UNI-T®

Instruments.uni-trend.com

SCPI Programming Manual

UDP6900 Series Programmable Digital Control Power

Copyright

Trademark Information

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

Software Version

1.00.0905

Software upgrade may have some change and add more Function, please subscribe UNI-T website to get the new version or contact UNI-T.

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.
- The information provided in this manual is subject to change without notice.
- UNI-T shall not be liable for any errors that may be contained in this manual. For any incidental or consequential damages arising out of the use or the information and deductive Functions provided in this manual.
- No part of this manual shall be photocopied, reproduced or adapted without the prior written permission of UNI-T.

Product Certification

UNI-T has certified that the product conforms to China national product standard and industry product standard as well as ISO9001:2008 standard and ISO14001:2004 standard. UNI-T will go further to certificate product to meet the standard of other member of the international standards organization.

Chapter 1 SCPI Introduction

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.

Instruction Format

Command is consisting of a keyword, separator, parameter domain and end mark. Take the following command as an example.

:VOLTage:LEVel 25

VOLTage, LEVel is keyword, ":" and blank is separator, "25" is parameter (some commands have multiple parameters and separated by ","), the line separator or carriage return after the command is the end mark.

For the convenience of description, the following conventions are adopted for subsequent symbols.

- Square Brackets "[ ]"
  The contents in square brackets (command keywords) can be omitted.

- Braces "{ }"
  It represents the parameter in command string.

- Angle Braces "< >"
  The parameter enclosed in the angle brackets must be a numerical parameter.

- Vertical Bar "|"
  It is used to separate multiple parameters.

- End Mark
  The line separator <LF> (0x0A) or carriage return <CR> (0x0D).

Parameter Description

The data types of programming parameters are numeric, character, Boolean, and many other types. Regardless of the type, it is all expressed in ASCII. For more details, see the following table.

Symbol	Meaning	Example
<NR1>	Integer	123, 0123
<NR2>	Fixed floating point number	123., 12.3, 0.123, .123
<NR3>	Floating point number	123, 12.3, 123E+3
<NRF>	It may be <NR1>, <NR2> or <NR3>	
<Boolean>	Boolean data	0|1|ON|OFF

Shorthand Rule

All the commands are case-insensitive. The commands can be all input in uppercase letters or in lowercase letters. For abbreviations, it should enter all the uppercase letters that exist in the command syntax.

Data Return

Data return is divided into single data and batch data. The single data return is the corresponding parameter type, in which the real return type is presents by the scientific notation method. The part before e retains three figure behind the decimal point, and the e part retains three figure; the batch return must be obey IEEE 488.2# string data format, **'#'+ the length of character bits** [fixed to one character] + ASCII of **valid data length** + **valid data**+ **end mark** ['\n']. Such as, **#3123xxxxxxxxxxxxxxxxxxx\n** represents **123** bytes of valid batch data return format, where '**3**' means that "123" occupies 3 character bits.

Multi-SCPI

Multi-SCPI instruction adds address information to realize multi-machine SCPI communication on the basis of standard SCPI protocol.

Command Format

ADDR[space]<address>: {SCPI}

<address>=multi-computer address, value: 1~32, the broadcast address is 0.

Example

ADDR 1:*IDN?

->Uni-Trend,UDP6942B,[Serial Number],1.00.0807

Chapter 2 Communication Interface and Setting

The detailed description can refer to "Chapter 3 3.15.4 RS232 and RS485" in *UDP6900 series digital control power – user's manual.*

Chapter 3.SCPI Command

3.1 Command List

Measurement Command	Function
:MEASure:VOLTage?	Measure the currently output voltage
:MEASure:CURRent?	Measure the currently output current
:MEASure:POWEr?	Measure the currently output power
:MEASure:DVM?	Measure the voltage of the current voltmeter channel
:MEASure:ALL?	Measure the currently output voltage, current and power
Output Setting Command	**Function**
[:SOURce]:VOLTage[:LEVel][:IMMediate][:AMPLitude]	Set the output voltage value
[:SOURce]:CURRent [:LEVel][:IMMediate][:AMPLitude]	Set the output current value
[:SOURce]:VOLTage:PROTection[:LEVel]	Set the protective value for OVP (over-voltage protection) Function
[:SOURce]:CURRent:PROTection[:LEVel]	Set the protective value for OCP (over-current protection) Function
[:SOURce]:VOLTage:PROTection:STATe	Turn on or off the OVP Function
[:SOURce:]CURRent:PROTection:STATe	Turn on or off the OCP Function
[:SOURce:]VOLTage:PROTection:TRIPed?	Query the trigger state of OVP
[:SOURce:]VOLTage:PROTection:CLEar	Clear the trigger state of OVP
[:SOURce:]CURRent:PROTection:TRIPed?	Query the trigger state of OCP
[:SOURce:]CURRent:PROTection:CLEar	Clear the trigger state of OCP
[:SOURce:]VOLTage:STEP	Set the step value for the set voltage
[:SOURce:]VOLTage:UP	The set value of step-up voltage
[:SOURce:]VOLTage:DOWN	The set value of step-down voltage
[:SOURce:]CURRent:STEP	Set the step value for the set current
[:SOURce:]CURRent:UP	The set value of step-up current
[:SOURce:]CURRent:DOWN	The set value of step-down current

[:SOURce:]VOLTage:SLEW:RISing	Set the rising slope for the voltage
[:SOURce:]VOLTage:SLEW:FALLing	Set the falling slope for the voltage
[:SOURce:]CURRent:SLEW:RISing	Set the rising slope for the current
[:SOURce:]CURRent:SLEW:FALLing	Set the falling slope for the current
:OUTPut:MODE	Set the slope mode for the voltage
:OUTPut[:STATe]	Turn on or off the power output
:OUTPut:CVCC?	Query the state of output constant voltage and constant current
:OUTPut:OVP:VALue	Set the protective value of OVP Function
:OUTPut:OCP:VALue	Set the protective value of OCP Function
:OUTPut:OVP[:STATe]	Turn on or off the OVP Function
:OUTPut:OCP[:STATe]	Turn on or off the OCP Function
:OUTPut:OVP:TRIPed?	Query the trigger state of OVP
:OUTPut:OVP:CLEar	Clear the trigger state of OVP
:OUTPut:OCP:TRIPed?	Query the trigger state of OCP
:OUTPut:OCP:CLEar	Clear the trigger state of OCP
:OUTPut:TIMer	Set the switch of the timer
:OUTPut:TIMer:DATA	Set the duration of the timer
Command of List Mode	Function
:LISTout[:STATe]	Set the switch of the list mode and the state of the list mode
:LISTout:BASE	Set the basic parameter for the list mode
:LISTout:PARAmeter	Set the group parameter for the list mode
:LISTout:TEMPlet:CONSTruct	According to the set template parameters to build the group parameter for the list mode
:LISTout:TEMPlet:SELect	Set the template type for the list mode
:LISTout:TEMPlet:OBJect	Set the template object for the list mode
:LISTout:TEMPlet:STARt	Set the start group for the list mode
:LISTout:TEMPlet:POINTs	Set the number of the template for the list mode
:LISTout:TEMPlet:MAXValue	Set the maximum value of the template for the list mode
:LISTout:TEMPlet:MINValue	Set the minimum value of the template for the list mode
:LISTout:TEMPlet:INTErval	Set the interval of the template for the list mode
:LISTout:TEMPlet:INVErt	Set the invert mode for the template for the list mode
:LISTout:TEMPlet:WIDTh	Set the pulse width of the template for the list mode
:LISTout:TEMPlet:PERIod	Set the pulse mode for the list mode
:LISTout:TEMPlet:SYMMetry	Set the symmetry of the template for the list mode

:LISTout:TEMPlet:EXPRate	Set the exponential value of the template for the list mode
Delay Command	Function
:DELAY[:STATe]	Enable/Disable the delay
:DELAY:STARt	Set the start group for the delay
:DELAY:GROUPs	Set the number of group for the delay
:DELAY:CYCLEs	Set the cycles for the delay
:DELAY:ENDState	Set the stop state for the delay
:DELAY:STOP	Set the stop condition for the delay
:DELAY:PARAmeter	Set the group parameter for the delay
:DELAY:GENerate:STAT	Set the state code of the delay template
:DELAY:GENerate:FIX	Set the fixed duration of the delay template
:DELAY:GENerate:INC	Set the progressive increase for the delay template
:DELAY:GENerate:DEC	Set the progressive decrease for the delay template
Preset Command	Function
:PRESet#[:APPLy]	Set the parameter application for the preset group
:PRESet#:SET:VOLTage	Set the voltage for the preset group
:PRESet#:SET:CURRent	Set the current for the preset group
:PRESet#:SET:OVP	Set the protective value of OVP for the preset group
:PRESet#:SET:OCP	Set the protective value of OCP for the preset group
:PRESet#:SET:TIMer	Set the timer for the preset group
:PRESet#:SET:TIMer:DATA	Set the duration of timer for the preset group
Monitor Command	Function
:MONItor[:STATe]	Turn on or off the monitor
:MONItor:VOLTage	Set the voltage condition for the monitor
:MONItor:CURRent	Set the current condition for the monitor
:MONItor:POWER	Set the voltage condition for the monitor
:MONItor:DVM	Set the monitor for the digital voltmeter
:MONItor:LOGic	Set the logic of conditional judgment for the monitor
:MONItor:STOPway	Set the stop condition for the monitor
System Command	Function
:SYSTem:REMote	Set the power to remote control mode
:SYSTem:LOCal	Set the power to panel control mode
:SYSTem:BEEPer:TEST	Test the beeper, it will make one sound (the beeper must be turned on)
:SYSTem:BEEPer[:STATe]	Turn on or off the beeper
:SYSTem:BRIGhtness	Set the screen brightness for the power

:SYSTem:COMMunicate:COM:BAUD	Set the baud rate for the RS232 and RS485
:SYSTem:COMMunicate:COM:PROTocol	Set the communication protocol of RS232 and RS485
:SYSTem:COMMunicate:COM:ADDRess	Set the multi-machine address of RS232 and RS485
:SYSTem:COMMunicate:LAN:APPLy	Save and apply the set network parameter
:SYSTem:COMMunicate:LAN:DHCP[:STATe]	Turn on or off the DHCP Function
:SYSTem:COMMunicate:LAN:IPADdress	Set the IP address for the LAN port
:SYSTem:COMMunicate:LAN:SMASK	Set the subnet mask for the LAN port
:SYSTem:COMMunicate:LAN:GATEway	Set the gateway address for the LAN port
:SYSTem:ERRor[:NEXT]?	Acquire the error code and character string of SCPI
:SYSTem:ERRor:COUNt?	Acquire the error code and character string of SCPI
:SYSTem:VERSion?	Acquire the version of SCPI
SCPI Command	Function
*IDN?	Query the instrument information
*STB?	Query the state byte event register
*SRE	Set the state byte event enable register of SCPI
*ESR?	Query the status register of SCPI
*ESE	Set the event enable register of SCPI
:STATus:QUEStionable[:EVENt]?	Query the QUSE event register of SCPI
:STATus:QUEStionable:CONDition?	Query the QUSE status register of SCPI and return the current state
:STATus:QUEStionable:ENABle	Set the QUSE enable register of SCPI

3.2 Instruction Prase

:MEASure:VOLTage?

Function Measure the currently output voltage

Syntax :MEASure:VOLTage?

Example :MEASure:VOLTage? -> <OutVoltage>

:MEASure:CURRent?

Function Measure the currently output current

Syntax :MEASure:CURRent?

Example :MEASure:CURRent? -> <OutCurrent>

:MEASure:POWEr?

Function Measure the currently output power

Syntax :MEASure:POWEr?

Example :MEASure:POWEr? -> <OutPower>

:MEASure:DVM?

Function Measure the voltage of the current voltmeter channel

Syntax :MEASure:DVM?

Example :MEASure:DVM? -> <Dvm>

:MEASure:ALL?

Function Measure the currently voltage, current and power

Syntax :MEASure:ALL?

Example :MEASure:ALL? -> <OutVoltage>,<OutCurrent>,<OutPower>

[:SOURce]:VOLTage[:LEVel]

Function Set the output voltage

Syntax [:SOURce]:VOLTage[:LEVel]{<Value>|MINimum|MAXimum}

[:SOURce]:VOLTage[:LEVel]?

<Value>= output the set voltage, the unit is V.

Example :VOLTage <Value>

:VOLTage? -> <Value>

[:SOURce]:CURRent[:LEVel]

Function Set the output current

Syntax [:SOURce]:CURRent[:LEVel]{<Value>|MINimum|MAXimum}

[:SOURce]:CURRent[:LEVel]?

<Value>= output the set current, the unit is A.

Example :CURRent <Value>

:CURRent? -> <Value>

[:SOURce]:VOLTage:PROTection[:LEVel]

Function Set the protective value of OVP

Syntax [:SOURce]:VOLTage:PROTection[:LEVel]{<Value>|MINimum|MAXimum}

[:SOURce]:VOLTage:PROTection[:LEVel]?

<Value>= the protective value of OVP, the unit is V.

Example :VOLTage:PROTection <Value>

:VOLTage:PROTection? -> <Value>

[:SOURce]:CURRent:PROTection[:LEVel]

Function Set the protective value of OCP

Syntax [:SOURce]:CURRent:PROTection[:LEVel]{<Value>|MINimum|MAXimum}

[:SOURce]:CURRent:PROTection[:LEVel]?

<Value>= the protective value of OCP, the unit is A.

Example :CURRent:PROTection <Value>

:CURRent:PROTection? -> <Value>

[:SOURce]:VOLTage:PROTection:STATe

Function Set the switch of OVP

Syntax [:SOURce]:VOLTage:PROTection:STATe {<Boolean>}

[:SOURce]:VOLTage:PROTection:STATe?

<Boolean>=ON|OFF|0|1

Example :VOLTage:PROTection:STATe <Boolean>

:VOLTage:PROTection:STATe? -> <Boolean>

[:SOURce]:CURRent:PROTection:STATe

Function Set the switch of OCP

Syntax [:SOURce]:CURRent:PROTection:STATe {<Boolean>}

[:SOURce]:CURRent:PROTection:STATe?

<Boolean>=ON|OFF|0|1

Example :CURRent:PROTection:STATe <Boolean>

:CURRent:PROTection:STATe? -> <Boolean>

[:SOURce:]VOLTage:PROTection:TRIPed?

Function Query the trigger state of OVP

Syntax [:SOURce:]VOLTage:PROTection:TRIPed? <State>

<State>= 1 indicates triggered; 0 indicates not trigger.

Example :VOLTage:PROTection:TRIPed? -> <State>

[:SOURce:]VOLTage:PROTection:CLEar

Function Clear the trigger state of OVP

Syntax [:SOURce:]VOLTage:PROTection:CLEar

[:SOURce:]CURRent:PROTection:TRIPed?

Function Query the trigger state of OCP

Syntax [:SOURce:]CURRent:PROTection:TRIPed? <State>

<State>= 1 indicates triggered; 0 indicates not trigger.

Example :CURRent:PROTection:TRIPed? -> <State>

[:SOURce:]CURRent:PROTection:CLEar

Function Clear the trigger state of OCP

Syntax [:SOURce:]CURRent:PROTection:CLEar

[:SOURce:]VOLTage:STEP

Function Set the step-up voltage

Syntax [:SOURce:]VOLTage:STEP {<Value>}

[:SOURce:]VOLTage:STEP?

<Value>= the step-up voltage, the unit is V.

Example :VOLTage:STEP <Value>

:VOLTage:STEP? -> <Value>

[:SOURce:]VOLTage:UP

Function Increase the set voltage according to the step-up value

Syntax [:SOURce:]VOLTage:UP

[:SOURce:]VOLTage:DOWN

Function Decrease the set voltage according to the step-up value

Syntax [:SOURce:]VOLTage:DOWN

[:SOURce:]CURRent:STEP

Function Set the step-up of current

Syntax [:SOURce:]CURRent:STEP {<Value>}

[:SOURce:]CURRent:STEP?

<Value>= the step-up of current, the unit is A.

Example :CURRent:STEP <Value>

:CURRent:STEP? -> <Value>

[:SOURce:]CURRent:UP

Function Increase the set current according to the step-up value

Syntax [:SOURce:]CURRent:UP

[:SOURce:]CURRent:DOWN

Function Decrease the set current according to the step-up value

Syntax [:SOURce:]CURRent:DOWN

[:SOURce:]VOLTage:SLEW:RISing

Function Set the rising slope of voltage

Syntax [:SOURce]:VOLTage:SLEW:RISing {<Value>}

[:SOURce]:VOLTage:SLEW:RISing?

<Value>= the slope of voltage, the unit is V/s.

Example :VOLTage:SLEW:RISing <Value>

:VOLTage:SLEW:RISing? -> <Value>

[:SOURce:]VOLTage:SLEW:FALLing

Function Set the falling slope of voltage

Syntax [:SOURce]:VOLTage:SLEW:FALLing {<Value>}

[:SOURce]:VOLTage:SLEW:FALLing ?

<Value>= the slope of voltage, the unit is V/s.

Example :VOLTage:SLEW:FALLing <Value>

:VOLTage:SLEW:FALLing? -> <Value>

[:SOURce:]CURRent:SLEW:RISing

Function Set the rising slope of current

Syntax [:SOURce]:CURRent:SLEW:RISing {<Value>}

[:SOURce]:CURRent:SLEW:RISing?

<Value>= the slope of current, the unit is A/s.

Example :CURRent:SLEW:RISing <Value>

:CURRent:SLEW:RISing? -> <Value>

[:SOURce:]CURRent:SLEW:FALLing

Function Set the falling slope of current

Syntax [:SOURce]:CURRent:SLEW:FALLing {<Value>}

[:SOURce]:CURRent:SLEW:FALLing ?

<Value>= the slope of current, the unit is A/s.

Example :CURRent:SLEW:FALLing <Value>

:CURRent:SLEW:FALLing? -> <Value>

:OUTPut:MODE

Function Set the slope mode for the power

Syntax :OUTPut:MODE {<Mode>}

:OUTPut:MODE?

<Mode>=NORMAL (normal mode); VSR (the slope of voltage, the CV mode is priority); ISR (the slope of current, the CC mode is priority).

Example :OUTPut:MODE <Mode>

:OUTPut:MODE? -> <Mode>

:OUTPut:OCP:VALue

Function Set the protective value of OCP

Syntax :OUTPut:OCP:VALue {<Value>|MINimum|MAXimum}

:OUTPut:OCP:VALue?

<Value>= the protective value of OCP, the unit is A.

Example :OUTPut:OCP:VALue <Value>

:OUTPut:OCP:VALue? -> <Value>

:OUTPut:OVP[:STATe]

Function Set the switch of OVP

Syntax :OUTPut:OVP[:STATe]{<Boolean>}

:OUTPut:OVP[:STATe]?

<Boolean>= ON|OFF|0|1

Example :OUTPut:OVP <Boolean>

:OUTPut:OVP? -> <Boolean>

:OUTPut:OCP[:STATe]

Function Set the switch of OCP

Syntax :OUTPut:OCP[:STATe]{<Boolean>}

:OUTPut:OCP[:STATe]?

<Boolean>= ON|OFF|0|1

Example :OUTPut:OCP <Boolean>

:OUTPut:OCP? -> <Boolean>

:OUTPut:OVP:TRIPed?

Function Query the trigger state of OVP

Syntax :OUTPut:OVP:TRIPed? -><State>

<State>= 1 indicates triggered; 0 indicates not trigger.

Example OUTPut:OVP:TRIPed? -> <State>

:OUTPut:OVP:CLEar

Function Clear the trigger state of OVP

Syntax :OUTPut:OVP:CLEar

:OUTPut:OCP:TRIPed?

Function Query the trigger state of OCP

Syntax :OUTPut:OCP:TRIPed? -><State>

<State>= 1 indicates triggered; 0 indicates not trigger.

Example :OUTPut:OCP:TRIPed? -> <State>

:OUTPut:OCP:CLEar

Function Clear the trigger state of OCP

Syntax :OUTPut:OCP:CLEar

:OUTPut:TIMer

Function Set the switch of timer

Syntax :OUTPut:TIMer {<Boolean>}

:OUTPut:TIMer?

<Boolean>= ON|OFF|0|1

Example :OUTPut:TIMer <Boolean>

:OUTPut:TIMer? -> <Boolean>

:OUTPut:TIMer

Function Set the switch of timer

Syntax :OUTPut:TIMer {<Boolean>}

:OUTPut:TIMer?

<Boolean>= ON|OFF|0|1

Example :OUTPut:TIMer <Boolean>

:OUTPut:TIMer? -> <Boolean>

:OUTPut:TIMer:DATA

Function Set the duration of timer

Syntax :OUTPut:TIMer:DATA {<Value>}

:OUTPut:TIMer:DATA?

<Value>= the duration of timer, the unit is second (s).

Example :OUTPut:TIMer <Value>

:OUTPut:TIMer? -> <Value>

:LISTout[:STATe]

Function Set the switch of list mode and query the state of list mode

Syntax :LISTout[:STATe]{<Boolean>}

:LISTout[:STATe]?->{<Boolean>,<time>,<curGroup>,<endGroup>,<remainCycle>,<endState>}

<Boolean>= ON|OFF|0|1

<time>= the time remaining of the current group, the unit is second (s).

<curGroup>= the current group

<endGroup>= the stop group

<remainCycle>= the remain cycle

<endState>= the stop state: OFF indicates the output is disabled; LAST indicates maintain the output.

Example :LISTout <Boolean>

:LISTout? -> ON,0.1,000,009,00000,OFF

:LISTout:BASE

Function Set the basic parameter for list mode

Syntax :LISTout:BASE {<Start>,<Groups>,<Cycle>,<endState>}

:LISTout:BASE?->{<Start>,<Groups>,<Cycle>,<endState>}

<Start>= start group

<Groups>= the number of group

<Cycle>= the number of cycle, 0 indicates infinite loop

<endState>= the stop state: OFF indicates the output is disabled; LAST indicates maintain the output.

Example :LISTout:BASE <Start>,<Groups>,<Cycle>,<endState>

:LISTout:BASE? -><Start>,<Groups>,<Cycle>,<endState>

:LISTout:PARAmeter

Function Set the group parameter for list mode

Syntax :LISTout:PARAmeter {<No>,<Volt>,<Curr>,<Time>}

:LISTout:PARAmeter?->{<No>,<Volt>,<Curr>,<Time>}

<No>= group serial number

<Volt>= voltage, the unit is V.

<Curr>= current, the unit is A.

<Time>= duration, the unit is s.

Example :LISTout:PARAmeter <No>,<Volt>,<Curr>,<Time>

:LISTout:PARAmeter? 0,2

->#226000,10.000,12.000, 100.0;#226001,20.000,07.539, 2.0;

#226 indicates 26 occupies two bytes, the data segment has 26 data {000,10.000,12.000,100.0;}, and the serial number 0 indicates the voltage is 10V, the current is 12A and 100 seconds. The specific format refers to the section Data Return of the chapter 1.

:LISTout:TEMPlet:CONSTruct

Function Build the group parameter of the list mode according to the set template

Syntax :LISTout:TEMPlet:CONSTruct

:LISTout:TEMPlet:SELect

Function Set the template mode for list mode

Syntax :LISTout:TEMPlet:SELect {SINE|PULSE|RAMP|UP|DN|UPDN|RISE|FALL}

:LISTout:TEMPlet:SELect?->{SINE|PULSE|RAMP|UP|DN|UPDN|RISE|FALL}

SINE: sine template

PULSE: pulse template

RAMP: slope template

UP: stair-up template

DN: stair-down template

UPDN: stair-up and stair-down template

RISE: exponential rising template

FALL: exponential falling template

Example :LISTout:TEMPlet:SELect SINE

:LISTout:TEMPlet:SELect?->SINE

:LISTout:TEMPlet:OBJect

Function Set the template object of list mode

Syntax :LISTout:TEMPlet:OBJect {V|C}

:LISTout:TEMPlet:OBJect?->{V|C}

V= voltage template C= current template

Example :LISTout:TEMPlet:OBJect V

:LISTout:TEMPlet:OBJect?->V

:LISTout:TEMPlet:STARt

Function Set the start group of list mode

Syntax :LISTout:TEMPlet:STARt {<Start>}

:LISTout:TEMPlet:STARt?->{<Start>}

<Start>= the start group of the template

Example :LISTout:TEMPlet:STARt 0

:LISTout:TEMPlet:STARt?->0

:LISTout:TEMPlet:POINTs

Function Set the group number of list mode

Syntax :LISTout:TEMPlet:POINTs <Groups>

:LISTout:TEMPlet:POINTs?->{<Groups>}

<Groups>= the group number of the template

Example :LISTout:TEMPlet:POINTs 64

:LISTout:TEMPlet:POINTs?->64

:LISTout:TEMPlet:MAXValue

Function Set the maximum value of the template for list mode

Syntax :LISTout:TEMPlet:MAXValue {<Value>}

:LISTout:TEMPlet:MAXValue?->{<Value>}

<Value>= the maximum value of the template

Example :LISTout:TEMPlet:MAXValue 10.000

:LISTout:TEMPlet:MAXValue?->10.000

:LISTout:TEMPlet:MINValue

Function Set the minimum value of the template for list mode

Syntax :LISTout:TEMPlet:MINValue {<Value>}

:LISTout:TEMPlet:MINValue?->{<Value>}

<Value>= the minimum value of the template

Example :LISTout:TEMPlet:MINValue 1.000

:LISTout:TEMPlet:MINValue?->1.000

:LISTout:TEMPlet:INTErval

Function Set the duration of the template for list mode

Syntax :LISTout:TEMPlet:INTErval {<Value>}

:LISTout:TEMPlet:INTErval?->{<Value>}

<Value>= the duration of the template, the unit is s, and the minimum resolution is 0.1s.

Example :LISTout:TEMPlet:INTErval 1.0

:LISTout:TEMPlet:INTErval?->1.0

:LISTout:TEMPlet:INVErt

Function Set the invert mode for list mode

Syntax :LISTout:TEMPlet:INVErt {<Boolean>}

:LISTout:TEMPlet:INVErt?->{<Boolean>}

<Boolean>= ON indicate turn on the invert function; OFF indicate turn off the invert function.

(The template mode can only be set if it is SINE template or PULSE template or RAMP template, otherwise, it is valid.)

Example :LISTout:TEMPlet:INVErt ON

:LISTout:TEMPlet:INVErt?->ON

:LISTout:TEMPlet:WIDTh

Function Set the pulse width for list mode

Syntax :LISTout:TEMPlet:WIDTh {<Width>}

:LISTout:TEMPlet:WIDTh?->{<Width>}

<Width>= the pulse width, the unit is s and the range is 0.1~(Period-0.1). The "Period" is pulse period.

(It is valid when the template mode is PULSE template, otherwise, it is valid.)

Example :LISTout:TEMPlet:WIDTh 5.0

:LISTout:TEMPlet:WIDTh?->5.0

:LISTout:TEMPlet:PERIod

Function Set the pulse period of the pulse template for list mode

Syntax :LISTout:TEMPlet:PERIod {<Period>}

:LISTout:TEMPlet:PERIod?->{<Period>}

<Period>= the pulse width, the unit is s and the range is (Width+0.1), 99999.9. The "Width" is pulse width

(It is valid when the template mode is PULSE template, otherwise, it is valid.)

Example :LISTout:TEMPlet:PERIod 5.0

:LISTout:TEMPlet:PERIod?->5.0

:LISTout:TEMPlet:SYMMetry

Function Set the symmetry of the ramp template for list mode

Syntax :LISTout:TEMPlet:SYMMetry {<Value>}

:LISTout:TEMPlet:SYMMetry?->{<Value>}

<Value>= the symmetry of the ramp template, the range is 0~100.

(It is valid when the template mode is RAMP template, otherwise, it is valid.)

Example :LISTout:TEMPlet:SYMMetry 50.0

:LISTout:TEMPlet:SYMMetry?->50.0

:LISTout:TEMPlet:EXPRate

Function Set the exponential value of the exponent template for list mode

Syntax :LISTout:TEMPlet:EXPRate {<Value>}

:LISTout:TEMPlet:EXPRate?->{<Value>}

<Value>= the exponential value of the exponent template, the range is 0~10.

(It is valid when the template mode is Exponential RISE/ Exponential Fall template, otherwise, it is valid.)

Example :LISTout:TEMPlet:EXPRate 5.0

:LISTout:TEMPlet:EXPRate?->5.0

:DELAY[:STATe]

Function Set the switch of delay and query the state of delay

Syntax :DELAY[:STATe]{<Boolean>}

:DELAY[:STATe]?->{<Boolean>,<time>,<curGroup>,<endGroup>,<remainCycle>,<endState>}

<Boolean>= ON|OFF|0|1

<time>= the time remaining of the current group, the unit is second (s).

<curGroup>= the current group

<endGroup>= the stop group

<remainCycle>= the remain cycle

<endState>= the stop state: OFF indicates the output is disabled; LAST indicates maintain the output.

Example :DELAY <Boolean>

:DELAY? -> ON, 0.4,010,016,99999,OFF

:DELAY:STARt

Function Set the start group for delay

Syntax :DELAY:STARt {<Start>}

:DELAY:STARt?->{<Start>}

<Start>= the start group of delay

Example :DELAY:STARt 0

:DELAY:STARt?->0

:DELAY:GROUPs

Function Set the operation group for delay

Syntax :DELAY:GROUPs {<Groups>}

:DELAY:GROUPs?->{<Groups>}

<Groups>= the operation group of delay

Example :DELAY:GROUPs 64

:DELAY:GROUPs?->64

:DELAY:CYCLEs

Function Set the cycles of delay

Syntax :DELAY:CYCLEs {<Cycles>}

:DELAY:CYCLEs?->{<Cycles>}

<Cycles>= the operation group of delay, 0 indicates infinite cycle

Example :DELAY:CYCLEs 0

:DELAY:CYCLEs?->0

:DELAY:ENDState

Function Set the stop state for delay

Syntax :DELAY:ENDState {<endState>}

:DELAY:ENDState?->{<endState>}

<endState>= the stop state: OFF indicates the output is disabled; LAST indicates maintain the output. ON indicates the output is enabled.

Example :DELAY:ENDState OFF

:DELAY:ENDState?->OFF

:DELAY:STOP

Function Set the stop condition for delay

Syntax :DELAY:STOP {NONE|<V|>V|<C|>C|<P|>P [,<Value>]}

:DELAY:STOP?->{NONE|<V|>V|<C|>C|<P|>P [,<Value>]}

<Value>= the condition parameter, it can be omitted. The numerical value cannot be change if it is omitted.

Example :DELAY:STOP >V,15.000

:DELAY:STOP?->V,15.000

:DELAY:PARAmeter

Function Set the group parameter for delay

Syntax :DELAY:PARAmeter {<No>,<Boolean>,<Time>}

:DELAY:PARAmeter?->{<No>,<Boolean>,<Time>}

<No>= group serial number

<Boolean>= output state, ON: turn on the output; OFF: turn off the output

<Time>= duration, the unit is s and the minimum resolution is 0.1s.

Example :DELAY:PARAmeter 0,ON,1.0

:DELAY:PARAmeter? 0,1

->#215000,OFF 1.0;

#215 indicates 15 occupies two bytes, the data segment has 15 data {000, OFF 1.0;}, and the serial number 0 indicates the output is enabled, the duration is 1.0s. The specific format refers to the section Data Return of the chapter 1.

:DELAY:GENerate:STAT

Function Output the state with a group parameter which the delay generated by the specified status code

Syntax :DELAY:GENerate:STAT {<Start>,<Groups>, 01P|10P}

:DELAY:GENerate?->STAT,{<Start>,<Groups>, 01P|10P}

<Start>= generate the start group

<Groups>= generate the group number

<01P|10P>= 0 1P: turn off at first and then turn on; 10P: turn on at first and then turn off

Example :DELAY:GENerate:STAT 0,64,01P

:DELAY:GENerate?->STAT,0,64,01P

:DELAY:GENerate:FIX

Function Generate the duration of the delay group by the fixed duration

Syntax :DELAY:GENerate:FIX {<Start>,<Groups>, <Time_on>,<Time_off>}

:DELAY:GENerate?->FIX,{<Start>,<Groups>, <Time_on>,<Time_off>}

<Start>= generate the start group

<Groups>= generate the group number

<Time_on>= open duration, the unit is s and the minimum resolution is 0.1s.

<Time_off>= close duration, the unit is s and the minimum resolution is 0.1s.

Example :DELAY:GENerate:FIX 0,10,5,10

:DELAY:GENerate?->FIX,0,10,5,10

:DELAY:GENerate:INC

Function Generate the duration of the delay group by the progressive increase duration

Syntax :DELAY:GENerate:INC {<Start>,<Groups>, <Time_base>,<Time_step>}

:DELAY:GENerate?->INC,{<Start>,<Groups>, <Time_base>,<Time_step>}

<Start>= generate the start group

<Groups>= generate the group number

<Time_base>= duration base, the unit is s and the minimum resolution is 0.1s.

<Time_step>= the progressive increase of duration, the unit is s and the minimum resolution is 0.1s.

Example :DELAY:GENerate:INC 0,10,10,2

:DELAY:GENerate?->INC,0,10,10,2

:DELAY:GENerate:DEC

Function Generate the duration of the delay group by the progressive decrease duration

Syntax :DELAY:GENerate:DEC {<Start>,<Groups>, <Time_base>,<Time_step>}

:DELAY:GENerate?->DEC,{<Start>,<Groups>, <Time_base>,<Time_step>}

<Start>= generate the start group

<Groups>= generate the group number

<Time_base>= duration base, the unit is s and the minimum resolution is 0.1s.

<Time_step>= the progressive decrease of duration, the unit is s and the minimum resolution is 0.1s.

Example :DELAY:GENerate:DEC 0,10,100,1

:DELAY:GENerate?->DEC,0,10,100,1

:PRESet#[:APPLy]

Function Apply the preset parameter to the output parameter

Syntax :PRESet#[:APPLy]

the range is 0~7

Example :PRESet0:APPLy

:PRESet#:SET:VOLTage

Function Set the voltage for the preset group

Syntax :PRESet#:SET:VOLTage {<Volt>}

:PRESet#:SET:VOLTage?->{<Volt>}

<Volt>= voltage, the unit is V.

the range is 0~7

Example :PRESet0:SET:VOLTage 5.000

:PRESet0:SET:VOLTage?->5.000

:PRESet#:SET:CURRent

Function Set the current for the preset group

Syntax :PRESet#:SET:CURRent {<Curr>}

:PRESet#:SET:CURRent?->{<Curr>}

<Curr>= current, the unit is A.

the range is 0~7

Example :PRESet7:SET:CURRent 5.000

:PRESet7:SET:CURRent?->5.000

:PRESet#:SET:OVP

Function Set the OVP for the preset group

Syntax :PRESet#:SET:OVP {<Boolean>,<Volt>}

:PRESet#:SET:OVP?->{<Boolean>,<Volt>}

<Boolean>=ON: turn on the OVP function; OFF: turn off the OVP function

<Volt>= voltage, the unit is V.

the range is 0~7

Example :PRESet1:SET:OVP ON,62.000

:PRESet1:SET:OVP?->ON,62.000

:PRESet#:SET:OCP

Function Set the OCP for the preset group

Syntax :PRESet#:SET:OCP {<Boolean>,<Curr>}

:PRESet#:SET:OCP?->{<Boolean>,<Curr>}

<Boolean>=ON: turn on the OCP function; OFF: turn off the OCP function

<Curr>= current, the unit is A.

the range is 0~7

Example :PRESet1:SET:OCP ON,15.500

:PRESet1:SET:OCP?->ON,15.500

:PRESet#:SET:TIMer

Function Set the switch of timer for the preset group

Syntax :PRESet#:SET:TIMer {<Boolean>}

:PRESet#:SET:TIMer?->{<Boolean>}

<Boolean>=ON: turn on the timer; OFF: turn off the timer

the range is 0~7

Example :PRESet0:SET:TIMer ON

:PRESet0:SET:TIMer?->ON

:PRESet#:SET:TIMer:DATA

Function Set the duration of timer for the preset group

Syntax :PRESet#:SET:TIMer:DATA {<Time>}

:PRESet#:SET:TIMer:DATA?->{<Time>}

<Time>= turn off the duration of timer, the unit is s.

the range is 0~7

Example :PRESet0:SET:TIMer:DATA 5.000

:PRESet0:SET:TIMer:DATA?->5.000

:PRESet#:SET:TIMer:DATA

Function Set the duration of timer for the preset group

Syntax :PRESet#:SET:TIMer:DATA {<Time>}

:PRESet#:SET:TIMer:DATA?->{<Time>}

<Time>= turn off the duration of timer, the unit is s.

the range is 0~7

Example :PRESet0:SET:TIMer:DATA 5.000

:PRESet0:SET:TIMer:DATA?->5.000

:MONItor[:STATe]

Function Enable or disable the monitor

Syntax :MONItor[:STATe]{<Boolean>}

:MONItor[:STATe]?->{<Boolean>}

<Boolean>=ON: turn on the monitor; OFF: turn on the monitor

Example :MONItor:STATe ON

:MONItor:STATe?->ON

:MONItor:VOLTage

Function Set the condition of output voltage for the monitor

Syntax :MONItor:VOLTage {<V|>V|NONE ,[<Volt>]}

:MONItor:VOLTage?->{<V|>V|NONE ,[<Volt>]}

Logical condition

<V: when the output voltage is less than the set parameter, it is true.

>V: when the output voltage is greater than the set parameter, it is true.

NONE: it is always true.

<Volt>= voltage, it can be omitted. The set parameter cannot be change when it is omitted. It will not return when the condition is NONE.

Example :MONItor:VOLTage <V, 15.000

:MONItor:VOLTage?-><V, 15.000

:MONItor:CURRent

Function Set the condition of output current for the monitor

Syntax :MONItor:CURRent {<C|>C|NONE ,[<Curr>]}

:MONItor:CURRent?->{<C|>C|NONE ,[<Curr>]}

Logical condition

<C: when the output current is less than the set parameter, it is true.

>C: when the output current is greater than the set parameter, it is true.

NONE: it is always true.

<Curr>= current, it can be omitted. The set parameter cannot be change when it is omitted. It will not return when the condition is NONE.

Example :MONItor:CURRent <C, 1.000

:MONItor:CURRent?-><C, 1.000

:MONItor:POWER

Function Set the condition of output power for the monitor

Syntax :MONItor:POWER {<P|>P|NONE ,[<Power>]}

:MONItor:POWER?->{<P|>P|NONE ,[<Power>]}

Logical condition

<V: when the output power is less than the set parameter, it is true.

>V: when the output power is greater than the set parameter, it is true.

NONE: it is always true.

<Power>= power, it can be omitted. The set parameter cannot be change when it is omitted. It will not return when the condition is NONE.

Example :MONItor:POWER <P, 15.000

:MONItor:POWER?-><P, 15.000

:MONItor:DVM

Function Set the condition of DVM voltage for the monitor

Syntax :MONItor:DVM {<DVM|>DVM|NONE ,[<DVM>]}

:MONItor:DVM?->{<DVM|>DVM|NONE ,[<DVM>]}

Logical condition

<DVM: when the DVM voltage is less than the set parameter, it is true.

>DVM: when the DVM voltage is greater than the set parameter, it is true.

NONE: it is always true.

<DVM>=DVM voltage, it can be omitted. The set parameter cannot be change when it is omitted. It will not return when the condition is NONE.

Example :MONItor:DVM >DVM, 10.000

:MONItor:DVM?-> >DVM, 10.000

:MONItor:LOGic

Function Enable or disable the monitor

Syntax :MONItor:LOGic {<No>,<Logic>}

:MONItor:LOGic? {<No>}->{<Logic>}

<No>= the number of logical symbol, and the range is 1~3

<Logic>= logic, AND: And operation "&"; OR: OR operation "|".

Example :MONItor:LOGic 1,AND

:MONItor:LOGic? 1->AND

:MONItor:STOPway

Function Set the trigger mode for the monitor

Syntax :MONItor:STOPway {<Type>,<Boolean>}

:MONItor:STOPway? {<Type>}->{<Boolean>}

<Type>=OUTOFF: the output is disabled; MSG: message prompt; BEEPER: beeper alarm

<Boolean>=ON: turn on OFF: turn off

Example :MONItor:STOPway OUTOFF, ON

:MONItor:STOPway? OUTOFF->ON

:MONItor:STOPway MSG, OFF

:MONItor:STOPway? MSG->OFF

:SYSTem:REMote

Function Set the remote control for the power. The key is locked in this mode, unlock the key by manual and it return to the panel control mode.

Syntax :SYSTem:REMote

:SYSTem:LOCal

Function Set the panel control for the power.

Syntax :SYSTem:LOCal

:SYSTem:BEEPer:TEST

Function Test the beeper function. Sending this command will cause the beeper to sound once, it only valid when the beeper is turned on.

Syntax :SYSTem:BEEPer:TEST

:SYSTem:BEEPer[:STATe]

Function Turn on or off the beeper

Syntax :SYSTem:BEEPer[:STATe]{<Boolean>}

:SYSTem:BEEPer[:STATe]?->{<Boolean>}

<Boolean>=ON: turn on the beeper; OFF: turn off the beeper

Example :SYSTem:BEEPer:STATe ON

:SYSTem:BEEPer:STATe?->ON

:SYSTem:BRIGhtness

Function Set the backlight brightness for the screen

Syntax :SYSTem:BRIGhtness {<Value>}

:SYSTem:BRIGhtness?->{<Value>}

<Value>= the brightness of screen, the range is 20~100.

Example :SYSTem:BRIGhtness 50

:SYSTem:BRIGhtness?->50

:SYSTem:COMMunicate:COM:BAUD

Function Set the communication baud rate for RS232 and RS485

Syntax :SYSTem:COMMunicate:COM:BAUD {<BaudRate>}

:SYSTem:COMMunicate:COM:BAUD?->{<BaudRate>}

<BaudRate>= the communication baud rate,

the range can set to 9600,14400,19200,38400,57600,115200.

Example :SYSTem:COMMunicate:COM:BAUD 115200

:SYSTem:COMMunicate:COM:BAUD?->115200

:SYSTem:COMMunicate:COM:PROTocol

Function Set the communication protocol for RS232 and RS485

Syntax :SYSTem:COMMunicate:COM:PROTocol {<Protocol>}

:SYSTem:COMMunicate:COM:PROTocol?->{<Protocol>}

<Protocol>= the communication protocol, 0: SCPI 1: Multi-SCPI 2: Modbus

Example :SYSTem:COMMunicate:COM:PROTocol 0

:SYSTem:COMMunicate:COM:PROTocol?->0

:SYSTem:COMMunicate:COM:ADDRess

Function Set the multimachine communication address for RS232 and RS485

Syntax :SYSTem:COMMunicate:COM:ADDRess {<Address>}

:SYSTem:COMMunicate:COM:ADDRess?->{<Address>}

<Address>= the communication address, the range is 1~32, 0 is the broadcast address.

Example :SYSTem:COMMunicate:COM:ADDRess 1

:SYSTem:COMMunicate:COM:ADDRess?->1

:SYSTem:COMMunicate:LAN:APPLy

Function Save and apply the set network parameter

Syntax :SYSTem:COMMunicate:LAN:APPLy

Note: The network parameter does not take effect immediately, it can only be valid after the command is executed and be saved.

:SYSTem:COMMunicate:LAN:DHCP[:STATe]

Function Turn on or off the DHCP function

Syntax :SYSTem:COMMunicate:LAN:DHCP[:STATe] {<Boolean>}

:SYSTem:COMMunicate:LAN:DHCP[:STATe]?->{<Boolean>}

<Boolean>=ON: turn on the DHCP; OFF: turn off the DHCP

Example :SYSTem:COMMunicate:LAN:DHCP ON

:SYSTem:COMMunicate:LAN:APPLy
:SYSTem:COMMunicate:LAN:DHCP?->ON

:SYSTem:COMMunicate:LAN:IPADdress

Function Set the IP address for the LAN port

Syntax :SYSTem:COMMunicate:LAN:IPADdress {<Address>}
:SYSTem:COMMunicate:LAN:IPADdress?->{<Address>}
<Address>= the IP address with the format of "x.x.x.x"

Example :SYSTem:COMMunicate:LAN:IPADdress "192.168.1.100"
:SYSTem:COMMunicate:LAN:APPLy
:SYSTem:COMMunicate:LAN:IPADdress?->192.168.1.100

:SYSTem:COMMunicate:LAN:SMASK

Function Set the subnet mask for the LAN port

Syntax :SYSTem:COMMunicate:LAN:SMASK {<Address>}
:SYSTem:COMMunicate:LAN:SMASK?->{<Address>}
<Address>= the IP address with the format of "x.x.x.x"

Example :SYSTem:COMMunicate:LAN:SMASK "255.255.255.0"
:SYSTem:COMMunicate:LAN:APPLy
:SYSTem:COMMunicate:LAN:SMASK?->255.255.255.0

:SYSTem:COMMunicate:LAN:GATEway

Function Set the gateway address for the LAN port

Syntax :SYSTem:COMMunicate:LAN:GATEway {<Address>}
:SYSTem:COMMunicate:LAN:GATEway?->{<Address>}
<Address>= the IP address with the format of "x.x.x.x"

Example :SYSTem:COMMunicate:LAN:GATEway "192.168.1.1"
:SYSTem:COMMunicate:LAN:APPLy
:SYSTem:COMMunicate:LAN:GATEway?->192.168.1.1

:SYSTem:ERRor[:NEXT]?

Function Acquire the error code and the character string of SCPI

Syntax :SYSTem:ERRor[:NEXT]?->{<errorNum>,<errorString>}
<errorNum>= error code
<errorString>= character string

Example :SYSTem:ERRor[:NEXT]?
->0,"No error"

:SYSTem:ERRor:COUNt?

Function Acquire the error code and the character string of SCPI

Syntax :SYSTem:ERRor:COUNt?->{<errorCount>}

<errorCount>= the count of error queue

Example :SYSTem:ERRor:COUNt?

->0

:SYSTem:VERSion?

Function Acquire the version of SCPI

Syntax :SYSTem:VERSion?->{<Version>}

<Version>= the version of SCPI

Example :SYSTem:VERSion?

1999.0

*IDN?

Function Query the instrument's information

Syntax *IDN?

Example *IDN?

-> Uni-Trend, UDP6942B,0000000000000,1.00.0905

Explanation The returned format of the instrument is <Manufacturer>,<Model>,<Serial Number>,<Software Version>.

STB (Status Byte Register)

The status byte register records the trigger state of the other registers. When the register & enable register is not 0, the corresponding bit of STB will be set, the register will not be latched, it will change with the event.

Bit	Decimalism		Definition	Description
0		R01	Not Used	
1	2	PRO	Protection Event Flag	
2	4	QMA	Error/Event queue message available	Error event message queue is available
3	8	QES	Questionable status	Generate the questionable status register
4	16	MAV	Message Available	
5	32	ESR	Standard Event Status Register	Generate the standard event status register
6	64	SRQ	Service Request	
7		OPS	Not Used	

*STB?

Function Query the status byte event register, it automatically clear the flag bit after each read, return in decimalism

Syntax *STB?

Example *STB? -> 4

Explanation If the return value is 4, the status byte register is set to Bit 2; this means that the error queue is not empty, i.e., an error has been generated.

See the description of the Status Byte Register STB for details.

*SRE

Function Set the status byte even enable register of SCPI

If the SRE register & STB register are not 0, then SRQ bit of the STB register is 1

Syntax *SRE <Value>

*SRE?-><Value>

<Value>= the value of status byte enable register

Example *SRE 128

*SRE?->128

ESR (Event Register)

The event register records SCPI and the operation error events of power supply. When an event occurs, the register latches the event and clears the register only when Query register or sending a *CLS clear command.

Bit	Decimalism		Definition	Description
0		OPC	Operation complete	The default is 0
1		Not Used		The default is 0
2	4	QER	Query error	Query the error
3	8	DER	Device dependent error	Device error, such as self-inspection error
4	16	EER	Execution error (e.g. range error)	Execution error
5	32	CER	Command error (e.g. syntax error)	Command error, such as syntax error
6		Not Used		The default is 0
7	128	PON	Power On	Repower the power supply

*ESR?

Function Query the event register of SCPI, it automatically clear the flag bit after each read, return in decimalism

Syntax *ESR?

Example *ESR? -> 128

Explanation If the returned value is 128, it indicates the SCPI event register is set to Bit 7; this means the power on event has occurred

See the description of ESR (Event Register) for details.

*ESE

Function Set the SCPI event enable register

If ESE register &ESR are not 0, then the ESR bit of STB register is 1

Syntax *SRE <Value>

*SRE?-><Value>

<Value>= the value of status byte enable register

Example *ESE 128

*ESE?->128

QUES(Query Status Register)

The QUES status register provides information about the operating status of the power supply, such as the operation mode of constant current and constant voltage, OTP(over-temperature protection), OCP(over-current), and other status changes.

Bit	Decimalism		Definition	Description
0	1	CV	CV Mode	The operation mode is CV (constant voltage)
1	2	CC	CC Mode	The operation mode is CC (constant current)
2		Not Used	Not Used	
3		Not Used	Not Used	
4	16	OTP	Over Temperature	Over temperature protection (OTP)
5		Not Used	Not Used	
6		Not Used	Not Used	
7		Not Used	Not Used	
8	256	OSP	Over Sense	Line loss compensation is too large(OSP)
9	512	OVP	Over Voltage	OVP(over voltage protection)
10	1024	OCP	Over Current	OCP(over current protection)

:STATus:QUEStionable[:EVENt]?

Function Query the QUSE status event register of SCPI, it automatically clear the flag bit after each read, return in decimalism

Syntax :STATus:QUEStionable[:EVENt]?

Example :STATus:QUES? -> 512

Explanation If the returned value is 512, then the QUSE status event register of SCPI is set to Bit 9; this means the event of OVP has occurred

See the description of QUSE(Query Status Register) for details.

:STATus:QUEStionable:CONDition?

Function Query the QUSE status event register of SCPI, it return the current state

Syntax :STATus:QUEStionable:CONDition?

Example :STATus:QUES:COND? -> 1

Explanation If the returned value is 1, then the QUSE status event register of SCPI is set to Bit 0; this means the power mode is CV

See the description of QUSE(Query Status Register) for details.

:STATus:QUEStionable:ENABle

Function Set the QUSE status event enable register of SCPI, if the QUES (status event register) & QUES (status event enable register) are not 0, then set the QUES bit of STB to 1

Syntax :STATus:QUEStionable:ENABle <Value>

:STATus:QUEStionable:ENABle?-><Value>

<Value>= the value of QUES (time enable register)

Example :STATus:QUEStionable:ENABle 512

:STATus:QUEStionable:ENABle?->512