Programmable DC Switching Power Supply

# Programming Manual

(*FTG Series*)

Version V2.02

2020-04

## Preface

This manual contains reference information for programming the FTG series programmable switching DC power supply Unit (PSU, i.e. Power Supply Unit) over the remote interface using the SCPI programming language.

## Related Information

The applications in this manual assume that you know how to connect the power supply to the computer. Please refer to the user manual for the specific online method.
Part of the content in the manual is related to specific accessories. If you need some special accessories or the accompanied accessories are not enough to meet your needs, please contact Faithtech Technology sales or after-sales service department.

## Announcement

For product latest information, please log on to Faithtech's official website http://www.faithtech.cn for inquiries.

## Version History

| Date | Version | Amendment |
|---|---|---|
| 2016-03 | 1.00 | Complete this manual |
| 2019-04 | 1.01 | Correction of certain examples |
| 2020-04 | 2.02 | Revise and append some prgramming examples |

# Table of Contents

**1. Communication Interface........ 1**

1.1. Introduction........ 1

1.2. Configure communication interface........ 1

Connect RS232........ 2

Connect LAN........ 2

Connect GPIB........ 3

1.3. Switching protocol........ 3

1.4. Enter remote control mode........ 3

1.5. Others........ 4

**2. SCPI Status Registers........ 5**

2.1. Command introduction........ 5

2.2. Program syntax........ 5

Command keyword........ 6

Root Specifier........ 6

Command separators........ 6

Query syntax........ 7

Command terminator........ 7

2.3. Paramter Format........ 8

2.4. Status system........ 8

Channel Status Register Group........ 9

Standard Event Status Register Group........ 1 0

Status Byte Register........ 1 0

2.5. Command version information........ 1 1

**3. SCPI Command Description........ 1 2**

3.1. IEEE488.2 common command reference........ 1 2

*CLS........ 1 2

*ESE........ 1 3

*ESR?........ 1 3

*IDN?........ 1 3

*OPC........ 1 3

*PSC........ 1 4

*RST........ 1 4

*SRE........ 1 4

*STB?........ 1 5

*SAV........ 1 5

*RCL........ 1 5

*TST?........ 1 6

*WAI........ 1 6

3.2. CONFigure command........ 1 7

CONFigure:FOLD:BACK........ 1 7

CONFigure:FOLD:TIME........ 1 7

CONFigure:APG:MODE........ 1 7

CONFigure:APG:VOLTage........................................ 17
CONFigure:INHibit........................................ 18
CONFigure:AUTO:LOAD........................................ 18
CONFigure:AUTO:OUTPut........................................ 18
3.3. OUTPut command........................................ 19
OUTPut[:STATe]........................................ 19
OUTPut:FUNCtion........................................ 19
OUTPut:PROTect:VOLTage........................................ 19
OUTPut:PROTect:CURRent........................................ 19
OUTPut:PROTect:POWer........................................ 20
OUTPut:PROTect:CLEar........................................ 20
3.4. SOURce command........................................ 20
SOURce:VOLTage[:LEVel]........................................ 20
SOURce:VOLTage:LIMit:HIGH........................................ 20
SOURce:VOLTage:LIMit:LOW........................................ 21
SOURce:CURRent[:LEVel]........................................ 21
SOURce:CURRent:LIMit:HIGH........................................ 21
SOURce:CURRent:LIMit:LOW........................................ 22
3.5. SEQuence command........................................ 22
SEQuence:STATus........................................ 22
SEQuence:RUN:NUMBer........................................ 22
SEQuence:EDIT:NUMBer........................................ 22
SEQuence:EDIT:COUNt........................................ 23
SEQuence:EDIT:CYCLe........................................ 23
SEQuence:EDIT:LINK........................................ 23
SEQuence:EDIT:SAVE........................................ 23
SEQuence:EDIT:STEP........................................ 24
SEQuence:EDIT:VOLTage........................................ 24
SEQuence:EDIT:CURRent........................................ 24
SEQuence:EDIT:DELay........................................ 24
3.6. CP command........................................ 25
CP:VOLTage........................................ 25
CP:CURRent........................................ 25
CP:POWer........................................ 25
CP:RESPonse........................................ 25
3.7. MEASure command........................................ 26
MEASure:VOLTage?........................................ 26
MEASure:CURRent?........................................ 26
MEASure:POWer?........................................ 26
MEASure:TEMPerature?........................................ 26
3.8. STATus command........................................ 27
STATus:CHANnel:CONDition?........................................ 27
STATus:CHANnel[:EVENt]?........................................ 27
STATus:CHANnel:ENABle........................................ 27

3.9. SYSTem command........ 27
SYSTem:ERRor?........ 27
SYSTem:LOCal........ 28
SYSTem:VERSion?........ 28
**4. SCPI error information........ 29**
4.1. Introduction........ 29
4.2. Check error information........ 29
4.3. Command errors........ 29
4.4. Execution errors........ 31
4.5. Query errors........ 31
**5. Programming example........ 32**
5.1. VI static output........ 32
5.2. Query measurement parameter........ 32
5.3. Edit SEQ file........ 32
5.4. Run SEQ File........ 33
5.5. Constant power (CP) output........ 33

# 1. Communication Interface

## 1.1. Introduction

Faithtech FTG series programmable DC switching power supply provides various remote communication interfaces such as RS232 port (standard), LAN (standard), GPIB (optional), etc.. You can connect to the power supply through a dedicated cable with the computer, the computer can control the source.

Table 1-1 Communication Interfaces

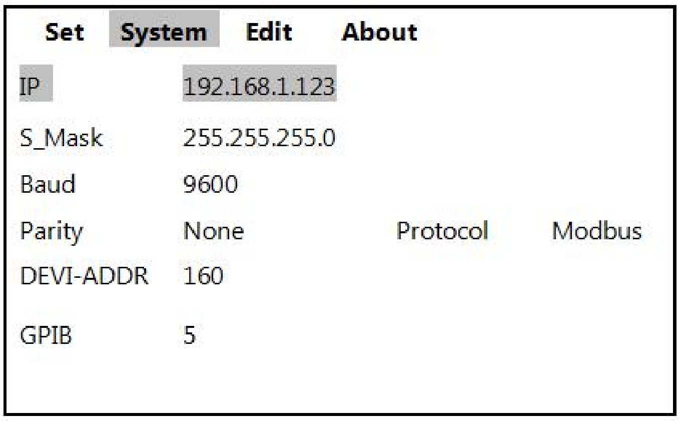
| Remote Controller | Interface | Explanation |
|---|---|---|
| PC | RS232 | Universal serial port |
| | LAN | Standard ethernet |

### ◎ Caution:

You can only select one communication method at a time. Default is RS232.

## 1.2. Configure communication interface

This section describes in detail the configuration method of each communication interface. These configurations can only be set via the front panel keyboard of the power supply. For more detailed configuration introduction, please refer to the user manual. Press the "Menu" key to enter the menu, under the "System" column, select the "Communication" item, and press the "Enter" key to enter the communication configure interface.



IP 192.168.1.123
S_Mask 255.255.255.0
Baud 9600
Parity None
Protocol Modbus
DEVI-ADDR 160
GPIB 5

Figure 1-1 Communication Configure Interface

Use the knob or direction keys to move the cursor to the setting item, and press the "Enter" key to enter the parameter editing mode. Enter the number keys and decimal point keys to edit the IP address, and turn the knob to select the baud rate and verification mode. The user presses the "Enter" key to confirm the editing parameters, presses the "Esc" key to exit. The communication configuration information is stored in the instrument internal non-volatile memory, and the configuration will not be affected by shutting down or recalling the preset settings. After changing the communication parameters, the device needs to be restarted for the changes to take effect.

## Connect RS232

The serial port is a universal asynchronous serial communication interface that conforms to the RS232 level specification and does not support any flow control. RS232 uses 9600 baud rate by default, which can be set to 4800, 19200, 38400 or 115200bps. The baud rate of the power supply and the computer must be the same. The factory default parity is off (no check). If the parity is enabled, the RS232 interface will use odd or even parity to verify data. For the RS232 interface, only TxD and RxD signals can transmit data, and the pin signals are described in the following table.

Table 1-2 RS232 PIN signal

| Pin NO. | Input/Output | Description |
|---|---|---|
| 1 | --- | N.C. |
| 2 | Input | RxD |
| 3 | Output | TxD |
| 4 | --- | DSR |
| 5 | --- | GND |
| 6 | --- | DTR |
| 7 | --- | CTS |
| 8 | --- | RTS |
| 9 | --- | N.C. |

## Connect LAN

The FTG series power supply has an Ethernet communication interface, adopts UDP communication mode, and the default port number is 7000. Before starting communication, the user needs to set the IP address and subnet mask, and ensure that the address of the PC and the power supply are in the same network segment, and that there is no duplicate IP address with the power supply in the network segment, otherwise the connection will not be correct.

### Connect GPIB

GPIB interface and IEEE488.2 GPIB connection cable are optional accessories, please contact authorized sales agent or Faithtech in case of need.
Use a IEEE488.2 GPIB cable to connect GPIB interfaces of power supply and PC. Please ensure that the screws have been screwed down in order to have a full connection. Then press “Menu” button to enter the system menu to set the address. GPIB address is saved in nonvolatile memory.

## 1.3. Switching protocol

FTG series support SCPI and Modbus-RTU protocol, select the correct protocol base on your requirements.

---

**◎ Caution:**

Restart the power supply after you changed the protocol, so as to let the changes take effect.

---

## 1.4. Enter remote control mode

After the power supply receives any correct SCPI command, it enters the remote control mode.
In the remote control mode: the local keyboard is locked, the key operation is invalid, and the power can only be controlled by programming commands; the front panel screen displays real-time status information such as voltage, current, and power, etc.. There are two ways to exit the remote control mode:

- Press "Enter" key, the system returns to local operation mode.
- Send the programming command "SYSTem:LOCal" to make the power supply return to local mode.

## 1.5. Others

For instructions on related software operations, driver installation and communication operations, please refer to the user manual and the accompanying software instructions. For the latest information about the software and drivers, please log in to Faithtech's website http://www.faithtech.cn for inquiries.

# 2. SCPI Status Registers

## 2.1. Command introduction

SCPI commands can be divided to common and subsystem commands.
Common commands are defined by the IEEE 488.2 standard to perform common interface functions. They begin with an * and consist of three letters (command) or three letters and a ? (query).
Subsystem commands are specific to instrument functions. They can be a single command or a group of commands. The groups are comprised of commands that extend one or more levels below the root.

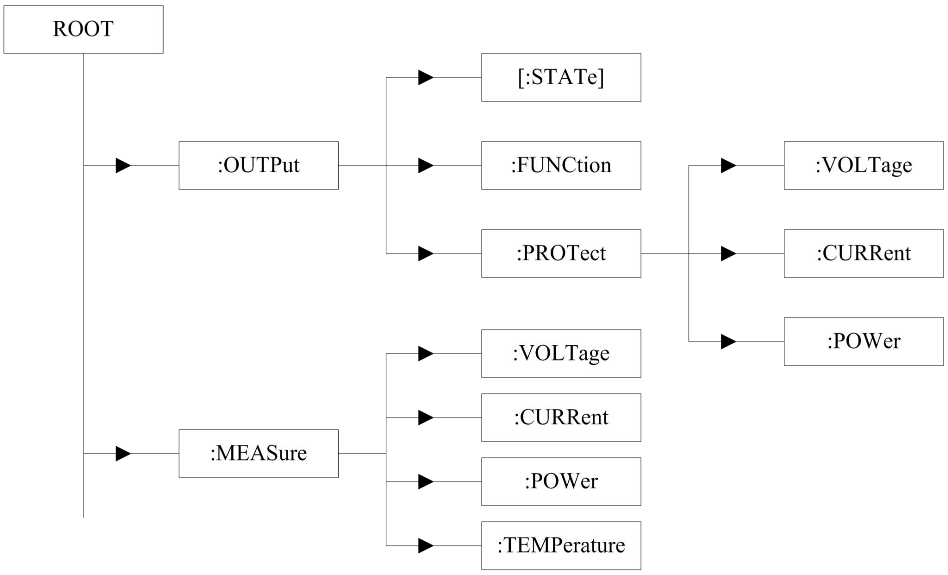


Figure 2-1 Command Levels

## 2.2. Program syntax

FTG series programmable DC power supply SCPI commands are inherited and expanded from IEEE488.2 standards. SCPI commands are constructed with keyword, seperator, paramter and terminator. Below is an example,

CURRent:STATic:L1 10.0

In this command, CURRent, STATic, L1 are command keywords, ":" and space character are seperator, 10.0 is parameter (some commands have multiple parameters seperated with comma ","), A <carriage return> at the end of this command is the command terminator.
Throughout this document, the following conventions are used for the SCPI command syntax:

- Square brackets ([]) indicate optional keywords or parameters. The braces are not sent with the command string.
- Braces ({}) enclose parameters within a command string.
- Triangle brackets (<>) indicate that you must substitute a value or a code for the enclosed parameter.
- A vertical bar (|) separates one of two or more alternative parameters.

## Command keyword

Each command keyword has two formats: long mnemonic and short mnemonic. Short mnemonic is an abbreviation for long mnemonic. Each mnemonic does not exceed 12 characters (including any number suffixes that may appear). The power supply only accepts precise long or short mnemonics. The rules for generating mnemonics are as follows:
The long mnemonic consists of a word or phrase. If it is a word, the entire word constitutes a mnemonic; if it is a phrase, the first character of each word and the entire last word constitute a mnemonic.

| | |
|---|---|
| CONFIGURE —— | CONFigure |
| Main Value —— | MVALue |

The short mnemonic is generally composed of the first 4 characters of the long mnemonic.

| | |
|---|---|
| CONFigure —— | CONF |

If the character length of the long mnemonic is less than or equal to 4, the long and short mnemonics are the same; if the length of the long mnemonic is greater than 4, and the fourth character is a vowel, the short mnemonic will discard this vowel and becomes 3 characters.

| | |
|---|---|
| SAVE —— | SAVE |
| TIMer —— | TIM |

The mnemonic is not case sensitive.

## Root Specifier

When it precedes the first header of a message unit, the colon becomes the root specifier. It tells the command parser that this is the root or the top node of the command tree.

## Command separators

### Colon “:”

A colon (:) is used to separate a command keyword from a lower-level keyword, such as command “CURR:MVAL 10”; also, when it precedes the first header of a message unit, the colon becomes the root specifier, it tells the command parser that this is the root or the top node of the command tree.

**Space**

Used for seperating command and parameter.

**Semicolon “;”**

A semicolon (;) is used to separate two commands within the same subsystem, and can also minimize typing. For example, sending the following command string,

“CURR:RANG 0;MVAL 10”

is the same as sending the following two commands:

“CURR:RANG 0”

“CURR:MVAL 10”

**Comma “,”**

A comma “,” is used to seperate parameters, as below command,

“CAL:STAT ON,6900”

## Query syntax

You can query the value of most parameters by adding a question mark (?) to the command.

For example, the following command sets the output voltage to 80 V:

VOLTage 80

You can query the value by executing:

VOLTage?

After the power supply receives the query command and completes the analysis, it executes the command and generates a response message. The response message is first written into the output buffer. If the current remote interface is a GPIB interface, it will wait for the controller to read the response; otherwise, the response message will be sent to the interface immediately.

Most of the setting commands have corresponding query syntax. If a command that cannot be queried is received, the power supply will report the error message "-115 Command can not query" and nothing is returned.

## Command terminator

There are two types of command terminators: new line character (ASCII symbol LF，ASCII value 10) and EOI (available only in GPIB interface). Command string termination will always reset the current SCPI command path to the root level.

## 2.3. Paramter Format

Table 2-1 Parameter types

| Symbol | Explanation | Data Example |
|---|---|---|
| <NR1> | Integer value | 123 |
| <NR2> | Float value | 123.，12.3，0.12，1.23E4 |
| <NRf> | Could be NR1 or NR2。 | |
| <NRf+> | Extended type, including <NRf>, MIN, MAX | |
| <Bool> | Boolean value | 1\|0\|ON\|OFF |
| <CRD> | String, such as CURR。 | |
| <AARD> | Return ASCII data. Allowed for undefined 7-Bit ASCII. It includes a command terminator. | |

## 2.4. Status system

The status system records various conditions and states of the power supply in each status register group. The structure of this status system is shown in the figure below. The status system includes the standard event status register group, the channel status register group, and the status byte register. Each register group consists of multiple registers, including condition register, event register, and enable register.

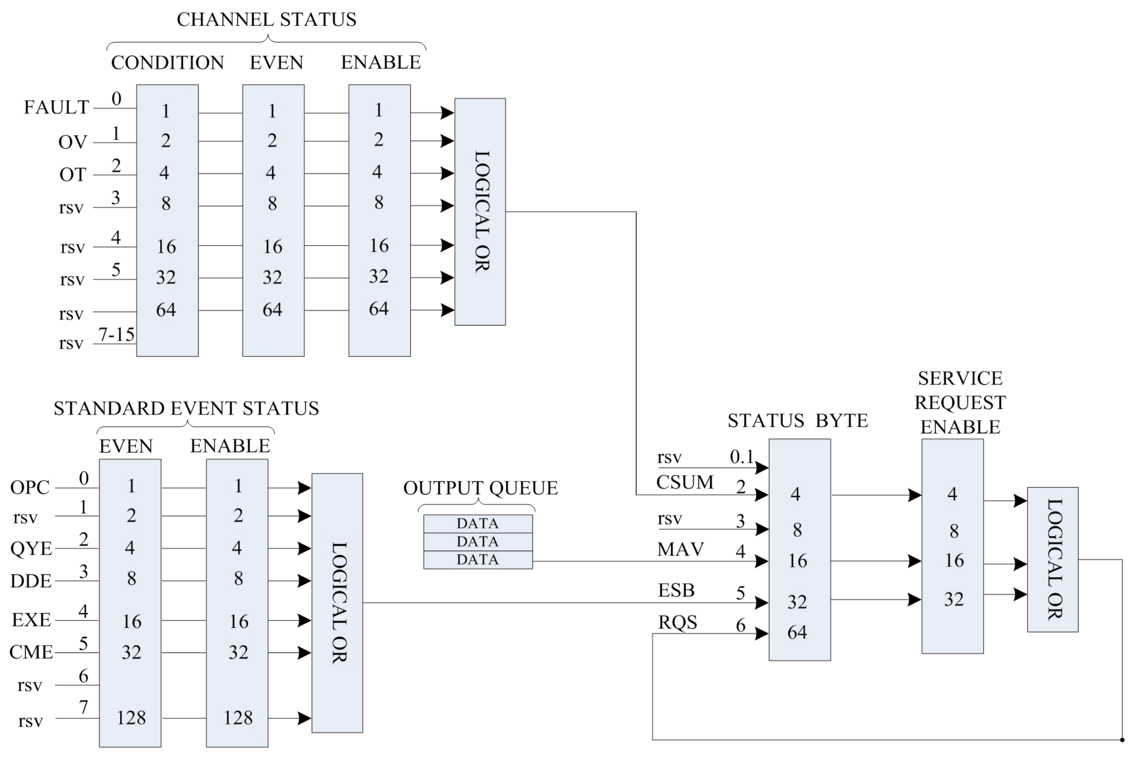


Figure 2-2 Status system of FTG power supply

## Channel Status Register Group

The Channel Status Register group reflects the real-time status and events of the power supply, including the Channel Condition register, PTR filter register, NTR filter register, Channel Event register, and Channel Event Enable register.

The Channel Condition register records the real-time status of the power supply. The main content is the alarm information of the power supply, including over-current status, over-voltage status, etc. The detailed definition is shown in the following table.

Table 2-2 Channel Condition Register Bit Explanation

| Bit | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
|---|---|---|---|---|---|---|---|---|
| Explain | FAULT | OV | OT | rsv | rsv | rsv | rsv | rsv |
| FAULT：Power supply module fault<br>OV：Overvoltage protection<br>OT：Over-temperature protection<br>rsv：Reserved | | | | | | | | |

The channel event register records the status change event of the power supply, and the meaning of each binary bit corresponds to the bit of the channel condition register. The channel event register can be cleared by the related query command or "*CLS" command. After clearing, it will restart to record new events.

## Standard Event Status Register Group

Standard Event Status Register Group records important events that occur during power supply analyzing programming commands or executing operations, including Standard Event register and Standard Event Enable register.
The definition of each bit of the standard event register is compatible with the IEEE 488.2 standard, and the detailed definition is as follows:

Table 2-3 Standard Event Register Bit Explanation

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | rsv | rsv | CME | EXE | DDE | QYE | rsv | OPC |
| OPC All operations & commands completed<br>QYE Query Error<br>DDE Device specific Error<br>EXE Excecution Error<br>CME Command Error | | | | | | | | |

Bits in the Standard Event register are automatically cleared by a query of that register (such as *ESR?) or by sending the *CLS (clear status) command. Querying an event register returns a decimal value which corresponds to the binary-weighted sum of all bits set in the register. The Standard Event ENABle register is used to define which bits of the Standard Event register will latch ESB (bit 5) of the Status Byte register.

## Status Byte Register

The Status Byte Register records important states that IEEE 488.2 bus-compatible devices need to support. Its status bits record whether there are currently unserviced events, errors, standard events, etc. in the power supply.
The bits definition of the Status Byte Register are fully compatible with IEEE 488.2 specifications, details are as follows:

Table 2-4 Status Byte Register Bits

| Bit | | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | | RQS | ESB | MAV | rsv | CSUM | rsv | rsv |
| CSUM The summary bit for the Channel Status Register group<br>MAV (Message Available) This is set when there is data in the Output Queue<br>ESB The summary bit for the Standard Event Status Register group<br>RQS Exsit Request for service<br>rsv Reserved | | | | | | | | |

## 2.5. Command version information

The version information of programming commands can only be queried remotely. Implement the following command via remote interface to check PSU's programming command version:

SYSTem:VERSion?

The return message format "YYYY.V", and "YYYY" stands for year, "V" stands for version code.

# 3. SCPI Command Description

## 3.1. IEEE488.2 common command reference

This section summarizes the mandatory subset of IEEE 488.2 commands required for any SCPI compliant instrument.

| Common command | Description |
|---|---|
| *CLS | Clears all event registers, besides status byte register and error queue |
| *ESE {<value>} | Programs bits in the Standard Event Enable register |
| *ESR? | Query the Standard Event Register |
| *IDN? | Returns the UNIQUE identification of the PSU |
| *OPC | Operation Complete Command used for program synchronization |
| *RCL {<profile>} | Recalls the PSU state stored in the specified storage location |
| *RST | Reset PSU to the initial state |
| *SAV {<profile>} | Stores the current PSU state in the specified storage location |
| *SRE | Programs bits in the Service Request Enable register |
| *STB? | Query the Status Byte register |
| *PSC | Define power on state for service request enable register and standard event enable register |
| *TST? | Returns Self-Test results |
| *WAI | Waits until all pending commands are completed |

### *CLS

Clear command. This command clears all event registers in the PSU:

- ✧ Standard Event Register
- ✧ Channel Event Register
- ✧ Status Byte Register
- ✧ Error Queue

Command syntax: *CLS

Parameter: None

Query syntax: None

## *ESE

This command sets the Standard Event Enable register bits in the PSU. A 1 in the bit position enables the corresponding event. All of the enabled events of the Standard Event Enable Register are logically ORed sets the Event Summary Bit (ESB) of the Status Byte Register.

Command syntax: *ESE <NR1>

Parameter: 0～255

The value when power on: refer to *PSC command.

Example: *ESE 128

Query syntax: *ESE?

Returned parameter: <NR1> (returns a decimal value which corresponds to the binary-weighted sum of all bits in the Standard Event Enable register)

Related commands: *ESR?、*PSC、*STB?

## *ESR?

Standard Event Register Query. Reading the Standard Event register clears it.

Query syntax: *ESR?

Parameter: None

Returned parameter: <NR1> (returns a decimal value which corresponds to the binary-weighted sum of all bits in the Standard Event register)

Related commands: *CLS、*ESE、*OPC

## *IDN?

Queries the manufacturer, model name, reserved code, and firmware version of the PSU.

Query syntax: *IDN?

Parameter: None

Return parameter example: Faith,FTG050-100-50,0,V1.00

## *OPC

It causes the PSU to set the OPC bit (bit 0) of the Standard Event register. OPC bit value is "1", that is all commands before *OPC operation have been completed.

Command syntax: *OPC

Parameter: None

Query syntax: *OPC?

Returned parameter: <NR1>

Related commands: *TRG  *WAI

## *PSC

It determines to save or not for the Service Request Enable Register and Standard Event Enable Register at the moment when PSU is powered on.
1: Not save. When powered on, the PSU will clear the Service Request Enable Register and Standard Event Enable Register.
0: Save. When powered on, the data will not be cleared.

Command syntax: *PSC <bool>

Parameter: 0 | 1

Usage example: *PSC 1

Query syntax: *PSC?

Return: OFF | ON（return *PSC current status）

Related commands: *ESE, *SRE

## *RST

Reset Command. Restores the PSU to its initial factory default state.

Command syntax: *RST

Parameter: None

Returned parameter: 无

Related commands: 无

## *SRE

Sets or queries the Service Request Enable register. The Service Request Enable register determines which bits of the Status Byte register are able to generate service requests.
The RQS bit of the Status Byte Register will be set "1" when the same bit are "1" for both Status Byte Register and Service Request Enable Register.

Command syntax: *SRE  <NR1>

Parameter: 0～255

Query syntax: *SRE?

Returned parameter: <NR1>

Related commands: *ESE *ESR *PSC

## *STB?

Query Status Byte register. The Status Byte Register bits are cleared when it is read.

Command syntax: *STB?

Parameter: None

Returned parameter: <NR1>

Related commands: *CLS *ESE *ESR

## *SAV

This command stores the current instrument state in the specified storage location. Any state previously stored in the same location is overwritten without generating any errors. The PSU has 20 storage locations in non-volatile memory which are available to the user for storing PSU states. (location 1 ~ 20)

Command syntax: *SAV <NR1>

Parameter: 1～20

Usage example: *SAV 3

Query syntax: None

Related commands: *RCL

### ◎ Caution:

*SAV command takes an excution time of 500ms, do not operate the instrument while implementing this command.

## *RCL

This command recalls the PSU state stored in the specified storage location. The PSU has 20 storage locations in non-volatile memory to store PSU states. It is not possible to recall the PSU state from a storage location that is empty or was deleted.

Command syntax: *RCL <NR1>

Parameter: 1～20

Usage example: *RCL 3

Query syntax: None

Related commands: *SAV

---

## ◎ Caution:

*RCL command takes an excution time of 500ms, do not operate the instrument while implementing this command.

---

## *TST?

Self-Test Query. The self-test query causes an internal self-test, and returns whether or not the PSU completed the self-test without detected errors.

If all tests passed:

```
*TST?
0
```

If one or more tests failed:

```
*TST?
1
```

Use “SYSTem:ERRor?” to obtain error code.

Command syntax: *TST?

Parameter: None

Return: <NR1>

Related commands: None

## *WAI

The Wait-to-Continue Command causes the PSU to wait until all pending commands are completed before executing any other commands.

Command syntax: *WAI

Parameter: None

Relating commands: *OPC

## 3.2. CONFigure command

### CONFigure:FOLD:BACK

Set foldback protection mode, when power supply switches output mode between CC and CV, the output will be turned off.

Command syntax: CONFigure:FOLD:BACK <NR1>

Parameter: 0 | 1 | 2 | OFF | CV2CC | CC2CV

Example: CONF:FOLD:BACK 1

Query syntax: CONFigure:FOLD:BACK?

Returned parameter: <NR1>

### CONFigure:FOLD:TIME

Set foldback protection delay time.

Command syntax: CONFigure:FOLD:TIME <NRf>

Parameter: 0.1～600.0

Unit: s (Second)

Example: CONF:FOLD:TIME 1

Query syntax: CONFigure:FOLD:TIME?

Returned parameter: <NRf>[Unit=s]

### CONFigure:APG:MODE

Set analog programming mode.

Command syntax: CONFigure:APG:MODE <NR1>

Parameter: 0 | 1 | 2 |3 | 4 | OFF | U | I | U&I | P

Unit: None

Example: CONF:APG:MODE 1

Query syntax: CONFigure:APG:MODE?

Returned parameter: <NR1>

### CONFigure:APG:VOLTage

Set analog programming reference voltage.

Command syntax: CONFigure:APG:VOLTage <NR1>

Parameter: 0 | 1 | REF5 | REF10

Unit: None

Example: CONF:APG:VOLT 1

Query syntax: CONFigure:APG:VOLTage?

Returned parameter: <NR1>

## CONFigure:INHibit

Set external signal control behavior for PSU output.

Command syntax: CONFigure:INHibit <NR1>

Parameter: 0 | 1 | 2 | TRIGGER | TOGGLE | HOLD

Unit: None

Example: CONF:INH 1

Query syntax: CONFigure:INHibit?

Returned parameter: <NR1>

## CONFigure:AUTO:LOAD

Turn on/off power supply auto load function, when this function is ON, the PSU will save the parameters before it is powered off, and automatically load the previously saved paramters when it is powered on.

Command syntax: CONFigure:AUTO:LOAD <NR1>

Parameter: 0 | 1 | OFF | ON

Unit: None

Example: CONF:AUTO:LOAD ON

Query syntax: CONFigure:AUTO:LOAD?

Returned parameter: <NR1>

## CONFigure:AUTO:OUTPut

Set the output status of the PSU at the moment when it is powered on. If auto load function is ON, at the same time auto output is set as ON, then the PSU will automatically turn on output when it is powered on.

Command syntax: CONFigure:AUTO:OUTPut <NR1>

Parameter: 0 | 1 | OFF | ON

Unit: None

Example: CONF:AUTO:OUTP ON

Query syntax: CONFigure:AUTO:OUTPut?

Returned parameter: <NR1>

## 3.3. OUTPut command

### OUTPut[:STATe]

Turn on/off output, also used for turn on/off test functin output.

Command syntax: OUTPut[:STATe] <bool>

Parameter: 0 | 1 | OFF | ON

Example: OUTP ON

Query syntax: OUTPut?

Returned parameter: <NR1>

### OUTPut:FUNCtion

Switch power supply test function.

Command syntax: OUTPut:FUNCtion <NR1>

Parameter: 0 | 1 | 2 | VI | SEQ | CP

Example: OUTP:FUNC VI

Query syntax: OUTPut:FUNCtion?

Returned parameter: <NR1>

### OUTPut:PROTect:VOLTage

Set power supply overvoltage protection threshold.

Command syntax: OUTPut:PROTect:VOLTage <NRf>

Parameter: MIN～MAX

Unit: V (Volt)

Example: OUTP:PROT:VOLT 10

Query syntax: OUTPut:PROTect:VOLTage?

Returned parameter: <NRf>[Unit=V]

### OUTPut:PROTect:CURRent

Set power supply overcurrent protection threshold.

Command syntax: OUTPut:PROTect:CURRent <NRf>

Parameter: MIN～MAX

Unit: A (Ampere)

Example: OUTP:PROT:CURR 10

Query syntax: OUTPut:PROTect:CURRent?

Returned parameter: <NRf>[Unit=A]

### OUTPut:PROTect:POWer

Set power supply overpower protection threshold.

Command syntax: OUTPut:PROTect:POWer <NRf>

Parameter: MIN～MAX

Unit: W (Watt)

Example: OUTP:PROT:POW 1000.0

Query syntax: OUTPut:PROTect:POWer?

Returned parameter: <NRf>[Unit=W]

### OUTPut:PROTect:CLEar

Clear the protection state or fault of the power supply.

Command syntax: OUTPut:PROTect:CLEar

Example: OUTP:PROT:CLE

Parameter: None

Query syntax: None

## 3.4. SOURce command

### SOURce:VOLTage[:LEVel]

Set static output voltage.

Command syntax: SOURce:VOLTage[:LEVel] <NRf>

Parameter: MIN～MAX

Unit: V (Volt)

Example: SOUR:VOLT 25.0

Query syntax: SOURce:VOLTage[:LEVel]?

Returned parameter: <NRf>[Unit=V]

### SOURce:VOLTage:LIMit:HIGH

Set static output voltage upper limit, in order to protect the DUT.

Command syntax: SOURce:VOLTage:LIMit:HIGH <NRf>

Parameter: MIN～MAX

Unit: V (Volt)

Example: SOUR:VOLT:LIM:HIGH 80.0

Query syntax: SOURce:VOLTage:LIMit:HIGH?

Returned parameter: <NRf>[Unit=V]

## SOURce:VOLTage:LIMit:LOW

Set static output voltage lower limit, in order to protect the DUT.

Command syntax: SOURce:VOLTage:LIMit:LOW <NRf>

Parameter: MIN～MAX

Unit: V (Volt)

Example: SOUR:VOLT:LIM:LOW 10.0

Query syntax: SOURce:VOLTage:LIMit:LOW?

Returned parameter: <NRf>[Unit=V]

## SOURce:CURRent[:LEVel]

Set static output current.

Command syntax: SOURce:CURRent[:LEVel] <NRf>

Parameter: MIN～MAX

Unit: A (Ampere)

Example: SOUR:CURR 5.0

Query syntax: SOURce:CURRent[:LEVel]?

Returned parameter: <NRf>[Unit=A]

## SOURce:CURRent:LIMit:HIGH

Set static output current upper limit, to protect the DUT.

Command syntax: SOURce:CURRent:LIMit:HIGH <NRf>

Parameter: MIN～MAX

Unit: A (Ampere)

Example: SOUR:CURR:LIM:HIGH 30.0

Query syntax: SOURce:CURRent:LIMit:HIGH?

Returned parameter: <NRf>[Unit=A]

## SOURce:CURRent:LIMit:LOW

Set static output current lower limit, to protect the DUT.

Command syntax: SOURce:CURRent:LIMit:LOW <NRf>

Parameter: MIN～MAX

Unit: A (Ampere)

Example: SOUR:CURR:LIM:LOW 1.5

Query syntax: SOURce:CURRent:LIMit:LOW?

Returned parameter: <NRf>[Unit=A]

## 3.5. SEQuence command

### SEQuence:STATus

Query sequence running status, return sequence cycle times and current step number.

Command syntax: SEQuence:STATus?

Parameter: None

Example: SEQ:STAT?

Returned parameter: <NR1>,<NR1>

### SEQuence:RUN:NUMBer

Set the file number in sequnce test mode.

Command syntax: SEQuence:RUN:NUMBer <NR1>

Parameter: 1～8

Example: SEQ:RUN:NUMB 1

Query syntax: SEQuence:RUN:NUMBer?

Returned parameter: <NR1>

### SEQuence:EDIT:NUMBer

Set the file number in sequence edit mode.

Command syntax: SEQuence:EDIT:NUMBer <NR1>

Parameter: 1～8

Example: SEQ:EDIT:NUMB 1

Query syntax: SEQuence:EDIT:NUMBer?

Returned parameter: <NR1>

## SEQuence:EDIT:COUNt

Set file length for the sequence file being edited.

Command syntax: SEQuence:EDIT:COUNt <NR1>

Parameter: 1～50

Example: SEQ:EDIT:COUN 10

Query syntax: SEQuence:EDIT:COUNt?

Returned parameter: <NR1>

## SEQuence:EDIT:CYCLe

Set cycle times for the sequence being edited. Set to 0 means infinit loop.

Command syntax: SEQuence:EDIT:CYCLe <NR1>

Parameter: 0～60000

Example: SEQ:EDIT:CYCL 1

Query syntax: SEQuence:EDIT:CYCLe?

Returned parameter: <NR1>

## SEQuence:EDIT:LINK

Set linked sequence for the sequence being edited. 0 means no link.

Command syntax: SEQuence:EDIT:LINK <NR1>

Parameter: 0～8

Example: SEQ:EDIT:LINK 0

Query syntax: SEQuence:EDIT:LINK?

Returned parameter: <NR1>

## SEQuence:EDIT:SAVE

Save the sequnce file that is being edited.

Command syntax: SEQuence:EDIT:SAVE

Parameter: None

Example: SEQ:EDIT:SAVE

Query syntax: None

## SEQuence:EDIT:STEP

Set the step number for the present step being edited.

Command syntax: SEQuence:EDIT:STEP <NR1>

Parameter: 1～100

Example: SEQ:EDIT:STEP 1

Query syntax: SEQuence:EDIT:STEP?

Returned parameter: <NR1>

## SEQuence:EDIT:VOLTage

Set output voltage for present step in sequence file.

Command syntax: SEQuence:EDIT:VOLTage <NRf>

Parameter: MIN～MAX

Unit: V (Volt)

Example: SEQ:EDIT:VOLT 12.0

Query syntax: SEQuence:EDIT:VOLTage?

Returned parameter: <NRf>[Unit=V]

## SEQuence:EDIT:CURRent

Set output current for step in sequence file.

Command syntax: SEQuence:EDIT:CURRent <NRf>

Parameter: MIN～MAX

Unit: A (Ampere)

Example: SEQ:EDIT:CURR 5.0

Query syntax: SEQuence:EDIT:CURRent?

Returned parameter: <NRf>[Unit=A]

## SEQuence:EDIT:DELay

Set delay time of the present step in sequence file.

Command syntax: SEQuence:EDIT:DELay <NRf>

Parameter: MIN～MAX

Unit: s (Second)

Example: SEQ:EDIT:DEL 1.0

Query syntax: SEQuence:EDIT:DELay?

Returned parameter: <NRf>[Unit=s]

## 3.6. CP command

### CP:VOLTage

Set maximum output voltage in CP output mode.

Command syntax: CP:VOLTage <NRf>

Parameter: MIN～MAX

Unit: V (Volt)

Example: CP:VOLT 100.0

Query syntax: CP:VOLTage?

Returned parameter: <NRf>[Unit=V]

### CP:CURRent

Set maximum output current in CP output mode.

Command syntax: CP:CURRent <NRf>

Parameter: MIN～MAX

Unit: A (Ampere)

Example: CP:CURR 50.0

Query syntax: CP:CURRent?

Returned parameter: <NRf>[Unit=A]

### CP:POWer

Set constant output power in CP output mode.

Command syntax: CP:POWer <NRf>

Parameter: MIN～MAX

Unit: W (Watt)

Example: CP:POW 500.0

Query syntax: CP:POWer?

Returned parameter: <NRf>[Unit=W]

### CP:RESPonse

Set response speed in CP output mode.

Command syntax: CP:RESPonse <NR1>

Parameter: 1～100

Unit: None

Example: CP:RESP 50

Query syntax:　CP:RESPonse?

Returned parameter: <NR1>

## 3.7. MEASure command

### MEASure:VOLTage?

Enquire the actual output voltage.

Query syntax: MEASure:VOLTage?

Example: MEAS:VOLT?

Returned parameter: <NRf>[Unit=V]

### MEASure:CURRent?

Enquire the actual output current.

Query syntax: MEASure:CURRent?

Example: MEAS:CURR?

Returned parameter: <NRf>[Unit=A]

### MEASure:POWer?

Enquire the actual output power.

Query syntax: MEASure:POWer?

Example: MEAS:POW?

Returned parameter: <NRf>[Unit=W]

### MEASure:TEMPerature?

Enquire the actual module temperature.

Command syntax: MEASure:TEMPerature?

Example: MEAS:TEMP?

Returned parameter: <NRf>[Unit=℃]

## 3.8. STATus command

### STATus:CHANnel:CONDition?

Query the Channel Condition register.

Command syntax: STATus:CHANnel:CONDition?

Parameter: None

Example: STAT:CHAN:COND?

Returned parameter: <NR1>

### STATus:CHANnel[:EVENt]?

Query the Channel Event register. The Channel Event register is cleared after read.

Command syntax: STATus:CHANnel[:EVENt]?

Parameter: None

Example: STAT:CHAN?

Returned parameter: <NR1>

### STATus:CHANnel:ENABle

Programs the Channel Event Enable Register bits.

Command syntax: STATus:CHANnel:ENABle <NR1>

Parameter: 0～65535

Example: STAT:CHAN:ENAB 65535

Query syntax: STATus:CHANnel:ENABle?

Returned parameter: <NR1>

## 3.9. SYSTem command

### SYSTem:ERRor?

Query the Error Queue.

Command syntax: SYSTem:ERRor?

Example: SYST:ERR?

Returned parameter: <NR1>,<SRD>

## SYSTem:LOCal

Exsit remote control mode, enter local control.
Command syntax: SYSTem:LOCal

Parameter: None

Example: SYST:LOC

Query syntax: None

## SYSTem:VERSion?

Query system version information.
Command syntax: SYSTem:VERSion?

Parameter: None

Example: SYST:VERS?

Returned parameter: <NR2>

# 4. SCPI error information

## 4.1. Introduction

Any errors that occur during the work process are recorded in the error queue until the error queue is full. The error information can be read through the panel menu or programming commands.

Errors are retrieved in the order of first-in, first-out, and the first error returned is the earliest error. Each time it is read, one error item is deleted from the error queue. If there is no error currently, i.e. the error queue is empty, the power supply will return the message "+0 No error" when sending the query command.

## 4.2. Check error information

In remote control mode, execute the following command to read and clear an error message in the error queue:

SYSTem:ERRor?

The information returned by this command is a string, such as:

"+101 Invalid character"

This error message indicates that there are invalid characters in the command string received by the power supply. If all error information is read or no error occurs when query the error queue, executing the "SYSTem:ERRor?" command will return information:

"+0 No error"

This message means that there is no error or that the error message has all been cleared.

The following subsections will describe in detail the meanings of error codes which is returned by the power supply.

## 4.3. Command errors

| | |
|---|---|
| -100 | Command error |
| -101 | Invalid character |
| -102 | Syntax error |
| -103 | Invalid separator |
| -104 | Data type error |
| -105 | GET not allowed |
| -106 | Semicolon unwanted |
| -107 | Comma unwanted |

| | |
|---|---|
| -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 rang |
| -115 | Command can not query |
| -116 | Command must query |
| -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 |
| -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 |
| -180 | Macro error |
| -181 | Invalid outside macro definition |
| -183 | Invalid inside macro definition |
| -184 | Macro parameter error |

## 4.4. Execution errors

| | |
|---|---|
| -200 | Execution error |
| -220 | Parameter error |
| -221 | Setting conflict |
| -222 | Data out of range |
| -224 | Illegal paramter value |
| -225 | Out of memory |
| -232 | Invalid format |
| -240 | Hardware error |
| -242 | Calibration data lost |
| -243 | NO reference |
| -256 | File name not found |
| -259 | Not selected file |
| -295 | Input buffer overflow |
| -296 | Output buffer overflow |

## 4.5. Query errors

| | |
|---|---|
| -350 | Query overflow |
| -400 | Query error |

# 5. Programming example

## 5.1. VI static output

```
OUTP OFF                     //Must turn off output before swtching output mode
OUTP:FUNC VI                 //Switch to static VI output mode
SOUR:VOLT 10                 //Set output votlage is 10V
SOUR:CURR 10                 //Set output current is 10A
OUTP ON                      //Turn on output
SOUR:VOLT 20                 //Set output voltage to 20V
```

## 5.2. Query measurement parameter

```
MEAS:VOLT?                   //Display the measured voltage
MEAS:CURR?                   //Display the measured current
MEAS:POW?                    //Display the meausred power
MEAS:VOLT?;CURR?;POW?        //Display the measured voltage, current, power
```

## 5.3. Edit SEQ file

```
OUTP OFF                     //Must turn off output before edit sequence file
SEQ:EDIT:NUMB 1              //Edit sequence file, file number is 1
SEQ:EDIT:COUN 3              //Set sequence length is 3
SEQ:EDIT:CYCL 1              //Set sequence running cycles is 1
SEQ:EDIT:LINK 0              //Set SEQ link file, 0 means no linked sequence file
SEQ:EDIT:STEP 1              //Edit step 1
SEQ:EDIT:VOLT 5.0            //Set output voltage is 5V
SEQ:EDIT:CURR 1.0            //Set output current is 1A
SEQ:EDIT:DEL 1.0             //Set step delay is 1s
SEQ:EDIT:STEP 2              //Edit step 2
SEQ:EDIT:VOLT 10.0           //Set output voltage is 10V
SEQ:EDIT:CURR 2.0            //Set output current is 2A
SEQ:EDIT:DEL 2.0             //Set step delay is 2s
SEQ:EDIT:STEP 3              //Edit step 3
SEQ:EDIT:VOLT 15.0           //Set output voltage is 15V
SEQ:EDIT:CURR 3.0            //Set output current is 3A
SEQ:EDIT:DEL 3.0             //Set step delay is 3s
```

```
SEQ:EDIT:SAVE                    //Save the current sequence file
```

## 5.4. Run SEQ File

```
OUTP OFF                          //Must turn off output before swtching output mode
OUTP:FUNC SEQ                     //Switch to sequence test output mode
SEQ:RUN:NUMB 1                    //Select sequence file, file number 1
OUTP ON                           //Turn on output
SEQ:STAT?                        //Query current sequence step and running cycles
```

## 5.5. Constant power (CP) output

```
OUTP OFF                          //Must turn off output before swtching output mode
OUTP:FUNC CP                      //Switch to CP output mode
CP:VOLT 20.0                     //Set maximum voltage 20V
CP:CURR 50.0                     //Set maximum current   50A
CP:POW 100.0                      //Set constant output power 100W
CP:RESP 100                      //Set constant power response speed 100%
OUTP ON                           //Turn on output
CP:POW 200.0                      //Change output power to 200W
```