a selection of parameters; only one may be chosen.
Capital letter	Uppercase letters indicate the abbreviated form of a command.
□	Space character. Represents a required space; shown for readability.

1.1.3 Command Tree Structure

SCPI commands follow a tree-structured hierarchy with three levels. (Note: The parser supports access to any level of the command tree.)

The top level consists of subsystem commands. Subordinate commands are accessible only after entering the parent subsystem. Colons (:) delimit successive levels in the command path.

Figure 1-1 Command Tree Structure



Example

```
ROOT:CCC:DDD ppp
ROOT      Subsystem command
  CCC      Second level
    DDD      Third level
      ppp      Parameter
```

1.2 Command and Parameter

A command consists of a **Command** and [**Parameter**], separated by a space (ASCII:20H).

Example AAA:BBB1.234
 Command [Parameter]

1.2.1 Command

The command header may be in long form or abbreviated form. The long form enhances readability. The abbreviated form improves programming efficiency.

1.2.2 Parameter

1. No parameters required: Some commands consist of headers only.
 For example, AAA:BBB
2. Character data: Parameters may be character strings. Abbreviated forms shall conform to the Command Abbreviation Rules defined previously.
 For example, AAA:BBB1.23
3. Numeric values: Parameters may be set numeric values.

<integer>	Decimal integer: 123, +123, -123
<float>	Floating-point numbers in multiple formats: <Fixed> (Floating-point number): 1.23, -1.23 <Scientific> (Scientific notation): 1.23E+4, +1.23e-4 <Suffix>: 1.23k, 1.23MA, 1.23G, 1.23u

Table 1-1 Metric Prefixes (Unit Multipliers)

Numeric Value	Multiplying Power
1E18 (EXA)	EX
1E15 (PETA)	PE
1E12 (TERA)	T
1E9 (GIGA)	G
1E6 (MEGA)	MA
1E3 (KILO)	K
1E-3 (MILLI)	M
1E-6 (MICRO)	U
1E-9 (NANO)	N
1E-12 (PICO)	P
1E-15 (PEMTO)	F
1E-18 (ATTO)	A



Because SCPI is case-sensitive, the abbreviated form differs from the full

command name.

For example:

- "1M" indicates 1 milli (10^{-3}), not 1 mega (10^6).
 - "1MA" indicates 11 mega (10^6).
-

1.2.3 Separator

The command parser accepts only specified separators; all others generate an Invalid Separator error.

;	Semicolon: Separates multiple commands For example, AAA:BBB 100.0;CCC:DDD
:	Colon: Separates hierarchy levels; resets to root For example, AAA:BBB:CCC 123.4;DDD:EEE 567.8
?	Question mark: Indicates query commands For example, AAA?
	Space: Separates header from parameters For example, AAA:BBB 1.234

1.3 Command Reference

All commands are organized by subsystem, as listed below.

MEASure	Measurement Subsystem
SYSTem	System Subsystem
FILE	File Subsystem
LIST	List Subsystem

Common Commands:

IDN?	Queries instrument identification
*RST	Resets instrument to default state

1.4 Test Command

- [MEASure]:TEST

Description: Executes the instrument self-test.

Command Syntax:	TEST // Executes the instrument self-test.
Query Syntax:	TEST?
Response:	0 (Off), 1 (On)

- [MEASure]:RESET

Description: Stops the test.

Command Syntax: `RESET` // Stops the test.

- **[MEASure]:TESTing:DATA?**

Description: Queries the test results.

Command Syntax: `TESTing:DATA?` // Queries the final test results.
`TESTing:DATA? 1` // Queries step 1 result.
`TESTing:DATA? 200` // Queries step 200 result.
`TESTing:DATA? 0` // Queries all test results.

Query Response: `01,DCW,ABORT,1.10,0.4110,0.0`

1.5 System Command

- **[SYSTem]:LANGuage**

Description: Sets the system language.

Command Syntax: `LANG <int>`

Parameter: 0-2

Examples: `LANG 0` // Sets the system language to Simplified Chinese.
`LANG?`

Response: `0` // Simplified Chinese

- **[SYSTem]:WARNing:BEEP:LEVel**

Description: Sets the beeper volume.

Command Syntax: `WARNing:BEEP:LEVel <int>`

Parameter: 0-3

Examples: `WARNing:BEEP:LEVel 3` // Sets beeper to Off.
`WARNing:BEEP:LEVel?`

Response: `3` // Returns the beeper state: 0(off).

- **[SYSTem]:REMOTE:CONTRol**

Description: Sets the instrument's remote control state.

Command Syntax: `REMOTE:CONTRol <int>`

Parameter: 0 (Off), 1 (On)

Examples: `REMOTE:CONTRol 0` // Sets the remote control to off.
`REMOTE:CONTRol?`

Response: `0` // Returns the remote-control state: 0(off).

- **[SYSTem]:STEP:MODE**

[SYSTem]:STEP:MODE Set step mode.

Command Syntax: STEP:MODE <int>

Parameter: 0-2

Examples: STEP:MODE 0 //Set step mode to normal

STEP:MODE?

Response: 0 //Step mode 0

- **[SYSTem]:FAILure:STOP**

Description: Sets stop-on-failure state.

Command Syntax: FAILure:STOP <int>

Parameter: 0 (Off), 1(On)

Examples: FAILure:STOP 0 // Sets stop-on-failure state to off.

FAILure:STOP?

Response: 0 // Returns stop-on-failure state: 0 (off).

- **[SYSTem]:SCREEN:SAVER**

Description: Sets the power-on screen state.

Command Syntax: SCREEN:SAVER <int>

Parameter: 0-2

Examples: SCREEN:SAVER 0 // Sets the power-on screen to menu.

SCREEN:SAVER?

Response: 0 // Menu

- **[SYSTem]:ACK:RETURN**

Description: Sets ACK return state.

Command Syntax: ACK:RETURN <int>

Parameter: 0 (Off), 1(On)

Examples: ACK:RETURN 0 // Sets ACK return state to off.

ACK:RETURN?

Response: 0 // Returns ACK return state: 0 (off).

- **[SYSTem]:CONTInue:VOLT**

[SYSTem]:CONTInue:VOLT Set the state of ProVOLT

Command Syntax: `CONT:VOLT <int>`

Parameter: 0/1

Examples: `CONT:VOLT 0` //Set the ProVOLT state to off

`CONT:VOLT?`

Response: 0 // The ProVOLT state is off

- **[SYSTem]:AUTO:SAVE**

[SYSTem]:AUTO:SAVE Set the Auto Save state

Command Syntax: `AUTO:SAVE <int>`

Parameter: 0/1

Examples: `AUTO:SAVE 0` //Set the Auto Save state to off

`AUTO:SAVE?`

Response: 0 // Auto Save state is off

- **[SYSTem]:VOLT:ADJUST**

[SYSTem]:VOLT:ADJUST Set VOLT ADJUST state

Command Syntax: `VOLT:ADJUST <int>`

Parameter: 0/1

Examples: `VOLT:ADJUST 0` // Set VOLT ADJUST state to off

`VOLT:ADJUST?`

Response: 0 // VOLT ADJUST state is off

- **[SYSTem]:CLEAR:RESET**

[SYSTem]:CLEAR:RESET Set Clear Reset State

Command Syntax: `CLEAR:RESET <int>`

Parameter: 0/1

Examples: `CLEAR:RESET 0` // Set Clear Reset State to off

`CLEAR:RESET?`

Response: 0 // Clear Reset State is off

- **[SYSTem]:REFRESH**

[SYSTem]:REFRESH Set Screen Refresh state

Command Syntax: REFRESH <int>

Parameter: 0/1

Examples: REFRESH 0 // Set Screen Refresh state to step

REFRESH?

Response: 0 //Screen Refresh state is step

1.6 File Command

- **[FILE]:NEW <int>,<string>**

Description: Creates a new file.

Command Syntax: NEW <int>,<string>

Parameter: 01-200, "xxxx" (file name)

Examples: NEW 1,"test1" // Creates file 1 with the name "test1".

- **[FILE]:LOAD<int>**

Description: Loads the specified file.

Command Syntax: LOAD <int>

Parameter: 01-200 (file number)

Examples: LOAD 2 // Loads file 2.

- **[FILE]:LOAD?**

Description: Queries the current file or the specified file.

Query Syntax: LOAD?

Response: 1, test1

Syntax: [FILE]:LOAD? <int> // Returns the current file.

Examples: LOAD? 1 // Queries file 1.

Response: 1, test1

- **[FILE]:DELeTe <int>**

Description: Deletes the specified file or the current file.

Command Syntax: DEL <int>

Parameter: 01-200 (or no parameter)

Examples: DEL 2 or DEL // Deletes the file 2 (If the parameter is omitted, the current file is deleted.)

- **[FILE]:SAVE**

Description: Saves the current file.

Command Syntax:

SAVE

- **[FILE]:STEP:TOTal?**

Description: Queries the number of steps in the specified file.

Syntax:

STEP:TOTal? // Queries the number of steps in the current file.

STEP:TOTal? 2 // Queries the number of steps in the specified file.

Response:

0-200

- **[FILE]:TOTal?**

Description: Queries the number of files.

Syntax:

FILE:TOTal?

Response:

0-100

- **[FILE]:STEP**

Description: Selects the test step.

Command Syntax:

[FILE]:STEP <int>

Parameter:

0-200

Examples:

[FILE]:STEP 1 // Selects Step 1.

[FILE]:STEP? // Queries the test step.

Response:

1 // Returns Step 1.

- **[FILE]:DELeTe:STEP**

Description: Deletes the test step.

Command Syntax:

DEL:STEP

- **[FILE]:PROMPT:CLEAR**

Description: Clears the prompt message.

Command Syntax:

PROMPT:CLEAR

- **[FILE]:PROMPT<string>**

Description: Sets the prompt message.

Command Syntax:

PROMPT<string>

Parameter:

"xxx" (prompt message)

Examples:	PROMPT "test"	// Sets the prompt message to "test".
	PROMPT? <int>	// Queries the prompt message.
Response:	test	// Returns test.

1.7 Test List Command

- [LIST]:ADD:ACW

Description: Adds ACW (AC withstand test) parameters to the test list.

Command Syntax: ADD:ACW <volt>,<curr hi >,< curr li >,<rise_t>,<dwel_l_t>,<fall_t>,<ARC_level>,<zero_ofst>,<freq>,<ARC_detect>,<curr_range>

Parameter:	<volt>	// Output voltage
	< curr hi >	// Upper limit of current
	< curr li >	// Lower limit of current
	<rise_t>	// Rise time
	<dwel_l_t>	// Test duration (dwell time)
	<fall_t>	// Fall time
	<ARC_level>	// Arc detection level
	<zero_ofst>	// Zero compensation offset
	<freq>	// Output voltage frequency
	<ARC_detect>	// Arc detection state
	<curr_range>	// Range

Examples: LIST:ADD:ACW 2000,10,0,1.0,1.0,1.0,5,0,60,OFF,AUTO

- [LIST]:ADD:OSC

Description: Adds OSC (Open/Short Check) parameters to the test list.

Command Syntax: ADD:OSC <open>,<short>,<sampling>

Parameter:	< open >	// Open-circuit
	< short >	// Short-circuit
	< sampling >	// Sampling standard

Examples: LIST:ADD:OSC 10,0,1

- [LIST]:ADD:DCW

Description: Adds DCW (DC withstand test) parameters to the test list.

Command Syntax: ADD:DCW <volt>,< curr hi >,< curr li >,<rise_t>,<dwel_l_t>,<fall_t>,<li_limit_charge>,<ARC_level> ,<zero_ofst>,< hi_limit_ramp >,<ARC_detect>,<

Parameter:	<curr_range>,< low_current_range >	
	<volt>	// Output voltage
	< curr hi >	// Upper limit of current
	< curr li >	// Lower limit of current
	<rise_t>	// Rise time
	<dwel_t>	//Test duration (dwell time)
	<fall_t>	//Fall time
	<li_limit_charge>	// Lower limit of charging current
	<ARC_level>	// Arc detection level
	<zero_ofst>	//Zero compensation offset
	< hi_limit_ramp >	// Upper limit of ramp
	<ARC_detect>	// Arc detection state
	<curr_range>	// Range
	< low_current_range >	// Low range
Examples:	LIST:ADD:DCW 1500,10000,0,1.0,1.0,1.0,0.0,5,0,0,OFF,AUTO,OFF	

- **[LIST]:ADD:IR**

Description: Adds IR (Insulation Resistance) parameters to the test list.

Command Syntax:	ADD:IR <volt>,< resistance hi >,< resistance li >,<rise_t>,<delay_t>,<dwel_t>,<fall_t>,<li_limit_charge>	
Parameter:	<volt>	// Output voltage
	< resistance hi >	// Upper limit of resistance
	< resistance li >	// Lower limit of resistance
	< rise_t >	// Rise time
	< delay_t >	// Delay time
	< dwel_t >	// Test duration (dwell time)
	< fall_t >	// Fall time
	< li_limit_charge >	// Lower limit of charging current
Examples:	LIST:ADD:IR 1500,0,1,1.0,1.0,1.0,1.0,0	

- **[LIST]:ADD:ACW:DEFault**

Description: Adds a default ACW test item to the test list.

Command Syntax:	ADD:ACW:DEF
-----------------	-------------

- **[LIST]:ADD:OSC:DEFault**

Description: Adds a default OSC test item to the test list.

Command Syntax: `ADD:OSC:DEF`

- **[LIST]:ADD:DCW:DEFault**

Description: Adds a default DCW test item to the test list.

Command Syntax: `ADD:DCW:DEF`

- **[LIST]:ADD:IR:DEFault**

Description: Adds a default IR test item to the test list.

Command Syntax: `ADD:IR:DEF`

- **[LIST]:STEP:CONTENT?**

Description: Queries the parameter of the test step.

Syntax: `STEP:CONTENT?` // Queries the parameter of the current test step.
`STEP:CONTENT? 1` // Queries the parameter of the first step.

Response: `0001,ACW,5000,20.00,0.000,0.5,0.0,0.5,5,0.000,50,OFF,Auto`

Note: Only the UT5470 and UT5475 supports ACW mode. the parameter indicates step number, test mode, output voltage, upper limit of current, lower limit of current, ramp-up time, test duration (dwell time), ramp-down time, arc detection level, zero compensation offset, frequency, arc detection state, and range, respectively.

- **[LIST]:SETup:VOLtage<int>**

Description: Sets the output voltage.

Command Syntax: `SET:VOL<int>`

Parameter: ACW:50-10000; DCW:50-12000; IR:50-6000; unit: V

Examples: `SET:VOL 100` // Sets the voltage to 100 V.
`SET:VOL?` // Queries the voltage.

Response: `100` // Returns the voltage of 100 V.

- **[LIST]:SETup:CURRent:HIGH<float>**

Description: Sets the upper limit of current.

Command Syntax: `SET:CURR:HIGH<float>`

Parameter: ACW:0.001-20.00; unit: mA
DCW:0.02-10000; unit: uA

Examples: `SET:CURR:HIGH 1` // Sets the upper limit of current to 1 mA.
`SET:CURR:HIGH?` // Queries the upper limit of current.

Response: 1 // Returns the upper limit of current of 1 mA.

- **[LIST]:SETup:CURRent:LOW<float>**

Description: Sets the lower limit of current.

Command Syntax: SET:CURR:LOW<float>

Parameter: ACW:0.000-20.00; unit: mA
DCW:0.0-10000; unit: uA

Examples: SET:CURR:LOW 1 // Sets the lower limit of current to 1 mA.
SET:CURR:LOW? // Queries the lower limit of current

Response: 1 // Returns the lower limit of current of 1 mA.

- **[LIST]:SETup:RESISTance:HIGH <float>**

- Description: Sets the higher limit of resistance.

Command Syntax: SET:RESIST:HIGH <float>

Parameter: IR: 0.0-50000; unit: MΩ

Examples: SET:RESIST:HIGH 10 // Sets the upper limit of resistance to 10MΩ.
SET:RESIST:HIGH? // Queries the upper limit of resistance.

Response: 10 // Returns the upper limit of resistance of
10MΩ

- **[LIST]:SETup:RESISTance:LOW <float>**

- Description: Sets the lower limit of resistance.

Command Syntax: SET:RESIST:LOW <float>

Parameter: IR: 0.1-50000; unit: MΩ

Examples: SET:RESIST:LOW 1 // Sets the lower limit of resistance to 1MΩ.
SET:RESIST:LOW? // Queries the lower limit of resistance.

Response: 1 // Returns the lower limit of resistance of 1MΩ

- **[LIST]:SETup:ZERO<float>**

Description: Sets the zero compensation offset.

Command Syntax: SET:ZERO<float>

Parameter: 0, execute zero-clearing

Examples:	SET:ZERO 0	//Sets the zero compensation offset to 0.
	SET:ZERO?	// Queries the zero compensation offset.
Response:	0.002	// Returns the zero compensation offset of 0.002 mA.

- **[LIST]:SETup:ZERO:START**

Description: Executes automatic zero compensation offset.

Command Syntax: SET:ZERO:START

- **[LIST]:SETup:RAMP<float>**

Description: Sets the ramp-up time.

Command Syntax: SET:RAMP<float>

Parameter: ACW:0.3-999.9; DCW:0.4-999.9; IR:0.4-999.9; unit: s

Examples: SET:RAMP 1 // Sets the ramp-up time to 1 s.
SET:RAMP? // Queries the ramp-up time.

Response: 1 // Returns the ramp-up time of 1 s.

- **[LIST]:SETup:DWELL<float>**

Description: Sets the test duration (dwell time).

Command Syntax: SET:DWELL<float>

Parameter: 0.1-999.9; 0=Continuous test, unit: s

Examples: SET:DWELL 1 // Sets the test duration (dwell time) to 1 s.
SET:DWELL? // Queries the test duration (dwell time).

Response: 1 // Returns the test duration (dwell time) of 1 s.

- **[LIST]:SETup:DOWN<float>**

Description: Sets the ramp-down time.

Command Syntax: SET:DOWN<float>

Parameter: 0.0-999.9; unit: s

Examples: SET:DOWN 1 // Sets the ramp-down time to 1 s.
SET:DOWN? // Queries the ramp-down time.

Response: 1 // Returns the ramp-down time of 1 s.

- **[LIST]:SETup:DELAY<float>**

Description: Sets the delay time.

Command Syntax: SET:DELAY<float>

Parameter: IR mode: 0.5-999.9; unit: s

Examples:	SET:DELAY 1	// Sets the delay time to 1 s.
	SET:DELAY?	// Queries the delay time.
Response:	1	// Returns the delay time of 1 s.

- **[LIST]:SETup:ARC:LEVel<int>**

Description: Sets the arc detection level.

Command Syntax:	SET:ARC:LEV<int>
Parameter:	1-9 (9 = maximum sensitivity)
Examples:	SET:ARC:LEV 9 // Sets the arc detection level to 9.
	SET:ARC:LEV? // Queries the arc detection level.
Response:	9 // Returns the arc detection level of 9.

- **[LIST]:SETup:FREQuency<int>**

Description: Sets the frequency.

Command Syntax:	SET:FREQ<int>
Parameter:	0/1 (50/60Hz)
Examples:	SET:FREQ 0 // Sets the frequency to 50 Hz.
	SET:FREQ? // Queries the frequency.
Response:	0 // Returns the frequency of 50 Hz.

- **[LIST]:SETup:ARC:DETECT<int>**

Description: Sets the arc detection state.

Command Syntax:	SET:ARC:DETECT<int>
Parameter:	0 (Off), 1 (On)
Examples:	SET:ARC:DETECT 0 // Sets the arc detection state to off.
	SET:ARC:DETECT? // Queries the arc detection state.
Response:	0 // Returns the arc detection state: 0 (off).

- **[LIST]:SETup:RANGE<int>**

Description: Sets the current measurement range mode.

Command Syntax:	SET:RANGE<int>
Parameter:	0 (Auto), 1 (Fixed)
Examples:	SET:RANGE 0 // Sets the current range to Auto mode.
	SET:RANGE? // Queries the current range mode.
Response:	0 // Returns the current range: auto.

- **[LIST]:SETup:OPEN:JUDGE<int>**

Description: Sets the open-circuit judgment threshold for OSC.

Command Syntax:	SET:OPEN:JUDGE <int>
Parameter:	10-100
Examples:	SET:OPEN:JUDGE 50 // Sets the open-circuit judgment threshold for OSC to 50%.
	SET:OPEN:JUDGE? // Queries the open-circuit judgment threshold.
Response:	50 // Returns the threshold value of 50%.

- **[LIST]:SETup:SHORT:JUDGE<int>**

Description: Sets the short-circuit judgment threshold for OSC.

Command Syntax:	SET:SHORT:JUDGE <int>	
Parameter:	100-500; 0=Off	
Examples:	SET:SHORT:JUDGE 100	// Sets the short-circuit judgment threshold for OSC to 100%.
	SET:SHORT:JUDGE?	// Queries the short-circuit judgment threshold.
Response:	100	// Returns the threshold value of 100%.

- **[LIST]:SETup:SAMPling STANDard<float>**

Description: Sets the sampling standard for OSC.

Command Syntax:	SET:SAMP:STAND <float>	
Parameter:	0.001-40; unit: nF	
Examples:	SET:SAMP:STAND 1	// Sets the sampling standard for OSC to 1 nF.
	SET:SAMP:STAND?	// Queries the sampling standard.
Response:	1	// Returns the sampling standard of 1 nF.

- **[LIST]:SETup:RAMP:HIGH<float>**

Description: Sets the upper current limit during ramp-up in DCW mode.

Command Syntax:	SET:RAMP:HIGH <float>	
Parameter:	0-10000; 0=Off; unit: uA	
Examples:	SET:RAMP:HIGH 1000	// Sets the ramp-up current limit to 1 mA in DCW mode.
	SET:RAMP:HIGH?	// Queries the ramp-up current limit.
Response:	1000	// Returns the ramp-up current limit of 1000.

- **[LIST]:SETup:CHARGE:LOW<float>**

Description: Sets the lower limit of charging current.

Command Syntax:	SET:CHARGE:LOW <float>	
Parameter:	DCW: Low range (on): 0.0-35; low range (off): 0.0-350; unit: uA IR:0.0-3.5, 0=Off; unit: uA	
Examples:	SET:CHARGE:LOW 10	// Sets the lower limit of charging current to 10 uA.
	SET:CHARGE:LOW?	// Queries the lower limit of charging current.
Response:	10	// Returns the lower limit of charging current: 10 uA.

- **[LIST]:SETup:CHARGE:LOW:START**

Description: Automatically exeutes the lower limit of charging current test.

Command Syntax:	SET:CHARGE:LOW:START
-----------------	----------------------

- **[LIST]:SETup:LOW:RANGE<int>**

Description: Sets the low range.

Command Syntax:	SET:LOW:RANGE<int>	
Parameter:	0(Off), 1(On)	
Examples:	SET:LOW:RANGE 0	// Sets the low range to off.

	SET:LOW:RANGE?	// Queries the low range state.
Response:	0	// Returns the low range state: 0 (off)

1.8 Address Subsystem

- ADDR <addr>::[cmd]

Description: Specifies the target device address and sends a control command to that device.

Command Syntax:	ADDR <addr>::[cmd] <NRF+>
Parameter:	addr: Device address (1-254) cmd: SCPI command
Examples:	ADDR 1::SET:VOL 100 // Sends the command to device address 1 and sets the output voltage to 100 V. ADDR 200::SET:VOL 500 // Sends the command to device address 200 and sets the output voltage to 500 V.
Note:	This command is used for multi-device control over an RS-485 network. The device address corresponds to the station number configured in the system. For single-device, the address subsystem can be omitted.

1.9 *RST Subsystem

Figure 1-2 *RST Subsystem Tree



Description: Resets the instrument to its default state.

Command Syntax:	*RST
Examples:	Request >*RST <NL> No return values >

1.10 IDN? Subsystem

Figure 1-3 IDN? Subsystem Tree



Description: Returns the instrument identification string.

Command Syntax:	IDN?
Query Response:	<Manufacturer>,<MODEL>,<SN>,<Revision>,
Examples:	Request > IDN? <NL> Response > UNI-T,UT5470,CWV70251400001,REV A1.0<NL>

1.11 ERRor Subsystem

Description: Provides access to the instrument's error queue.

Syntax:	ERRor?
Query Response:	Error string
Examples:	Query > ERR?<NL> Response > no error <NL>

The corresponding error codes are listed in the table below.

Error Code	Description
*E00	No error
*E01	Bad command
*E02	Parameter error
*E03	Missing parameter
*E04	buffer overrun
*E05	Syntax error
*E06	Invalid separator
*E07	Invalid multiplier
*E08	Numeric data error
*E09	Value too long
*E10	Invalid command
*E11	Unknow error

2. Modbus (RTU) Protocol

2.1 Data Format

Following the Modbus (RTU) communication protocol, the instrument responds to commands from the host computer and returns a standard response frame.

2.1.1 Command Frame

Figure 2-1 Modbus Command Frame Format

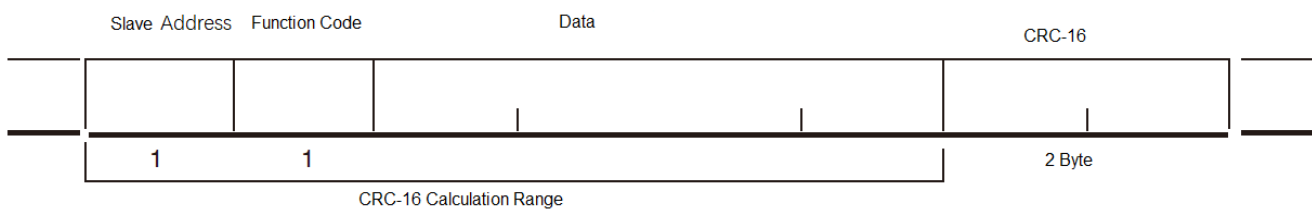


Table 2-1 Command Code

	A minimum silent interval of 3.5-character times is required between frames.
Slave Address	1 byte - Modbus supports 00-0x63 slave station 0x00 is used for broadcast (unified command) - If the instrument does not include an optional RS-485 interface, the default slave address is 0x01.
Function Code	1 byte 0x03: Read holding registers (multiple registers) 0x04: =03H, not used 0x06: Write single register (can be replaced by 0x10) 0x08: Diagnostic Echo Test (debug only) 0x10: Write holding registers (multiple registers)
Data	Specifies the register address, quantity, and contents
CRC-16	2 bytes, LSB - CyclicRedundancy Check (CRC-16): The CRC is calculated over all data starting from the slave address up to the last data byte.
	A minimum silent interval of 3.5-character times is required between frames.

2.1.2 Response Frame

Except for broadcast commands (slave address 0x00), all other slave addresses will return a response frame.

Figure 2-2 Normal Response Frame Format

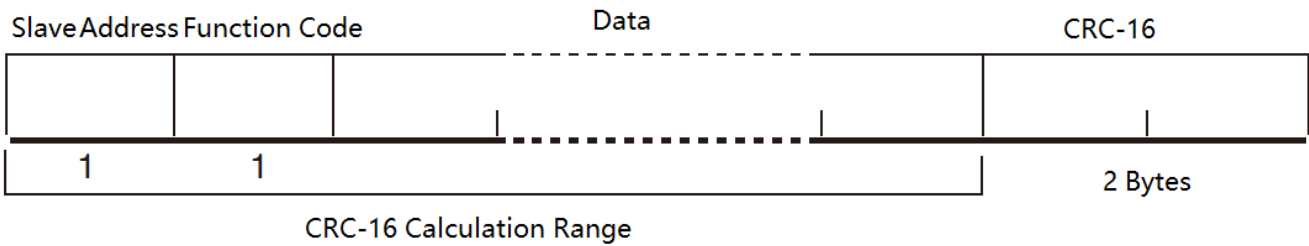


Figure 2-3 Exception Response Frame Format

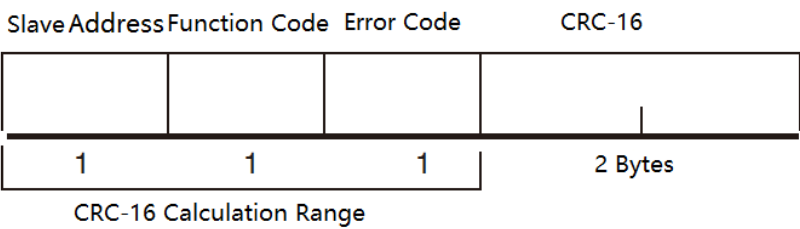


Table 2-2 Exception Response Code

Slave Address	1 byte - Returns the original slave address
Function Code	1 byte - In an exception response, the function code equals the request function code with Bit 7 set (logically ORed with 0x80). For example, 0x03 OR 0x80 = 0x83
Error Code	Exception Code: 0x01: Illegal function (function code not supported) 0x02: Illegal data address (register address does not exist) 0x03: Illegal data value (invalid data) 0x04: Device failure (execution error)
CRC-16	2 bytes, LSB - CyclicRedundancy Check (CRC-16): The CRC is calculated over all data starting from the slave address up to the last data byte.

2.1.3 No Response

The instrument does not process or return a response in the following cases. A communication timeout may occur.

- Slave address error: The received slave address does not match the configured device address.
- Transmission error: Data corruption occurs during communication.
- CRC-16 error: CRC checksum verification fails.

4. Frame format error

Example: For function code 0x03, the expected frame length is 8 bytes; if the received frame length is less or greater than 8 bytes, it is considered invalid.

5. Broadcast address (0x00): When the slave address is 0x00, the command is treated as a broadcast message. The instrument does not return any response.

2.1.4 Error Code

Table 2-3 Error Code

Code	Name	Description	Priority
0x01	Illegal function	Function code does not exist	1
0x02	Illegal data address	Register does not exist	2
0x03	Illegal data value	Invalid register count or byte number	3
0x04	Device failure (execution error)	Illegal data, written data outside valid range	4

2.2 Function Code

The instrument currently supports only the following codes. Other codes are not supported.

Table 2-4 Function Code

Function Code	Name	Description
0x03	Read holding registers	Reads data from multiple consecutive holding registers
0x10	Write holding registers	Writes data to multiple consecutive holding registers

2.3 Register and Data Format

Register size: 16 bits (2 bytes) per register.

For example, for the speed register (0x3002), the data is 2 bytes, and the numerical value must be written as 0x0001.

Data Format

The instrument supports the following data formats:

1. 1 register (16-bit integer)

For example, 0x64 → 00 64

2. 2 register (32-bit integer)

For example, 0x12345678 → 12 34 56 78

3. 2 register (32-bit float)

For example, 3.14 → 40 48 F5 C3

2.4 Read Holding Registers

Figure 2-4 Read Holding Registers (0x03) Request Frame Format

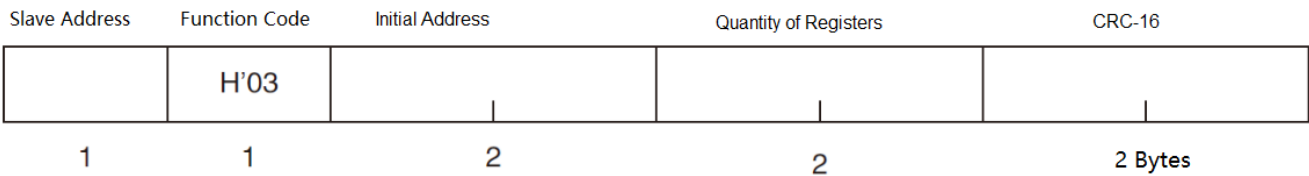
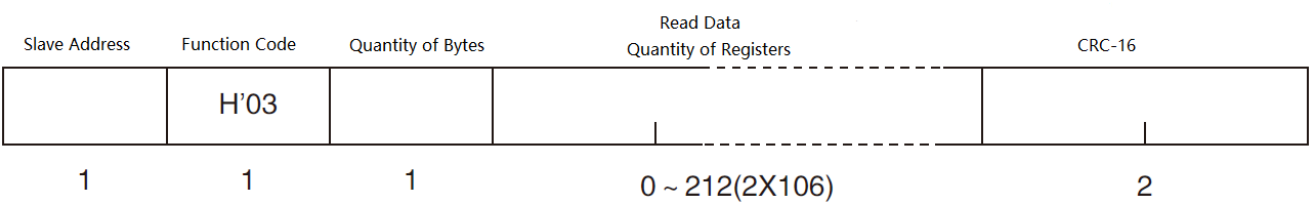


Table 2-5 Read Holding Registers (0x03) Request Fields

Code	Field	Description
	Slave address	If the RS485 address is not specified, the default value is 0x01.
0x03	Function code	
	Starting address	The starting address of register. Refer to the Modbus instruction set for valid address ranges.
	Quantity of registers	Number of registers to read: 0x0001-0x006A (1-106). Ensure all requested registers exist; otherwise returns exception code 0x02.
CRC-16	Check code	The CRC-16 checksum is used to verify data integrity in the Modbus frame.

Figure 2-5 Read Holding Register (0x03) Response Frame Format



Code	Field	Description
	Slave address	Returns the original slave address
0x03 or 0x83	Function code	Normal: 0x03 Exception: 0x83
	Byte count	Data bytes to follow = 2 × register quantity For example, reading 1 register returns Byte Count = 0x02, followed by 2 data bytes.
	Data	Register values

CRC-16	Check code	The CRC-16 checksum is used to verify data integrity in the Modbus frame.
--------	------------	---

2.5 Write Holding Registers

Figure 2-6 Write Holding Registers (0x10) Request Frame Format

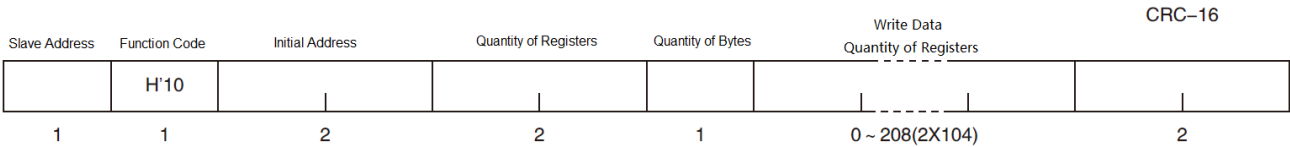
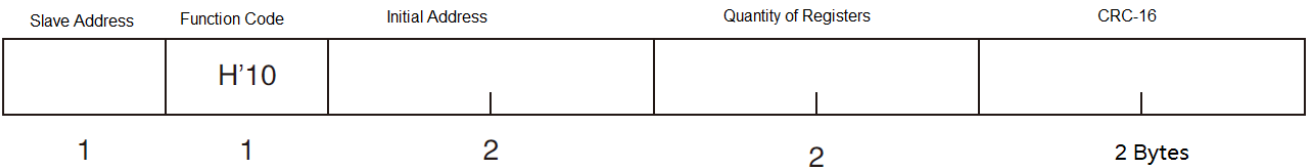


Table 2-6 Write Holding Registers(0x10) Request Fields

Code	Field	Description
	Slave address	If the RS485 address is not specified, the default value is 0x01.
0x10	Function code	
	Starting address	The starting address of register. Refer to the Modbus instruction set for valid address ranges.
	Quantity of registers	Number of registers to write: 0x0001-0x0068 (1-104). Ensure all requested registers exist; otherwise returns exception code 0x02.
	Byte count	Data bytes to follow = 2 × register quantity
CRC-16	Check code	The CRC-16 checksum is used to verify data integrity in the Modbus frame.

Figure 2-7 Write Holding Registers (0x10) Response Frame Format



Code	Field	Description
	Slave address	Returns the original slave address
0x10 or 0x90	Function code	Normal: 0x10 Exception: 0x90
	Starting address	The starting address of register. Refer to the Modbus instruction set for valid address ranges.
	Quantity of	

	registers	
CRC-16	Check code	The CRC-16 checksum is used to verify data integrity in the Modbus frame.

3. Modbus(RTU) Set

3.1 Register Overview

1. Unless otherwise specified, all values in request and response frames are **hexadecimal**.
2. For online floating-point conversion: http://www.binaryconvert.com/convert_float.html
3. Register access types are indicated as follows: RW (read-write), R (read-only), and W (write-only).
- 4.

UT5470/UT5472 Register Address

Register		Size	Function	Data (Hexadecimal)	Format	R/ W	Valid Values
Address	Byte Count						
0200	1	2	Current position	16-bit (2 bytes) integer		RW	1-200
0201	1	2	Maximum position	16-bit (2 bytes) integer		R	1-200
0202	1	2	New step	16-bit (2 bytes) integer		W	Write 0x0001 to step forward
0203	1	2	Step delete	16-bit (2 bytes) integer		W	Write 0x0001 to delete step
0204	1	2	Test mode	16-bit (2 bytes) integer		RW	UT5470: AC, OSC UT5472: DC, IR
0205	2	4	Output voltage	32-bit float (4 bytes)		RW	UT5470: AC: 50-10000 (Decimal) OSC: Fixed 100 (Decimal) UT5472: DC: 50-12000 (Decimal) IR: 50-6000 (Decimal) (Default unit: V)
0207	2	4	Upper limit of current/ resistance	32-bit float (4 bytes)		RW	UT5470: AC: 0.001-20.00mA (Decimal) UT5472: DC: 0.1-10000uA

						(Decimal) IR: 0.0-50000MΩ (Decimal)
0209	2	4	Lower limit of current /resistance	32-bit float (4 bytes)	RW	UT5470: AC: 0.000-20.00mA (Decimal) UT5472: DC: 0.0-10000uA (Decimal) IR: 0.1-50000MΩ (Decimal)
020B	2	4	Ramp-up time	32-bit float (4 bytes)	RW	0.3-999.9 (Decimal)
020D	2	4	Test duration (dwell time)	32-bit float (4 bytes)	RW	0.1-999.9, 0 (Decimal)
020F	2	4	Ramp-down time	32-bit float (4 bytes)	RW	0-999.9 (Decimal)
0211	2	4	Output frequency	32-bit (2 bytes) integer	RW	UT5470: AC: 50/60 Hz (Decimal)
0212	2	4	Delay time	32-bit float (4 bytes)	RW	UT5472: IR: 0.5-999.9s (Decimal)
0213	1	2	Arc detection state	16-bit (2 bytes) integer	RW	0: Off 1: On
0214	1	2	Arc detection level	16-bit (2 bytes) integer	RW	1-9 (Decimal)
0215	1	2	Range mode	16-bit (2 bytes) integer	RW	0: Auto 1: Fixed
0216	1	2	Startup clear	16-bit (2 bytes) integer	RW	1: On
0217	2	4	Zero clear	32-bit float (4 bytes)	RW	0
0219	1	2	Open-circuit detection level	16-bit (2 bytes) integer	RW	10-100%
021A	1	2	Short-circuit detection level	16-bit (2 bytes) integer	RW	100-500%, 0
021B	2	2	Sampling rate	32-bit float (4 bytes)	RW	0.100-40.00 nF
021D	1	1	Current file	16-bit (2 bytes) integer	RW	1-100
021E	1	1	Save file	16-bit (2 bytes) integer	RW	1: Save file
021F	1	1	Delete file	16-bit (2 bytes) integer	RW	1: Delete file
0220	1	2	Test control	16-bit (2 bytes) integer	RW	1: Start 2: Stop
0221	1	2	Test state	16-bit (2 bytes) integer	R	0: Idle

						1: Testing
0222	1	2	Read the measurement mode of step 1	16-bit (2 bytes) integer	R	
0223	1	2	Read the sorting result of step 1	16-bit (2 bytes) integer 0001: Pass 0002: Abort 0003: High Limit 0004: Low Limit 0005: Short Protection 0006: Arc protection 0007: Temperature protection 0008: Ground protection 0009: Low voltage protection 000A: Overvoltage protection 000B: Open circuit 000C: Short circuit 000D: Ramp upper limit 000E: Charge lower limit	R	
0224	2	4	Read the voltage measured in step 1	32-bit float (4 bytes)		
0226	2	4	Read the current/resistance measured in step 1	32-bit float (4 bytes)		

0228	2	4	Read the measurement time of step 1	32-bit float (4 bytes)		
022A	1	2	Read the measurement mode of step 2	16-bit (2 bytes) integer		
022B	1	2	Read the sorting result of step 2	16-bit (2 bytes) integer		
022C	2	4	Read the voltage measured in step 2	32-bit float (4 bytes)		
022E	2	4	Read the current/resistance measured in step 2	32-bit float (4 bytes)		
0230	2	4	Read the measurement time of step 2	32-bit float (4 bytes)		

UT5475 Register Address

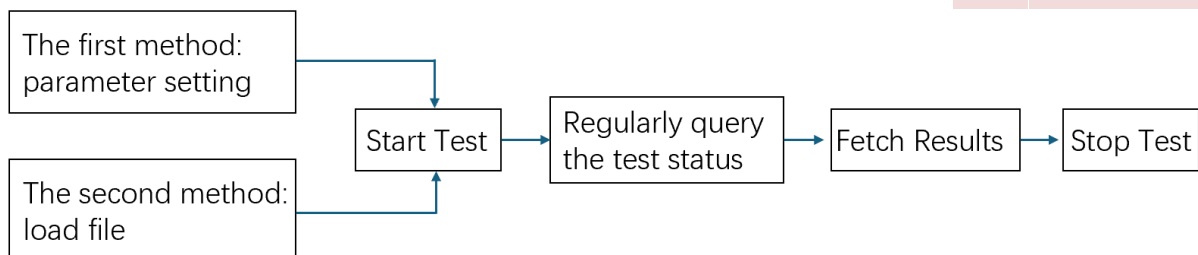
Register		Size	Function	Data Format (Hexadecimal)	R/ W	Valid Values
Address	Byte Count					
0200	1	2	Current position	16-bit (2 bytes) integer	RW	1-200
0201	1	2	Maximum position	16-bit (2 bytes) integer	R	1-200
0202	1	2	New step	16-bit (2 bytes) integer	W	Write 0x0001 to step forward
0203	1	2	Step delete	16-bit (2 bytes) integer	W	Write 0x0001 to delete step
0204	1	2	Test mode	16-bit (2 bytes) integer	RW	UT5475: AC、OSC、DC、IR
0205	2	4	Output voltage	32-bit float (4 bytes)	RW	AC : 50~10000(Decimal) OSC : Fixed 100(Decimal) DC : 50~12000(Decimal) IR : 50~6000

						(Decimal) (Default unit: V)
0207	2	4	Upper limit of current/ resistance	32-bit float (4 bytes)	RW	AC: 0.001-20.00mA (Decimal) DC: 0.1-10000uA (Decimal) IR: 0.0-50000MΩ (Decimal)
0209	2	4	Lower limit of current /resistance	32-bit float (4 bytes)	RW	AC: 0.000-20.00mA (Decimal) DC: 0.0-10000uA (Decimal) IR: 0.1-50000MΩ (Decimal)
020B	2	4	Ramp-up time	32-bit float (4 bytes)	RW	0.3-999.9 (Decimal)
020D	2	4	Test duration (dwell time)	32-bit float (4 bytes)	RW	0.1-999.9, 0 (Decimal)
020F	2	4	Ramp-down time	32-bit float (4 bytes)	RW	0-999.9 (Decimal)
0211	2	4	Output frequency	32-bit (2 bytes) integer	RW	AC: 50/60 Hz (Decimal)
0213	2	4	Delay time	32-bit float (4 bytes)	RW	IR: 0.5-999.9s (Decimal)
0215	1	2	Arc detection state	16-bit (2 bytes) integer	RW	0: Off 1: On
0216	1	2	Arc detection level	16-bit (2 bytes) integer	RW	1-9 (Decimal)
0217	1	2	Range mode	16-bit (2 bytes) integer	RW	0: Auto 1: Fixed
0218	1	2	Startup clear	16-bit (2 bytes) integer	RW	1: On
0219	2	4	Zero clear	32-bit float (4 bytes)	RW	0
021B	1	2	Open-circuit detection level	16-bit (2 bytes) integer	RW	10-100%
021C	1	2	Short-circuit detection level	16-bit (2 bytes) integer	RW	100-500%, 0
021D	2	2	Sampling rate	32-bit float (4 bytes)	RW	0.100-40.00 nF
021F	1	1	Current file	16-bit (2 bytes) integer	RW	1-100
0220	1	1	Save file	16-bit (2 bytes) integer	RW	1: Save file
0221	1	1	Delete file	16-bit (2 bytes) integer	RW	1: Delete file
0222	1	2	Test control	16-bit (2 bytes) integer	RW	1: Start

						2: Stop
0223	1	2	Test state	16-bit (2 bytes) integer	R	0: Idle 1: Testing
0224	1	2	Read the measurement mode of step 1	16-bit (2 bytes) integer	R	
0225	1	2	Read the sorting result of step 1	16-bit (2 bytes) integer 0001: Pass 0002: Abort 0003: High Limit 0004: Low Limit 0005: Short Protection 0006: Arc protection 0007: Temperature protection 0008: Ground protection 0009: Low voltage protection 000A: Overvoltage protection 000B: Open circuit 000C: Short circuit 000D: Ramp upper limit 000E: Charge lower limit	R	
0226	2	4	Read the voltage measured in step 1	32-bit float (4 bytes)		
0228	2	4	Read the current/resistance	32-bit float (4 bytes)		

			measured in step 1			
022A	2	4	Read the measurement time of step 1	32-bit float (4 bytes)		
022C	1	2	Read the measurement mode of step 2	16-bit (2 bytes) integer		
022D	1	2	Read the sorting result of step 2	16-bit (2 bytes) integer		
022E	2	4	Read the voltage measured in step 2	32-bit float (4 bytes)		
0230	2	4	Read the current/resistance measured in step 2	32-bit float (4 bytes)		
0232	2	4	Read the measurement time of step 2	32-bit float (4 bytes)		

3.2 Modbus Command Flow



3.3 Parameter Setting

3.3.1 Current Step Test Mode

Write Holding Registers

1	2	3	4	5	6	7	8	9	10	11
01	10	02	04	00	01	02	00	00	84	14
Slave Address	Write	Register Address		Register Quantity		Byte Count	Data		CRC16	

B8-B9:

UT5470

0000: ACW

0001: OSC

UT5472

0000: DCW

0001: IR

UT5475

0000:ACW

0001:OSC

0002:DCW

0003:IR

Write Response

1	2	3	4	5	6	7	8
01	10	02	04	00	01	41	B0
Slave Address	Write	Register Address		Register Quantity		CRC16	

3.3.2 Output Voltage of Test Mode

Write Holding Registers

1	2	3	4	5	6	7	8	9	10	11	12	13
01	10	02	05	00	02	04	42	C8	00	00	BF	76
Slave Address	Write	Register Address	Register Quantity	Byte Count	Data					CRC16		

B8-B11: Hexadecimal value for floating-point conversion. Example: 42C80000 = 100 (Decimal), i.e,100 V.

AC: 50-10000V (Decimal)

DC: 50-12000V (Decimal)

IR: 50-6000V (Decimal)

Write Response

1	2	3	4	5	6	7	8
01	10	02	05	00	02	50	71

Slave Address	Write	Register Address	Register Quantity	CRC16
---------------	-------	------------------	-------------------	-------

3.3.3 Multiple Test Parameters

Write Holding Registers

1	2	3	4	5	6	7	8-11	12-15	16-19	20	21
01	10	02	05	00	06	0C	42C80000	41400000	40000000	D7	87
Slave Address	Write	Register Address	Register Quantity		Byte Count	Output Voltage	Upper Limit of Current /Resistance	Lower Limit of Current /Resistance	CRC16		

B8-B11: Output voltage, 42C80000 (floating-point number) = 100 V

B12-B15: Upper limit of current/resistance, 41400000 (floating-point number) = 12 mA

B16-B19: Lower limit of current/resistance, 40000000 (floating-point number) = 2 mA

Write Response

1	2	3	4	5	6	7	8
01	10	02	05	00	06	51	B2
Slave Address	Write	Register Address		Register Quantity		CRC16	

3.4 Read Measured Results

Example: Reading UT5470 test results from Step 1 and Step 2.

Parameters:

- Step 1 Test Mode: AC
- Step 2 Test Mode: AC
- Voltage Unit: kV (default)
- Current Unit: mA (default)

3.4.1 Reading Step 1 Measured Voltage

The register 0222 stores the measured voltage from Step 1.

Request

1	2	3	4	5	6	7	8
---	---	---	---	---	---	---	---

01	03	02	22	00	01	25	B8
Slave Address	Read	Register		Register Quantity		CRC-16	

Response

1	2	3	4	5	8	9
01	03	02	00	00	B8	44
01	03	Byte Count	Integer		CRC-16	

Among them, B4~B5 are integers. The measurement mode is 0x0000, corresponding to AC withstand voltage.

3.4.2 Reading Step 1 Measured Current/Resistance

Register 0x0223 is used to obtain the measurement data of the step 1 sorting result.

Request

1	2	3	4	5	6	7	8
01	03	02	23	00	01	74	78
Slave Address	Read	Register		Register Quantity		CRC-16	

Response

1	2	3	4	5	8	9
01	03	02	00	01	79	84
01	03	Byte Count	Integer		CRC-16	

Among them, B4~B5 are integers. A return result of 0x0001 indicates "PASS".

3.4.3 Reading Step 1 Sorting Results

Register 0x0224 is used to obtain the voltage measurement data of step 1.

Request

1	2	3	4	5	6	7	8
01	03	02	24	00	02	85	B8
Slave Address	Read	Register		Register Quantity		CRC-16	

Response

1	2	3	4	5	6	7	8	9
01	03	04	3D	D2	B7	08	21	90
01	03	Byte Count	Single-precision floating-point				CRC-16	

			number	
--	--	--	--------	--

Among them, B4~B7 are single-precision floating-point numbers, with byte order AA BB CC DD.

Measurement data: 3D D2 B7 08 converted to floating-point number: 0x3DD2B708 = 0.102888 (decimal), which is 0.103 kV.

3.4.4 Reading Step 1/2 Test Results

Register 0x0226 is used to obtain the current/resistance measurement data of step 1.

Request

1	2	3	4	5	6	7	8
01	03	02	26	00	02	24	78
Slave Address	Read	Register	Register Quantity			CRC-16	

Response

1	2	3	4	5	6	7	8	9
01	03	04	3A	6A	3E	B5	07	20
01	03	Byte Count	Single-precision floating-point number				CRC-16	

Among them, B4~B7 are single-precision floating-point numbers. 0x3A6A3EB5 = 0.000893574 (decimal), which is 0.001 mA.

3.4.5 Get the measurement time of step 1

Register 0x0228 is used to obtain the measurement time of step 1.

Request

1	2	3	4	5	6	7	8
01	03	02	28	00	02	45	BB
Slave Address	Read	Register	Register Quantity			CRC-16	

Response

1	2	3	4	5	6	7	8	9
01	03	04	3F	00	00	00	F6	27
01	03	Byte Count	Single-precision floating-point number				CRC-16	

Among them, B4~B7 are single-precision floating-point numbers. 0x3F000000 = 0.5 (decimal), which is 0.5 seconds.

3.4.6 Simultaneously obtain the measurement results of step 1 and step 2.

Register 0x0222 is used to obtain the measurement results of step 1 and step 2.

Referring to this example, measurement data of multiple steps can be obtained simultaneously.

Request

1	2	3	4	5	6	7	8
01	03	02	22	00	10	E5	B4
Slave Address	Read	Register		Register Quantity		CRC-16	

Response

1	2	3	4-5	6-7	8-11	12-15	16-19
01	03	20	0000	0001	3DD2B708	3A6A3EB5	3F000000
01	03	Byte Count	Step 1 mode	Step 1 sorting	Step 1 voltage	Step 1 current	Step 1 time

20-21	22-23	24-27	28-31	32-35	36	37
0000	0001	3F804F83	3A6B4459	3F000000	79	19
Step 2 mode	Step 2 sorting	Step 2 voltage	Step 2 current	Step 2 time	CRC-16	

Where,

B4~B5: Step 1 Mode — AC Withstand Voltage

B6~B7: Step 1 Sorting — PASS

B8~B11: 0x3DD2B708 Step 1 Voltage = 0.103 kV

B12~B15: 0x3A6A3EB5 Step 1 Current = 0.001 mA

B16~B19: 0x3F000000 Step 1 Time = 0.5 s

B20~B21: Step 2 Mode — AC Withstand Voltage

B22~B23: Step 2 Sorting — PASS

B24~B27: 0x3F804F83 Step 2 Voltage = 1.002 kV

B28~B31: 0x3A6B4459 Step 2 Current = 0.001 mA

B32~B35: 0x3F000000 Step 2 Time = 0.5 s

Start/Stop Test

Write

1	2	3	4	5	6	7	8	9	10	11
01	10	02	2 0	00	01	02	00	01	43	30

Station ID	Write	Register	Register Quantity	Byte Count	Data	CRC16
------------	-------	----------	-------------------	------------	------	-------

B8~B9:

0001: Start

0002: Stop

Write Response

1	2	3	4	5	6	7	8
01	10	02	20	00	01	01	BB
Slave Station	Write	Register	Register Quantity	CRC16			

3.5 Start/Stop Test

Write Holding Registers

1	2	3	4	5	6	7	8	9	10	11
01	10	02	20	00	01	02	00	01	43	30
Slave Address	Write	Register	Register Quantity	Byte Count	Data	CRC16				

B8~B9:

0001: Start

0002: Stop

Write Response

1	2	3	4	5	6	7	8
01	10	02	20	00	01	01	BB
Slave Address	Read	Register	Register Quantity	CRC16			