S1465 Series Signal Generator

Programming Manual

Saluki Technology Inc.

The document applies to the signal generator of the following models:

- S1465A signal generator, 100kHz - 3GHz
- S1465B signal generator, 100kHz - 6GHz
- S1465C signal generator, 100kHz - 10GHz
- S1465D signal generator, 100kHz - 20GHz
- S1465F signal generator, 100kHz - 40GHz
- S1465H signal generator, 100kHz - 50GHz
- S1465L signal generator, 100kHz - 67GHz
- S1465A-V signal generator, 100kHz - 3GHz
- S1465B-V signal generator, 100kHz - 6GHz
- S1465C-V signal generator, 100kHz - 10GHz
- S1465D-V signal generator, 100kHz - 20GHz
- S1465F-V signal generator, 100kHz - 40GHz
- S1465H-V signal generator, 100kHz - 50GHz
- S1465L-V signal generator, 100kHz - 67GHz

Standard pack and accessories:

Part No.	Name
1	Main machine
2	User manual
3	Power cable assembly

Options of S1464(-V) series signal generator in addition to standard accessories :

➢ **S1465 series options:**

Option ID	Description	Function	Match
S1465-H01A-A	115dB programmable step attenuator	To expand output power dynamic range	Only A and B options
S1465-H01A-C	115dB programmable step attenuator	To expand output power dynamic range	Only C and D options
S1465-H01A-F	115dB programmable step attenuator	To expand output power dynamic range	Only F options

S1465-H01B	90dB programmable step attenuator	To expand output power dynamic range	Only H and L options
S1465-H02A	Analog modulation	Additional analog modulation, including AM, FM, ΦM, and low-frequency output	All models
S1465-H02B	Pulse modulation	Additional pulse modulation, with the minimum pulse width of 100ns	All models
S1465-H02C	Narrow pulse modulation	Additional pulse modulation, with the minimum pulse width of 20ns	All models, including H02B
S1465-H03	Analog sweep	Additional analog sweep (slope sweep)	All models
S1465-H04	Ultra low phase noise	To reduce phase noise, 10GHz@10kHz: -120dBc/Hz	All models
S1465-H05	High-power output	To increase the maximum output power	All models
S1465-H06	Enhanced high-power output	To increase the maximum output power of 10MHz-20GHz substantially	Only S1465D option
S1465-H80	S87230 USB power probe	For power measurement and calibration (9kHz-6GHz)	All models
S1465-H81	S87231 USB power probe	For power measurement and calibration (10MHz-18GHz)	All models
S1465-H82	S87232 USB power probe	For power measurement and calibration (50MHz-26.5GHz)	All models
S1465-H83	S87233 USB power probe	For power measurement and calibration (50MHz-40GHz)	All models
S1465-H91	N RF output port	To change RF output port to N (female)	Only S1465D option
S1465-H92	Rear panel RF output	To move RF output port to rear panel	All models
S1465-H93	Front handle kit	Front panel mounting handle	All models
S1465-H94	Rack installation kit	Kit for installing instrument on the cabinet	All models
S1465-H95	Commercial calibration certificate	Instrument is entrusted to metrology service	All models

➢ **S1465-V series options:**

Option ID	Description	Function	Match
S1465V-H01A	115dB programmable step attenuator	To expand output power dynamic range	Only A/B/C/D/E/F-V options
S1465V-H01B	90dB programmable step attenuator	To expand output power dynamic range	Only H/L-V options
S1465V-H02A	Analog modulation	Additional analog modulation, including AM, FM, ΦM, and low-frequency output	All models
S1465V-H02B	Pulse modulation	Additional pulse modulation, with the minimum pulse width of 100ns	All models
S1465V-H02C	Narrow pulse modulation	Additional pulse modulation, with the minimum pulse width of 20ns	All models, including H02B
S1465V-H03	Analog sweep	Additional analog sweep (slope sweep)	All models
S1465V-H04	Ultra low phase noise	To reduce phase noise, 10GHz@10kHz: -120dBc/Hz	All models
S1465V-H05	High-power output	To increase the maximum output power	All models
S1465V-H31	Large modulation bandwidth	Extend built-in modulation bandwidth to 200MHz	All models
S1465V-H32	Internal baseband external IQ input	Extend built-in baseband memory to 8GB	ALL models
S1465V-H33	Broadband external IQ input	Add broadband external IQ input function	Only S1465C/D/F-V options
S1465V-H80	S87230 USB power probe	For power measurement and calibration (9kHz-6GHz)	All models
S1465V-H81	S87231 USB power probe	For power measurement and calibration (10MHz-18GHz)	All models
S1465V-H82	S87232 USB power probe	For power measurement and calibration (50MHz-26.5GHz)	All models

S1465V-H83	S87233 USB power probe	For power measurement and calibration (50MHz-40GHz)	All models
S1465V-H91	N RF output port	To change RF output port to N (female)	Only S1465D-V option
S1465V-H92	Back panel RF output	To move RF output port to rear panel	All models
S1465V-H94	Machine frame	/	All models
S1465V-H95	Calibration report	/	All models

Preface

Thank you for choosing Saluki Technology Products.

We devote ourselves to meeting your demands, providing you high-quality measuring instrument and the best after-sales service. We persist with “superior quality and considerate service”, and are committed to offering satisfactory products and service for our clients.

Document No.

S1465-03-02

Version

Rev01 2019.04

Document Authorization

The information contained in this Programming Manual is subject to change without notice. The power to interpret the contents of and terms used in this document rests with Saluki.

Saluki Tech owns the copyright of this document which should not be modified or tampered by any organization or individual or reproduced or transmitted for the purpose of making profit without its prior permission, otherwise Saluki will reserve the right to investigate and affix legal liability of infringement.

Product Quality Assurance

The warranty period of the product is 36 months from the date of delivery. The instrument manufacturer will repair or replace damaged parts according to the actual situation within the warranty period.

Product Quality Certificate

The product meets the indicator requirements of the document at the time of delivery. Calibration and measurement are completed by the measuring organization with qualifications specified by the state, and relevant data are provided for reference.

Quality/Settings Management

Research, development, manufacturing and testing of the product comply with the requirements of the quality and environmental management system.

Contacts

Service Tel:	+886. 909 602 109
Website:	www.salukitec.com
Email:	sales@salukitec.com
Address:	No. 367 Fuxing N Road, Taipei 105, Taiwan (R.O.C.)

Content

1 About the Manual

This manual introduces the methods for remote control of S1465 series signal generator and application of SCPI. Meanwhile, in order to make it convenient for users to quickly master the remote control programming methods, some programming examples are listed, and the basic concept of I/O function library is introduced. To facilitate your skillful use of such instrument, please read carefully and follow this manual in advance for correct operation.

SCPI (Standard Commands for Programmable Instruments) define the standards and methods for remote control of the instrument, and are the remote control programming language for programmable electronic test and measuring instruments. SCPI are based on the IEEE-488.2 standard and form. Please refer to http://www.scpiconsortium.org for details. The manual describes the program control commands of S1465 series signal generators in details.

The chapters of the programming manual include:

- **Remote Control**

The methods for remote control of the instrument are summarized to make users get familiar with remote control quickly. It is divided into three parts: remote control basics, introducing program related concepts, software configuration, program port, SCPI, etc.; instrument port configuration method, introducing the connection method and software configuration method for program ports of S1465 series signal generator; I/O function library, introducing the basic concept of instrument driver and basic installation instructions of IVI-COM/IVI-C driver.

- **SCPI**

Common commands, instrument commands and compatible commands are introduced, and the functions, parameters and examples of SCPI are described.

- **Programming Examples**

The basic programming examples and advanced programming examples are provided in the way of text description and example code, and the explanation is provided to make it convenient for users to quickly master the remote control programming method of signal generator.

- **Error Description**

Error description and method to obtain after-sales services are included.

- **Annexes**

Necessary reference information related to program control of S1465 series signal generator is provided, including zoom table of SCPI and zoom table of errors.

2 Remote Control

This section introduces briefly the program control foundation, program control interface and configuration method and basic VISA interface programming method of the S1465 series signal generator, as well as the concept and classification of the I/O instrument driver library, so as to facilitate users' remote control. Specific contents are as follows:

- **Remote control basis**
- **Remote interface and configuration**
- **I/O function library**

2.1 Remote Control Basics

2.1.1 Remote Interface

Instruments with remote control function generally support two kinds of remote control interface: LAN and GPIB, and the type of port supported by the specific model of instrument is determined by the function of the instrument.

It is shown in the table below:

Table 2.1 Remote control interface type and VISA addressing character string

Program control interface	VISA addressing character string	Description
LAN (Local Area Network)	**VXI-11 protocol:** TCPIP::host_address[::LAN_device_name][::INSTR] **Raw socket protocol:** TCPIP::host_address::port::SOCKET	The operator realizes remote control by connecting the instrument via the network port on the rear panel of the instrument. For specific protocols, see the following: "2.1.1.1LAN interface"
GPIB (IEC/IEEE Bus Interface)	GPIB::primary address[::INSTR] 4	The operator realizes remote control by connecting the instrument via the port on the rear panel of the instrument. It follows the IEC 625.1/IEEE 418 bus interface standard. For details, see the following: "2.1.1.2 GPIB interface"

2.1.1.1 LAN interface

The signal generator can realize remote control via the computer in the 10Base-T and 100Base-T LANs, and various instruments form a system in the LAN, which is controlled by the computer in the LAN. The signal generator needs to be provided with port connectors, network cards, associated network protocols and relevant network services first before realizing remote control in the

LAN, and the master computer in the LAN needs also to be provided with instrument control software and VISA library in advance. Three working modes of the network card are as follows:

- 10Mbit/s Ethernet IEEE802.3;
- 100Mbit/s Ethernet IEEE802.3u;
- 1Gbit/s Ethernet IEEE802.3ab.

Control computer and signal generator need to be connected to a common TCP/IP protocol network via network interface. The cable between the computer and the signal generator is a commercial RJ45 cable (shielded or unshielded Category 5 twisted pair). During data transmission, data packet transmission will be adopted, and LAN transmission is faster. The cable length between the computer and the signal generator shall generally not exceed 100 m (100Base-T and 10Base-T). For more information about LAN communication, refer to: http://www.ieee.org. The knowledge of LAN interface is introduced hereinafter.

1) IP address

\When the signal generator is remotely controlled via LAN, the physical network connection shall be guaranteed to be smooth. The address of the signal generator is set to the subnet where the main control computer is located via menu “Local IP”. For example, if the IP address of the master computer is 192.168.12.0, the IP of the signal generator shall be set to 192.168.12.XXX, whereas XXX is the figure between 1 - 255. **The default network port number used by the signal generator for communication is 5001.**

Only the IP address is required for network connection, and the VISA addressing character string format is as follows:

TCPIP::host address[::LAN device name][::INSTR] or

TCPIP::host address::port::SOCKET

Whereas:

- TCPIP is the network protocol used;
- host address: IP address or host name of the instrument, for identification and control of the controlled instrument;
- The LAN device name defines the handle number of the protocol and subset (optional);
  - The VXI-11 protocol is adopted for the 0# equipment;
  - The newer high speed LAN instrument protocol is adopted for the 0# high speed LAN instrument;
- The INSTR is the instrument resource type (optional);
- The port marks the socket port number;
- SOCKET is the socket resource type of the original network.

Example:

- If the instrument IP is 192.1.2.3, the valid resource character string of the VXI-11 protocol is as follows:

TCPIP::192.1.2.3::INSTR

- The following can be used for establishing raw socket connection:

TCPIP::192.1.2.3::5001::SOCKET

Tip

Method of recognizing multiple instruments in the program control system

If multiple instruments are connected to the network, they can be identified by their individual IP address and associated resource string. The main control computer uses the respective VISA resource string for instrument identification.

2) VXI-11 protocol

The VXI-11 standard is based on the ONC RPC (Open Network Computing Remote Procedure Call) protocol, which is the network/transport layer of the TCP/IP protocol. Since the TCP/IP network protocol and relevant network services are configured in advance, such connection-oriented communication can follow the principle of exchanging in sequence and recognize connection interruption during communication, ensuring no information missing.

3) Socket communication

The TCP/IP protocol connects the signal generator to the network via LAN sockets. Socket is a basic method used in computer network programming that enables applications using different hardware and operating systems to communicate in the network. This method allows for two-way communication between the signal generator and the computers via ports.

The sockets is a specially-written software class that defines the information necessary for network communication (such as the IP addresses and device port numbers) and integrates some basic operations in network programming. Sockets can be used after installing packaged libraries in the operating system. Two commonly used socket libraries are the Berkeley socket library for UNIX the Winsock library for Windows.

The socket in the signal generator is compatible with Berkeley socket and Winsock through Application Program Interface (API). In addition, other standard sockets are also compatible through the API. When the signal generator is controlled using SCPI, the socket program created in the program issues command. The socket port number of the signal generator must be set before LAN socket is used. For the signal generator, the socket port number is set to 5,000.

2.1.1.2 GPIB interface

The GPIB interface is a widely-used instrument remote interface currently, which can be connected with different kinds of instruments through the GPIB cable and can establish the test system with the main control computer. To realize remote control, the main control computer shall be preinstalled with the GPIB bus card, driver and VISA library. During communication, the main control computer will address the controlled instrument through the GPIB bus address firstly. The user can set the GPIB address and ID for querying strings, and the GPIB communication language can be set to the SCPI form by default. The operation of the GPIB and its relevant interface is defined and described in details in the ANSI/IEEE standard 488.1-1987 and the ANSI/IEEE standard 488.2-1992. For details, refer to IEEE website: http://www.ieee.org. i

GPIB processes information in bytes at the data transmission speed of up to 8MBps, which is fast. Since the data transmission speed is limited by the distance between the equipment/system and computer, pay attention to the following during GPIB connection:

- Up to 15 instruments can be set up via the GPIB interface;
- The total length of the transmission cable cannot exceed 15m nor double the number of instruments in the system. In general, the maximum length of the transmission cable between the devices cannot exceed 2 meters.
- In case of multiple instruments connected in parallel, you need to use “OR” connectors.
- The terminal of the IEC bus cable shall be connected to the instrument or controller computer.

2.1.2 Message

Messages transmitted on the data cable can be classified into two types as follows:

1) Interface message

During communication between the instrument and master computer, the attention line must be pulled down so that the interface message can be transmitted to the instrument via the data cable. Only instruments with the GPIB bus function can transmit the interface message.

2) Instrument message

For the detailed structure and syntax of the instrument message, see Section "2.1.3 SCPI". The instrument message can be divided into command and instrument response according to the different transmission directions. All remote interfaces use instrument message in the same method unless otherwise stated.

a. Command:

A command (programming message) is a message transmitted from the main control computer to the instrument for remote instrument control and status information query. The command is divided into the following two types:

➢ Based on the effect on the instrument:

- Setting command: to change the instrument setting status, for example, to reset or set frequencies.
- Query command: to query and return data, for example, to recognize instruments or query parameter values. The query command is ended with a suffix question mark.

➢ Based on the definition in the standard:

- Commonl command: to define the function and syntax by IEEE488.2, applicable to all types of instruments (if realized) for such functions as register management in standard status, reset and self test.
- Instrument control command: to be the instrument feature command for realizing instrument functions. For example, to set frequencies.

b. Instrument response:

Instrument response (response message and service request) is the query result information sent from the instrument to the computer. The information contains the measurement result and instrument status.

2.1.3 SCPI

2.1.3.1 SCPI command brief introduction

SCPI (Standard Commands for Programmable Instruments) is the command set that is established based on the standard IEEE488.2 and applicable to all instruments. The purpose is to provide the same function with same program control command, realizing universality of program control commands.

The SCPI command consists of the command header and one or multiple parameters, which are separated by blank. The command header contains one or multiple key fields. The command acts as a query command if it is suffixed with a question mark. The command can be divided into the common command and instrument-specific command, with different syntactic structure for each other. The SCPI command has the following features:

1) The SCPI is established for the test functions rather than instrument operation description.

2) The SCPI reduces the repetition of the realization process of similar test functions, thus ensuring the programming compatibility;

3) Remote control message is defined in a layer that is independent of the communication physical layer hardware.

4) The SCPI is irrelated with the programming methods and languages, and the SCPI test program is easy to be transplanted;

5) The SCPI is scalable so that it is applicable to measurement control on different scales.

6) Scalability makes SCPI a “Live” standard.

If you are interested in learning more about SCPI, please refer to:

IEEE Standard 488.1-1987, IEEE Standard Digital Interface for

Programmable Instrumentation. New York, NY, 1998.

IEEE Standard 488.2-1987, IEEE Standard Codes, Formats, Protocols and Comment

Commands for Use with ANSI/IEEE Std488.1-1987. New York, NY, 1998

Standard Commands for Programmable Instruments(SCPI) VERSION 1999.0.

For details on the program control command set, classification and description of the S1465 series signal generators, see the following:

1) “3 SCPI ” of the Manual;
2) Appendix B Lookup Table of SCPI in this manual.

2.1.3.2 SCPI command description

1) General terms

The following terms apply to this section. In order to better understand the content hereinafter, you need to understand the exact definitions of these terms.

Controller

The controller is any computer used to communicate with the SCPI device. A controller may be a PC, a small computer, or a card on the card cage. Some AI devices can also be used as controllers.

Equipment

The euqipment is any device that supports SCPI. Most of the devices are electronic measurement or excitation devices that use GPIB interfaces for communication.

Remote Control message

The remote control message is a combination of one or more correctly formatted SCPIs. It guides the equipment to measure and output the signal.

Response message

The response message is a data set that specifies the SCPI format. It is always sent from the equipment to the controller or listener to remind the controller of the internal condition or measured value of the equipment.

Command

A command is an instruction that satisfies the SCPI standard. The combination of commands controlling the devices forms a message. In general, a command includes keywords, parameters, and punctuation.

Event command

An event-type SCPI can't be queried. An event command generally has no corresponding key settings on front panel. Its function is to trigger an event at a particular moment.

Query

A query is a special type of command. When the controller is queried, it is necessary to return to the response message in conformity with syntax requirement of the controller. The query statement is always ended with a question mark.

2) Command type

There are two types of SCPIs: common commands and instrument-specific commands. Figure 2.1 shows the difference between two commands. Common commands are defined in IEEE 488.2 to manage macros, status registers, synchronization, and data storage. Common commands are easy to recognize as they all begin with an asterisk. For example, *IDN? , *OPC and *RST are common commands. Common commands don't belong to any instrument-specific command. The instrument uses the same method to interpret them without consideration to the current path setting.

It is very easy to identify instrument-specific commands because they contain a colon (:). Colons are used between the beginning of command expression and keywords, e.g.: FREQuency[:CW?]. Instrument-specific commands are divided into command subsets of corresponding subsystem according to the functional block inside the instrument. For example, the power subsystem (:POWer) contains the power-related command while the status subsystem (:STATus) contains the command for the status control register.

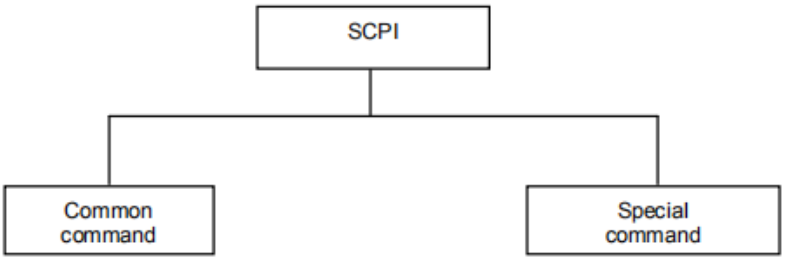

Figure 2.1 Types of SCPI commands

3) Instrument-specific command syntax

A typical command consists of a keyword prefixed with a colon. The keyword is followed by parameters. The following is an example of a syntax declaration:

[:SOURce]:POWer[:LEVel] MAXimum|MINimum

In the example above, the [:LEVel] in the command follow : POWer closely without any space. Following the [:LEVel]: MINimum|MAXimum is the parameter. There is a space between the command and its parameter. The description of other parts of the syntax expression are shown in Tables 2.2 and 2.3.

Table 2.2 Special characters in the command syntax

Symbol	Meaning	Example
|	The vertical between the keyword and the parameter means multiple options.	[:SOURce]:AM:SOURce EXTernal|INTernal EXTernal and INTernal ARE alternative choices.
[]	Square brackets indicate that the keyword or parameter contained is optional when constructing the command. The command will also be executed even if these implied keywords or parameters are ignored.	[:SOURce]:AM[:DEPTh]:EXPonential? SOURce and DEPTh are dispensable.

< >	The part inside the angle brackets can't be used literally. In the command, instead, it represents the part that must be contained.	[:SOURce]:FREQ:STOP <val> In this command, <val> must be replaced by an actual frequency and unit. For example::FREQ:STOP 3.5GHz
{ }	The part inside the braces indicates that the parameter is optional.	[:SOURce]:LIST:POWer <val>{,<val>} For example: LIST:POWer 5

Table 2.3 Command syntax

Characters, keywords and syntax	Example
Uppercase characters represent the minimum set of characters required to execute a command.	[:SOURce]:FREQuency[:CW]?, FREQ is the short format part of the command.
The lower case part of the command is optional; such flexible format is loaded "flexible listening". For more information, please refer to Section "Command parameter and response".	:FREQuency :FREQ,:FREQuency or :FREQUENCY; either of them is correct.
When a colon is between the two command mnemonics, it moves the current path in the command tree down by one level. For more information, please refer to the command path part in section "Command tree".	:TRIGger[:SEQuence]:SOURce? TRIGger is the topmost keyword of this command.
If the command contains multiple parameters, the comma is used for spacing between parameters. The parameter is not part of the command path, so it does not affect the levels of the path.	[:SOURce]:LIST:DWELI <val>{,<val>}
The semicolon is used to separate 2 adjacen commands, without affecting current command path.	:FREQ 2.5GHZ; :POW 10DBM
Blank characters, such as <space> or <tab>, are usually ignored as long as they do not appear between keywords or in keywords. However, you must separate the commands and parameters with blank characters, which does not affect the current path.	:FREQ uency or :POWer :LEVel6.2 is not allowed. :LEVel and 6.2 must be separated by a space. i.e. :POWer:LEVel 6.2

4) Command tree

Most remote control programming will adopt the instrument-specific command. SCPI adopts a structure similar with the file system for analyzing such command, and this command structure is called a command tree, as shown in Figure 2.2:

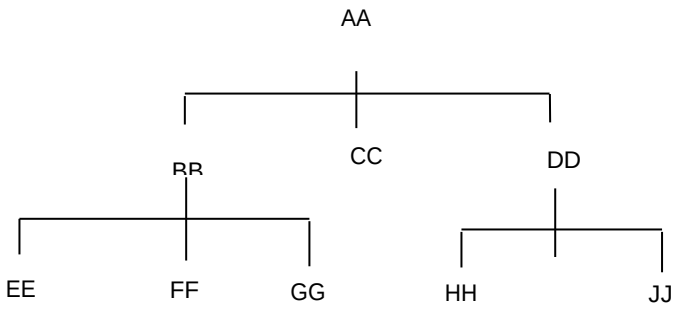

Figure 2.2 Simplified Command Tree Diagram

The command on the top is the root command, which is called the "root" for short. You must go to the next level of commands based on a specific path during command analysis. For example: :POWer:ALC:SOURce? where, POWer stands for AA,: ALC stands for BB, :SOURce stands for GG, and the whole command path is (:AA:BB:GG).

The software module (**command interpreter**) in the instrument software is used specially for analyzing each SCPI command received. The interpreter uses a series of rules that identify the paths of the command tree to divide the command into separate command elements. Since the same command keyword can exist in different paths, current command path will be kept after current command is analyzed, so as to make analysis on follow-up commands quicker and more efficient. After booting or *RST (reseting) the instrument, current command path is reset to root.

5) Command parameters and response

The SCPI define different data formats in the use of program and response messages to comply with the principles of "**flexible listening**" and "**accurate speaking**". For more information, please refer to IEEE488.2. "Flexible listening" means that the formats of the commands and parameters are flexible.

For example, the signal generator is set to a frequency reference status command, i.e. FREQuency:REFerence:STATe ON|OFF|1|0.

The following command formats are used to set the frequency reference function as “On”:

:FREQuency:REFerence:STATe ON, :FREQuency:REFerence:STATe 1,

:FREQ:REF:STAT ON, :FREQ:REF:STAT 1.

Different parameter type corresponds to one or more response data types. The value type parameter returns a data type when querying, and the response data are accurate and rigorous, which is called "**accurate speaking**.”

For example, if you query the power state (:POWer:ALC:STATe?), when it is on, the response data returned is always 1 whether the setting command sent previously is POWer:ALC:STATe 1 or :POWer:ALC:STATe ON.

Table 2.4 SCPI command parameters and response types

Parameter type	Response data type
Numerical	Real number or integer
Extended numerical	Integer
Discrete	Discrete
Boolean	Numeric boolean
String	String
Block	Definite Length Block
	Indefinite Length Block

Non-decimal numeric types	Hexadecimal
	Octal
	Binary

Numerical parameters

Numeric parameters can be used in both instrument-specific commands and common commands. Numeric parameters receive all common decimal notations, including positive/negative signs, decimal point, and scientific notation. If a device only accepts a specified numeric type, such as an integer, it will automatically round up the received numeric parameters.

The following are examples of numeric parameters:

0	No decimal point
100	Optional decimal point
1.23	Signed bit
4.56e<space>3	Index mark e can be followed by a space
-7.89E-01	Index marker e can be uppercase or lowercase
+256	Positive lookahead allowed
5	Decimal points can be used first

Extended numerical parameters

Most measurements related to instrument-specific commands use extended numeric parameters to specify physical quantities. Extended numerical parameters receive all numeric parameters and additional special values. All extended numeric parameters receive MAXimum and MINimum as parameter values. Other special values, For example: whether to receive the UP and DOWN values depends on the instrument analysis capacity, and all valid parameters are listed in the SCPI command table.

Note: Extended numeric parameters do not apply to common commands or subsystem command STATus.

Examples of extended numeric parameters:

101	Numeric parameter
1.2GHz	GHz can be used as an index (E009)
200MHz	MHz can be used as an index (E006)
-100mV	-100 millivolts
10DEG	10 degrees
MAXimum	Maximum effective setting
MINimum	Minimum effective setting
UP	Increase by a step
DOWN	Recrease by a step

Discrete parameters

When there are a finite number of parameter values to be set, discrete parameters are used for identification. A discrete parameter uses mnemonics to represent each valid setting. Like the SCPI mnemonics, the discrete parameter mnemonics can be set in long and short formats, with both capitalized and lowercase characters.

In the following example, discrete parameters are used with commands:

:TRIGger[:SEQuence]:SOURce BUS|IMMediate|EXTernal

BUS	GPIB, LAN, RS-232 trigger
IMMediate	Trigger immediately
EXTernal	Trigger externally

Boolean parameters

Boolean parameters represent a single binary condition that is either true or false. There are only four possible representations for a Boolean parameter.

ON	Logical true
OFF	Logical false
1	Logical true
0	Logical false

String parameters

String parameters allow ASCII strings to be sent as parameters. Single quotes and double quotes are used as separators.

The following are example of string parameters:

'This is Valid' 'This is also Valid' 'SO IS THIS'

Real response data

Large portions of measurement data are real. They are formatted as basic decimal notation or scientific notation. Most advanced programming languages all support these two formats.

Examples of real number response data:

1.23E+0

-1.0E+2

+1.0E+2

0.5E+0

0.23

-100.0

+100.0

0.5

Integer response data

The integer response data are a decimal expression of an integer with the sign bit. When the status register is queried, the integer response data will be mostly returned.

Examples of integer response data:

0 Sign digit optional

+100 Leading + allowed

-100 Leading - allowed

256 No decimal point

Discrete response data

Discrete response data are basically the same as discrete parameters, only that the return format of discrete response data is only a short form in uppercase.

Samples of discrete response data:

INTernal Amplitude level control mode is internal

EXTernal Amplitude level control mode is external

MMHead Amplitude level control mode is millimeter wave source module

Numeric Boolean response data

The Boolean response data returns a binary value of 1 or 0.

String response data

The string response data and string parameters are the same. The main difference is that the string response data use double quotes rather than single quotes as the separator. The string response data can also be inserted with double quotes inside which there can be no characters.

“This is a string”

“one double quote inside brackets: (“”)”

6) Systems of values in commands

The value of command can be entered in binary, decimal, hexadecimal or octal format. In the binary, hexadecimal, or octal format, a suitable identifier should be added in front of the value. In the decimal (default) format, an identifier isn’t required.

When the value without an indicator is entered, the equipment will ensure that it is entered in decimal format. The identifiers required in all formats are listed as follows:

- #B indicates that this number is a binary value.
- #H indicates that this number is a hexadecimal value.
- #Q indicates that this number is an octal value.

The following are various representations of the decimal number 45 in SCPI commands:

#B101101

#H2D

#Q55

The following example sets the RF output power to 10 dBm (or a value of the equivalent value of the currently selected unit, such as DBUV or DBUVEMF) with a hexadecimal value of 000A.

:POW #H000A

When using a non-decimal format, a measurement unit, such as DBM or mV, is not used with the value.

7) Command line structure

Since one command line can contain multiple SCPI commands, the following methods can be used to indicate the end of current command line:

- Enter;
- Enter and EOI;
- EOI and the last data byte.

Commands in command line are separated by semicolons, and commands for different subsystems begin with a colon. For example:

MEM:COPY:NAME Test1, MeasurementXY;FREQ:CW 5GHz.

This command line contains two commands, in which the first one belongs to the MEM subsystem and the second one belongs to the FREQ subsystem. If the adjacent commands belong to the same subsystem, the command path will be partially repeated and the command can be abbreviated. For example: For example:

FREQ:CW 5GHZ;FREQ:OFFSet 3GHz

This command line contains two commands, both of which belong to the FREQ subsystem and have the same level. Therefore, the second command can begin with the subordinate to FREQ and may not begin with a colon, which can be abbreviated to the following command line:

FREQ:CW 5GHz;OFFS 3GHzc

8) Unit description

The programmble commands provided herein support the frequency units Hz, kHz, MHz and GHz. The programmable commands related to the frequency support parameters without unit, and if the frequency unit is not provided, the default unit is Hz; the power unit dBm is supported currently, and if the parameter unit is not provided, the default unit is dBm; the gain and attenuation unit dB is supported, and if the parameter unit is not provided, the default unit is dB; the time units ns, us, ms and s are supported, and if the parameter unit is not provided, the default unit is s. See the range of parameters in each command for the unit type of programmable command parameters provided herein.

2.1.4 Command Sequence and Synchronization

IEEE488.2 defines the difference between the overlapping and sequential commands as follows:

- Sequential commands are sequences of commands that are executed continuously. Usually, each command is executed fast.
- Overlapped commands indicate that the previous command is not executed automatically before the next command is executed. Normally overlapped commands take longer to process and allow the program to process other events synchronously.

Even if multiple commands are set in a command line, they are not necessarily executed in the order in which they are received. In order to ensure that the commands are executed in a certain order, each command must be sent as a separate command line.

Example: Command line contains set and query commands

If multiple commands in a command line contain query commands, the query result is unpredictable. The following command returns a fixed value:

:FREQ:STAR 1GHZ;STOP 100;:FREQ:STAR?

Return value: 1000000000 (1GHz)

The following command returns an unfixed value:

:FREQ:STAR 1GHz;STAR?;STOP 1000000

The returned result may be the current frequency start value, because the host program will delay execution of the command. If the host program executes the command after receiving it, the returned value may also be 1 GHz. **This signal generator has not supported the overlapped command yet.**

Tip

The setting command and query command are sent separately.

General rules: In order to ensure the correctness of the returned result from the query command, the setting command and the query command shall be sent in different program control messages.

2.1.4.1 Preventing commands from being subject to overlapping execution

In order to prevent the overlapped execution of commands, multiple threads or commands: *OPC, *OPC? or *WAI can be used. These three commands can be executed only after the hardware is set. During programming, the computer can be forced to wait for some time to synchronize certain events.

The details are separately described below:

- **Controller program uses multiple threads**

Multi threads are used to wait for completion of the command and achieve synchronization of GUI and program control, that is, a single thread waits for completion of *OPC?, without impeding the execution of the GUI or remote control thread.

- **The usage of the three commands in synchronous execution is shown in the table below:**

Table 2.5 Command syntax

Method	Execution action	Programming method
*OPC	Set the operation completion bit is set.	Set ESE BIT0; Set SRE BIT5; Send the overlapping command and *OPC; Wait for the service request signal (SRQ). SRQ represents the completion of execution of the overlapped command.
*OPC?	Stop executing the current command 1. The command is returned only when the operation completion bit in the ESR is set, which indicates that the previous command is processed.	Terminate the processing of the current command before executing other commands. Send this command directly after the current command.

*WAI	Before executing *WAI, wait until all commands are sent and continue processing the uncompleted commands.	Terminate the processing of the current command before executing other commands. Send this commanddirectlyafter the current command.

If the processing time of the overlapped command is short, the command *WAI or *OPC can be used after use of the overlap command to achieve command synchronization. In order to synchronously execute other tasks when the computer or instrument is waiting for the completion of execution of overlapped commands, the following synchronization technologies can be adopted:

- **OPC and service request**

1) Set the OPC mask bit (bit0) of ESE to *ESE 1;
2) Set the bit5 of SRE to *SRE 32 to enable the ESB service request;
3) Send the overlapped command and *OPC;
4) Wait for the service request signal.

SRQ represents the completion of execution of the overlapped command

- **OPC? and service request**

1) Set the bit4 of SRE to *SRE 16 to enable the MAV service request;
2) Send overlapped commands and *OPC? ;
3) Wait for the service request signal.

SRQ represents the completion of execution of the overlapped command

- **Event Status Register (ESE)**

1) Set the OPC mask bit (bit0) of ESE to *ESE 1;
2) Send the overlapped command only and do not send *OPC, *OPC or *WAI;
3) Send "*OPC;*ESR?" in the timer for cyclic query of completion status of operation.

If the return value (LSB) is equal to 1, this indicates that the overlap command has been executed.

- ***OPC? And short timeout**

1) Send the overlapped command only and do not send *OPC, *OPC or *WAI
2) Send "<short timeout>; *OPC?" in the timer, so as to query the completion status of operation circularly;
3) If the return value (LSB) is equal to 1, this indicates that the overlap command has been executed. In case of timeout, the device is in the operational process.
4) Reset the timeout value to the old value;
5) Send the command "SYStem:ERRor?" to clear the error queue and delete the "-410, Query Interrupted" message.

If the return value (LSB) is equal to 1, this indicates that the overlap command has been executed.

2.1.5 Status Report System

The status report system stores all operation status information and error information of current instrument. Such information is stored in the status register and error list respectively, which can be queried via the program control interface.

2.1.5.1 Structure of status register

Please refer to the hierarchical chart of the status register:

Figure 2.3 Hierarchical structure of the status register

Status registers are described by classification below:

1) STB, SRE

The status byte (STB) register and associating mask register (service request enabling register (SRE)) form the highest-level register of the status report system. STB saves almost all operation statuses of the instrument by collecting lower-level register information.

2) ESR, SCPI status register

STB receives the information from the following registers:

- Values from interaction between the event status register (ESR) and event status enabling (ESE) mask register.
- The SCPI status register includes the STATus:OPERation and STATus:QUEStionable registers

（SCPI definition), which contain specific operation information of the instrument. All SCPI status registers have the same internal structure

3) IST,PPE

Similar with SRQ, IST ("Individual Status") marks a separate bit consisting of all statuses of the instrument. The associated parallel poll enable register (PPE) determines the STB data bits for IST marking.

4) Output buffer

The output buffer stores the message returned by the instrument to the controller. It doesn't belong to the status reporting system but determines the value of the MAV bit of STB.

Please refer to "Section 2.1.5 Status Reporting System" of the programming manual for details of the above registers.

Tip

SRE, ESE

The service request enabling register (SRE) can be used for STB enabling. Similarly, the ESE can be used for ESR enabling.

2.1.5.2 Structure of SCPI Status Register

Each standard SCPI register consists of five parts. Each part contains 16 bits and is functionally independent. For example, one bit is assigned for each hardware status and valid for all five parts of the register. If Bit15 is set to 0, the value of the register is positive integer data.

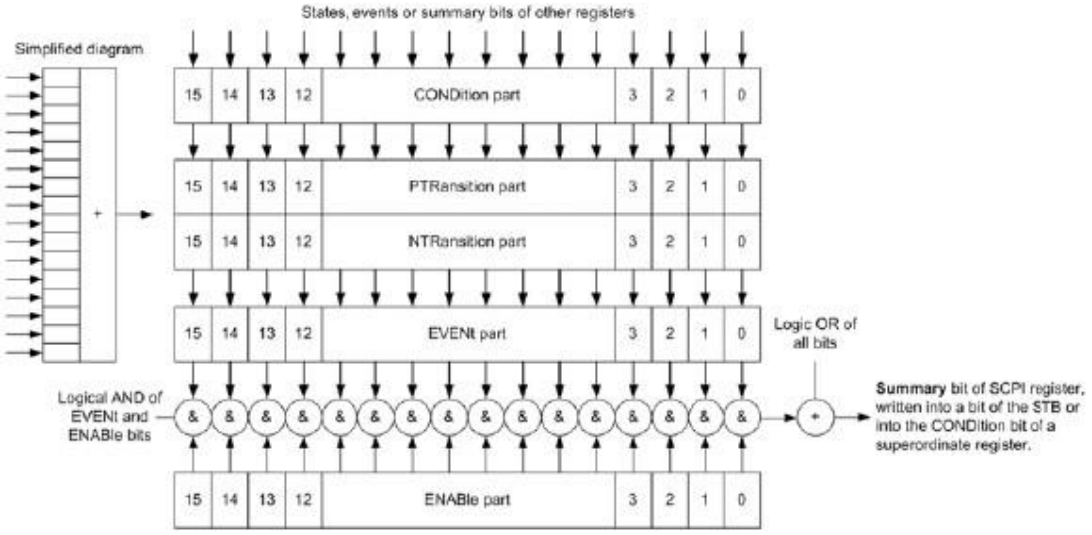

Figure 2.4 Structure of status register

It can be seen from the figure above that the status register consists of five parts, which are respectively described as follows:

- **Condition register**

In this part, the summary bit of hardware or lower registers are directly written, reflecting the working state of the current instrument. This register is read only, not writable. It reads but not clearing the value.

- **Positive and negative conversion register**

The two transfer registers define the state transition bit of the condition register stored in the event register.

The positive conversion register is similar to a conversion filter. When an event bit of the condition register is changed from 0 to 1, the associated PTR bit determines if the event bit should be set to 1, as described below:

If the PTR bit is equal to 1, the event bit should be set.

If the PTR bit is equal to 0, the event bit should not be set.

The positive conversion register is readable and writable, and its reading will not clear any value.

The negative conversion register is similar to a conversion filter. When an event bit of the condition register is changed from 1 to 0, the associated NTR bit determines if the event bit should be set to 1, as described below:

If the NTR bit is equal to 1, the event bit should be set.

If the NTR bit is equal to 0, the event bit should not be set.

The positive conversion register is readable and writable, and its reading will not clear any value.

- **Event register**

This part indicates whether the event has occurred since the last reading and whether the content of the condition register is stored. It represents only the event passed through the transition register. It can only be changed by the instrument and read by the user. The value will be cleared after reading. The value of this part is often equal to the value of the entire register.

- **Enable register**

This part determines whether the related event bit acts on the final summary data. The bits of each enable part is the sum of related enable bits. The logical operation result of this part is or not the summary bit.

-- Enable bit = 0: the related event bit does not act on the data sum.

-- Enable bit = 1: the related event bit acts on the data sum.

This part is readable and writable. It reads but not clearing the value.

- **Data bit sum**

The data sum bit of each register consists of event and enable parts. The result gets into the condition part of the high level register. The instrument automatically generates data sum bit for each register so that events can cause different levels of service requests.

2.1.5.3 Description of Status Register

The status registers are detailed as follows:

1) Status byte (STB) and service request enable register (SRE)

IEEE488.2 defines the status byte (STB). The rough instrument status is reflected by collecting the information of lower registers. Bit6 is equal to the summary data of other status byte bits. The result of a comparison between the status byte and the condition part of the SCPI register may be assumed to be the top in the SCPI hierarchy. The value of status byte may be read through common command "*STB?" or serial query.

The status byte is connected to the service request enable register (SRE). Each bit of the status byte corresponds to a bit in SRE. Bit6 of SRE is ignored. If one of the bits in SRE is set and the related STB bit changes from 0 to 1, a service request (SRQ) will be generated. Common command "*SRE" is used to set SRE, and common command "*SRE?" used to read SRE. The status byte is described in Table 2.6 Description of status byte:

Table 2.6 Description of status byte

Bit	Meaning
0..1	Not used.
2	The error queue is non-null. Set to this bit when a new error is inserted into the error queue. If related SRE bit enables the bit, a service request will be generated when a new error is generated in the error queue, so that the error can be identified and the error can be queried. Such method effectively reduces errors in program control.
3	Data sum bit of the status query register. Set to this bit only when the event bit of the inquiry status register and the related enable bit are set to 1. This bit represents a queriable status of the instrument. Specific instrument status information can be obtained by querying the inquiry status register of the status register.
4	MAV bit (message available). Set to this bit if the output queue information is readable. This bit is used when the controller queries instrument information.
5	ESB bit. Data sum bit of the event status register. The bit can be set if one bit of the event status register is set and the enable event enables the corresponding bit in the register. If the position bit is 1, it means that the instrument has a severe error, and the specific error information can be obtained by querying the event status register.
6	MSS bit (master status summary bit). Set to this bit when the instrument triggers a service request.
7	Data sum bit of the operation status register. The bit can be set if the event bit of the operation status register and the corresponding enable bit are set to 1. This bit indicates that the instrument executes an operation, and the specific operation type can be obtained by querying the operation status register.

2) Event status register (ESR) and events status enable register (ESE)

IEEE488.2 defines ESR. The command "*ESR?" may be used to read the event status register (ESR). ESE belongs to the enable part of SCPI register. If one of the bits is 1 and one of the bits in the response ESR changes from 0 to 1, the ESB bit of STB should be set to 1. The command "*ESE" may be used to set ESE, and the command "*ESE?" used to read ESE.

Table 2.7 Description of event status byte

Bit	Meaning
0	Operation completed Set to this bit when the previous commands have been executed and the command *OPC has been received.
1	Not used.

2	Query error Set to this bit when the controller reads the instrument data without sending the query command, or sends a new command before reading the query data. It indicates that there is a query error, for which the query cannot be executed.
3	Instrument error Set to this bit when there is an instrument error. Error code range: -300 to -399, or positive error code. Specific errors can be found in relevant information in the error queue.
4	Execution error Set to this bit when a syntactically correct command is received but cannot be executed, and an error with code ranging from -200 to -300 is generated in the error queue.
5	Command error Set to this bit when the syntax of the command received is incorrect. Error code range: -100 to -200. Specific errors can be found in relevant information in the error queue.
6	User request Set to this bit when the instrument is switched to manual control mode.
7	Power ON Set to this bit when the instrument power is turned on.

3) Status: inquiry register

The register contains instrument status that does not meet specification requirements. The register value may be queried through the command "STAT:QUES:COND" or "STAT:QUES:EVEN". The register is described in Table 2.8 below.

Table 2.8 Description of status: inquiry register

Bit	Meaning
0-2	Not used.
3	If the local power setting is wrong, set this bit.
4	If the time base is not hot, set this bit.
5	If the frequency of local oscillator or the reference frequency of any active channel is wrong, set this bit.
6	Not used.
7	If the local modulation setting is wrong, set this bit.
8	If the instrument is uncalibrated (the “Uncalibrated” prompt message is displayed on the interface), set this bit.
9	In case of self-test error, set this bit.
10-14	Not used
15	The bit is always 0.

Tip

Query register

Status: the inquiry register has collected the information of all lower sub-registers (for example, bit2 has collected all time related information). Since each path corresponds to a separate sub-register, in case of a status bit error of the inquiry register, it is

required to go back to the sub-register of the path to check for the specific error source. By default, the sub-register status being queried belongs to the currently selected path.

4) STAT: QUES: FREQ register

The information about local oscillator and reference frequency is stored in this register. Each active channel corresponds to a separate frequency register. The value of this register can be read via the command "STATus:QUEStionable:FREQuency:CONDition?" or "STATus:QUEStionable: FREQuency [:EVENt]?".

The register is described in Table 2.9 below.

Table 2.9 Description of status: inquiry: frequency register

Bit	Meaning
0	Not used
1	Local oscillator unlocked Set to this bit when the local oscillator is out of lock. Meanwhile, the prompt "LO UNL" will be displayed on the user interface.
2..7	Not used.
8	External reference Set to this bit when an external reference oscillator is set but no available external reference signal is actually connected. In such case, the frequency synthesizer is out of lock, and the frequency accuracy is low.
9..14	Not used.
15	The bit is always 0.

5) STAT: QUES: POW register

The register contains information about power overload when operating the instrument. Each active path corresponds to a separate power register. The register value may be read by using the command

"STATus:QUEStionable: POWer:CONDition?" or "STATus:QUEStionable:POWer [:EVENt]?. POWer [:EVENt]?".

The register is described in Table 2.10 below.

Table 2.10 Description of status: inquiry: power register

Bit	Meaning
0	Overload If the power of RF input signal is overloaded, set this bit. This will cause signal distortion without damaging the instrument, and at the same time, the user interface will show the prompt message "RF overload".
3	Input overload If the signal power of RF input connector is out of the rated range, set this bit. At the same time, the user interface shows the prompt message "Input overload". The RF input and the input mixer are separated to protect the instrument. In case of repeated measurement, reduce the power level of RF input connector.
4..14	Not used.
15	The bit is always 0.

2.1.5.4 Application of Status Reporting System

The status reporting system is used to monitor the status of one or more instruments in a test system. In order to correctly realize the function of the status reporting system, the controller in the test system must receive and evaluate the information of all instruments. Standard methods used include:

- Service request (SRQ) initiated by the instrument;
- Serial query of all instruments in the bus system, initiated by the controller in the system, in order to find the initiator of the service request and the reason.
- Perform parallel query of all instruments;
- Program command to query the status of specific instruments;
- Query of error queue.

1) Service request

In some cases, the instrument sends a service request (SRQ) to the controller to obtain the controller's service, and the controller initiates an interrupt to enter the corresponding interrupt handler. According to Figure 2.4, an SRQ is typically initiated by one or more status bytes and by bits 2, 3, 4, 5 or 7 of the related enable register (SRE). These bits, in turn, make up advanced registers, error queues or output buffers. In order to use all the service requests as far as possible, all bits in enable registers SRE and ESE should be set to 1.

Example: When the command "*OPC" is used after sweep, the SRQ signal will be generated.

a. Call the function InstrWrite to write the command "*ESE 1", and set to ESE bit0 (operation completed).

b. Call the function InstrWrite to write the command "*SRE 32", and set to SRE bit5 (ESB).

c. Call the function InstrWrite to write the command "*INIT;*OPC", and SRQ is generated after the operation is completed.

The instrument will generate an SRQ after settings are completed.

SRQ can only be initiated by the instrument. In case of an instrument error, the controller program should allow a service request to be made to the instrument and handled by a dedicated interrupt service program. Please refer to Section 4.2.1.1. "Service Request" for specific routines.

2) Serial query

Similar to the command *STB, serial query is used to query the status byte of the instrument. Serial query adopts the method of interface message, so the query speed is fast. IEEE 488.2 defines the specific method for serial query. The method is mainly used to quickly obtain the status of one or more instruments connected with the controller in the test system.

3) Parallel query

In the test system, the controller sends an information bit to the data cable through a command, and can query 8 instruments at the same time. The data configured on the data cable of the instrument is a logical "0" or "1". In addition to the conditions under which the SRE register determines the SRQ to be generated, the bits of parallel poll enable register (PPE) and STB register should be subject to AND operation. The result obtained is sent to the controller of parallel query as the response result after OR operation and NOT operation, or the result may be obtained through the command *IST.

In parallel query, first the instrument should be set to the parallel query status through the command PPC, which allocates one data cable to the instrument and determines whether the bit is reversed in response. The PPE register is used when executing parallel query. Parallel query is mainly used for the controller to quickly locate which instrument has sent the service request. Therefore, the same values should be set for the registers SRE and PPE.

4) Instrument status query

The following two commands may be used to query each part of the status register:

- Command * IDN? is used to query the advanced register;
- The status system command is used to query the SCPI register (for example: STATus:QUEStionable…).

The returned value of the queried register is usually in decimal format, which is used by the controller program for detection. In order to obtain a more detailed description of the SRQ cause, the parallel query will be carried out after the SRQ generally.

Description of response data bit

The STB and ESR registers contain 8 bits, and the SCPI register contains 16 bits. The returned value of the query status register is in decimal format. The decimal value is equal to the sum of each bit and respective weight.

The relationship between the bit and the weight is shown in the figure below:

Data Bit	7	6	5	4	3	2	1	0
Weight	128	64	32	16	8	4	2	1

Figure 2.5 Relationship between the bit and the weight

5) Error Queue

Each error status of the instrument corresponds to an entry in the error queue, which contains a specific error message text that can be viewed through the error log or queried through the program command: SYSTem:ERRor[:NEXT]?. If there is no error in the error queue, the query returns 0, "No error".

The error queue should be queried in the controller service request handler because a more accurate description of the cause of the error can be obtained than in the status register. Especially in the test phase of the controller program, the error queue should be frequently queried to clarify the error command record sent by the controller to the instrument.

2.1.5.5 Reset Status Reporting System

Commands and events for the reset status reporting system are listed below. In addition to the commands *RST and SYSTem:PRESet, other commands will not change the function settings of the instrument. Similarly, DCL will not change the set state of the instrument. Details are shown in the table below:

Table 2.11 Reset status reporting system

Function Event	Power ON/OFF (Powered status cleared)		DCL, SDC (Instrument cleared, instrument selected to be cleared)	*RST or SYSTem: PRESet	STATus: PRESet	*CLS
	0	1				
Clear STB, ESR	—	Yes	—	—	—	Yes
Clear SRE, ESE	—	Yes	—	—	—	—

Clear PPE	—	Yes	—	—	—	—
Clear the event part of the register	—	Yes	—	—	—	Yes
Clear the enable part of the operation and inquiry registers. Fill the enable part of other registers with 1.	—	Yes	—	—	Yes	—
Fill the positive transition part with 1. Clear the negative transition part.	—	Yes	—	—	Yes	—
Clear the error queue	Yes	Yes	—	—	—	Yes
Clear the output buffer	Yes	Yes	Yes	—	—	—
Clear the command processing and input buffers	Yes	Yes	Yes	—	—	—

2.1.6 Programming Precautions

1) Please initialize the instrument status first before changing the setting

During remote instrument setting, initialize the instrument status (for instance, to send " *RST") first before realizing required status setting.

2) Command sequence

Generally, the setting command and query command must be sent separately. Otherwise, the returned value of the query command will vary with current operation sequence of the instrument.

3) Failure response

The service request can only be initiated by the instrument itself. The master program in the test system shall instruct the instrument to initiate the service request actively in case of any failure, and then enter corresponding service termination program for handling.

4) Error queue

When handling the service request each time, the master program shall query the instrument error queue rather than the status register, so as to obtain the error cause with higher accuracy. Especially during the test phase of the master program, it shall often query the error queue, so as to obtain the error command sent by the master program to the instrument.

2.2 Remote Control and its Configuration

2.2.1 LAN

SICL-LAN is used to control S1465 series signal generator in Local Area Network (LAN).

Note

Operation of the USB master port connector on the front panel

The type-A connector is used as the USB master port connector on the front panel, and the port is used to connect with the flash disk of the USB port in the S1465 series signal generators for upgrading the built-in software of the instrument, which can also be connected to the USB keyboard and mouse for controlling the signal generator. This port cannot be used for remote control of the instrument.

2.2.1.1 Connection Establishment

Connect the S1465 signal generator and the external master program (the computer) to the LAN via a network cable, as shown in Figure 2.6.

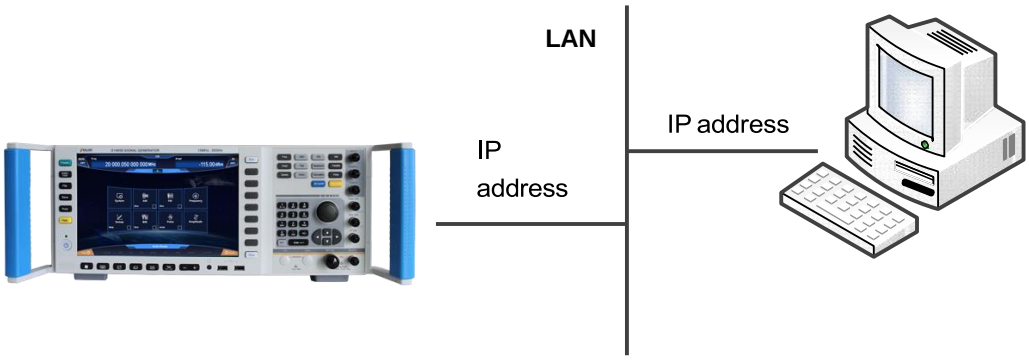

Figure 2.6 LAN interface connection diagram

2.2.1.2 Interface Configuration

Physical connection of the network should be guaranteed for remote control on the signal generator via the LAN. Because DHCP, domain name access and wide area network connection are not supported, the network program setting of signal generator is relatively simple.

Click [system] → [LAN Port] to go to the interface shown in Figure 2.7. Set "IP address", "Subnet mask" and "Default gateway" to the subnet where the master controller is located.

Figure 2.7 LAN interface setting

Note

Make sure the signal generator are in normal physical connection via 10Base-T LAN or 100Base-T LAN cables.

Since the signal generator supports only the setup of single LAN control system and setting of static IP address rather than DHCP or host access via DNS or domain name server, the user does not need to modify the subnet mask, which will be set to 255.255.255.0 by default in the instrument.

2.2.2 GPIB

2.2.2.1 Connection Establishment

Connect the S1465 signal generator to the external master program (the computer) via the GPIB cable, as shown in Figure 2.8.

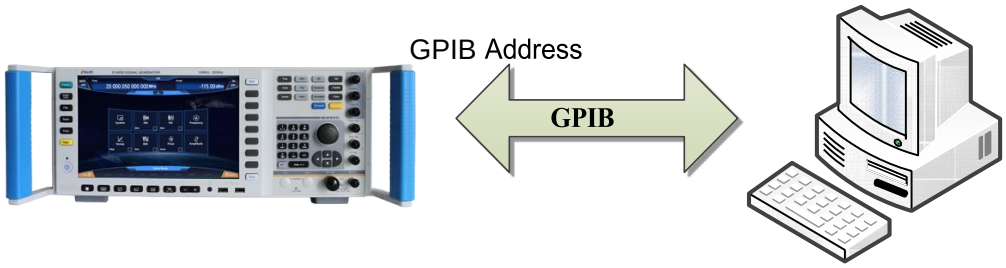

Figure 2.8 GPIB interface connection diagram

2.2.2.2 Interface Configuration

Users may need to modify the GPIB address when using the signal generator to establish the system. The default GPIB address of the machine is 19. Methods for modification of the GPIB address are described below:

Click [system] → [GPIB Interface] to enter the interface shown in Fig. 2.9, so as to make changes in the local GPIB address input box by using the numeric keys on the front panel.

Figure 2.9 GPIB interface setting

2.3 I/O Library

2.3.1 I/O Library Overview

I/O library is the software program library programmed for the instrument in advance, namely, the instrument driver, which is the software intermediate layer between the computer and the instrument hardware. It consists of the function library, utility program, tool suite and is the set of series software code modules, which is corresponding to a planned operation, such as instrument configuration, reading from the instrument, writing into the instrument and triggering the instrument. It remains in the computer, acting as the bridge and link between the computer and instrument. With high-level modular libraries facilitating programming, users do not need to learn complex lower-level programming protocol for specific instrument any more. Instrument driver is the key for rapid test and measurement application development.

From the aspect of function, a general instrument driver usually consists of a functional body, an interactive developer interface, a program developer interface, a subprogram interface and an I/O interface as shown in Fig. 2.10.

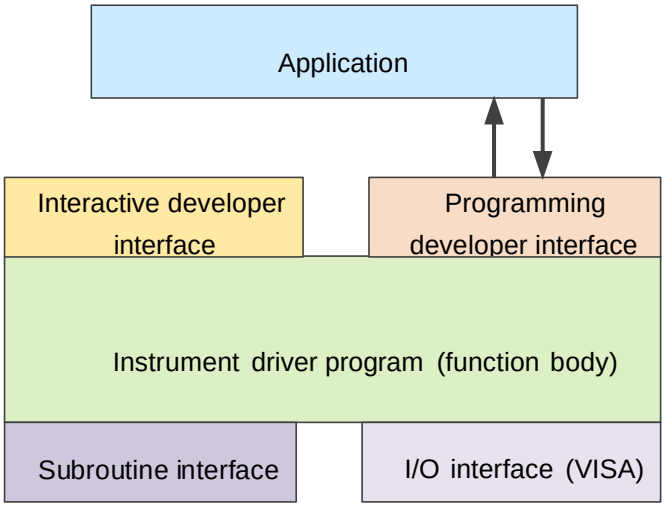

Figure 2.10 Instrument driver structure model

The detailed descriptions are shown below:

1) Function body. It is the main function part of the instrument driver, which can be interpreted as the framework program of the instrument driver.
2) Interactive developer interface. The application development environment supporting instrument driver development always provides graphical interactive developer interface for user convenience. For example, the function panel in Labwindows/CVI is a type of interactive developer interface. In the function panel, each parameter of the instrument driver functions is expressed in the form of graphical control.
3) Programming developer interface. It is the software interface used by the application program for loading the instrument driver function, for example, the dynamic link library file .dll of the instrument driver in the Windows system.
4) I/O interface. It is used to realize actual communication between the instrument driver and instrument. The bus-specific I/O software (such as GPIB) or the generally-standardized I/O software applied cross multiple buses (VISA I/O) can be used.
5) Subroutine interface. It is the software interface used by the instrument driver to access other supporting libraries, such as the database and FFT function. The instrument driver will use the subroutine interface to call other software module, operating system, program control code library and analysis function library, so as to complete relevant functions.

2.3.2 I/O Library Installation and Configuration

With the development of the test field application from the traditional instrument to the virtual instrument, the instrument driver has experienced different development processes in order to solve the instrument interchangeability and test program reusability of the automatic test system. Currently, the IVI (Interchangeable Virtual Instruments) driver is popularly applied. Based on the IVI specification, a new instrument programming interface is defined, the class driver and VPP architecture are inserted onto the VISA so that the test application is completely independent of the instrument hardware, and unique instrument simulation, range detection and status buffer functions are added, which improves the system operation efficiency and truly achieves the instrument interchange.

IVI driver consists of two types, namely, IVI-C and IVI-COM. IVI-COM is based on the Microsoft component object model (COM) technology and adopts the COM API mode; IVI-C is based on ANSI C and adopts the C API mode. Such two drive types are designed by following the instrument type specified in the IVI specification with the same application development environment, including Visual Studio, Visual Basic, Agilent VEE, LabVIEW and CVI/ LabWindows.

Currently it is necessary to provide two types of drivers in order to meet the demands of different users in different development environments. The IVI driver of the signal generator uses Nimbus Driver Studio to produce IVI-COM and IVI-C drivers as well as program installation package. For specific installation and configuration, please refer to documents accompanied with the control card and I/O Library of your choice.

The IVI drivers installed are divided into: IVI intrinsic function group and instrument class function group (basic function group and extended function group). For details about functional classification, functions and attributes, please refer to the accompanied help document of the driver.

Tip

Configuration port and I/O library installation

When a computer is to be used to control the signal generator, please confirm that the necessary ports and I/O library have been properly installed and configured.

Tip

Use of I/O library

Once installed, the attached IVI-COM/C driver installation package will automatically install the driver function panel, help documents, and sample programs of the driver functions to facilitate the users to develop and integrate the program control functions.

3 SCPI

The specific contents are as follows:

- **Command instruction**
- **Common commands**
- **Instrument subsystem command**

3.1 Command Instruction

This chapter provides detailed command reference information for remote control, including:

- Complete syntax format and parameter list;
- Syntax diagram for non-standard SCPI;
- Detailed function description and related command description;
- Supported command formats (settings or queries);
- Parameter description, including: data type, value range and default value (unit);
- Key path;
- Model of instrument in the same class of instrument that is compatible with the command. If not specified, it indicates that the current command only applies to S1465.
- Other instructions.

.In the sections about common commands and instrument subsystem commands, the commands in sequence are listed first for convenient query and use.

3.2 Common Commands

Common commands are used to control instrument status registers, status reports, synchronization, data storage and other common functions. The use and function of common commands apply to different instruments. All common commands may be identified by the first "*" in the command word, and are defined in detail in IEEE488.2.

The followings are the explanation and instruction of the common commands in IEEE488.2.

- *IDN?
- *RCL
- *RST
- *SAV
- *TRG

Tip

Command use:

Unless otherwise specified, commands may be used for setting or query.

The condition that one command is only used for setting or querying, or initiating an event, if any, will be indicated separately in the command instruction.

***IDN?**

Function: This command is used for returning to instrument identification.

Return value: <ID>: "manufacturer, <instrument model>, <serial number>, <firmware version number>"

Example: CETC41,1465,2017008,1.0.11

Note: for query only.

***RCL <Value>**

Function: This command is used for recalling instrument status from the designated internal register of the signal generator.

Parameter: range [0, 99].

Note: for setting only.

***RST**

Function: This command is used for accomplishing reset function of the signal generator.

Note: for setting only.

***SAV <Value>**

Function: This command is used for storing the current status of the signal generator in the designated internal register.

Parameter: range [0, 99].

Note: for setting only.

***TRG**

Function: This command is used for selecting the bus in the signal generator as the trigger source.

Note: for setting only.

3.3 Instrument Subsystem Command

This section details the subsystem commands of S1465 series signal generator.

- OUTPut
- FREQuency
- POWer
- PHASe
- LIST
- LFOutput
- SWEep
- PULM
- AMPLitude Modulation

- FREQuency Modulation
- PHASe Modulation
- Digital Modulation
- MEMory
- ROSCillator
- SYSTem

3.3.1 OUTPut Subsystem

The output subsystem command is used to control the state of RF output signal.

The following commands are used to select the operating mode, including

- :OUTPut:BLANking[:STATe]
- OUTPut[:STATe]
- OUTPut:MODulation[:STATe]

:OUTPut:BLANking[:STATe] <State>

Function: This command sets the state of RF Blank. When RF Blank is turned on, if the signal generator is in spot frequency state, RF output signal will be turned off during frequency switching; if the signal generator is in sweep state, RF output signal will be turned off during frequency band switching and RTC.

Setting format: OUTPut:BLANking[:STATe] ON|OFF|1|0

Query format: OUTPut:BLANking[:STATe]?

Parameter:

<State> Boolean data is as follows:

ON | 1: Blank is turned on

OFF | 0: Blank is turned off.

Example: OUTPut:BLANking 1

The Preceding example sets the signal generator Blank to ON state.

Reset status: 0

:OUTPut[:STATe] <State>

Function description: This command is used to enable the RF output of the signal generator.

Setting format: :OUTPut[:STATe] ON|OFF|1|0

Query format: :OUTPut[:STATe] ?

Parameter Description:

<State> Boolean data, which is taken as follows:

ON | 1: RF output

OFF | 0: RF OFF.

Example: OUTPut 1

The preceding example sets RF output of the signal generator.

Reset state: 0

Key Entry: 【RF ON/OFF】

:OUTPut:MODulation[:STATe] <State>

Function description: This command is used for setting the state of the modulation main switch.

Setting format: OUTPut:MODulation[:STATe] ON|OFF|1|0

Query format: OUTPut:MODulation[:STATe]?

Parameter Description:

<State> Boolean data, which is taken as follows:

ON | 1: Modulation ON

OFF | 0: modulation OFF.

Example: OUTPut:MODulation 1 set the modulation state to ON.

Reset state: 0

Key Entry: 【Modulation ON/OFF】

3.3.2 FREQuency Subsystem

The frequency subsystem command is used to control the frequency common function of the RF output signal.

The following commands are used to select the operating mode, including:

- [:SOURce]:FREQuency[:CW|FIXed]
- [:SOURce]:FREQuency:MODE
- [:SOURce]:FREQuency:MULTiplier
- [:SOURce]:FREQuency:OFFSet
- [:SOURce]:FREQuency:REFerence
- [:SOURce]:FREQuency:REFerence:STATe
- [:SOURce]:FREQuency:STEP
- [:SOURce]:FREQuency:STARt
- [:SOURce]:FREQuency:STOP
- [:SOURce]:FREQuency[:CW|FIXed]:AUTO

[:SOURce]:FREQuency[:CW] <Frequency>

Function: This command is used for setting the output frequency of the signal generator in CW mode. For setting other frequency modes, please refer to the command ":FREQuency:MODE".

Setting format: [:SOURce]:FREQuency[:CW|FIXed] <val>

Query format: [:SOURce]:FREQuency[:CW|FIXed]?

Parameter:

<Frequency> Output frequency in CW mode. Model Range

S1465A [100kHz~3GHz]
S1465B [100kHz~6GHz]
S1465C [100kHz~10GHz]
S1465D [100kHz~20GHz]
S1465E [100kHz~40GHz]
S1465H [100kHz~50GHz]
S1465L [100kHz~67GHz]

Example: [:SOURce]:FREQuency[:CW|FIXed] 10GHz Set the frequency of the signal generator to 10 GHz.

Reset state: Freq Start + (Freq Stop - Freq Start)/2

Key Entry: 【Frequency】—>[CW]

[:SOURce]:FREQuency:MODE <Mode>

Function: This command sets the frequency mode of the signal generator.

Setting format: [:SOURce]:FREQuency:MODE FIXed|CW|SWEep|LIST

Query format: [:SOURce]:FREQuency:MODE?

Parameter:

<Mode> Discrete data, Current Sweep Type to be configured. The values are as follows:

FIXed|CW The setting of these two discrete parameters is the same in the signal generator in meaning, that is, the controlled signal generator outputs CW (spot frequency) signal, which will terminate the frequency sweep signal currently output by the instrument.

For setting spot frequency, please refer to the commands ":FREQuency[:CW]"and ":FREQuency[:FIXed]".

SWEep This parameter sets current frequency mode as sweep mode. Sweep mode in the software, whether step sweep or ramp sweep, is determined by: "SWEep:GENeration". The default is step sweep.

LIST Current Sweep Type is set to List mode. If the current list is empty, the signal generator will prompt the list is empty. The signal generator can start sweep only when there is at least one sweep point in the list.

Example: :FREQuency:MODE LIST Set the signal generator to List Sweep mode.

Reset status: CW

[:SOURce]:FREQuency:MULTiplier <FreqMult>

Function: This command sets Freq Mul for signal source. When set Freq Mul is greater than 1, a multiplier indicator "*" is displayed in frequency display area. At this time, displayed frequency value= RF output

frequency value ×multiplier. However, the actual frequency output is still the frequency with no multiplier added. When Freq Mul is set to 1, the indicator disappears.

Setting format: [:SOURce]:FREQuency:MULTiplier <val>

Query format: [:SOURce]:FREQuency:MULTiplier?

Parameter:

<FreqMult> Freq Mul.

Range: 1[1，36].

Example: :FREQuency: MULTiplier 8 Freq Mul of the signal generator is 8.

Reset status: 1

Key Entry: 【Frequency】—>[Basic Config]->[Freq Mul]

[:SOURce]:FREQuency:OFFSet <FreqOffs>

Function: When Frequency Offset is not set to zero, an offset indicator "*" is displayed in frequency display area, and the display value becomes the frequency value plus offset. At this time, the displayed frequency value= RF output frequency ×multiplier+ frequency offset. However, the actual frequency output is still the frequency with no multiplier and frequency offset added. When Freq Offset is set to zero, the indicator disappears.

Setting format: [:SOURce]:FREQuency:OFFSet <val>

Query format: [:SOURce]:FREQuency:OFFSet?

Parameter:

<FreqOffs> Freq Offset

Range: 0Hz [-325GHz, +325GHz].

Example: :FREQuency:OFFSetr 10GHz Freq Offset of the signal generator is 10 MHz.

Reset status: 0Hz

Key Entry: 【Frequency】—>[Basic Config]->[Freq Offset]

[:SOURce]:FREQuency:REFerence <FreqRef>

Function: This command is used for setting the frequency reference function. The set value can function well when the frequency reference switch is on. See the command ":FREQuency:REFerence:STATe". Frequency reference indicator "*" is displayed in the frequency display area. In this case, the frequency reference value will be subtracted from the set CW output signal. For example: The current CW output frequency is 1 GHz. If Freq Ref is set to 1 GHz, the CW output frequency displayed at this time will be based on 0 Hz frequency reference, so frequency display area will display 1 MHz; if CW frequency is set to 1 MHz, frequency display area will display 1 MHz and the actual output frequency is 1.001 GHz.

Setting format: [:SOURce]:FREQuency:REFerence <val>

Query format: [:SOURce]:FREQuency:REFerence?

Parameter:

<FreqRef> Freq Ref.

Model Range

S1465A [0Hz~3GHz]
S1465B [0Hz~6GHz]
S1465C [0Hz~10GHz]
S1465D [0Hz~20GHz]
S1465E [0Hz~40GHz]
S1465H [0Hz~50GHz]
S1465L [0Hz~67GHz]

Example: :FREQuency:REFerence 10GHz Set the relative frequency of the signal generator to 10 GHz.

Reset status: 0Hz

Key Entry: 【Frequency】—>[Base Config]->[Freq Ref]

[:SOURce]:FREQuency:REFerence:STATe <State>

Function: This command sets Freq Ref Switch to ON or OFF state. After the frequency reference switch is ON, when the CW frequency of the signal generator is changed, the frequency value displayed in the frequency display area is based on this frequency reference. For setting frequency reference, please see the command ":FREQuency:REFerence"; when the switch is OFF, the frequency value displayed in the frequency display area is the actual CW frequency of the signal generator.

Setting format: [:SOURce]:FREQuency:REFerence:STATe ON|OFF|1|0

Query format: [:SOURce]:FREQuency:REFerence:STATe?

Parameter:

<State> Boolean data is as follows:

ON | 1: Freq Ref Switch is turned on,

OFF | 0: Freq Ref Switch is turned off.

Example: :FREQuency:REFerence:STATe 1 Freq Ref of the signal generator is turned on.

Reset status: 0

Key Entry: 【Frequency】—>[Basic Config]->[Freq Ref Switch]

[:SOURce]:FREQuency:STEP <FreqStep>

Function: Set the step value of the frequency. After setting this value, for setting CW frequency of the signal generator, please see the command ":FREQuency:CW" or ":FREQuency:FIXed". When changing the frequency by UP|DOWN, the frequency variation will change based on the current set step value.

Setting format: [:SOURce]:FREQuency:STEP <val>

Query format: [:SOURce]:FREQuency:STEP?

Parameter:

<FreqStep> Freq Step.

Model Range

S1465A [0Hz~2.9999GHz]
S1465B [0Hz~5.9999GHz]
S1465C [0Hz~9.9999GHz]
S1465D [0Hz~19.9999GHz]
S1465E [0Hz~39.9999GHz]
S1465H [0Hz~49.9999GHz]
S1465L [0Hz~66.9999GHz]

Example: :FREQuency: STEP 1MHz Freq Step of the signal generator is 1 MHz.

Reset status: 100MHz

Key Entry: 【Frequency】—>[Basic Config]->[Freq Step]

[:SOURce]:FREQuency:STARt <StartFreq>

Function: This command sets Freq Start of Step Sweep. See the command ":FREQuency:STOP".

Setting format: [:SOURce]:FREQuency:STARt <val>

Query format: [:SOURce]:FREQuency:STARt?

Parameter:

<StartFreq> Freq Start of sweep.

Model Range

S1465A [100kHz~3GHz]
S1465B [100kHz~6GHz]
S1465C [100kHz~10GHz]
S1465D [100kHz~20GHz]
S1465E [100kHz~40GHz]
S1465H [100kHz~50GHz]
S1465L [100kHz~67GHz]

Example: :FREQuency:STARt 1MHz Freq Start of Step/Analog Sweep of the signal generator is 1 MHz.

Reset status: 100kHz

Key Entry: 【Sweep】—>[Step Sweep]/[List Sweep]-> [Freq Start]

[:SOURce]:FREQuency:STOP <StopFreq>

Function: This command sets Freq Stop of Step Sweep. See the command "FREQuency:STARt".

Setting format: [:SOURce]:FREQuency:STOP <val>

Query format: [:SOURce]:FREQuency:STOP?

Parameter:

<StopFreq> Freq Stop of sweep.

Model Range

S1465A [100kHz~3GHz]
S1465B [100kHz~6GHz]

S1465C [100kHz~10GHz]
S1465D [100kHz~20GHz]
S1465E [100kHz~40GHz]
S1465H [100kHz~50GHz]
S1465L [100kHz~67GHz]

Example: :FREQuency:STOP 100MHz Freq Stop of Step/Analog Sweep of the signal generator is 100 MHz.

Reset status: It is related to the model. For 67GHz model, it is 67GHz.

Key Entry: 【Sweep】—>[Step Sweep]/[List Sweep]-> [Freq Stop]

[:SOURce]:FREQuency[:CW|FIXed]:AUTO <State>

Function: This command is used for set frequency follow-up. When this function is enabled, the output CW frequency of the signal generator will change according to the editing of corresponding frequency value in the internal user flatness list. See the commands ":CORRection:FLATness:PAIR", ":CORRection:FLATness:FILL:STARt" and ":CORRection:FLATness:FILL:STOP".

Setting format: [:SOURce]:FREQuency[:CW|FIXed]:AUTO ON|OFF|1|0

Query format: [:SOURce]:FREQuency[:CW|FIXed]:AUTO?

Parameter:

<State> Boolean data is as follows:

ON | 1: Frequency follow-up ON,

OFF | 0: Frequency follow-up OFF.

Example: [:SOURce]:FREQuency[:CW|FIXed]:AUTO 1 Frequency follow-up of the signal generator is ON.

Reset status: 0

Key Entry: 【Calibration】—>[Power Accuracy Calibration] —>[Frequency Follow-up]

3.3.3 POWer Subsystem

The power subsystem command is used for controlling the general functions of RF output signal power level.

The following commands are used for setting operation modes:

- [:SOURce]:POWer:ALC:LEVel
- [:SOURce]:POWer:ALC:SEARch
- [:SOURce]:POWer:ALC:SOURce
- [:SOURce]:POWer:ALC:SOURce:EXTernal:COUPling
- [:SOURce]:POWer:ALC[:STATe]
- [:SOURce]:POWer:ATTenuation
- [:SOURce]:POWer:ATTenuation:AUTO
- [:SOURce]:POWer[:LEVel][:IMMediate][:AMPLitude]
- [:SOURce]:POWer[:LEVel][:IMMediate]:OFFSet

- [:SOURce]:POWer:REFerence
- [:SOURce]:POWer:REFerence:STATe
- [:SOURce]:POWer:STEP
- [:SOURce]:POWer:ALC:BANDwidth|BWIDth
- [:SOURce]:POWer:ALC:BANDwidth|BWIDth:AUTo

[:SOURce]:POWer:ALC:LEVel <AlcLevel>

Function: This command sets ALC level when Attenuation Style is set to Manual mode. For selection of manual/automatic mode of attenuator, see the command ":POWer:ATTenuation:AUTO".

Setting format: [:SOURce]:POWer:ALC:LEVel <value>

Query format: [:SOURce]:POWer:ALC:LEVel?

Parameter:

<AlcLevel> ALC level.

Range: 0dBm[-20dBm, +30dBm].

Example: :POWer:ALC:LEVel 5dBm ALC level is 5 dBm.

Reset status: 0dBm

Key Entry: 【Amplitude】—>[Atten Config]—>[ALC]

[:SOURce]:POWer:ALC:SEARch <Mode>

Function: This command activates or deactivates the internal amplitude auto search of the signal generator when ALC loop is turned on. Amplitude search will enable amplitude to stabilize the signal generator on the output amplitude selected by users and keep the internal modulator in drive state when ALC Loop is turned off. For ALC loop state, see the command ":POWer:ALC[:STATe]".

Setting format: [:SOURce]:POWer:ALC:SEARch ON|OFF|1|0|ONCE

Query format: [:SOURce]:POWer:ALC:SEARch?

Parameter:

<Mode> Discrete data. Auto amplitude search state is as follows:

OFF | 0: This command terminates Auto mode, and Search Style is Manual.

ON | 1: Amplitude is automatically searched with the change of RF output amplitude or frequency. Search Style is then in Auto mode

ONCE Do Search once at current RF output frequency.

Example: :POWer:ALC:SEARch 1 Search Style is Auto.

Reset status: Manual

Key Entry: 【Amplitude】—> [ALC Loop Config]—>[Search Style Auto Manual]/[Search Out]

[:SOURce]:POWer:ALC:SOURce <Mode>

Function: This command allows users to select ALC Level Control for the signal generator as appropriate,

including INT, EXT and Source Module.

Setting format: [:SOURce]:POWer:ALC:SOURce INTernal|DIODe

Query format: [:SOURce]:POWer:ALC:SOURce?

Parameter:

<State > Discrete data. Level Control is as follows:

INTernal: Level Control is INT,

DIODe external diode detector level control,

Example: :POWer:ALC:SEARch:REFerence INT Level Control is INT.

Reset status: Internal

Key Entry: 【Amplitude】—> [Level Control]->[Level Control]

[:SOURce]:POWer:ALC:SOURce:EXTernal:COUPling <CouplingValue>

Function: This command sets EXT Detector Couple. When Level Control is EXT diode detector, this command enables you to set the coupling value for external level control. For level control, see the command ":POWer:ALC:SOURce".

Setting format: [:SOURce]:POWer:ALC:SOURce:EXTernal:COUPling <value>

Query format: [:SOURce]:POWer:ALC:SOURce:EXTernal:COUPling?

Parameter:

<CouplingValue> EXT Detector Couple

Range: 16dBm [-90dBm， +90dBm].

Example: :POWer:ALC:SOURce:EXTernal:COUPling 16dBm Ext Detector Couple is 16dBm.

Reset status: 16.00dBm

Key Entry: 【Amplitude】—> [Level Control]->[Ext Detector Couple]

[:SOURce]:POWer:ALC[:STATe] <State>

Function: This command enables you to turn ALC loop ON or OFF. ALC Loop is mainly to correct amplitude drift and to make the signal generator output amplitude level not change with time and temperature.

Setting format: [:SOURce]:POWer:ALC[:STATe] ON|OFF|1|0

Query format: [:SOURce]:POWer:ALC[:STATe]?

Parameter:

<State> Boolean data is as follows:

ON | 1: ALC Loop is turned off,

OFF | 0: ACL Loop is turned on.

Example: :POWer:ALC 1 This example indicates ALC Loop state is set as ALC-OFF.

Reset status: 1

Key Entry: 【Amplitude】—> [ALC Loop Config]—>[ALC Loop State ALC-OFF ALC-ON]

[:SOURce]:POWer:ATTenuation <Atten>

Function: This command is used for setting power amplitude attenuation value of the mechanical attenuator of the signal generator. The value set by this command will be valid only when the attenuator is set to Manual. For setting auto/manual mode of attenuator, see the command ":POWer:ATTenuation:AUTO".

The minimum attenuation step set by this command is 5 dB, that is, users can only set attenuation to 0 dB, 5 dB, 10 dB, 15 dB by step of 5 dB. After Attenuation is set, the signal generator output amplitude is the current ALC minus the currently set attenuation.

Setting format: [:SOURce]:POWer:ATTenuation <value>

Query format: [:SOURce]:POWer:ATTenuation?

Parameter:

<Atten> Amplitude attenuation.

Range: 115dB[0dB，115dB].

Example: :POWer:ATTenuation 15dB Attenuation is 15 dB.

Reset status: 115dB

Key Entry: 【Amplitude】—>[Atten Config]—>[Attenuation]

[:SOURce]:POWer:ATTenuation:AUTO <State>

Function: This command sets the control state of internal programmable step attenuator: Auto or Manual. In Auto mode, the signal generator automatically sets the value of amplitude attenuator based on the current output amplitude; in Manual mode, the current amplitude attenuation of attenuator does not change during the process of changing amplitude output level.

Setting format: [:SOURce]:POWer:ATTenuation:AUTO ON|OFF|1|0

Query format: [:SOURce]:POWer:ATTenuation:AUTO?

Parameter:

<State> Boolean data is as follows:

ON | 1: Attenuation is AUTO,

OFF | 0: Attenuation is Manual.

Example: :POWer:ATTenuation:AUTO 0 Attenuation Style is Manual.

Reset status: 1

Key Entry: 【Amplitude】—>[Attenuation Config]—>[Attenuation Style Auto Manual]

[:SOURce]:POWer[:LEVel][:IMMediate][:AMPLitude] <Ampl>

Function: This command sets the output amplitude level of the signal generator.

Setting format: [:SOURce]:POWer[:LEVel][:IMMediate][:AMPLitude] <value>

Query format: [:SOURce]:POWer[:LEVel][:IMMediate][:AMPLitude]?

Parameter:

<Ampl> Amplitude level value.

Range: -135dBm [-135dBm， +30dBm].

Example: :POWer 0dBm The output amplitude level is 0 dBm.

Reset status: -115dBm

Key Entry: 【Amplitude】

[:SOURce]:POWer[:LEVel][:IMMediate]:OFFSet <PowOffset>

Function: The command is the actual output amplitude offset of the signal generator. When the value is non-zero, an offset indicator "*" is displayed in amplitude display area and the displayed amplitude value is the actual amplitude power plus the amplitude offset. Such amplitude offset does not change the actual output amplitude of the signal generator. Only the displayed amplitude value is changed.

Setting format: [:SOURce]:POWer[:LEVel][:IMMediate]:OFFSet <value>

Query format: [:SOURce]:POWer[:LEVel][:IMMediate]:OFFSet?

Parameter:

<PowOffset> Ampl Offset.

Range: 0dB [-100dB， +100dB].

Example: :POWer:OFFS -10dB Ampl Offset is -10 dB.

Reset status: 0dB

Key Entry: 【Amplitude】—>[Basic Config]->[Ampl Offset]

[:SOURce]:POWer:REFerence <PowRef>

Function: Ampl Ref can be set when Ampl Ref Switch is turned on. For the state of power reference switch, see the command ":POWer:REFerence:STATe". When Ampl Ref Switch is turned on, an indicator "*" is displayed in amplitude display area and the displayed amplitude value = actual output amplitude - amplitude reference value.

For example: When current CW output power is 1dBm, if the power reference is set to 1dBm, the displayed CW output power is based on this power reference. Therefore, the power display area shows 0dBm, and the actual output frequency of the signal generaor remains 1dBm.

Setting format: [:SOURce]:POWer:REFerence <value>

Query format: [:SOURce]:POWer:REFerence?

Parameter:

<PowRef> Ampl Ref.

Range: 0dBm [-135dBm， +30dBm].

Example: :POWer:REFerence -10dBm Ampl Ref is -10 dBm.

Reset status: 0dBm

Key Entry: 【Amplitude】—>[Basic Config]->[Ampl Ref]

[:SOURce]:POWer:REFerence:STATe <State>

Function: This command sets the state of Ampl Ref Switch. When the power reference switch is ON and the power

reference value is not zero, if the power level of the signal generator is changed, the power value displayed in the power display area will be based on this power reference. For setting power reference, see the command ":POWer:REFerence"; when the power reference switch is OFF, the power value displayed in the power display area will be the actual CW output power of the signal generator.

Setting format: [:SOURce]:POWer:REFerence:STATe ON|OFF|1|0

Query format: [:SOURce]:POWer:REFerence:STATe?

Parameter:

<State> Boolean data is as follows:

ON | 1: Ampl Ref is turned on,

OFF | 0: Ampl Ref is turned off.

Example: :POWer:REFerence:STATe 1 Ampl Ref is turned on.

Reset status: 0

Key Entry: 【Amplitude】—>[Basic Config]->[Ampl Ref Switch ON OFF]

[:SOURce]:POWer:STEP <PowStep>

Function: Set the step value of the power. After setting this value, for setting CW power of the signal generator, see the command ":POWer[:LEVel][:IMMediate][:AMPLitude]". When changing the power by UP|DOWN, the power variation will change based on the variation of the current set step value.

Setting format: [:SOURce]:POWer:STEP <value>

Query format: [:SOURce]:POWer:STEP?

Parameter:

<PowStep> Ampl Ref.

Range: 0.10dB [0.01dB， 20dB].

Example: :POWer:STEP 1dB Ampl Step is 1 dB.

Reset status: 0.10dB

Key Entry: 【Amplitude】—>[Basic Config]->[Ampl Step]

[:SOURce]:POWer:ALC:BANDwidth|BWIDth<AlcBandWidth>

Function: This command sets ALC (automatic leveling control) Band which enables the signal generator to output different frequency bands. There are four ALC loop bandwidth settings in different states, 100 Hz, 1 kHz, 10 kHz and 100 kHz.

Notes:

1. When ALC Band is set to Auto, see the commands ":POWer:ALC:BANDwidth:AUTO" and ":POWer:ALC:BANDwidth:AUTO"; when it is improperly set, this setting is invalid;

2. When the internal baseband of the instrument is turned on, ACL Band is invalid, and the appropriate bandwidth should to be selected by baseband. For details, please refer to Section "5.2.5 Baseband"

of the User Manual of S1465 series Microwave Synthesized Signal Generator.

Setting format: [:SOURce]:POWer:ALC:BANDwidth|BWIDth 100Hz|1kHZ|10kHz|100kHz

Query format: [:SOURce]:POWer:ALC:BANDwidth|BWIDth?

Parameter:

<AlcBandWidth > Discrete data. ALC Band is as follows:

100Hz | 0: ALC Band is 100 Hz,

1kHz | 1: ALC Band is 1 kHz,

10kHz | 2: ALC Band is 10 kHz,

100kHz | 3: ALC Band is 100 kHz.

Example: [:SOURce]:POWer:ALC:BANDwidth|BWIDth 100Hz

ALC loop bandwidth is 100Hz.

Reset status: 10kHz

Key Entry: 【Amplitude】—>[ALC Band]

[:SOURce]:POWer:ALC:BANDwidth|BWIDth:AUTO <State>

Function: This command is used for setting ALC(automatic leveling control) loop bandwidth mode. At auto mode, the signal generator selects proper ALC loop bandwidth automatically; at manual mode, ALC loop bandwidth is set by the user. For details, see the command

"[:SOURce]:POWer:ALC:BANDwidth|BWIDth ".

Setting format: [:SOURce]:POWer:ALC:BANDwidth|BWIDth:AUTO ON|OFF|1|0

Query format: [:SOURce]:POWer:ALC:BANDwidth|BWIDth:AUTO?

Parameter:

<State> Boolean data is as follows:

ON | 1: ALC Band is in Auto mode,

OFF | 0: ALC Band is in Manu mode.

Example: [:SOURce]:POWer:ALC:BANDwidth|BWIDth:AUTO 1 ALC bandwidth mode is set to Manual.

Reset status: 1

Key Entry: 【Amplitude】—>[ALC Band Manual Auto]

3.3.4 LIST Subsystem

The LISTsubsystem command is used for setting the list sweep function of RF output signal. The commands and parameters of the subsystem are:

The following commands are used for setting operation modes:

- [:SOURce]:LIST:DIRection
- [:SOURce]:LIST:DWELl

- [:SOURce]:LIST:FREQuency
- [:SOURce]:LIST:FILL:POINts
- [:SOURce]:LIST:FILL:STARt
- [:SOURce]:LIST:FILL:STOP
- [:SOURce]:LIST:POWer
- [:SOURce]:LIST:RETRace
- [:SOURce]:LIST:TRIGger:SOURce
- [:SOURce]:LIST:FILL:POWer
- [:SOURce]:LIST:FILL:DWELl
- [:SOURce]:LIST:FILL:EXECute
- [:SOURce]:LIST:DELete

[:SOURce]:LIST:DIRection <Direc>

Function: This command sets Step Sweep Indirection. Users can select both Forward and Backward modes: Forward indicates sweep from the first point to the last point in list, and Backward indicates sweep from the last point to the first point in current list.

Setting format: [:SOURce]:LIST:DIRection UP|DOWN

Query format: [:SOURce]:LIST:DIRection?

Parameter:

<Direc> Discrete data. Step Sweep Indirection is as follows:

UP Start Forward sweep from the first point in list,

DOWN Start Backward sweep from the last point in list.

Example: :LIST:DIRection UP Set Step Sweep Indirection to Forward.

Reset status: UP

Key Entry: 【Sweep】—>[List Sweep]—>[Step Sweep Indirection Forward Backward]

[:SOURce]:LIST:DWELl <Val>{,{Val}}

Function: This command sets the dwell time of each sweep point in current list. If users need to set different dwell time, it is necessary to enter the corresponding dwell time for each point in list, i.e. the dwell time parameter value of list sweep points in turn, separated by commas. If the number of points input by users is less than that of the current list points, the number of points where the dwell time is not entered uses the current default.

Setting format: [:SOURce]:LIST:DWELl <val>{,{val}}

Query format: [:SOURce]:LIST:DWELl?

Parameter:

<Val> Dwell Time of List Sweep Points.

Range: 1ms [1ms， 60s].

Example: :LIST:DWELI 30ms， 20ms

Set the dwell time of the first point in list to 30 ms, and the second point to 20ms.

Key Entry: 【Sweep】—>[List Sweep]—>[Edit List...]—>[Time(ms)]

[:SOURce]:LIST:FREQuency <Val>{,{Val}}

Function: This command is used for setting the CW frequency of each sweep point in the current list. If the user needs to set different frequency value, every frequency point must be set with the corresponding frequency value. To this end, it is only necessary to input the frequency values of the sweep points successively with a comma between values. If the number of points input by the user is less than that of the current list, the points without list frequency will take the current default value.

Setting format: [:SOURce]:LIST:FREQuency <val>{,{val}}

Query format: [:SOURce]:LIST:FREQuency?

Parameter:

<Val> Frequency of list sweep point.

Model Range

S1465A [100kHz~3GHz]
S1465B [100kHz~6GHz]
S1465C [100kHz~10GHz]
S1465D [100kHz~20GHz]
S1465E [100kHz~40GHz]
S1465H [100kHz~50GHz]
S1465L [100kHz~67GHz]

Example: :LIST:FREQuency The CW frequency in 300MHz， 1GHz， 500MHz lists are set to 300MHz, 1GHz, 500MHz successively.

Key Entry: 【Sweep】—>[List Sweep]—>[Edit List...]—>[Freq(MHz)]

[:SOURce]:LIST:FILL:POINts <Num>

Function: After this command is set, frequency point will be inserted according to Freq Start, Freq Stop and Equal Freq Interval. It should be noted that, if you don‘t want the frequency points in the list to increase or decrease in sequence, please don't use this command. For related commands, see “:LIST:FILL:STARt”,”:LIST:FILL:STOP”.

Setting format: [:SOURce]:LIST:FILL:POINts <num>

Query format: [:SOURce]:LIST:FILL:POINts?

Parameter:

<Num> Insert Counts

Range: 3[2， 801].

Example: :LIST:FILL:POINts 100 Set frequency points to 100 in list.

Reset status: 3

Key Entry: 【Sweep】—>[List Sweep]—>[Insert Counts]

[:SOURce]:LIST:FILL:STARt <FreqStart>

Function: This command sets Freq Start of List Sweep, which is used with Freq Stop and Insert Counts to automatically generate list sweep points. For setting the stop frequency and sweep points of the list, see the commands" :LIST:FILL:STOP", ":LIST:FILL:POINts", ":LIST:FILLL:POWer", "LIST:FILL:DWELI".

Setting format: [:SOURce]:LIST:FILL:STARt <val>

Query format: [:SOURce]:LIST:FILL:STARt?

Parameter:

<FreqStart> Freq Start of List Sweep.

Model Range

S1465A [100kHz~3GHz]
S1465B [100kHz~6GHz]
S1465C [100kHz~10GHz]
S1465D [100kHz~20GHz]
S1465E [100kHz~40GHz]
S1465H [100kHz~50GHz]
S1465L [100kHz~67GHz]

Example: :LIST:FILL:STARt 300MHz Set Freq Start of List Sweep to 300 MHz.

Key Entry: 【Sweep】—>[List Sweep]-> [Freq Start]

[:SOURce]:LIST:FILL:STOP <FreqStop>

Function: This command is used for setting the stop frequency for list sweep. It is used together with the start frequency and number of points of the list to generate the list sweep point automatically. For setting start frequency and number of sweep points of the list, see the commands

":LIST:FILL:STARt", ":LIST:FILL:POINts", ":LIST:FILLL:POWer", "LIST:FILL:DWELI".

Setting format: [:SOURce]:LIST:FILL:STOP <val>

Query format: [:SOURce]:LIST:FILL:STOP?

Parameter:

<FreqStop> Freq Stop of List Sweep.

Model Range

S1465A [100kHz~3GHz]
S1465B [100kHz~6GHz]
S1465C [100kHz~10GHz]
S1465D [100kHz~20GHz]
S1465E [100kHz~40GHz]
S1465H [100kHz~50GHz]
S1465L [100kHz~67GHz]

Example: :LIST:FILL:STOP 1GHz Set Freq Stop of List Sweep to 1 GHz.

Key Entry: 【Sweep】—>[List Sweep]-> [Freq Stop]

[:SOURce]:LIST:POWer <Val>{,{Val}}

Function: This command sets the amplitude of each sweep point in current list. If users need to set different offset for each list point, it is necessary to enter the corresponding offset value for each point in list, i.e. the amplitude offset of list sweep points in turn, separated by commas. If the number of points input by users is less than that of the current list points, the number of points where the list power offset is not entered uses the current default.

Setting format: [:SOURce]:LIST:POWer <val>{,{val}}

Query format: [:SOURce]:LIST:POWer?

Parameter:

<Val> Ampl Offset of List Sweep Points.

Range: 0dBm [-100dB， +100dB].

Example: :LIST:POWer 1dB，0.2dB，1.3dB，2.5dB，-3.6dB

Set the power offset in the list to 1dB，0.2dB，1.3dB，2.5dB，-3.6dB successively.

Key Entry: 【Sweep】—>[List Sweep]—>[Edit List...]—>[Offset(dB)]

[:SOURce]:LIST:RETRace <State>

Function: This command is used for setting RTC switch. After accomplishing the single list sweep, the output frequency of the signal generator stays on the first or the last point of the list. This command is only available in the single sweep mode.

Setting format: [:SOURce]:LIST:RETRace ON|OFF|1|0

Query format: [:SOURce]:LIST:RETRace?

Parameter:

<State> Boolean data is as follows:

ON | 1: RTC Switch ON; after accomplishing sweep, the output frequency stays on the first frequency point of the list.

OFF | 0: RTC Switch OFF; after accomplishing sweep, the output frequency stays on the last frequency point of the list.

Example: :LIST:RETRace 0 RTC Switch OFF; after accomplishing the single list sweep, the output CW frequency of the signal generator will stay on the last frequency point of the list.

Reset status: 0

Key Entry: 【Sweep】—>[Sweep Mode]->[RTC Switch ON OFF]

[:SOURce]:LIST:TRIGger:SOURce <Source>

Function: This command is used for setting the trigger source for initiating list sweep. The trigger sources are available in four modes, namely, Auto, Bus, Ext and Key. For related command, see the command

"TRIGger[:SEQuence]:SOURce".

Setting format: [:SOURce]:LIST:TRIGger:SOURce IMMediate|BUS|EXTernal|KEY

Query format: [:SOURce]:LIST:TRIGger:SOURce?

Parameter:

<Source> Discrete data. List Trig is as follows:

IMMediate, the trigger signal is always true, when a sweep is completed, the systme automatically triggers the next sweep.

BUS Bus, the trigger source is from GPIB group execute trigger, or triggered by"*TRG" command.

EXTernal Ext, the trigger signal is from the trigger input connector on the rear panel.

KEY Key, the trigger signal is from the trigger key on the front panel.

Example: :LIST:TRIGger:SOURce BUS Set the trigger source for list sweep to Bus.

Reset status: IMM

Key Entry: 【Sweep】—>[List Sweep]—>[List Trig]

[:SOURce]:LIST:FIII:POWer <Val>

Function: This command is used for setting the list sweep power offset. It is used together with the start frequency and number of points of the list to generate the list sweep point. For setting start frequency and number of sweep points of the list, see the commands ":LIST:FILL:STARt", ":LIST:FILL:POINts", ":LIST:FILLL:STOP", "LIST:FILL:DWELI"

Setting format: [:SOURce]:LIST:FILL:POWer <val>

Query format: [:SOURce]:LIST:FILL:POWer?

Parameter:

<val> Power offset for all list sweep points.

Range: 0dBm [-100dB，+100dB]

Example: :LIST:FILL:POWer 10dB Set the list sweep power offset to 10dB.

Key Entry: 【Sweep】—> [List Sweep]—>[All List Ampl Offset]

[:SOURce]:LIST:FIII:DWELI <Val>

Function: This command is used for setting the dwell time for all list sweep points. It is used together with the start frequency and number of points of the list to generate the list sweep point automatically. For setting start frequency and number of sweep points of the list, see the commands ":LIST:FILL:STARt", ":LIST:FILL:POINts", ":LIST:FILLL:STOP", "LIST:FILL:POWer"..

Setting format: [:SOURce]:LIST:FILL:DWELI <val>

Query format: [:SOURce]:LIST:FILL:DWELI?

Parameter:

<val> Dwell time for all list sweep points.

Range: 1ms [1ms，60s].

Example: :LIST:FILL:POWer 10ms Set the dwell time for all list sweep points to 10ms.

Key Entry: 【Sweep】—> [List sweep]—>[All List Dwell Time]

[:SOURce]:LIST:FIll:DWELl <Val>

Function: This command is used for setting the dwell time for all list sweep points. It is used together with the start frequency and number of points of the list to generate the list sweep point automatically. For setting start frequency and number of sweep points of the list, see the commands ":LIST:FILL:STARt","LIST:FILL:POINts","LIST:FILLL:STOP","LIST:FILL:POWer".

Setting format: [:SOURce]:LIST:FILL:DWELl <val>

Query format: [:SOURce]:LIST:FILL:DWELl?

Parameter:

<val> Dwell time for all list sweep points. Range:

1ms [1ms，60s].

Example: :LIST:FILL:POWer 10ms Set the dwell time for all list sweep points to 10ms.

Key Entry: 【Sweep】—> [List sweep]—>[All List Dwell Time]

[:SOURce]:LIST:FIll:EXECute

Function: This command is used for generating list sweep points according to set start frequency, stop frequency, number of points of list, power offset for all points and dwell time for all points; see the commands ":LIST:FILL:STARt", ":LIST:FILL:POINts", ":LIST:FILLL:STOP", "LIST:FILL:POWer", "LIST:FILL:STOP"

Setting format: [:SOURce]:LIST:FILL:DWELl <val>

Example: :LIST:FILL:EXECute Accomplish automatic filling of list.

Key Entry: 【Sweep】—>[List Sweep]—>[Auto Fill]

Note: For setting only.

[:SOURce]:LIST:DELete <Mode>

Function: This command is used for deleting the points in the list of List Sweep.

Setting format: [:SOURce]:LIST:FILL:EXECute

Parameter: Discrete data. Options for deleting the list points are:

CURRent: Current point

ALL: All points

Example: :LIST:DELete ALL Delete all points in the list.

Key Entry: 【Sweep】—>[List Sweep]—>[Edit List]—>[Del All]

Note: For setting only.

3.3.5 LFOutput Subsystem

The following commands are used for setting operation modes:

- [:SOURce]:LFOutput:AMPLitude

- [:SOURce]:LFOutput:FREQuency
- [:SOURce]:LFOutput:FREQuency:ALTernate
- [:SOURce]:LFOutput:FREQuency:ALTernate:AMPLitude:PERCent
- [:SOURce]:LFOutput:NOISe
- [:SOURce]:LFOutput:RAMP
- [:SOURce]:LFOutput:SHAPe
- [:SOURce]:LFOutput:STATe
- [:SOURce]:LFOutput:SWEep:TIME

[:SOURce]:LFOutput:AMPLitude <Ampl>

Function: This command is used for setting the output signal amplitude from the LF output BNC (Bayonet Nut Connector) of the signal generator

Setting format: [:SOURce]:LFOutput:AMPLitude <val>(unit:VPP|Mvpp|VRMS)

Query format: [:SOURce]:LFOutput:AMPLitude?

Parameter:

<Ampl> LF output signal amplitude.

Range: 2.000Vpp[0.002Vpp,4.000Vpp].

Example: :LFOutput:AMPLitude 1Vpp LF output signal amplitude is set to 1 Vpp.

Reset status: 2.000Vpp

Key Entry: 【Frequency】—>[LF Out]—>[LF Ampl]

[:SOURce]:LFOutput:FREQuency <Frequency

Function: This command is used for setting LF output. It should be noted that, when LF Waveform is set to SweepSinc or DualSinc, this command will set the start frequency of sweep sine and the frequency 1 of dual sine. For setting LF waveform, see the command "LFOutput:SHAPe".

Setting format: [:SOURce]:LFOutput:FREQuency <val>

Query format: [:SOURce]:LFOutput:FREQuency?

Parameter:

<Frequency> LF output signal frequency.

Range: 400Hz[0.01Hz， 10MHz].

Freq Start of Sweep Sin

Range: 400Hz[0.01Hz,9.99999999MHz]. When inputting 10MHz, there will be no error indication in the software, and the start frequency of sweep sine will be set to 9.99999999MHz automatically.

DualSinc Frequency1

Range: 400Hz[0.01Hz,10MHz].

Example: :LFOutput:FREQuency 1MHz Set LF output signal frequency to 1 MHz.

Reset status: 400Hz

Key Entry:

- 【Frequency】—>[LF Out]—>[LF]
- 【Frequency】—>[Sweep Sinc]—>[Freq Start]
- 【Frequency】—>[Dual Sinc>>—>[Frequency1]

[:SOURce]:LFOutput:FREQuency:ALTernate <Frequency>

Function: This command is used for setting the stop frequency of sweep sine or the frequency 2 of dual sine when LF Waveform is set to SweepSinc or DualSinc. For setting LF waveform, see the command "LFOutput:SHAPe". If LF Waveform is not set to SweepSinc or DualSinc, the value of frequency 2 of dual sine will be modified as default.

Setting format: [:SOURce]:LFOutput:FREQuency:ALTernate <val>

Query format: [:SOURce]:LFOutput:FREQuency:ALTernate?

Parameter:

<Frequency> Frequency2 of dual sine.

Range: 400Hz[0.01Hz，10MHz].

Stop frequency of sweep sine or

Range: 1MHz[0.02Hz，10MHz]. If the stop frequency of sweep sine is set to 0.01Hz, there will be no error indication in the software, and the stop frequency of sweep sine will be set to 0.02Hz automatically.

Example: :LFOutput:FREQuency:ALTernate 1MHz Set the stop frequency of sweep sine or the frequency 2 of dual sine to 1MHz.

Reset status: 400Hz

Key Entry:

- 【Frequency】—>[Sweep Sinc]—>[Freq Stop]
- 【Frequency】—>[Dual Sinc]—>[Frequency2]

[:SOURce]:LFOutput:FREQuency:ALTernate:AMPLitude:PERCent <Percent>

Function: This command is used for setting the percentage of amplitude of second tone to that of total output signal in dual sine waveform when LF Waveform is set to DualSinc; for example, if the second tone accounts for 20% of the total waveform power, the first tone will account for 80% of the total power output.

Setting format: [:SOURce]:LFOutput:FREQuency:ALTernate:AMPLitude:PRECent <val>

Query format: [:SOURce]:LFOutput:FREQuency:ALTernate:AMPLitude:PRECent?

Parameter:

<Percent> Amplitude percentage of frequency 2 of dual sine.

Range: 50 [0，100].

Example: :LFOutput:FREQuency:ALTernate:AMPLitude:PRECent 20

Set the percentage of second waveform of dual sine to the total signal output power to 20%.

Reset status: 50%

Key Entry: 【Frequency】—> [Dual Sinc]—>[Freq 2 Ampl Percent]

[:SOURce]:LFOutput:NOISe <Mode>

Function: This command is used for setting signal output types when LF Waveform is set to White noise, which includes: While noise and Gauss. For setting LF waveform, see the command ":LFOutput:SHAPe".

Setting format: [:SOURce]:LFOutput: NOISe UNIForm|GAUSsian

Query format: [:SOURce]:LFOutput: NOISe?

Parameter:

<Mode> Discrete data. Options for type of LF noise signal are:

UNIForm White noise,

GAUSsian Gauss.

Example: :LFOutput: NOISe GAUSsian LF noise is of Gauss type.

Reset status: UNIF

Key Entry: 【Frequency】—>[LF Out]—>[LF Waveform>>]—>[White noise]

[:SOURce]:LFOutput:RAMP <Mode>

Function: This command is used for setting signal output types when LF Waveform is set as Zigzag, which include Zigzag-up and Zigzag-down. For setting LF waveform, see the command —:LFOutput:SHAPe ‖.

Setting format: [:SOURce]:LFOutput:RAMP POSitive|NEGative

Query format: [:SOURce]:LFOutput:RAMP?

Parameter:

<Mode> Discrete data. Options for type of Zigzag signal are: POSitive

Zigzag-up,

NEGative Zigzag-down.

Example: :LFOutput:RAMP NEGative LF Zigzag signal is of Zigzag-down type.

Reset status: POS

Key Entry: 【Frequency】—>[LF Out]—>[LF Waveform]—>[Zigzag>>]

[:SOURce]:LFOutput:SHAPe <Mode>

Function: This command sets LF signal output waveform. There are seven waveforms for selection: Sinc, Square, Triangle, Zigzag, White noise, SweepSinc and DualSinc.

Setting format: [:SOURce]:LFOutput:SHAPe SINE|SQUare|TRIangle|RAMP
|NOISe|SWEPtsine|DUALsine

Query format: [:SOURce]:LFOutput:SHAPe?

Parameter:

<Mode> Discrete data. LF Waveform is as follows: SINE Sine,

SQUare Square, TRIangle Triangle, RAMP Zigzag, NOISe Noise, SWEPtsine SweepSinc, DUALsine DualSinc.

Example: :LFOutput:SHAPe TRIangle LF waveform of the signal generator is triangle.

Reset status: SINE

Key Entry:【Frequency】—>[LF Out]—>[LF Waveform>>]

[:SOURce]:LFOutput:STATe <State>

Function: This command is used for setting LF Output of the signal generator.

Setting format: [:SOURce]:LFOutput:STATe ON|OFF|1|0

Query format: [:SOURce]:LFOutput:STATe?

Parameter:

<State> Boolean data is as follows:

ON | 1: LF Output is turned on, and LF signal output is turned on accordingly. OFF | 0: LF Output is turned off, and LF signal output is turned off accordingly.

Example: :LFOutput:STATe OFF Turn off LF signal output.

Reset status: 0

Key Entry: 【Frequency】—>[LF Out]—>[LF Output ON OFF]

[:SOURce]:LFOutput:SWEep:TIME <Time>

Function: This command is used for setting sweep time when LF Waveform is set to SweepSinc.

Setting format: [:SOURce]:LFOutput:SWEep:TIME <val>

Query format: [:SOURce]:LFOutput:SWEep:TIME?

Parameter:

<Time> Sweep time of sweep sine.

Range: 10ms[100us，2s].

Example: :LFOutput:SWEep:TIME 200ms Sweep time of sweep sine is set to 200ms.

Reset status: 10ms

Key Entry: 【Frequency】—>[Sweep Sinc>>]—>[Sweep Time]

3.3.6 SWEep Subsystem

SWEep subsystem commands are used for controlling the sweep frequency functions of RF output signal. The commands and parameters of the subsystem are:

The following commands are used for setting operation modes:

- [:SOURce]:SWEep:DIRection

- [:SOURce]:SWEep:DWELl
- [:SOURce]:SWEep:POINts
- [:SOURce]:SWEep:STEP
- [:SOURce]:SWEep:TIME:AUTO
- [:SOURce]:SWEep:TIME
- [:SOURce]:SWEep:GENeration
- [:SOURce]:SWEep:TRIGger:SOURce

[:SOURce]:SWEep:DIRection <Direction>

Function: This command sets Step Sweep Indirection, including: Forward and Backward. Forward indicates Step Sweep from Freq Start to Freq Stop, and Backward indicates sweep from Freq Stop to Freq Start.

Setting format: [:SOURce]:SWEep:DIRection UP|DOWN

Query format: [:SOURce]:SWEep:DIRection?

Parameter:

<Direction> Discrete data. Step Sweep Indirection is as follows:

UP : Forward,

DOWN : Backward.

Example: :SWEep:DIRection DOWN Step sweep direction is set as backward.

Reset status: UP

Key Entry: 【Sweep】—>[Sweep Mode>>]—>[Step Sweep>>]—>[Step Sweep Indirection Forward Backward]

[:SOURce]:SWEep:DWELl <DwellTime>

Function: This command is used for setting the dwell time of step sweep which refers to the pause time during the sweep process of current step frequency points. The dwell time set by the user will be valid when the trigger source of step sweep is set as Auto. For setting trigger source, see the command "SWEep:TRIGger:SOURce".

Setting format: [:SOURce]:SWEep:DWELl <value>

Query format: [:SOURce]:SWEep:DWELl?

Parameter:

<Val> Step dwell.

Range: 10.000ms[1ms， 60s].

Example: :SWEep:DWELl 1s Set All Step Dwell Time to 1 s.

Reset status: 10.000ms

Key Entry: 【Sweep】—>[Step Sweep]—>[Step Dwell]

[:SOURce]:SWEep:POINts <Num>

Function: This command sets the current step counts.

Setting format: [:SOURce]:SWEep:POINts <val>

Query format: [:SOURce]:SWEep:POINts?

Parameter:

<Num> Step Counts

Range: 11[2，801].

Example: :SWEep:POINts 101 Set Step Counts to 101.

Reset status: 11

Key Entry: 【Sweep】—>[Step Sweep]—>[Step Counts]

[:SOURce]:SWEep:STEP <FreqStep>

Function: This command is used for setting the frequency step of step sweep. After the start and stop frequency of step sweep as well as the step value are set by the user, step sweep will start according to the step value. For setting start and stop frequency of step sweep, see the commands ":FREQuecny:STARt", ":FREQuenct:STOP" .

Setting format: [:SOURce]:SWEep:STEP <value>

Query format: [:SOURce]:SWEep:STEP?

Parameter:

<FreqStep> Frequency step of step sweep.

Range: (Stop frequency – Start frequency)/ (Max. step points – 1) — (Stop frequency – Start frequency) / (Min. step points – 1).

Example: :SWEep:STEP 1MHz Set Step Sweep interval to 1MHz.

Reset status: (Stop frequency – Start frequency)/ (Min. step points - 1).

Key Entry: None

Caution

Frequency step of step sweep

To ensure equispaced step points, the set frequency step value will generally be slightly adjusted. It will not be adjusted only when it can be divided exactly by the sweep width.

[:SOURce]:SWEep:TIME:AUTO <state>

Function: This command is used for setting ramp sweep time type

Setting format: [:SOURce]:SWEep:TIME:AUTO ON|OFF|1|0

Query format: [:SOURce]:SWEep:TIME:AUTO?

Parameter:

<State> Boolean data is as follows:

ON | 1: Sweep time type is set to Auto.

OFF | 0: Sweep time type is set to Manual.

Example: :SWEep:TIME:AUTO 1 Set sweep time type to Auto

Reset status: 1

Key Entry: 【Sweep】—>[Ramp Sweep] —>[Sweep Dwell Time Type] —>[Auto Manual]

[:SOURce]:SWEep:TIME <state>

Function: When ramp sweep time type is set to Manual, this command is used for setting sweep time

Setting format: [:SOURce]:SWEep:TIME <val>

Query format: [:SOURce]:SWEep:TIME?

Parameter:

<Time> Sweep time.

Range: 100ms[100ms， 200s].

Example: :SWEep:TIME 200ms Set sweep time to Auto

Reset status: 2346.667ms

Key Entry: 【Sweep】—>[Ramp Sweep] —>[Sweep Dwell Time]

[:SOURce]:SWEep:GENeration <Mode>

Function: When sweep mode is set, this command is used for setting sweep mode

Setting format: [:SOURce]:SWEep:GENeration ANALog|STEPped

Query format: [:SOURce]:SWEep:GENeration?

Parameter:

<Mode> Discrete data. Options for sweep mode are: ANALog

Ramp Sweep,

STEPped Step Sweep.

Example: :SWEep:GENeration STEPped Set sweep mode to step sweep.

Reset status: STEP

Key Entry: None

[:SOURce]:SWEep:TRIGger:SOURce <Mode>

Function: This command sets Step Trig. It includes Auto, Bus, Ext and Key. For related command, see the command "TRIGger[:SEQuence]:SOURce".

Setting format: [:SOURce]:SWEep:TRIGger:SOURce IMMediate|BUS|EXTernal|KEY

Query format: [:SOURce]:SWEep:TRIGger:SOURce?

Parameter:

<Mode> Discrete data. Step Trig is as follows:

IMMediate Auto, the trigger signal is always true, when a sweep is completed, the system automatically triggers the next sweep.

BUS Bus, the trigger source is from GPIB group execute trigger, or triggered by *TRG command.

EXTernal Ext, the trigger signal is from the trigger input connector on the rear panel.

KEY |Trigger key; trigger source comes from the trigger key of the front panel.

Example: :SWEep: TRIGger:SOURce BUS Set Step Trig to Bus mode.

Reset status: IMM

Key Entry: 【Sweep】—>[Step Sweep]—>[Step Trig]

3.3.7 PULM Subsystem

PULM subsystem commands are used for controlling the pulse modulation functions of RF output signal. The commands and parameters of the subsystem are:

The following commands are used for setting operation modes:

- [:SOURce]:PULM:EXTernal:POLarity
- [:SOURce]:PULM:INTernal:DELay
- [:SOURce]:PULM:INTernal:FREQuency
- [:SOURce]:PULM:INTernal:PERiod
- [:SOURce]:PULM:INTernal:PWIDth
- [:SOURce]:PULM:SOURce
- [:SOURce]:PULM:STATe
- [:SOURce]:PULM:INTernal:JITTered:MODE
- [:SOURce]:PULM:INTernal:JITTered:PERCent
- [:SOURce]:PULM:INTernal:PTRain:DATA
- [:SOURce]:PULM:INTernal:PTRain:DELete
- [:SOURce]:PULM:INTernal:PTRain:POINts
- [:SOURce]:PULM:INTernal:PTRain:PRESet
- [:SOURce]:PULM:INTernal:SLIDing:STEP
- [:SOURce]:PULM:INTernal:SLIDing:POINts
- [:SOURce]:PULM:INTernal:STAGger:INSert
- [:SOURce]:PULM:INTernal:STAGger:POINts
- [:SOURce]:PULM:INTernal:STAGger:DELete
- [:SOURce]:PULM:INTernal:STAGger:PRESet
- [:SOURce]:PULM:LFM:BWIDth
- [:SOURce]:PULM:LFM:DIRection
- [:SOURce]:PULM:LFM:STATe

[:SOURce]:PULM:EXTernal:POLarity <Mode>

Function: The command logically inverts the external input pulse signal, i.e. when Pulse Source is EXT, the pulse signal input from pulse input port on the front panel of the signal generator is TTL high level signal or is inverted to TTL low level signal. For selecting pulse source, see the command ":PULM:SOURce".

Setting format: [:SOURce]:PULM:EXTernal:POLarity INVerted|NORMal

Query format: [:SOURce]:PULM:EXTernal:POLarity?

Parameter:

<Mode> Discrete data. Input Direction is as follows:

NORMal Input Direction is turned off and the input pulse signal is TTL high level. INVerted Input Direction is turned on and the input pulse signal is TTL low level.

Example: :PULM:ENTernal:POLarity INV External input pulse signal is inverted to TTL low level.

Reset status: NORM

Key Entry: 【Pulse】—>[Basic Config]—>[Input Direction ON OFF]

[:SOURce]:PULM:INTernal:DELay <DelayTime>

Function: This command is used for setting the pulse delay of the pulse modulation. The actual settable maximum delay depends on the current pulse period set by the user. In addition, it should be noted that, the set pulse delay is valid only when the pulse source is set to Auto, Square, D-Pulse or Trig. And when it is set to Trig, there is 100ns inherent delay in the pulse delay. For related commands, see the commands ":PULM:INTernal:PERiod", ":PULM:SOURce".

Setting format: [:SOURce]:PULM:INTernal:DELay <val>

Query format: [:SOURce]:PULM:INTernal:DELay?

Parameter:

<DelayTime> Pulse delay time of pulse modulation.

Non-Trig Mode: 0s[0ns，42.000000000s], Trig Mode: 0s[100ns，42.000000000s].

Example: :PULM:INTernal:DELay 1ms Set pulse delay to 1 ms

Reset status: 0s

Key Entry: 【Pulse】—>[Basic Config]—>[Delay]

[:SOURce]:PULM:INTernal:FREQuency <Frequency>

Function: This command sets PRF of pulse modulation. When the pulse source is set to Square, the duty ratio of the pulse signal output is 50%. This command may be used for changing the frequency of the square signal. For setting pulse source, see the command —:PULM:SOURce ‖.

Setting format: [:SOURce]:PULM:INTernal:FREQuency <val>

Query format: [:SOURce]:PULM:INTernal:FREQuency?

Parameter:

<Frequency> PRF of pulse modulation.

Range: 1kHz [0.023Hz， 25MHz]

Example: :PULM:INTernal:FREQuency 1MHz Set PRF to 1 MHz.

Reset: 1kHz

Key Entry: 【Pulse】—>[Basic Config]—>[PRF]

[:SOURce]:PULM:INTernal:PERiod <Period>

Function: This command sets the period of pulse signal generated inside the signal generator. If the set period is smaller than or equal to the current pulse width, the pulse width will be automatically adjusted to be smaller than pulse period. For setting pulse width, see the command ":PULM:INTernal:PWIDth".

Setting format: [:SOURce]:PULM:INTernal:PERiod <value>

Query format: [:SOURce]:PULM:INTernal:PERiod?

Parameter:

<Percent> Pulse period.

Range: 1.000000ms[40ns， 42.000000000s].

Example: :PULM:INTernal:PERiod 10ms Pulse signal period is 10 ms.

Reset status: 1.000000ms

Key Entry: 【Pulse】—>[Basic Config]—>[Period]

[:SOURce]:PULM:INTernal:PWIDth <PWidth>

Function: This command sets Width of pulse signal generated inside the signal generator. If the set pulse width is greater than or equal to the current pulse period, the pulse width will be automatically adjusted to be smaller than the current pulse period. Besides, if the set pulse width is less than 1 us, it is recommended to execute power search function. When pulse source is set to Staggered, the pulse width in staggered list will be a unified value, and should also be changed through this command.

Setting format: [:SOURce]:PULM:INTernal:PWIDth <val>

Query format: [:SOURce]:PULM:INTernal:PWIDth?

Parameter:

<PWidth> Pulse signal width.

Range: 50.000us [20ns， 41.999999990s].

Example: :PULM:INTernal:PWIDth 10us Set the pulse signal width to 10 us.

Reset status: 50.000us

Key Entry: 【Pulse】—>[Basic Config]—>[Width]

[:SOURce]:PULM:SOURce <Mode>

Function: This command sets Source of Pulse Modulation, including: EXT, Scalar, Auto, Square, D-Pulse, Pulse Train, Gate, Trig, Jittered, Staggered and Sliding. In Scalar mode, it is not allowed to change associated pulse parameters, and the signal generator will automatically output pulse signal of 18-microsecond pulse

width and 36-microsecond period.

Setting format: [:SOURce]:PULM:SOURce EXTernal|SCALar|INTernal|SQUare|DOUBlet |PTRain|GATEd|TRIGgered|JITTered|STAGger|SLIDing

Query format: [:SOURce]:PULM:SOURce?

Parameter:

<Mode> Discrete data. Source is as follows:

EXTernal	Source is EXT,
SCALar	Source is Scalar and outputs 27.8 kHz square wave,
INTernal	Pulse Source is Auto,
SQUare	Source is Square.
DOUBlet	Source is D-Pulse.
PTRain	Source is Pulse Train.
GATEd	Source is Gate.
TRIGgered	Auto mode is activated. In this mode, the period is the external sync pulse period, and the pulse width is the local pulse width setting.
JITTered	Source is Jittered.
STAGger	Source is Staggered.
SLIDing	Source is Sliding.

Example: :PULM:SOURce SQUare Set Source to Square.

Reset status: INT

Key Entry: 【Pulse】—>[Basic Config]—>[Source]

[:SOURce]:PULM:STATe <State>

Function: This command sets whether to output the pulse modulation signal of the signal generator.

Setting format: [:SOURce]:PULM:STATe ON|OFF|1|0

Query format: [:SOURce]:PULM:STATe?

Parameter:

<State> Boolean data is as follows:

ON | 1: Pulse Modulation is turned on,

OFF | 0: Pulse Modulation is turned off.

Example: :PULM:STATe 1 Pulse Modulation is turned on.

Reset: 0

Key Entry: 【Pulse】—>[Basic Config]—>[Pulse Modulation ON OFF]

[:SOURce]:PULM:INTernal:JITTered:MODE <Mode>

Function: This command is used for setting the jitter modes of pulse modulation period, which include Random and Gaussion. At Random mode, the pulse period changes from 0 to 10% at random; at Gaussion mode, it changes from 0 to 10% in Gaussian distribution.

Setting format: [:SOURce]:PULM:INTernal:JITTered:MODE UNIForm|GAUSsian

Query format: [:SOURce]:PULM:INTernal:JITTered:MODE?

Parameter:

<Mode> Discrete data. Options for PRF jitter mode for pulse modulation are: UNIForm | 0: Random,

GAUSsian | 1: Gauss.

Example: [:SOURce]:PULM:INTernal:JITTered:MODE GAUS Set PRF jitter mode to Gaussion.

Reset status: GAUS

Key Entry: 【PULSE】—>[Jittered]—>[Dither Style]

[:SOURce]:PULM:INTernal:JITTered:PERCent <Percent>

Function: This command is used for setting PRF jitter percentage of pulse modulation signal. When the pulse source is set to Jittered, this percentage will change the pulse period while the pulse width will remain unchanged.

Setting format: [:SOURce]:PULM:INTernal:JITTered:PERCent <val>

Query format: [:SOURce]:PULM:INTernal:JITTered:PERCent?

Parameter:

<Percent> PRF jitter percentage of pulse modulation.

Range: 0[0， 10].

Example: [:SOURce]:PULM:INTernal:JITTered:PERCent 5 Pulse jitter percentage is 5%.

Reset status: 10

Key Entry: 【PULSE】—>[Jittered]—>[Dither Percent]

[:SOURce]:PULM:INTernal:PTRain:DATA <PlsWidth>,<PlsPerd>{PlsWidth, PlsPerd …}

Function: When the pulse source for pulse modulation of the signal generator is set to M-Pulse, this command is used for setting the pulse train function, which includes the following parametric components: Pulse width and period. It supports at most 1024 pulse trains. Pulse width and period should be set in pairs; otherwise, this command parameter will be invalid.

Setting format: [:SOURce]:PULM:INTernal:PTRain:DATA <pls_width>,<pls_period>

{pls_width， pls_period…}

Parameter:

<PlsWidth> Pulse width of pulse train.

Range: 50us[0.02us， 42s].

< PlsPerd> Pulse period of pulse train.

Range: 1ms[0.03us， 42s].

Example: [:SOURce]:PULM:INTernal:PTRain:DATA 100us,1ms,200us,2ms

Set pulse trains with pulse width of 100us and pulse period of 1ms, and those with pulse width of 200us and pulse period of 2ms.

Key Entry: 【PULSE】—>[Pulse Train]—>[Edit Pulse Train]

Note: for setting only.

[:SOURce]:PULM:INTernal:PTRain:DELete <Index>

Function: This command is used for deleting any index point in the pulse trains list of the signal generator. If the index number to be deleted exceeds the range of list points, this operation will be invalid. In this case, the user may query the current list points before deletion operation. If the queried number of points is 1, for valid deletion of the pulse trains list, the index value should be set to zero. For setting the points of pulse train, see the command "PULM:INTernal:PTRain:POINts".

Setting format: [:SOURce]:PULM:INTernal:PTRain:DELete <num>

Parameter:

<Index> Index of pulse trains list.

Range: 0[0， 1023].

Example: [:SOURce]:PULM:INTernal: PTRain:DELete 1 Delete the point with index number of 1 in the current pulse trains list.

Key Entry: None

Note: for setting only.

[:SOURce]:PULM:INTernal:PTRain:POINts?

Function: This command is used for querying points of current pulse train. "Zero" indicates that the current list is empty, and "Nonzero" indicates the actual points in the current list. It should be noted that, if the user operates the interface manually, the index in the pulse trains list will start from 0; that is, if the queried number of points in the list is 1, the index actually shown in the list is 0.

Query format: [:SOURce]:PULM:INTernal:PTRain:POINts?

Return value:

<Num> Integer data; returned points of pulse trains list.

Range: 0[0， 1023].

Example: [:SOURce]:PULM:INTernal:PTRain:POINts? Query points of current pulse trains list.

Reset status: 0

Key Entry: None

Note: for query only.

[:SOURce]:PULM:INTernal:PTRain:PRESet

Function: This command is used for deleting all points in the pulse trains list. If the current list is empty, no operation will be conducted. The user may query the points in the current list before deletion operation. For the

command related to pulse train points, see "PULM:INTernal:PTRain:POINts".

Setting format: [:SOURce]:PULM:INTernal:PTRain:PRESet

Example: [:SOURce]:PULM:INTernal:PTRain:PRESet Delete all point in the pulse train list.

Key Entry: 【PULSE】—>[Pulse Train]—>[Edit Pulse Train] —>[Del All]

Note: for setting only.

[:SOURce]:PULM:INTernal:SLIDing:STEP <StepTime>

Function: When the pulse source is set to Sliding, this command is used for setting pulse sliding step. Based on the current pulse period, the pulse signal will increase progressively with the set sliding step. For example, when the step point is set to 1024, current pulse period is 1ms and sliding step is 100us, the pulse period will change between 1ms and 103.4ms.

Setting format: [:SOURce]:PULM:INTernal:SLIDing:STEP <val><time unit>

Query format: [:SOURce]:PULM:INTernal:SLIDing:STEP?

Parameter:

<StepTime> Sliding step.

Range: 100ns[0s，42ms].

Example: [:SOURce]:PULM:INTernal:SLIDing:STEP 100us Set pulse sliding step to 100us.

Reset: 100ns

Key Entry: 【PULSE】—>[Sliding]—>[Sliding Step]

[:SOURce]:PULM:INTernal:SLIDing:POINts <Num>

Function: When the pulse source is set to Sliding, this command is used for setting pulse sliding points.

Setting format: [:SOURce]:PULM:INTernal:SLIDing:POINts <Num>

Query format: [:SOURce]:PULM:INTernal:SLIDing:POINts?

Parameter:

<StepTime> Sliding points.

Range: [2，1024].

Example: [:SOURce]:PULM:INTernal:SLIDing:POINts 10 Set sliding points to 10.

Reset: 1024

Key Entry: 【PULSE】—>[Sliding]—>[Sliding Counts]

[:SOURce]:PULM:INTernal:STAGger:INSert <Index>,<PlsPerd>

Function: This command is used for inserting pulse staggered point. The user should input index and pulse period successively. The multiple of at most five is supported for the staggered pulse of the signal generator. Therefore, the user may query the current staggered points before using this command. When the maximum number of staggered points is exceeded, current set staggered point will not be inserted to this staggered list. In addition, pulse width in the list will be a unified value. For related commands, see

"PULM:INTernal:STAGger:POINts", ":PULM:INTernal:PWIDth".

Setting format: [:SOURce]:PULM:INTernal:STAGger:INSert <index>,<pls_period>

Parameter:

<Index> Index of PRF staggered list points.

Range: 0[0， 4].

< PlsPerd> Pulse period.

Range: 1.000000ms[40ns， 42.000000000s].

Example: [:SOURce]:PULM:INTernal:STAGger:INSert 1， 1ms

Insert a staggered point with 1ms period in index 1 of the current PRF staggered list.

Key Entry: 【PULSE】—>[Staggered]—>[EditStagg List…] —>[Insert]

Note: for setting only.

[:SOURce]:PULM:INTernal:STAGger:POINts?

Function: This command is used for querying current PRF staggered points. "Zero" indicates that there is no staggered point in the current list, and "Nonzero" indicates the actual points in the current list. It should be noted that, if the user operates the interface manually, the index in the Stagg List will start from 0; that is, if the queried number of points in the list is 1, the index actually shown in the list is 0.

Query format: [:SOURce]:PULM:INTernal:STAGger:POINts?

Return value:

<Num> Integer data; returned points of Stagg List.

Range: 0[0， 5].

Example: [:SOURce]:PULM:INTernal:STAGger:POINts?

Query current PRF staggered points.

Reset status: 0

Key Entry: None

Note: for query only.

[:SOURce]:PULM:INTernal:STAGger:DELete <Index>

Function: This command is used for deleting any index in Stagg Listof the signal generator. If the index point to be deleted exceeds the range of list points, this operation will be invalid. In this case, user may query the current list points before deletion operation. It should be noted that, if the queried staggered point number is 1, for valid deletion, the index shall be set to zero. For points in Stagg List, see the command "PULM:INTernal:STAGger:POINts".

Setting format: [:SOURce]:PULM:INTernal:STAGger:DELete <num>

Parameter:

<Index> Index of Stagg List.

Range: 0[0， 4].

Example: [:SOURce]:PULM:INTernal:STAGger:DELete 1 Delete the point with index 1 in the current Stagg List.

Reset status: 0

Key Entry: 【PULSE】—>[Staggered]—>[EditStagg List…] —>[Del Current] Note: for setting only.

[:SOURce]:PULM:INTernal:STAGger:PRESet

Function: This command is used for deleting all points in Stagg List. If there is no staggered point in current list, this operation will be invalid. In this case, user may query the staggered points in the current list before deletion operation. For querying points in Stagg List, see the command "PULM:INTernal:STAGger:POINts".

Setting format: [:SOURce]:PULM:INTernal:STAGger:PRESet

Example: [:SOURce]:PULM:INTernal:STAGger:PRESet Delete all points in Stagg List.

Key Entry: 【PULSE】—>[Staggered]—>[EditStagg List…] —>[Del All]

Note: for setting only.

[:SOURce]:PULM:LFM:BWIDth <LfmBwidth>

Function: This command is used for setting LFM bandwidth. After setting, the output signal centers on the output carrier frequency of the signal generator, with the FM range equal to carrier frequency ± 1/2 bandwidth.

Setting format: [:SOURce]:PULM:LFM:BWIDth <val><freq unit>

Query format: [:SOURce]:PULM:LFM:BWIDth?

Parameter:

<LfmBwidth> LFM bandwidth.

Range: 4MHz[0Hz， 200MHz].

Example: [:SOURce]:PULM:LFM:BWIDth 20MHz LFM bandwidth is 20MHz.

Key Entry: 【Signal】—>[LFM]—> [BandWidth]

[:SOURce]:PULM:LFM:DIRection <Mode>

Function: This command is used for setting LFM direction. When LFM Direction is set to Increase, the FM signal output of the signal generator will increase progressively from negative to positive bandwidth; when the direction is set to Decrease, it will change from positive to negative bandwidth. After turning off LFM, the instrument will turn off pulse modulation and IQ modulation automatically.

Setting format: [:SOURce]:PULM:LFM:DIRection POSitive|NEGative

Query format: [:SOURce]:PULM:LFM:DIRection?

Parameter:

<Mode> Discrete data. Options for LFM Direction are:

POSitive | 0: Increase,

NEGative | 1: Decrease.

Example: [:SOURce]:PULM:LFM:DIRection POS LFM Direction is Increase.

Reset status: POS

Key Entry: 【Signal】—>[LFM]—> [LFM Direction]

[:SOURce]:PULM:LFM:STATe <State>

Function: This command is used for setting LFM. When LFM is set to ON, the signal generator will initiate pulse modulation and IQ modulation automatically. After turning off LFM, the instrument will turn off pulse modulation and IQ modulation automatically.

Setting format: [:SOURce]:PULM:LFM:STATe ON|OFF|1|0

Query format: [:SOURce]:PULM:LFM:STATe?

Parameter:

<State> Boolean data is as follows:

ON | 1: LFM ON,

OFF | 0: LFM OFF.

Example: [:SOURce]:PULM:LFM:STATe 1 LFM is ON.

Reset status: 0

Key Entry: 【Signal】—>[LFM]—> [LFM ON OFF]

3.3.8 AMPLitude MODulation Subsystem

The following commands are used for setting AM operation mode:

- [:SOURce]:AM[:DEPTh]:EXPonential
- [:SOURce]:AM[:DEPTh][:LINear]
- [:SOURce]:AM:INTernal:FREQuency
- [:SOURce]:AM:INTernal:FREQuency:ALTernate
- [:SOURce]:AM:INTernal:FREQuency:ALTernate:AMPLitude:PERCent
- [:SOURce]:AM:INTernal:NOISe
- [:SOURce]:AM:INTernal:RAMP
- [:SOURce]:AM:INTernal:SHAPe
- [:SOURce]:AM:INTernal:SWEep:TIME
- [:SOURce]:AM:MODE
- [:SOURce]:AM:SOURce
- [:SOURce]:AM:STATe
- [:SOURce]:AM:TYPE

[:SOURce]:AM[:DEPTh]:EXPonential <AmDepthExp>

Function: When AM Type is EXP, set AM Depth of AM signal in dB. For AM type, see the command ":AM:TYPE".

Setting format: [:SOURce]:AM:DEPTh:EXPonential <.val>

Query format: [:SOURce]:AM: DEPTh:EXPonential?

Parameter:

<AmDepthExp> AM Depth (EXP).

Range: 0.00dB[0.00dB， 40.00dB].

Example: :AM: DEPTh:EXPonential 10dB AM Depth is 10 dB.

Reset status: 30dB

Key Entry: 【AM】—>[Basic Config]—>[AM Depth]

[:SOURce]:AM[:DEPTh][:LINear] <AmDepthLine>

Function: This command is used for setting AM depth in percent. The set value is valid only when AM type is set to Linear. See the command ":AM:TYPE".

Setting format: [:SOURce]:AM: DEPTh [:LINear] <.val>

Query format: [:SOURce]:AM: DEPTh [:LINear]?

Parameter:

<AmDepthExp> AM depth (linear)

Range: 30[0， 100].

Example: :AM:DEPTh 10 This example indicates that the set linear AM depth is 10%

Reset status: 30

Key Entry: 【AM】—>[Basic Config]—>[AM Depth]

[:SOURce]:AM:INTernal:FREQuency <Frequency>

Function: This command is used for setting the internal modulation rate for AM in the signal generator. In addition, through this command, the first tone can be set when AM waveform is set to DualSinc, and the start frequency can be set when AM waveform is set to SweepSinc. For AM signal output waveform, see the command ":AM:INTernal:SHAPe".

Setting format: [:SOURce]:AM:INTernal:FREQuency <val>

Query format: [:SOURce]:AM:INTernal:FREQuency?

Parameter:

<Frequency> AM rate.

Range: 1kHz[5mHz， 1MHz].

Freq Start of SweepSinc.

Range: 10mHz[10mHz,0.99999999MHz]

DualSinc frequency1.

Range: 1kHz[10mHz,1MHz].

Example: :AM:INTernal:FREQuency 100kHz Internal modulation rate for AM is 100kHz.

Reset status: AM rate: 1kHz.

Start frequency of sweep sine: 10mHz.

Frequency 1 of dual sine: 1kHz

Key Entry: 【AM】—>[Basic Config]—>[AM Rate]

[:SOURce]:AM:INTernal:FREQuency:ALTernate <Frequency>

Function: This command is used for setting the second tone when AM waveform is set to DualSinc, or the stop frequency when AM waveform is set to SweepSinc. For AM signal output waveform, see the command ":AM:INTernal:SHAPe".

Setting format: [:SOURce]:AM:INTernal:FREQuency:ALTernate <.val>

Query format: [:SOURce]:AM:INTernal:FREQuency:ALTernate?

Parameter:

<Frequency> Frequency2 when AM waveform is set to DualSinc, or the stop frequency when AM waveform is set to SweepSinc.

Range: Dual sine 1kHz[0.02Hz，1.000000000MHz].

Sweep sine 1kHz[0.02Hz，1.000000000MHz].

Example: :AM:INTernal:FREQuency:ALTernate 500kHz Set alternate frequency to 500kHz.

Key Entry: 【AM】—>[Sweep Sinc]—>[Freq Stop],

【AM】—>[Dual Sinc]—>[Frequency2].

[:SOURce]:AM:INTernal:FREQuency:ALTernate:AMPLitude:PERCent<Pert>

Function: This command is used for setting the percentage of amplitude of second tone to that of total output signal in dual sine waveform when AM Waveform is set to DualSinc; for example, if the second tone accounts for 20% of the total waveform power, the first tone will account for 80% of the total power output.

Setting format: [:SOURce]:AM:INTernal:FREQuency:ALTernate:AMPLitude:PERCent <val>

Query format: [:SOURce]:AM:INTernal:FREQuency:ALTernate:AMPLitude:PERCent?

Parameter:

<Pert> Amplitude percentage of frequency 2 when AM Waveform is set to DualSinc.

Range: 50[0，100].

Example: :AM:INTernal:FREQuency:ALTernate:AMPLitude:PERCent 20

Set the percentage of second waveform of dual sine to the total signal output power to 20%.

Reset status: 50

Key Entry: 【AM】—>[Dual Sinc]—> [Freq 2 Ampl Percent]

Key path: [Arb] —> [Trigger] —> [External Clock Frequency]

3.3.12 MEMOry Subsystem

The following commands are used to select the memory mode, including

- :MEMory:COPY:NAME
- :MEMory:DELete:NAME
- :MEMory:MOVE
- :MEMory:DATA

➢ **:MEMory:COPY:NAME <SrcName>,<DestName>**

Function description:

This command is used to copy the data in one file to another file. If the source file and the destination file are not in the same folder, the file name may be the same. When copying any arbitrary file, the tag file related to the file will be copied together. It should be noted that the absolute path must be attached to the file name.

Setting format: MEMory:COPY:NAME <src_name>,<dest_name>

Parameter description:

<SrcName> string type, name of source file

<DestName> string type, name of destination file.

Example: :MEMory:COPY:NAME

"c:\\data\\user\\seq1.dat","c:\\data\\user\\seq2.dat"

Copy the data in seq1.dat to seq2.dat.

Key path: [File] —> [Copy]

Description: For setting only.

➢ **:MEMory:DELete:NAME <FileName>**

Function description:

This command is used to delete user file in the signal generator.

Setting format: MEMory:DLEete:NAME <file_name>

Parameter description:

<FileName> Character string type. User file saved in the signal generator.

Example: MEMory:DELete:NAME "c:\\arb1.seq"

Delete file arb1.seq in disk C.

Key path: NONE

Description: For setting only.

➢ **:MEMory:MOVE <FileName>**

Function description:

This command is used to rename the file in the signal generator. It should be noted that the absolute path must be attached to the file name.

Setting format: MEMory:MOVE <Sourfile_name>, <Desfile_name>

Parameter description:

<FileName> Character string type. File name saved in the signal generator.

Example: MEMory:MOVE "c:\data\arb1.seq","c:\data\arb2.seq"

Rename file arb1.seq to arb2.seq

Description: For setting only.

➢ **:MEMory:DATA <FileName>,<#AB\n><DataBlock>**

Function description:

This command is used to download arbitrary data into the instrument in the way of data block through the communication interface of the signal generator, and save it in the instrument with the name of <file_name>. The command can only be used to transmit the data with the number of bytes below 1000000000, that is, the total number of bits of transmitted data is less than 10, and the data is binary data.

Setting format: MEMory:DATA <file_name>, <data_block>

Parameter description:

<FileName> Character string type. Name of arbitrary file saved in the signal generator specified by users. Users are not entitled to specify the absolute path.

<#AB\n> # and \n are fixed formats, where # indicates the beginning of the data, \n is the placeholder, and A and B represents the length of the data. For example, in #210\n, 3 indicates that the bytes account for 2 bits, 10 represents the total number of bytes, and <DataBlock> following 10 represents 10 bytes of data.

<DataBlock> data block to be transmitted.

Example: [:SOURce]:MEMory:DATA arb1.dat, #41024\n jklasdj…

It indicates that 1024 bytes of data are sent to the signal generator and saved in the generator with the file name of arb1.dat.

Description: For setting only.

3.3.13 ROSCillator Subsystem

The oscillator subsystem command is used to realize functions of the signal generator related to time base.

- [:SOURce]:ROSCillator:REFerence

➢ **[:SOURce]:ROSCillator: REFerence <Val>**

Function description:

This command is used to adjust internal reference of the signal generator by setting internal calibration parameter, so as to make the frequency output more accurate. It should be noted that within 2h after starting the signal generator, the instrument should be preheated.

Please do not change the reference value easily. For more detailed information, please refer to the user's manual for 1465 series signal generator.

Setting format: [:SOURce]:ROSCillator:REFerence <val>

Query format: [:SOURce]:ROSCillator: REFerence?

Parameter description:

<Val> internal calibration parameter.

Range: [0, 65535].

Example: ROSCillator: REFerence 30000

Adjust the internal reference accuracy value to 30000.

3.3.14 SYSTem Subsystem

The system subsystem command is used to realize functions of the signal generator related to its performance.

The following commands are used to select the operating mode, including:

- :DIAGnostic:INFormation:CCOunt:PON
- :DIAGnostic:INFormation:OTIMe
- :DIAGnostic:SNUM
- :SYSTem:COMMunicate:GPIB:ADDRess
- :SYSTem:COMMunicate:GTLocal
- :SYSTem:DEVice:LANGuage
- :SYSTem:COMMunicate:LAN:IP
- :SYSTem:COMMunicate:LAN:SUBNet
- :SYSTem:COMMunicate:LAN:GATeway
- :SYSTem:ERRor[:NEXT]
- :SYSTem:PRESet:TYPE

➢ **:DIAGnostic:INFormation:CCOunt:PON**

Function description:

Query the accumulative startup times of the instrument

Query format: DIAGnostic:INFormation:CCOunt:PON?

Example: DIAGnostic:INFormation:CCOunt:PON?

This example shows that the cumulative number of times the signal generator has been started is queried.

Description: For query only.

➢ **:DIAGnostic:INFormation:OTIMe**

Function description:

Query the instrument firmware date and time stamp

Query format: DIAGnostic:INFormation:OTIMe?

Example: DIAGnostic:INFormation:OTIMe?

This example shows that the cumulative number of hours the signal generator has been started is queried.

Description: For query only.

- **:DIAGnostic:SNUM?**

Function description:

This command is used to read the system serial number of the signal generator.

Query format: DIAGnostic:SNUM?

Returned value: Serial number

Example: DIAGnostic:SNUM

This example shows that the system serial number of the signal generator is queried.

Description: For query only.

- **:SYSTem:COMMunicate:GPIB:ADDRess <Address>**

Function description:

This command is used to set GPIB address of the signal generator, which is 19 by default. To ensure normal communication, the local GPIB address should be different from that of other devices in the same test system.

Setting format: SYSTem:COMMunicate:GPIB:ADDRess <val>

Query format: SYSTem:COMMunicate:GPIB:ADDRess?

Parameter description:

<Address> integer data, GPIB address.

Range: 19 [0, 30].

Example: SYSTem:COMMunicate:GPIB:ADDRess 19

Set GPIB address of the signal generator to 19.

Reset state: 19

Key path: [System] —> [GPIB Port] —> [Local GPIB Addr]

- **:SYSTem:COMMunicate:GTLocal**

Function description:

This command is used to switch the signal generator to local operation mode. In this mode, users can operate the buttons on the front panel of the instrument, and the indication of remote control operation mode on the instrument operation interface will disappear.

Setting format: SYSTem:COMMunicate:GTLocal

Example: SYSTem:COMMunicate:LAN:IP 172.141.114.114

:SYSTem:COMMunicate:GTLocal

Switch the signal generator from remote control state to local state.

Description: For setting only.

- **:SYSTem:DEVice:LANGuage <Mode>**

Function description:

This command is used to set the language displayed on the interface of the signal generator. At present, the instrument supports Chinese and English interface. The default interface is Chinese. After the language switch is completed, the interface can be switched in real time, but when the command is used for query, the returned value is the state value before the switch. The instrument should be restarted to make the returned value displayed correctly.

Setting format: SYSTem:DEVice:LANGuage CHINese|ENGLish

Query format: SYSTem:DEVice:LANGuage?

Parameter description:

<Mode> discrete data. The values of interface language are as follows:

CHINeses Chinese

ENGLish English

Example: SYSTem:DEVice:LANGuage ENGLish

Set the interface of the instrument to English.

Reset state: Keep the current language.

Key path: [System] —> [Base Config] —> [Language/LANG]

- **:SYSTem:COMMunicate:LAN:IP <Address>**

Function description:

This command is used to set IP address of the signal generator. The parameters are expressed in dotted decimal notation.

Setting format: SYSTem:COMMunicate:LAN:IP <ipstring>

Query format: SYSTem:COMMunicate:LAN:IP?

Parameter Description:

<Address> string type, network IP address expressed in dotted decimal notation

Example: SYSTem:COMMunicate:LAN:IP “172.141.114.114”

Set IP address of the signal generator to 172.141.114.114.

Key path: [System] —> [LAN Port] —> [Local Machine IP Addr]

- **:SYSTem:COMMunicate:LAN:SUBNet <Address>**

Function description:

This command is used to set subnet mask address of the signal generator. The parameters are expressed in dotted decimal notation.

Setting format: SYSTem:COMMunicate:LAN:SUBNet <ipstring>

Query format: SYSTem:COMMunicate:LAN: SUBNet?

Parameter Description:

<Address> string type, network IP address expressed in dotted decimal notation

Example: SYSTem:COMMunicate:LAN: SUBNet “255.255.255.0”

Set subnet mask address of the signal generator to 255.255.255.0.

Key path: [System] —> [LAN Port] —> [Net Mask]

- **:SYSTem:COMMunicate:LAN:GATeway <Address>**

Function description:

This command is used to set network gateway address of the signal generator in external network access LAN. The parameters are expressed in dotted decimal notation.

Setting format: SYSTem:COMMunicate:LAN:GATeway <ipstring>

Query format: SYSTem:COMMunicate:LAN:GATeway?

Parameter Description:

<Address> string type, network IP address expressed in dotted decimal notation

Example: SYSTem:COMMunicate:LAN: GATeway "172.141.114.254"

Set network gateway address of the signal generator to 172.141.114.254.

Key path: [System] —> [LAN Port] —> [Default Gate]

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

Function description:

This command is used to query the errors in error queue of the signal generator. When an error is queried, it will be deleted from the error queue. If there is no error in the error queue, users will find the message: "+0, No ERROR".

Query format: SYSTem:ERRor[:NEXT]?

Returned value: <ErrorInfo>: "error code, error".

Example: SYSTem:ERRor[:NEXT]?

This example shows that the errors of the signal generator are queried.

Description: For query only.

- **:SYSTem:PRESet:TYPE <Mode>**

Function description:

This command is used to set the reset state of the signal generator, including manufacturer, user and last state. In manufacturer mode, the instrument will return to the default state of the manufacture after reset. The user mode is the reset state defined by the user. After the instrument is reset, it will return to the instrument state specified by the user. The last state is the state before the instrument is reset.

Setting format: SYSTem:PRESet:TYPE NORMal|USER|LAST

Query format: SYSTem:PRESet:TYPE?

Parameter description:

<Mode> discrete data. The values of reset state are as follows:

NORMal	|	0: manufacturer
USER	|	1: user
LAST	|	2: last state

Example: SYSYem:PRESet:TYPE USER set the reset state of the signal generator to user.

Reset state: NORMal

Key path: [System] —> [Reset] —> [Reset type]

4 Programming Examples

4.1 Basic Operation Examples

The following examples show the basic methods for using the VISA library to realize remote control programming of the instrument, with C++ language as an example.

4.1.1 VISA Library

VISA is the generic term of the standard I/O function library and relevant specifications. Whereas, the VISA library functions are a set of functions convenient for loading, and the core function can control various instruments, regardless of the instrument interface type and operation method of various I/O interface software. Such library functions are used to compile the instrument drive program, and complete the command and data transmission between the computer and instrument, realizing remote instrument control. The addressing strings ("VISA resource strings") can be initialized to establish instrument connection with program control ports (LAN, USB and GPIB).

To realize remote control, the VISA library must be installed first. The VISA library is encapsulated with the transmission function at the bottom layer with VXI, GPIB, LAN and USB interfaces, so as to facilitate users ' direct loading. Programming interfaces supported by the signal generator is GPIB and LAN. **These interfaces can be used with VISA libraries and programming languages for remote control of signal generators.** The Agilent I/O Library provided by Agilent Company is widely used as the bottom-layer I/O library.

Figure 4.1 shows the relationship between program control interfaces, VISA libraries, programming languages and signal generators with the GPIB interface as an example.

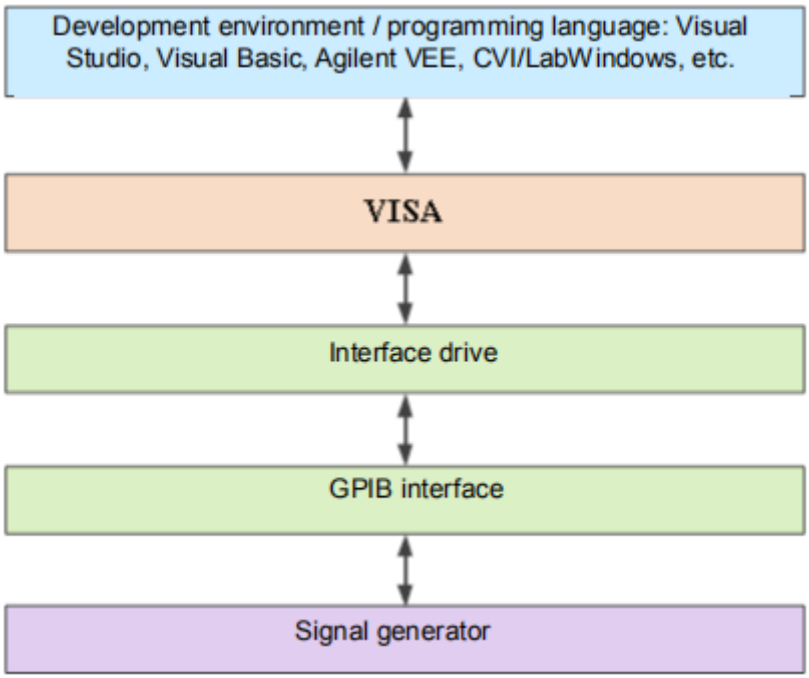

Figure 4.1 Program control software/ hardware layers

4.1.2 Example Runtime Environment

4.1.2.1 Configuration Requirements

The programming examples described in this chapter have run successfully on a computer configured as follows.

- IBM compatible PC above Pentium class;
- Windows 2000 or Windows XP;
- Visual Studio 2010/2012 integrated development environment;
- PCI-GPIB interface card of NI or GPIB interface card of Agilent;
- VISA library of NI or Agilent;
- GPIB card;
- Network card;
- Available serial ports COM1 and COM2.

4.1.2.2 Files Included

To run an example program written in C/C++, you must include the required files in a project in VC6.0.

If you use the VISA library, you must:

- Add visa32.lib to the source file;
- Add visa.h to the header file.

If you use the NI-488.2 library, you must:

- Add GPIB-32.OBJ file to the source file;
- Add windows.h file to the header file;
- Add Deci-32.h file to the header file.

For more information about the NI-488.2 library and VISA library, please refer to the websites of NI and Agilent respectively.

4.1.3 Initialize and Set Default State

To start the program, the VISA resource manager must be initialized, so as to open and establish the communication connection between the VISA library and instrument. Specific steps are shown below:

4.1.3.1 Generating Global Variables

First, generate the global variables to be loaded by other program modules, for example, the instrument handle variable. The programs shown below must contain the following global variables:

```
ViSession analyzer;
ViSession defaultRM;
Const char analyzerString [VI_FIND_BUFLEN] = "GPIB0::20::INSTR";
Const analyzerTimeout = 10000;
```

Whereas, the constant sourceString represents the instrument descriptor, "GPIB0" represents the controller, and "20" represents the instrument connected to the controller. If it is assumed that the instrument is connected to the LAN and the IP address is "192.168.1.1", the value of the variable should be:

Const char sourceString [VI_FIND_BUFLEN] = " TCPIP::192.168.1.1::INSTR ";

4.1.3.2 Initialize the Controller

/***/

The following example shows the way to open and establish the communication connection between the VISA library and instrument (with instrument descriptor specified).

//Initialize the controller: Open the default explorer and return the instrument handle source.

/***/

```
void InitController()
{
    ViStatus status;
    status = viOpenDefaultRM(&defaultRM);
    status = viOpen(defaultRM, sourceString, VI_NULL, VI_NULL, &source);
}
```

4.1.3.3 Instrument Initialization

/***/

The following examples show how to initialize the default state of the instrument and empty the status register.

/***/

```
void InitDevice()
{
    ViStatus status;
    long retCnt;
    status = viWrite(source, "*CLS", 4, &retCnt); //reset the status register
    status = viWrite(source, "*RST", 4, &retCnt); //reset the instrument
    status = viWrite(source, "freq 1ghz", 9, &retCnt); //set the continuous wave frequency of the signal generator to 1GHz
}
```

4.1.4 Send Setting Command

/***/

The following example below shows the way to set the point frequency and amplitude of S1465 series signal generator.

/***/

```
void SimpleSettings()
```

```
{
    ViStatus status;
    long retCnt;
    //Set the point frequency to 128MHz
    status = viWrite(source, "FREQENCY:CW 128MHz", 18, &retCnt);
    //Set the amplitude to -10dBm
    status = viWrite(source, "POW -10dBm", 10, &retCnt);
}
```

4.1.5 Read the Status of Measuring Instrument

/**/

The following examples show how to read the set state of the instrument.

/**/

```
void ReadSettings()
{
    ViStatus status; long retCn;
    char rd_Buf_CW[VI_READ_BUFLEN];    // #define VI_READ_BUFLEN 20
    char rd_Buf_LVL[VI_READ_BUFLEN];

    //Query the point frequency
    status = viWrite(source, "FREQ:CW?", 8, &retCnt);
    Sleep(10);
    status = viRead(source, rd_Buf_CW, 20, &retCnt);
    //Query the amplitude
    status = viWrite(source, "POW?", 4, &retCnt);
    Sleep(10);
    status = viRead(source, rd_Buf_LVL, 20, &retCnt);
    //Print the debugging information
    sprint("Cw is %s", rd_Buf_CW);
    sprint("POW is %s", rd_Buf_ LVL);
}
```

5.3.5 Command Synchronization

The following example of sweeping process is taken to show the command synchronization method:

```
void SweepSync()
```

```
{
    ViStatus status; long retCn; ViEventType etype; ViEvent eevent;
    int stat;
    char OpcOk [2];
    /*****************************************************************/
    /* command INITiate[:IMMediate] to start single sweeping (when continuous sweeping is closed: INIT:CONT OFF)*/
    /* After single sweeping, execute the next command in the command buffer zone
    */
    /*****************************************************************/
    status = viWrite(analyzer, "INIT:CONT OFF", 13, &retCnt);
    //Method 1 to wait for sweeping completion: use *WAI
    status = viWrite(analyzer, "ABOR;INIT:IMM;*WAI", 18, &retCnt);

    //Method 2 to wait for sweeping completion: use *OPC?
    status = viWrite(analyzer, "ABOR;INIT:IMM; *OPC?", 20, &retCnt);
    status = viRead(analyzer, OpcOk, 2, &retCnt); //wait for *OPC to return “1”

    //Method 3 to wait for sweeping completion: use *OPC
    //To use the GPIB service request, set "Disable Auto Serial Poll” to "yes"
    status = viWrite(analyzer, "*SRE 32", 7, &retCnt);
    status = viWrite(analyzer, "*ESE 1", 6, &retCnt); //enable service request ESR
    //Set the event enabling position, and end the operation.
    status = viEnableEvent(analyzer, VI_EVENT_SERVICE_REQ, VI_QUEUE, VI_NULL);
    //Enable the SRQ event
    status = viWrite(analyzer, "ABOR;INIT:IMM;*OPC", 18, &retCnt);
    //Start sweeping together with OPC
    status = viWaitOnEvent(analyzer, VI_EVENT_SERVICE_REQ, 10000, &etype, &eevent)
    //Wait for service request
    status = viReadSTB(analyzer, &stat);
    status = viClose(eevent); //close event handle
    // disable SRQ event
    status = viDisableEvent(analyzer, VI_EVENT_SERVICE_REQ, VI_QUEUE);
    //Main program continues……
}
```

4.2 Advanced Operation Examples

4.2.1 Set Point Frequency at LAN Interface and Query

/**/

To use the following examples correctly, you must match your host address to the IP address of the signal source. (Network design examples in this manual are implemented in VC6.0 by using WINSOCK components to establish socket.)

/**/

```
#include “stdafx.h
#include <afxscok.h>
#include <stdio.h>
#include <stdlib.h>
#ifdef _UNICODE
#define SG_IP_ADDR        L”192.168.1.199”      //IP address of the signal generator
#else
#define SG_IP_ADDR        ”192.168.1.199”       //IP address of the signal generator
#endif
#define SG_SOCKET_PORT 5001          //port number of the signal generator
void ShowMsg(PCHAR lpszText)
{
      #ifdef _UNICODE
      AfxMessageBox((CString)lpszText);
      #else
      AfxMessageBox(lpszText);
      #endif
}
void SocketTest(void)
{
      CSocket client;
      int iFlag;
      char rgcBuf[256];
      int iBufLen;
      if (!AfxSocketInit())   //initialize the network port
      {
            ShowMsg("Initialization failed");
      }
      else
```

```
    {
    iFlag = client.Create();
    If (!iFlag)
    {
        ShowMsg("Socket creation failed");
    }
    else
    {
    ShowMsg("Socket creation successful");
    iFlag = client.Connect(SG_IP_ADDR, SG_SOCKET_PORT); //Connect the network port if (!iFlag)
    {
        ShowMsg("Connection failed");
    }
    //Set the point frequency to 1GHz
    sprintf(rgcBuf, "%s\n", "FREQ 1GHz");
    iBufLen = (int)strlen(rgcBuf);
    iFlag = client.Send(rgcBuf, iBufLen);
    if (!iFlag)
    {
        ShowMsg("Send failed");
    }
    }
    else
    {
        iFlag= client.Receive(rgcBuf, sizeof(rgcBuf), 0); //read from the network if (!iFlag)
        {
        ShowMsg("Receive failed");
        }
    }
  }
}
client.Close();
}
}
```

4.2.2 Set Point Frequency at GPIB Interface and Query

/***/

In this example, the functions of the VISA library are used to set the signal source to output point frequency of 500MHz and power of -2dbm, query the current frequency and power, start VC6.0, add necessary files, and input the following codes into your .cpp file

/***/

```
#include "stdafx.h"
#include "visa.h"
#include <iostream>
#include <stdlib.h>
#include <conio.h>
void main()
{
    ViSession defaultRM,vi;          //declare a variable of type VISession
    ViStatus vistatus = 0;       //for device communication
    char buff[256];  //declare a variable with character data stored
    vistatus = viOpenDefaultRM(&defaultRM);              //Open the GPIB task, address: 19
    vistatus=viOpen(defaultRM,"GPIB0::19::INSTR",VI_NULL,VI_NULL,&vi); if(vistatus)
    {
        printf("The task cannot be opened. Please recheck the device and connect \n");
        exit(0);
    }
    viPrintf(vi,"*RST\n");                    //reset the signal source
    viPrintf(vi,"FREQ 500MHZ\n");         //set the point frequency to 500MHz
    viPrintf(vi,"FREQ?\n");                   //query the point frequency
    viScanf(vi,"%t",buff);        //put the query results into an array
    printf("source CW freq is: %s\n",buff);       //display the point frequency
      viPrintf(vi,"POW     -2dBm\n");              //set the power level to -2dBm
    viPrintf(vi,"POW?\n");             //query the current power
    viScanf(vi,"%t",buff);        //put the query results into an array
    printf("source POW is: %s\n",buff);       //display the power
    viClear(vi);
    viClose(vi);
    viClose(defaultRM);
}
```

5 Error Description

This chapter will show you how to find problems and accept after-sales service, and explain error message of the signal generator.

- **Errors**
- **Method to Obtain After-sales Services**

5.1 Errors

The signal generator records the errors during the measurement in two ways: error queue displayed on front panel operation interface and error queue in SCPI (remote control mode), which are stored and managed respectively.

5.1.1 Local Errors

5.1.1.1 Error View

View via interface:

In case of any error prompt at the lower right of the signal source during use, it indicates that there is something wrong with the software or hardware of the signal source. You can basically judge the error type as per the error code, and take corresponding measures for troubleshooting.

The error display area of the signal source can only display one error prompt at a time. Since multiple errors can occur to the instrument, to see all errors, do the following:

Step 1. Click [system] and then [Machine error], and an error list window will pop up.

Step 2. The prompt will be displayed in the window.

Step 3. Use the mouse to browse the error and close the dialog window.

Step 4. Select "Clear error list" to clear historical errors.

5.1.1.2 Error Description

If an error is detected during the measurement of the signal generator, an alarm or error will be displayed on the right side of the status indicating area (error abbreviation + detailed error description).

Table 5.1 List of local error description

Key error field	Detailed error description
Unleveled	For overpower or no power
Reference loop unlocked	The reference loop signal inside the signal generator is out of lock.
Decimal loop unlocked	The decimal loop signal inside the signal generator is out of lock.
Local oscillator loop unlocked	The local oscillator loop signal inside the signal generator is out of lock.
VCO loop unlocked	The VCO loop signal inside the signal generator is out of lock.

5.1.2 Program Errors

5.1.2.1 Error Format and Description

In remote control mode, errors are recorded in the error/event queue of the status reporting system, and can be queried with the comm and “SYSTem:ERRor?”. The format is as follows:

"<Error code>, "<Error in error queue>; <"Detailed error description>"

Example:

“-110,” a data breaking bounds; the input parameter is beyond the lower bound.

A negative error code defined by the SCPI standard. This type of error is not specified here.

5.1.2.2 Error Type

Error event corresponds to only one type of error. The types of errors are classified below (8257):

- **Query error (-499 to -400):** indicating that the output queue of the instrument controls and detects a message exchange protocol error described in Chapter 6 of IEEE 488.2. At this point, the query error bit (bit2) of the event status register is set (please refer to IEEE 488.2, 6.5 for details). The data cannot be successfully read from the output queue at this time.
- **Instrument characteristic error (-399 to -300, 201 to 703, and 800 to 810):** indicating that the instrument operation is not successful, and the reason may be abnormal hardware or firmware state. Such error codes are often used self-detection of the instrument. At this point, the instrument characteristic error bit (bit3) of the event status register is set.
- **Execution error (-299 to -200):** indicating that an error is detected during the measurement of the instrument. At this point, the execution error bit (bit4) of the event status register is set.
- **Command error (-199 to -100):** indicating a syntax error detected during command parsing of the instrument, usually due to an incorrect command format. At this point, the command error bit (bit5) of the event status register is set.

5.2 Method to Obtain After-sales Services

5.2.1 Contact Us

If there is a problem with S1465 series signal generator, first observe the error and save it, and solve the problem in advance. If the problem cannot be solved, contact the service and consultation center of the Company as per the contact information provided below and provide us with the error collected. We will coordinate with you to solve the problem as soon as possible.

Contact information:

Service Tel:	+886. 909 602 109
Website:	www.salukitec.com
Email:	sales@salukitec.com
Address:	No. 367 Fuxing N Road, Taipei 105, Taiwan (R.O.C.)

5.2.2 Package and Mailing

In case of any failure to the signal generator that is difficult to be eliminated, contact us by phone or fax. If it is confirmed by both parties that the signal generator has to be returned for repairing, pack the instrument with the original packing materials and case by following the steps below:

1) Prepare a detailed description of the failure of the signal generator and put it into the package along with the instrument.
2) Pack it with the original packing materials, so as to minimize possible damage.

3) Place cushions at the four corners of the outer packing carton, and place the instrument in the outer packing carton.
4) Seal the opening of the packing carton with adhesive tape and reinforce the packing carton with nylon tape.
5) Specify text like “Fragile! Do not touch! Handel with care!” and so on.
6) Please check by precision instrument.
7) Keep a copy of all shipping documents.

Note

Precautions on packing the signal generator

Using other materials instead of the original ones to pack the signal generator can damage the instrument. Never use polystyrene beads to pack the instrument due to two reasons, that is, they cannot provide sufficient protection on the instrument, and they can be sucked in to the instrument fan by the static electricity generated, resulting in instrument damage.

Tip

Instrument package and transportation

Please follow carefully the precautions described in “2.1.1.1 Unpacking” of the User Manual when transporting or handling the instrument (for example, damage occurred during delivery).

Annex

This section introduces the below information:

- **Annex A Zoom Table of SCPI Classified by Subsystem**
- **Annex B Zoom Table of Errors**

Annex A Zoom Table of SCPI Classified by Subsystem

Inde x	Command	Function
1	*IDN?	Universal command
2	*RCL	Universal command
3	*RST	Universal command
4	*SAV	Universal command
5	*CLS	Universal command
6	*ESE	Universal command
7	*ESR?	Universal command
8	*STB?	Universal command
9	*TRG	Universal command
10	*TST?	Universal command
11	:OUTPut[:STATe](?)	Set RF output on/off
12	OUTPut:MODulation[:STATe](?)	Set modulation master on/off
13	[:SOURce]:FREQuency[:CW|FIXed](?)	Set signal generator output frequency
14	[:SOURce]:FREQuency:MODE(?)	Set frequency generation mode
15	[:SOURce]:FREQuency:MULTiplier(?)	Set frequency multiplier
16	[:SOURce]:FREQuency:OFFSet(?)	Set frequency offset
17	[:SOURce]:FREQuency:REFerence(?)	Set relative frequency
18	[:SOURce]:FREQuency:REFerence:STATe(?)	Set relative frequency on/off
19	[:SOURce]:FREQuency:STEP(?)	Set frequency stepping
20	[:SOURce]:FREQuency:STARt(?)	Set step sweep start frequency
21	[:SOURce]:FREQuency:STOP(?)	Set step sweep end frequency
22	[:SOURce]:POWer:ALC:LEVel(?)	Set ALC level value
23	[:SOURce]:POWer:ALC:SEARch(?)	Set power search mode
24	[:SOURce]:POWer:ALC:SOURce(?)	Set power fixed-amplitude mode
25	[:SOURce]:POWer:ALC:SOURce:EXTernal:COUPling(?)	Set external detection coupling factor
26	[:SOURce]:POWer:REFerence:STATe(?)	Set ALC loop status
27	[:SOURce]:POWer:ATTenuation(?)	Set power attenuation
28	[:SOURce]:POWer:ATTenuation:AUTO(?)	Set power attenuation on/off

29	[:SOURce]:POWer[:LEVel][:IMMediate][:AMPLitude](?)	Set power level
30	[:SOURce]:POWer[:LEVel][:IMMediate]:OFFSet(?)	Set amplitude offset
31	[:SOURce]:POWer:REFerence(?)	Set relative amplitude
32	[:SOURce]:POWer:REFerence:STATe(?)	Set relative amplitude on/off
33	[:SOURce]:POWer:STEP(?)	Set amplitude stepping
34	[:SOURce]:POWer:ALC:BANDwidth|BWIDth	Set ALC loop bandwidth
35	[:SOURce]:POWer:ALC:BANDwidth|BWIDth:AUTo	Set ALC loop bandwidth selection mode
36	[:SOURce]:POWer:SWEep[:STATe]	Set amplitude sweep on/off
37	[:SOURce]:LIST:DIRection(?)	Set list sweep direction
38	[:SOURce]:LIST:DWELl	Set Dwell time for all points
39	[:SOURce]:LIST:FREQuency	Set list sweep frequency
40	[:SOURce]:LIST:FILL:POINts(?)	Set number of list sweep points
41	[:SOURce]:LIST:FILL:STARt(?)	Set list sweep start frequency
42	[:SOURce]:LIST:FILL:STOP(?)	Set list sweep end frequency
43	[:SOURce]:LIST:FILL:POWer(?)	Set list sweep amplitude
44	[:SOURce]:LIST:TRIGger:SOURce(?)	Set list sweeptrigger source
45	[:SOURce]:LIST:FILL:POWer(?)	Set amplitude offset for all list points
46	[:SOURce]:LIST:FILL:DWELl(?)	Set Dwell time for all list points
47	[:SOURce]:LIST:FILL:EXECute	Complete list fill-in
48	[:SOURce]:LIST:DELete	Delete list points
49	[:SOURce]:LFOutput:AMPLitude(?)	Set low frequency range
50	[:SOURce]:LFOutput:FREQuency(?)	Set low frequency rate
51	[:SOURce]:LFOutput:RAMP(?)	Set waveform direction of the low-frequency zigzag wave
52	[:SOURce]:LFOutput:SHAPe(?)	Set low frequency waveform
53	[:SOURce]:LFOutput:STATe(?)	Set low frequency on/off
54	[:SOURce]:LFOutput:OFFSet(?)	Set low frequency range offset
55	[:SOURce]:LFOutput:DUAL:FUNCtion:AMPLitude:PERCent?	Set the amplitude ratio of the dual function generator relative to audio 1
56	[:SOURce]:LFOutput:DUAL: FUNCtion[1]|2:FREQuency?	Set the value of frequency 1 (default) or frequency 2 of the dual function generator
57	[:SOURce]:LFOutput:DUAL:FUNCtion[1]|2:PERCent?	Set the pulse duty factor of the dual function generator relative to audio 1 (default) or audio 2
58	[:SOURce]:LFOutput:DUAL:FUNCtion: POFFset?	Set the phase offset of the dual function generator relative to audio 1
59	[:SOURce]:LFOutput:DUAL:FUNCtion:SHAPe?	Set the output waveform of the dual function generator

60	[:SOURce]:LFOutput:DUAL:FUNCtion:SHAPe:RAMP?	Set the signal output type when the output waveform of the dual function generator is ramp
61	[:SOURce]:LFOutput:FUNCtion[1]|2:FREQuency?	Set the output frequency of the function generator 1|2
62	[:SOURce]:LFOutput:FUNCtion[1]|2:PERCent?	Set the pulse duty factor of the function generator 1|2
63	[:SOURce]:LFOutput:FUNCtion[1]|2:SHAPe?	Set the output waveform of the function generator 1|2
64	[:SOURce]:LFOutput:FUNCtion[1]|2:SHAPe:RAMP?	Set the signal output type when the output waveform of the function generator 1|2 is ramp
65	[:SOURce]:LFOutput:NOISe[1]|2:TYPe?	Set the noise type of the noise generator 1|2
66	[:SOURce]:LFOutput:SWEep:FUNCtion:FREQuency:STARt?	Set the start frequency of the sweep function generator
67	[:SOURce]:LFOutput:SWEep:FUNCtion:FREQuency:STOP?	Set the stop frequency of the sweep function generator
68	[:SOURce]:LFOutput:SWEep:FUNCtion:SHAPe?	Set the output waveform of the sweep function generator
69	[:SOURce]:LFOutput:SWEep:FUNCtion:SHAPe:RAMP?	Set the signal output type when the output waveform of the sweep function generator is ramp
70	[:SOURce]:LFOutput:SWEep:FUNCtion:TIME?	Set the sweep time of the sweep function generator
71	[:SOURce]:LFOutput:SWEep:FUNCtion:TRIGger:MODE?	Set the trigger mode of the sweep function generator
72	[:SOURce]:LFOutput:SWEep:FUNCtion:TRIGger: PERiod?	Set the sweep timer period of the sweep function generator
73	[:SOURce]:LFOutput:SWEep:FUNCtion:TRIGger:TYPE?	Set the trigger type of the sweep function generator
74	[:SOURce]:SWEep:DIRection(?)	Set stepping sweep direction
75	[:SOURce]:SWEep:DWELl(?)	Set stepping Dwell time
76	[:SOURce]:SWEep:POINts(?)	Set number of stepping sweep points
77	[:SOURce]:SWEep:TRIGger:SOURce(?)	Set stepping sweep trigger source
78	[:SOURce]:SWEep:RETRace(?)	Set flyback on/off
79	[:SOURce]:SWEep:STEP:TYPE(?)	Set stepping sweep mode
80	[:SOURce]:SWEep:STARt:TRIGger(?)	Set start stepping sweep trigger mode
81	[:SOURce]:SWEep:MODE(?)	Set sweep mode
82	[:SOURce]:PULM:EXTernal:POLarity(?)	Pulse input inversion on/off
83	[:SOURce]:PULM:INTernal:DELay(?)	Set pulse delay
84	[:SOURce]:PULM:INTernal:FREQuency（？）	Set pulse repetition

85	[:SOURce]:PULM:INTernal:PERiod(?)	Set pulse modulation period
86	[:SOURce]:PULM:INTernal:PWIDth(?)	Set pulse modulation width
87	[:SOURce]:PULM:SOURce(?)	Set pulse source
88	[:SOURce]:PULM:STATe(?)	Set pulse modulation on/off
89	[:SOURce]:PULM:INTernal:JITTered:MODE(?)	Set pulse period jittering mode
90	[:SOURce]:PULM:INTernal:JITTered:PERCent (?)	Set jittering percentage of pulse repetition
91	[:SOURce]:PULM:INTernal:PTRain:DATA	Set pulse string points
92	[:SOURce]:PULM:INTernal:PTRain:DELete	Delete any index point in the pulse string list
93	[:SOURce]:PULM:INTernal:PTRain:POINts	Query number of current pulse string points
94	[:SOURce]:AM[1]|2[:DEPTh]:EXPonential(?)	Set signal AM depth for channel 1 or 2
95	[:SOURce]:AM[1]|2[:DEPTh][:LINear](?)	Set signal AM signal depth for channel 1 or 2 in percentage
96	[:SOURce]:AM[1]|2:INTernal:FREQuency(?)	Set internal AM rate for channel 1 or 2
97	[:SOURce]:AM[1]|2:INTernal:RAMP(?)	Set the signal output type for zigzag FM wave for channel 1 or 2
98	[:SOURce]:AM[1]|2:INTernal:SHAPe(?)	This command sets the AM signal output waveform.
99	[:SOURce]:AM[1]|2:STATe(?)	Set the signal generator AM channel 1 or 2 on/off
100	[:SOURce]:AM:MODE(?)	Set depth AM on/off
101	[:SOURce]:AM:SOURce(?)	Set AM input selection
102	[:SOURce]:AM:MODulation:STATe(?)	Set signal generator AM signal output status
103	[:SOURce]:AM:TYPE(?)	Set AM type
104	[:SOURce]:AM:EXTernal:COUPling(?)	Set AM external input coupling mode
105	[:SOURce]:AM:EXTernal:PATH(?)	Set AM external input channel
106	[:SOURce]:AM[1]|2:INTernal:DUAL:FUNCtion:AMPlitude:PERCent?	Set the amplitude ratio of the dual function generator relative to audio 1
107	[:SOURce]:AM[1]|2:INTernal:DUAL:FUNCtion[1]|2:FREQuency?	Set the frequency of the dual function generator relative to audio 1 when the waveform of AM Path 1 or Path 2 is dual function generator
108	[:SOURce]:AM[1]|2:INTernal:DUAL:FUNCtion[1]|2:PERCent?	Set the pulse duty factor of the dual function generator relative to audio 1 when the waveform of AM Path 1 or Path 2 is dual function generator
109	[:SOURce]:AM[1]|2:INTernal:DUAL: FUNCtion:POFFset?	Set the phase offset of the dual function generator relative to audio 1 when the waveform of AM Path 1 or Path 2 is dual function generator

110	[:SOURce]:AM[1]|2:INTernal:DUAL:FUNCtion:SHAPe?	Set the output waveform of the dual function generator when the waveform of AM Path 1 or Path 2 is dual function generator
111	[:SOURce]:AM[1]|2:INTernal:DUAL:FUNCtion:SHAPe:RAMP?	Set the signal output type when the waveform of AM Path 1 or Path 2 is dual function generator and the output waveform of the generator is ramp.
112	[:SOURce]:AM[1]|2:INTernal:FUNCtion[1]|2:FREQuency?	Set the output frequency of the function generator 1|2 when the waveform of AM Path 1 or Path 2 is function generator 1|2.
113	[:SOURce]:AM[1]|2:INTernal:FUNCtion[1]|2:PERCent?	Set the pulse duty factor of the function generator 1|2 when the waveform of AM Path 1 or Path 2 is function generator 1|2.
114	[:SOURce]:AM[1]|2:INTernal:FUNCtion[1]|2:SHAPe?	Set the output waveform of the function generator 1|2 when the waveform of AM Path 1 or Path 2 is function generator 1|2
115	[:SOURce]:AM[1]|2:INTernal:FUNCtion[1]|2:SHAPe:RAMP?	Set the signal output type when the waveform of AM Path 1 or Path 2 is function generator 1|2 and the output waveform of the generator 1|2 is ramp.
116	[:SOURce]:AM[1]|2:INTernal:NOISe:FUNCtion[1]|2:TYPE?	Set the noise type of the noise generator 1|2 when the waveform of AM Path 1 or Path 2 is noise generator 1|2.
117	[:SOURce]:AM[1]|2:INTernal:SWEep:FUNCtion:FREQuency:STARt?	Set the start frequency of the sweep generator when the waveform of AM Path 1 or Path 2 is sweep generator.
118	[:SOURce]:AM[1]|2:INTernal:SWEep:FUNCtion:FREQuency:STOP?	Set the stop frequency of the sweep generator when the waveform of AM Path 1 or Path 2 is sweep generator.
119	[:SOURce]:AM[1]|2:INTernal:SWEep:FUNCtion:SHAPe?	Set the sweep type of the sweep generator when the waveform of AM Path 1 or Path 2 is sweep generator.
120	[:SOURce]:AM[1]|2:INTernal:SWEep:FUNCtion:SHAPe:RAMP?	Set the signal output type when the waveform of AM Path 1 or Path 2 is sweep generator and the sweep type is ramp.
121	[:SOURce]:AM[1]|2:INTernal:SWEep:FUNCtion:TIME?	Set the sweep time of the sweep generator when the waveform of AM Path 1 or Path 2 is sweep generator.
122	[:SOURce]:AM[1]|2:INTernal:SWEep:FUNCtion:TRIGger:MODE?	Set the trigger mode of the sweep generator when the waveform of AM Path 1 or Path 2 is sweep generator.
123	[:SOURce]:AM[1]|2:INTernal:SWEep:FUNCtion:TRIGger:PERiod?	Set the sweep timer period of the sweep generator when the waveform of AM Path 1 or Path 2 is sweep generator and the trigger type is timed trigger.

124	[:SOURce]:AM[1]|2:INTernal:SWEep:FUNCtion:TRIGger:TYPE?	This command is used to set the trigger type of sweep generator when the waveform of AM Path 1 or Path 2 is sweep generator.
125	[:SOURce]:FM[1]|2:DEViation(?)	Set FM offset for signal channel 1 or 2
126	[:SOURce]:FM[1]|2:INTernal:FREQuency(?)	Set internal modulation rate of signal generator FM channel 1 or 2
127	[:SOURce]:FM[1]|2:INTernal:RAMP(?)	Set the zigzag wave type for zigzag FM wave for channel 1 or 2
128	[:SOURce]:FM[1]|2:INTernal:SHAPe(?)	Set FMsignal output waveform for channel 1 or 2
129	[:SOURce]:FM[1]|2:STATe(?)	Set signal generator FM channel 1 or 2 on/off
130	[:SOURce]:FM:SOURce(?)	Set FM source
131	[:SOURce]:FM:MODulation:STATe(?)	Set FM on/off
132	[:SOURce]:FM:EXTernalL:COUPling (?)	Set FM external input coupling mode
133	[:SOURce]:FM:EXTernal:PATH (?)	Set AM external input channel
134	[:SOURce]:FM[1]|2:INTernal:DUAL:FUNCtion:AMPlitude:PERCent?	Set the amplitude ratio of the dual function generator relative to audio 1 when the waveform of FM Path 1 or 2 is dual function generator
135	[:SOURce]:FM[1]|2:INTernal:DUAL:FUNCtion[1]|2:FREQuency?	Set the frequency of the dual function generator relative to audio 1 when the waveform of FM Path 1 or Path 2 is dual function generator
136	[:SOURce]:FM[1]|2:INTernal:DUAL:FUNCtion[1]|2:PERCent?	Set the pulse duty factor of the dual function generator relative to audio 1 when the waveform of FM Path 1 or Path 2 is dual function generator
137	[:SOURce]:FM[1]|2:INTernal:DUAL: FUNCtion:POFFset?	Set the phase offset of the dual function generator relative to audio 1 when the waveform of FM Path 1 or Path 2 is dual function generator
138	[:SOURce]:FM[1]|2:INTernal:DUAL:FUNCtion:SHAPe?	Set the output waveform of the dual function generator when the waveform of FM Path 1 or Path 2 is dual function generator
139	[:SOURce]:FM[1]|2:INTernal:DUAL:FUNCtion:SHAPe:RAMP?	Set the signal output type when the waveform of FM Path 1 or Path 2 is dual function generator and the output waveform of the generator is ramp
140	[:SOURce]:FM[1]|2:INTernal:FUNCtion[1]|2:FREQuency?	Set the output frequency of the function generator 1|2 when the waveform of FM Path 1 or Path 2 is function generator 1|2
141	[:SOURce]:FM[1]|2:INTernal:FUNCtion[1]|2:PERCent?	Set the pulse duty factor of the function generator 1|2 when the waveform of FM Path 1 or Path 2 is function generator 1|2

142	[:SOURce]:FM[1]|2:INTernal:FUNCtion[1]|2:SHAPe?	Set the output waveform of the function generator 1|2 when the waveform of FM Path 1 or Path 2 is function generator 1|2
143	[:SOURce]:FM[1]|2:INTernal:FUNCtion[1]|2:SHAPe:RAMP?	Set the signal output type when the waveform of FM Path 1 or Path 2 is function generator 1|2 and the output waveform of the generator 1|2 is ramp
144	[:SOURce]:FM[1]|2:INTernal:NOISe:FUNCtion[1]|2:TYPE?	Set the noise type of the noise generator 1|2 when the waveform of FM Path 1 or Path 2 is noise generator 1|2
145	[:SOURce]:FM[1]|2:INTernal:SWEep:FUNCtion:FREQuency:STARt?	Set the start frequency of the sweep generator when the waveform of FM Path 1 or Path 2 is sweep generator
146	[:SOURce]:FM[1]|2:INTernal:SWEep:FUNCtion:FREQuency:STOP?	Set the stop frequency of the sweep generator when the waveform of FM Path 1 or Path 2 is sweep generator
147	[:SOURce]:FM[1]|2:INTernal:SWEep:FUNCtion:SHAPe?	Set the sweep type of the sweep generator when the waveform of FM Path 1 or Path 2 is sweep generator
148	[:SOURce]:FM[1]|2:INTernal:SWEep:FUNCtion:SHAPe:RAMP?	Set the signal output type when the waveform of FM Path 1 or Path 2 is sweep generator and the sweep type is ramp
149	[:SOURce]:FM[1]|2:INTernal:SWEep:FUNCtion:TIME?	Set the sweep time of the sweep generator when the waveform of FM Path 1 or Path 2 is sweep generator
150	[:SOURce]:FM[1]|2:INTernal:SWEep:FUNCtion:TRIGger:MODE?	Set the trigger mode of the sweep generator when the waveform of FM Path 1 or Path 2 is sweep generator
151	[:SOURce]:FM[1]|2:INTernal:SWEep:FUNCtion:TRIGger:PERiod?	Set the sweep timer period of the sweep generator when the waveform of FM Path 1 or Path 2 is sweep generator and the trigger type is timed trigger
152	[:SOURce]:FM[1]|2:INTernal:SWEep:FUNCtion:TRIGger:TYPE?	This command is used to set the trigger type of sweep generator when the waveform of FM Path 1 or Path 2 is sweep generator
153	[:SOURce]:PM[1]|2:DEViation(?)	Set phase offset for phase modulation channel 1 or 2
154	[:SOURce]:PM[1]|2:INTernal:FREQuency(?)	Set internal modulation rate of signal generator phase modulation channel 1 or 2
155	[:SOURce]:PM[1]|2:INTernal:SHAPE:RAMP(?)	Set the zigzag wave direction for zigzag wave of phase modulation channel 1 or 2
156	[:SOURce]:PM[1]|2:INTernal:SHAPe(?)	Set the output waveform of phase modulation signal channel 1 or 2
157	[:SOURce]:PM[1]|2:STATe(?)	Set signal generator phase modulation channel 1 or 2 on/off

158	[:SOURce]:PM:SOURce(?)	Set phase modulation source
159	[:SOURce]:PM:MODulation:STATe(?)	Set signal generator phase modulation signal output status
160	[:SOURce]:PM:EXTernal:COUPling(?)	Set phase modulation external input coupling mode
161	[:SOURce]:PM:EXTernal:PATH(?)	Set phase modulation external input channel
162	[:SOURce]:PM[1]|2:INTernal:DUAL:FUNCtion:AMPlitude:PERCent?	Set the amplitude ratio of the dual function generator relative to audio 1 when the waveform of PM Path 1 or 2 is dual function generator
163	[:SOURce]:PM[1]|2:INTernal:DUAL:FUNCtion[1]|2:FREQuency?	Set the frequency of the dual function generator relative to audio 1 when the waveform of PM Path 1 or Path 2 is dual function generator
164	[:SOURce]:PM[1]|2:INTernal:DUAL:FUNCtion[1]|2:PERCent?	Set the pulse duty factor of the dual function generator relative to audio 1 when the waveform of PM Path 1 or Path 2 is dual function generator
165	[:SOURce]:PM[1]|2:INTernal:DUAL: FUNCtion:POFFset?	Set the phase offset of the dual function generator relative to audio 1 when the waveform of PM Path 1 or Path 2 is dual function generator
166	[:SOURce]:PM[1]|2:INTernal:DUAL:FUNCtion:SHAPe?	Set the output waveform of the dual function generator when the waveform of PM Path 1 or Path 2 is dual function generator
167	[:SOURce]:PM[1]|2:INTernal:DUAL:FUNCtion:SHAPe:RAMP?	Set the signal output type when the waveform of PM Path 1 or Path 2 is dual function generator and the output waveform of the generator is ramp
168	[:SOURce]:PM[1]|2:INTernal:FUNCtion[1]|2:FREQuency?	Set the output frequency of the function generator 1|2 when the waveform of PM Path 1 or Path 2 is function generator 1|2
169	[:SOURce]:PM[1]|2:INTernal:FUNCtion[1]|2:PERCent?	Set the pulse duty factor of the function generator 1|2 when the waveform of PM Path 1 or Path 2 is function generator 1|2
170	[:SOURce]:PM[1]|2:INTernal:FUNCtion[1]|2:SHAPe?	Set the output waveform of the function generator 1|2 when the waveform of PM Path 1 or Path 2 is function generator 1|2
171	[:SOURce]:PM[1]|2:INTernal:FUNCtion[1]|2:SHAPe:RAMP?	Set the signal output type when the waveform of PM Path 1 or Path 2 is function generator 1|2 and the output waveform of the generator 1|2 is ramp
172	[:SOURce]:PM[1]|2:INTernal:NOISe:FUNCtion[1]|2:TYPE?	Set the noise type of the noise generator 1|2 when the waveform of PM Path 1 or Path 2 is noise generator 1|2

173	[:SOURce]:PM[1]|2:INTernal:SWEep:FUNCtion:FREQuency:STARt?	Set the start frequency of the sweep generator when the waveform of PM Path 1 or Path 2 is sweep generator
174	[:SOURce]:PM[1]|2:INTernal:SWEep:FUNCtion:FREQuency:STOP?	Set the stop frequency of the sweep generator when the waveform of PM Path 1 or Path 2 is sweep generator
175	[:SOURce]:PM[1]|2:INTernal:SWEep:FUNCtion:SHAPe?	Set the sweep type of the sweep generator when the waveform of PM Path 1 or Path 2 is sweep generator
176	[:SOURce]:PM[1]|2:INTernal:SWEep:FUNCtion:SHAPe:RAMP?	Set the signal output type when the waveform of PM Path 1 or Path 2 is sweep generator and the sweep type is ramp
177	[:SOURce]:PM[1]|2:INTernal:SWEep:FUNCtion:TIME?	Set the sweep time of the sweep generator when the waveform of PM Path 1 or Path 2 is sweep generator
178	[:SOURce]:PM[1]|2:INTernal:SWEep:FUNCtion:TRIGger:MODE?	Set the trigger mode of the sweep generator when the waveform of PM Path 1 or Path 2 is sweep generator
179	[:SOURce]:PM[1]|2:INTernal:SWEep:FUNCtion:TRIGger:PERiod?	Set the sweep timer period of the sweep generator when the waveform of PM Path 1 or Path 2 is sweep generator and the trigger type is timed trigger
180	[:SOURce]:PM[1]|2:INTernal:SWEep:FUNCtion:TRIGger:TYPE?	This command is used to set the trigger type of sweep generator when the waveform of PM Path 1 or Path 2 is sweep generator
181	[:SOURce]:DM:IQADjustment::GAIN(?)	Set internal IQ gain balance
182	[:SOURce]:DM:IQADjustment:IOFFse(?))	Set I path offset value
183	[:SOURce]:DM:IQADjustment:QOFFse(?)	Set Q path offset value
184	[:SOURce]:DM:IQADjustment:QSKew(?)	Set phase angle between IQ vectors
185	[:SOURce]:DM:IQADjustment[:STATe](?)	Set IQ modulation on/off
186	[:SOURce]:DM:MODulation:ATTenuation(?)	Set IQ signal attenuation
187	[:SOURce]:DM:MODulation:ATTenuation:AUTO(?)	Set IQ signal attenuation on/off
188	[:SOURce]:DM:STATe(?)	Set IQ modulation on/off
189	[:SOURce]:DM:EXTernal:BWIDth[:STATe](?)	Set external broadband I/Q input on/off
190	[:SOURce]:DM:IQADjustment:OUTPut[:STATe](?)	Set I/Q input adjustment on/off
191	[:SOURce]:DM:IQADjustment:OUTPut:ATTen(?)	Set I/Q output adjustment attenuation
192	[:SOURce]:DM:IQADjustment:OUTPut:GAIN (?)	Set I/Q output adjustment gain balance
193	[:SOURce]:DM:IQADjustment:OUTPut:IOFFset(?)	Set I/Q output adjustment I offset
194	[:SOURce]:DM:IQADjustment:OUTPut:UIOFfset(?)	Set I/Q output adjustment I/ offset
195	[:SOURce]:DM:IQADjustment:OUTPut:QOFFset(?)	Set I/Q output adjustment Q offset
196	[:SOURce]:DM:IQADjustment:OUTPut:UQOFfset(?)	Set I/Q output adjustment Q/ offset

197	[:SOURce]:DM:IQADjustment:OUTPut:SKEW (?)	Set I/Q output adjustment orthogonal deviation
198	[:SOURce]:RADio:CUSTom:ALPHa(?)	Set baseband filter factor
199	[:SOURce]:RADio:CUSTom:DATA(?)	Set baseband modulation signal data source
200	[:SOURce]:RADio:CUSTom:DATA:PRAM	Select file code stream file
201	[:SOURce]:RADio:CUSTom:FILTer(?)	Set baseband filter type
202	[:SOURce]:RADio:CUSTom:DATA:FIX4?	Set the value of the code data when fixed 4-bit code is selected as the data source
203	[:SOURce]:RADio:CUSTom:IQData	Transmit IQ data set to the baseband memory
204	[:SOURce]:RADio:CUSTom:MODulation:FSK[:DEViation](?)	Set baseband type to frequency modulation deviation in the FSK mode
205	[:SOURce]:RADio:CUSTom:MODulation:MSK:PHASe(?)	Set baseband type to phase modulation deviation in the MSK mode
206	[:SOURce]:RADio:CUSTom:MODulation[:TYPE](?)	Set the baseband modulation type
207	[:SOURce]:RADio:CUSTom:MODulation:ASK:DEPTh:PERCent?	Set the modulation depth of ASK when the radio modulation type is ASK mode
208	[:SOURce]:RADio:CUSTom:MODulation:UFSK	Set the file when the radio modulation type is user FSK mode
209	[:SOURce]:RADio:CUSTom:MODulation:UIQ	Set the file when the radio modulation type is user IQ mode
210	[:SOURce]:RADio:CUSTom:SRATe(?)	Set baseband code element rate
211	[:SOURce]:RADio:CUSTom:STATe(?)	Set baseband on/off
212	[:SOURce]:RADio:CUSTom:POLarity[:ALL](?)	Set baseband signal phase polarity
213	[:SOURce]:RADio:CUSTom:DENCode(?)	Set command differential encoding on/off
214	[:SOURce]:RADio:CUSTom:VCO:CLOCk(?)	Set baseband sampling clock type
215	[:SOURce]:RADio:CUSTom:TRIGger:EXTernal:SOURce(?)	Set external source of the baseband trigger source
216	[:SOURce]:RADio:CUSTom:TRIGger:EXTernal:SOURce:DELay (?)	Set external delay time of the baseband trigger source
217	[:SOURce]:RADio:CUSTom:TRIGger:EXTernal:SOURce:DELay: STATe(?)	Set external trigger source delay on/off
218	[:SOURce]:RADio:CUSTom:TRIGger:EXTernal:SOURce:SLOPe(?)	Set external trigger source trigger polarity
219	[:SOURce]:RADio:CUSTom:TRIGger:SOURce(?)	Set baseband trigger source type
220	[:SOURce]:RADio:CUSTom:TRIGger:TYPE(?)	Set baseband trigger source trigger mode
221	[:SOURce]:RADio:CUSTom:TRIGger:TYPE:CONTinuous:TYPE(?)	Set baseband continuous trigger type
222	[:SOURce]:RADio:CUSTom:TRIGger:TYPE:GATE:ACTive(?)	Set baseband trigger mode type
223	[:SOURce]:RADio:MTONe:ARB:SETup	Select multitone file for loading

224	[:SOURce]:RADio:MTONe:ARB:SETup:STORe	Store multitone file
225	[:SOURce]:RADio:MTONe:ARB: SETup:TABLe	Configure multitone waveform sequence
226	[:SOURce]:RADio:MTONe:ARB: SETup:TABLe:FSPacing(?)	Set multitone frequency interval
227	[:SOURce]:RADio:MTONe:ARB: SETup:TABLe:NTONes(?)	Set quantity of multitone
228	[:SOURce]:RADio:MTONe:ARB: SETup:TABLe:PHASe:INITialize(?)	Set multitone start phase type
229	[:SOURce]:RADio:MTONe:ARB: SETup:TABLe:PHASeINITialize:SE	Set phase relationship between multitones
230	[:SOURce]:RADio:MTONe:ARB: SETup:TABLe:ROW(?)	Set multitone parameters for a line in the multitone modulation list
231	[:SOURce]:RADio:MTONe:ARB[:STATe](?)	Set multitone on/off
232	[:SOURce]:RADio:TTONe:ARB:ALIGnment(?)	Set dual-tone signal offset position
233	[:SOURce]:RADio:TTONe:ARB:FSPacing(?)	Set dual-tone frequency interval
234	[:SOURce]:RADio:TTONe:ARB[:STATe](?)	Set dual-tone on/off
235	[:SOURce]:RADio:ARB:MODE(?)	Set random wave mode
236	[:SOURce]:RADio:ARB[:STATe](?)	Set random wave on/off
237	[:SOURce]:RADio:ARB:SEQuence	Load random wave file
238	[:SOURce]:RADio:ARB:SEQuence:CLOCk (?)	Set random wave clock type
239	[:SOURce]:RADio:ARB:SCLock:RATE(?)	Set random wave clock frequency
240	[:SOURce]:RADio:ARB:TRIGger:TYPE(?)	Set random wave trigger mode
241	[:SOURce]:RADio:ARB:TRIGger:TYPE:CONTinuous[:TYPE](?)	Set random wave continuous trigger mode
242	[:SOURce]:RADio:ARB:TRIGger:TYPE:SINGle(?)	Set random wave single trigger mode
243	[:SOURce]:RADio:ARB:TRIGger:TYPE:SADVance[:TYPE](?)	Set random wave waveform segment trigger mode
244	[:SOURce]:RADio:ARB:TRIGger:TYPE:GATE:ACTive (?)	Set random wave gate trigger mode
245	[:SOURce]:RADio:ARB:TRIGger:SOURce (?)	Set random wave trigger source
246	[:SOURce]:RADio:ARB:VCO:CLOCk(?)	Set random wave trigger sampling clock
247	[:SOURce]:RADio:ARB:EXTernal:CLOCk:RATE(?)	Set random wave trigger external clock frequency
248	:MEMory:COPY:NAME	Copyfilesfrom signal generator
249	:MEMory:DLEete:NAME	Delete user files
250	:MEMory:MOVE	Rename files from signal generator
251	:MEMory:DATA	Transmit data file
252	:ROSCillator:ADJust:REFerence(?)	Set signalgenerator internal reference
253	:DIAGnostic:INFormation:CCOunt:PON？	Query the accumulative startup times of the instrument

254	:DIAGnostic:INFormation:OTIMe?	Query the instrument firmware date and time stamp
255	:DIAGnostic:SNUM?	Read the system serial number of the signal generator
256	:SYSTem:COMMunicate:GPIB:ADDRess(?)	Set signal generator GPIB address
257	:SYSTem:COMMunicate:GTLocal	Set signal generator to local mode

Annex B Zoom Table of Error Information

Key error field	Error Description
Unleveled	For overpower or no power.
Reference loop unlocked	The reference loop signal inside the signal generator is out of lock.
Decimal loop unlocked	The decimal loop signal inside the signal generator is out of lock.
Local oscillator loop	Local oscillator loop signal inside the signal generator is out of lock.
VCO loop unlocked	The VCO loop signal inside the signal generator is out of lock.
External reference	The signal generator is in an external reference state, which is not an error.

-END OF DOCUMENT-