Ceyear 思仪 1466 series Signal Generator Programming Manual

Ceyear Technologies Co., Ltd.

The Manual applies to the following series signal generators based on the firmware version of 1.0 and higher.

- 1466C signal Generator （6kHz ~ 13GHz）
- 1466D signal Generator （6kHz ~ 20GHz）
- 1466E signal Generator （6kHz ~ 33GHz）
- 1466G signal Generator （6kHz ~ 45GHz）
- 1466H signal Generator （6kHz ~ 53GHz）
- 1466L signal Generator （6kHz ~ 72GHz）
- 1466N Signal Generator (6kHz ~ 90GHz)
- 1466P signal Generator （6kHz ~ 110GHz）

Version: A.2 July. 2024, Ceyear Technologies Co., Ltd.

Service Consultation: 0532-86889847 400-1684191

Technical Support: 0532-86880796

Quality Supervision: 0532--86886614

Fax:0532--86889056

Website: www.ceyear.com

E-mail: techbb@ceyear.com

Address: No. 98, Xiangjiang Road, Huangdao District, Qingdao, Shandong Province

Zip Code: 266555

Foreword

Thank you for choosing and using 1466 series signal generator developed and produced by Ceyear Technologies Co., Ltd.! Integrating high, sophisticated and cutting-edge technologies, our products offer high cost performance among similar products.

We will take the responsibility to maximally meet your needs and provide you with high-quality measuring instruments and first-class after-sales service. We aim to provide "high quality and considerate service", and operate on the principle of making customers satisfactory with our products and services.

Manual No.

2.827.1286SCCN

Version

A.11 2022.11

Ceyear Technologies Co., Ltd.

Manual Authorization

The contents of this manual are subject to change without notice. The contents and terms used in this manual are interpreted by Ceyear Technologies Co., Ltd.

The copyright of the manual belongs to Ceyear Technologies Co., Ltd, no modification or alteration can be made to the manual contents by any unit or person without approval of the Institute, and no reproduction or propagation of the manual can be made for profits, otherwise, Ceyear Technologies Co., Ltd reserves the right of pursuing legal responsibilities from any infringer.

Product warranty

The warranty period of this product is 18 months from the date of shipment. The network analyzer manufacturer will repair or replace the damaged components as required within the warranty period. For this purpose, the user shall return the product to the manufacturer and prepay the mailing fee. The manufacturer will return the fee to the user together with the product after maintenance.

Product quality certificate

This product is guaranteed to meet the specifications in this manual from the date of shipment. The calibration and measurement are completed by measuring bodies with national qualification, with relevant data to be provided for reference by users.

Quality/Environmental Management

This product complies with the quality and environmental management systems during R&D, manufacturing and testing. Ceyear Technologies Co., Ltd. is qualified and has passed ISO 9001 and ISO 14001 management systems.

Safety Precautions

Notice

Notice mark, indicating some important information which will not cause danger. It reminds the user to pay attention to a certain operation process, operation method or the like. Failure to observe the rules or operate correctly may cause damage to the instrument or loss of important data. Proceed to the next step only after fully understanding and meeting the note conditions indicated.

Tips

The "Tips" symbol indicates information tips. It reminds the user to pay attention to the instrument or certain operation process, operation method or the like.
The purpose is to guide the instrument operator to use the instrument correctly.

Table of Contents

1. Manual navigation

This chapter introduces the function, chapter structure and main content of the programming manual for 1466 series signal generator, as well as the instrument related documents provided for users.

1.1. About the manual

This manual introduces the methods for remote control of 1466 series signal generator and application of SCPI. Meanwhile, in order to make it convenient for users to quickly master the programmed 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.**错误!超链接引用无效。错误!超链接引用无效。**

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 1466 series signal generator; I/O function library, introducing the basic concept of instrument driver and basic installation instructions of IVI-COM/IVI-C driver.

● **Remote Control Commands**

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

- **Annex**

  Necessary reference information related to programmed control of 1466 series signal generator is provided, including zoom table of SCPI and zoom table of error message.

1.2 Related Documents

Documents of 1466 series signal generator include:

- Quick Start Guide
- User's Manual
- Remote Control Manual
- Online Help

Quick Start Guide

This manual introduces the basic methods for configuration and start-up measurement of the instrument to enable users to quickly understand the characteristics of the instrument, and master the basic settings and operation methods. Main chapters include:

- Get Prepared
- Typical Applications
- Get Help

User's Manual

This manual describes the functions and operation methods of the analyzer in detail, including configuration, measurement, remote control and maintenance, etc. The purpose is to guide users to fully understand the functional characteristics of the product and master common testing methods of the network analyzer. Main chapters include:

- Manual Navigation
- Overview
- Quick Start
- Operation Guide
- Menus
- Remote Control
- Troubleshooting and Repair
- Technical Indicators and Testing Methods
- Annex

Programming Guide

This manual introduces remote programming basics, SCPI basics, SCPI, programming examples and I/O driver function library in detail. The purpose is to guide users to quickly and comprehensively master the remote control commands and methods of the instrument. Main chapters include:

- Remote Control
- Remote Control Commands
- Programming Examples
- Error Description
- Annex

Online Help

Online help is integrated in the instrument, providing fast text navigation help to make it convenient for users in local and remote control operation. Both the hard keys on the front panel of the instrument or the user interface tool bar offer corresponding shortcut keys to activate this function. Main chapters are identical to those of the User's Manual.

2 Remote Control

This chapter introduces the remote control basics, remote control interface and configuration methods of 1466 series signal generator, and briefly introduces the concept and classification of I/O instrument driver library. The purpose is to facilitate users to start to achieve remote control. Specific contents include:

2.1 Remote Control Basics

2.1.1 Remote Control Interface

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

The remote control interface and related VISA addressing string are described in the table below:

Table 2.1 Remote control interface type and VISA addressing string

Program control interface	**VISA addressing string**	**Description**
LAN (Local Area Network)	**VXI-11 protocol:** TC PIP::Addressograph[::LAN_device_name][::INSTR] **Raw socket protocol:** TC PIP::Addressograph::port::SOCKET	The controller realizes remote control by connecting the network analyzer with the network port on the

		rear panel of the network analyzer. For details of the protocol, please refer to: 2.1.1.1 LAN Interface
GPIB (IEC/IEEE Bus Interface)	GPIB::primary address[::INSTR]	The controller realizes remote control by connecting the network analyzer with the port on the rear panel of the network analyzer. The bus interface standard IEC 625.1/IEEE 418 is met. For details, please refer to: 2.1.1.2 GPIB Interface
USB (Universal Serial Bus)	USB::<vendor ID>::<product_ID>::<serial_number>[::INSTR]	Instrument rear panel port. For details, please refer to: 2.1.1.3 USB Interface

2.1.1.1 LAN Interface

The signal generator can be controlled remotely by computers in LAN 10Base-T or 100Base-T. Various instruments are combined into a system in LAN and controlled uniformly by computers in it. In order to realize remote control in LAN, the signal generator should be equipped with port connector, network card and relevant network protocol in advance, and provided with relevant network services. Meanwhile, the master computer in the network should also be equipped with instrument control software and VISA library in advance. The three working modes of the network card are:

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

The master computer and the signal generator should be connected to the common TCP/IP protocol network through the network port. The cable between the computer and the signal generator is a commercial RJ45 cable (Category 5 cable with or without shielding). During data transmission, the transmission speed of LAN is faster when data packet transmission is applied. Generally, the length of the cable between the computer and the signal generator should not exceed 100m (100Base-T and 10Base-T). For more information about LAN communications, please refer to http://www.ieee.org.错误!超链接引用无效。错误!超链接引用无效。

1) IP address

Physical connection of the network should be guaranteed for remote control on the signal generator via the LAN. It is just required to set the address into the subnet of the host computer through the menu "Local IP" of the signal generator. For example, if the IP address of the host computer is 192.168.12.0, the IP address of the signal generator should be set to 12.XXX, where XXX is the value between 1 and 255. The default network port number used by the signal generator for communication is 5025. Users can configure in the interface or remotely, with the configuration value range of [1024,65535].

When establishing a network connection, only the IP address is required. The VISA addressing string is as follows:

TC PIP::host address[::LAN device name][::INSTR] or
TCPIP: : host address: port: : SOCKET
Where:

- TCPIP represents the network protocol used;

- host address represents the IP address or host name of the instrument, and is used for identifying and controlling the controlled instrument;
- LAN device name defines the handle number of the protocol and subset (optional);
- VXI-11 protocol is selected for device 0;
- More recent high speed LAN instrument protocol is selected for high speed LAN instrument 0;
- INSTR represents the instrument resource type (optional);
- port identifies the socket port number;
- SOCKET represents the raw network socket resource class.

Example:

- The IP address of the instrument is 192.1.2.3, and the effective resource string of the VXI-11 protocol is:
- TCPIP::192.1.2.3::inst0::INSTR
- To establish a raw socket connection, use:
- TCPIP::192.1.2.3::5025::SOCKET

Tips

Method of recognizing multiple instruments in the remote control system

If multiple instruments are connected in the network, the individual IP address and related resource string are used to distinguish. The host computer applies its own 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. The TCP/IP network protocol and associated network services are pre-configured for communication. Such connection-oriented communication, which follows the sequential exchange and can identify the interruption of the connection, ensures no loss of information.

3) Socket communication

The TCP/IP protocol connects a signal source over a LAN socket in the network. As a basic method used in computer network programming, the socket allows applications using different hardware and operating systems to communicate over a network. With this method, two-way communication between the signal generator and the computer is realized through port.

As a software class programmed specially, the socket defines the IP address, device port number and other necessary information for network communication, 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.

Sockets in the signal generator are compatible with Berkeley sockets and Winsock through the application program interface (API). In addition, it is compatible with the API of other standard sockets. When SCPI are used to control the signal generator, the socket program established in the program issues the command. Before using a LAN socket, the socket port number of the signal generator must be set. The socket port number of the signal generator is 5025.

2.1.1.2 GPIB Interface

As an instrument remote control interface widely used at present, GPIB interface is connected to different types of instruments through GPIB cable, so as to build a test system with the host computer. In order to realize remote control, the host computer should be equipped with GPIB bus card, driver and VISA library in advance. During communication, the host computer first addresses the controlled instrument through the GPIB bus address. The user may set the GPIB address and ID query string, and the GPIB communication language may be in the form of SCPI by default.

GPIB and its associated interface operations are defined and described in detail in ANSI/IEEE Standard 488.1-1987 and ANSI/IEEE Standard IEEE Standard 488.2-1992. For details of the standards, please refer to the IEEE website: http://www.ieee.org/http://www.ieee.org.

GPIB processes information in bytes at the data transmission speed of up to 8MBps, which is fast. Since the data transmission rate is limited by the distance between the device/system and the computer, the following points should be noted when connecting GPIB:

- Up to 15 instruments may be built through GPIB interface.
- The total length of the transmission cable should not be more than 15 m or twice the number of instruments in the system. In general, the maximum length of the transmission cable between the devices shall not exceed 2 m;
- If multiple instruments are connected in parallel, a "live" cable is required.
- The end of the IEC bus cable shall be connected to the instrument or master computer.

2.1.1.3 USB Interface

To implement USB programming, a computer and signal generator need to be connected via a USB B-type port with the VISA library installed in advance. VISA automatically detects and configures the instrument to establish a USB connection without the need to enter the instrument address string or install a separate driver.

USB address:

Addressing string format: USB::<vendor ID>::<product ID>::<serial number>[::RAW]

Where:

- <vendor ID> Manufacturer code; This code is fixed at 0x3399 (Ceyear Technologies Co., Ltd.)

  .
- <product ID> represents the instrument code; 1466 系列信号发生器为 0x2800 for 1466 series signal generators
- <serial number> represents the serial number of the network analyzer;

Example:

USB::0x03399::0x2800::200600000::INSTR

0x03399: Manufacturer code (Ceyear Technologies Co., Ltd.);

0x2800: Instrument code (1466 series signal generator);

200600000: serial number of the network analyzer.

2.1.2 Message

The messages transmitted on the data cable are divided into the following two categories:

1) **Interface message**

During communication between the instrument and the host computer, the attention cable should be pulled down first, and then the interface message will be transmitted to the instrument through the data cable. Only instruments with GPIB bus function can send interface message.

2) **Instrument message**

The network analyzer message can be divided into two types as per the transmission direction, namely, command and instrument response. Unless otherwise stated, all remote control interfaces apply instrument message in the same way.

a) **Commands:**

Commands (programming messages) are messages sent by the host computer to the network analyzer for remote control of instrument functions and query of state information. Commands are divided into the following two categories:

- Based on the impact on the network analyzer:

  -- setting commands: change the set state of the instrument, such as reset or setting frequency.

  -- query commands: query and return data, for example: identify the instrument or query the parameter value. Query commands end with the suffix question mark.

- ➢ Based on the definition in the standard:

  -- common commands: with functions and syntax to be defined by IEEE488.2, they are applicable to all types of instruments (if realized)

  The purpose is for management of standard status register, reset and self-detection, etc.

  -- instrument control commands: instrument characteristic commands, used to realize instrument functions, such as setting frequency.

  The syntax also follows the specifications of SCPI.

b) **Instrument responses:**

Instrument responses (response message and service request) are the query result information sent by the instrument to the computer. Such information includes measurement results, instrument status, etc.

2.1.3 SCPI

2.1.3.1 Introduction to SCPI Command

SCPI (Standard Commands for Programmable Instruments) are a command set for all instruments established based on Standard IEEE488.2. The main purpose is to make the same function have the same program command to achieve the universality of remote control commands.

SCPI consist of a command header and one or more parameters. The command header is separated from the parameters by spaces and contains one or more key fields. A command with direct suffix question mark is a query command. Commands are divided into common commands and instrument commands that have different syntactic structures. SCPI have the following characteristics:

1) Programmed control commands are oriented to test function rather than describing instrument operation;
2) Programmed control commands reduce the repetition of similar test function realization process, and ensure the compatibility of programming.
3) Programmed control messages are defined in layers that are hardware independent of the communication physical layer;
4) Programmed control commands are independent of programming methods and languages. The test program of SCPI is easy to transplant.

5) Programmed control commands are scalable and can adapt to different scale of measurement control.

6) SCPI have been a “living” standard for their scalability.

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 program command set, classification and description of 1466 series signal generator, please refer to:

1) "3. Programmed Control Commands" in this Manual; 3 程控命令

2)"Annex A Zoom table of SCPI command" in the user's manual.

2.1.3.2 Description of SCPI

1) General terms

The following terms apply to this section. To better understand the chapters, you shall understand the exact definitions of the terms.

Controller

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

Device

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

Program message

A program message is the combination of one or more SCPI commands that have been correctly formatted. Program messages tell the devices how to measure and output the signals.

Response message

A response message is a set of data of specified SCPI formats. Response messages always come from the devices to controllers or listening devices. Response messages tell the controllers about the internal state or measured values of the devices.

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

Event-type remote control commands cannot be queried. An event command generally has no corresponding front panel key setting, and its function is to trigger an event at a specific time.

Query

A query is a special type of command. When a control device is queried, a response message appropriate to the controller syntax requirements is returned. A query statement always ends with a question mark.

2) Command type

There are two types of SCPI: common commands and instrument commands. Figure 2.1 shows the difference between the two commands. Common commands, defined by IEEE 488.2, are used to manage macros and status registers and for synchronization and data storage. Because common commands all start with an asterisk, they are easy to be recognized. For example, *IDN? *OPC, *RST are all common commands. Common commands are not part of any instrument commands, and the instrument interprets them in the same way regardless of the current path setting of the commands.

Instrument commands are easy to be recognized because they contain a colon (:). A colon is used in the beginning of an expression or between two keywords, for example: FREQuency[:CW?]. According to the internal function module of the instrument, instrument commands are divided into sub-sets of corresponding subsystem commands. For example, the power subsystem (:POWer) contains power-related commands, while the status subsystem (:STATus) contains commands for the status control register.

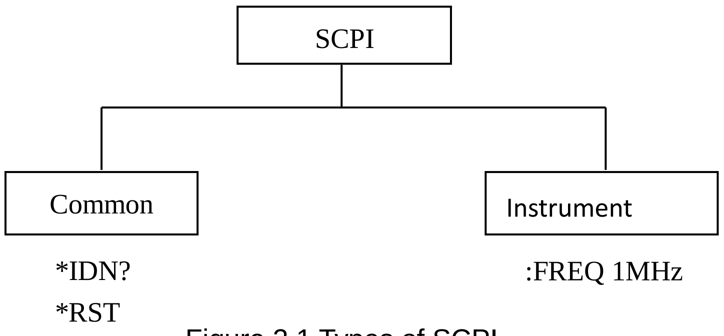


Figure 2.1 Types of SCPI

3) Instrument 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. MINimum|MAXimum immediately following [:LEVel] is the parameter. There is a space between the command and the parameter. Other parts of the syntax expression are described in Table 2.2 and Table 2.3.

Table 2.2 Special characters in command syntax

Symbol	Meaning	Example
|	The vertical bar between the keyword and the parameter represents multiple options.	[:SOURce]:AM:SOURce EXTernal|INTernal EXTernal and INTernal are optional
[]	A square bracket indicates that the contained keyword or parameter is optional when forming a command. The command will be executed even when such implied keyword or parameter is ignored.	[:SOURce]:AM[:DEPTh]:EXPonential? SOURce and DEPTh are optional.
< >	The part in angle brackets indicates that the command is not used in the literal sense. They represent the part that must be contained.	[:SOURce]:FREQ:STOP <val><uint> In the command, <val> and <uint> must be replaced must be replaced with actual frequency and unit. For example: :FREQ:STOP 3.5GHz

{ }	The part in 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 called "flexible listening". See the section "Parameters and Responses of Commands" for more information.	:FREQuency :FREQ,:FREQuency or :FREQUENCY, Either of them is correct.
When a colon is between two command mnemonics, it will move the current path down a level in the command tree. 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 more than one parameter, adjacent parameters are separated by commas. The parameter is not part of the command path, so it does not affect the path layer.	[:SOURce]:LIST:DWELl <val>{,<val>}
The semicolon is used to separate 2 adjacent 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. or within 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, namely, :POWer:LEVel 6.2.

4) Command tree

Most remote control programs apply instrument commands. When parsing such commands, SCPI apply a file system-like structure called command tree, as shown in Figure 2.2:

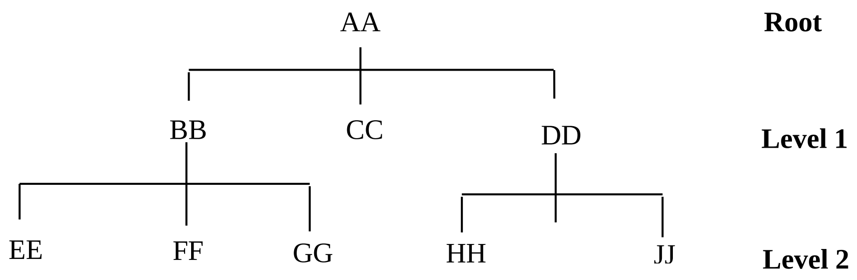


Figure 2.2 Simplified command tree

The top command is the root command, or "root" for short. When a command is parsed, follow a specific path to the next level of command according to the tree structure. For example, in :POWer:ALC:SOURce?, : POWer stands for AA, :ALC stands for BB, :SOURce stands for GG, and the entire command path is (:AA:BB:GG).

A software module in instrument software - **command interpreter**, is responsible for parsing each received SCPI. The command interpreter breaks commands into individual command elements by using a series of rules that distinguish the path of the command tree. After parsing the current command, keep the current command path unchanged. The advantage of this is to parse subsequent commands more quickly and efficiently since that the same command keyword may appear in different paths. After booting or *RST (reseting) the instrument, current command path is reset to root.

5) Command parameters and responses

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

For example, in setting frequency reference state command of signal generator: **:FREQuency:REFerence:STATe ON|OFF|1|0,** the following command formats show that the setting frequency reference function is ON:

The following command formats are all used to set the frequency reference function to on:

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

:FREQ:REF:STAT ON，:FREQ:REF:STAT 1.

Each parameter type has one or more corresponding response data types. During query, a data type will be returned for a numerical parameter, and the response data is precise and strict, known as "**precise speaking**."

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

Table 2.4 Parameter and response types of SCPI

Parameter type	**Response data type**
Numerical	Real number or integer
Extended numerical	Integer
Discrete	Discrete
Boolean	Digital boolean
String	String
Blocks	Finite-length blocks
	Infinite-length blocks
Non-decimal numeric types	Hexadecimal
	Octal
	Binary

Numerical parameters

Numeric parameters can be used in both instrument-specific commands and common commands. A numeric parameter receives all the usual decimal counting methods, including signs, decimals, and scientific notation. In case the device that only receives a value of specified numeric type (e.g., an integer), receives a floating-point data, a prompt will appear to indicate parameter error. All data of numeric types as shown below can be received if they are floating-point data types.

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 commands use extended numeric parameters to specify physical quantities. Extended numerical parameters receive all numeric parameters and additional special values. All the extended numeric parameters receive MAXimum and MINimum as parameter values. Whether other special values, such as UP and DOWN, will be received is determined by the ability of the instrument to parse. All effective parameters will be listed in the table of SCPI.

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

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 millivolt
10DEG	10°
MAXimum	Maximum effective setting
MINimum	Minimum effective setting
UP	Increase by a step
DOWN	Reduce by a step

Discrete parameters

When the number of parameter values to be set are finite, they are identified by discrete parameters. Discrete parameters use mnemonics to represent each valid setting. Like program command mnemonics, discrete parameter mnemonics have two formats, long and short, and allow for mixture of upper and lower cases.

In the following examples, discrete parameters and commands are used together.

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

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

Boolean parameters

A Boolean parameter represents a true or false binary condition, which can only have four possible values.

Examples of Boolean parameters

ON	Logically true
OFF	Logically false
1	Logically true

0	Logically 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

Most of the test data are of real number type, and their formats can be basic decimal notation or scientific notation, which are supported by most advanced programming languages.

Examples of real 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

An integer response data is a decimal expression of an integer value containing signed bit. When querying the state register, most of the response data returned are of integer type.

Examples of integer response data:

0	Sign bit optional
+100	Positive lookahead allowed
-100	Negative lookahead 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.

Examples of discrete response data:

INTernal	Stabilization mode is internal
EXTernal	Stabilization mode is external
MMHead	Stabilization type is millimeter wave source module

Digital bool response data

A binary value 1 or 0 is returned as Boolean response data.

String response data

String response data and string parameters are alike. The main difference is that the separators of string response data are double quotes instead of single quotes. Double quotes can also be embedded in string response data, and there may be no characters between the double quotes.

Here are some examples of string response data:

“This is a string”
“one double quote inside brackets: (“”)”

6) Number system of commands

The value of the command can be entered in binary, decimal, hexadecimal or octal format. When using binary, hexadecimal or octal format, a proper identifier is required before the value. Decimal format (the default format) does not require an identifier, and when a value is entered without a representation, the device will ensure that it is in decimal format. The following list shows the required representations for each format:

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

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

#B101101

#H2D

#Q55

The following example sets the RF output power to 10dBm (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

A command line may contain multiple SCPI. To indicate the end of the current command line, the following methods may be used:

- Enter;

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

Commands on the command line are separated by semicolons, and commands belonging to different subsystems begin with a colon. For example:

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

The command line contains two commands: the first one belongs to the MEM subsystem, and the second belongs to the FREQ subsystem. If adjacent commands belong to the same subsystem with repeated command path, they can be expressed in abbreviation. For example:

FREQ:CW 5GHZ; FREQ:OFFSet 3GHz

The command line contains two commands: both of them belong to the FREQ subsystem, with the same first level. Therefore, the second command can start from the next level of FREQ, and the colon for starting the command can be omitted. It can be abbreviated as follows:

FREQ:CW 5GHz; OFFS 3GHz

8) Measurement unit description

The program control commands provided in this manual support Hz, kHz, MHz and GHz as frequency units. Meanwhile, program control commands related to frequency support parameters without units. If there is no frequency unit, it will be Hz by default. Currently, dBm is supported as power unit. If the parameter has no unit, it will be dBm by default. dB is supported as gain and attenuation unit. If the parameter has no unit, it will be dB by default. ns, us, ms and s are supported as time units. If the parameter has no unit, it will be s by default. The unit types of program command parameters provided in this manual are shown in the parameter range of each command.

9) Description of suffix

Some of the commands included in the programmed control commands in this document contain a suffix, that is, the number following the command, which is usually used to indicate the setting of one of the multiple identical configurations. For example, even if [:SOURce[1]|2]:FREQuency:MODE, [:SOURce[1]|2]:RADio:ARB:MARKer[1]|2|3|4:ONTime is an omissible command, the suffix (if used) cannot be omitted. The SOURce2:FREQuency:MODE means the setting of the frequency generation mode of channel B, in which case SOURce cannot be omitted. If SOURce is omitted, it means the default suffix is used, which is the number in [], i.e., SOURce1 in this example. All SOURce2 in this document indicates RF channel B, and OUTPut2 also represents RF channel B.

10) Description of Option

Some of the programmed control commands in this document are related to the options of the instrument. When the instrument is installed with the corresponding options, the programmed control commands can take effect, otherwise an error message "command not defined" will be generated. Refer to the command subsystem or specific remarks for the relationship between commands and options.

2.1.4 Command Sequence and Synchronization

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

- Sequential commands are sequences of commands that are executed continuously. Each command is usually executed faster.
- An overlapping command is one that is not executed automatically before the next command is executed. It usually takes longer to process overlapping commands, and programs are allowed to process other events synchronously during the period.

Even if there are multiple setting commands on a command line, they may not be executed in the order they were received. To ensure that commands are executed in a certain order, each command must be sent on a separate command line.

Example: the command line contains setting and query commands

If multiple commands on a command line contain query commands, the query results are unpredictable. A fixed value is returned for the following command:

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

Returned value: 1000000000 (1GHz)

The following command returns an unfixed value:

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

The returned result may be the current start frequency value because the host program will delay execution of the command. If the host program executes after receiving the command, the returned result may also be 1GHz. **This signal generator does not support overlapping commands.**

Tips

Setting commands are sent separately from query commands

General rules: in order to ensure the correctness of returned results of query commands, setting commands and query commands should be sent in different program messages.

2.1.4.1 Prevent overlapping execution of commands

In order to prevent overlapping execution of commands, multithreading or commands *OPC, *OPC? or *WAI may be applied, which are executed only after the hardware setting is completed. During programming, the computer may force a period of time to synchronize certain events. The descriptions are shown below:

- **Master program using multi-threading**

Multi-threading is used to wait for command completion and synchronization between the UI and program control, that is, to wait for *OPC? Completion in separate threading without interfering GUI or program threading execution.

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

Table 2.5 Command syntax

Method	Action	Programming method
*OPC	After the command is executed, set it in the operation completion bit of the ESR register.	set ESE BIT0 位为 1 to 1; Set SRE BIT5 to 1; Send overlapping commands and *OPC; Wait for service request (SRQ) Service request represents that the overlapping command has been executed.
*OPC?	Stop executing the current command until 1 is returned. The command is returned only when it is in the operation completion bit of the ESR register, indicating that the previous command has been processed.	Terminate processing of the current command before executing other commands, and send the command directly after the current command.
*WAI	Before the execution of *WAI, wait for all commands to be sent before proceeding with unfinished commands.	Terminate the processing of the current command before executing other commands, and send the command directly after the current command.

In the case that the processing time of overlapping command is short, the command *WAI or *OPC may be used after the overlapping command to achieve command synchronization. In order to execute other tasks synchronously while the computer or instrument is waiting for the completion of overlapping commands, the following synchronization techniques may be applied:

- **OPC and service request**

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

Service request represents that the overlapping command has been executed.

➢ **OPC? and service request**

1) Set the bit4 of SRE: *SRE 16 enable MAV service request;

2) Send overlapping commands and *OPC? ;

3) Wait for service request.

Service request represents that the overlapping command has been executed.

➢ **Event status register (ESE)**

1) Set the OPC mask bit of ESE (bit0): *ESE 1;

2) Send only overlapping commands instead of *OPC, *OPC or *WAI;

3) Send "*OPC;*ESR?" in timer for cyclic query of the operation state.

The returned value (LSB) 1 indicates completion of the overlapping command.

➢ *** OPC? and short timeout**

1) Send only overlapping commands instead of *OPC, *OPC or *WAI

2) Send "<short timeout>; *OPC?" in timer for cyclic query of the operation state;

3) The returned value (LSB) equal to 1 indicates that the execution of the interleave command is complete. In case of timeout, it is during operation.

4) Reset the timeout value to the old value;

5) Send the command "SYStem:ERRor?" to clear the error queue, and delete the message "-410, query interrupt".

The returned value (LSB) 1 indicates completion of the overlapping command.

2.1.5 State Reporting System

The state reporting system stores all operation state information for the current instrument, including error message. They are stored in state registers and error queues respectively, and can be queried through a remote control interface.

2.1.5.1 Structure of State Register

Please refer to the hierarchical structure of status register shown below:

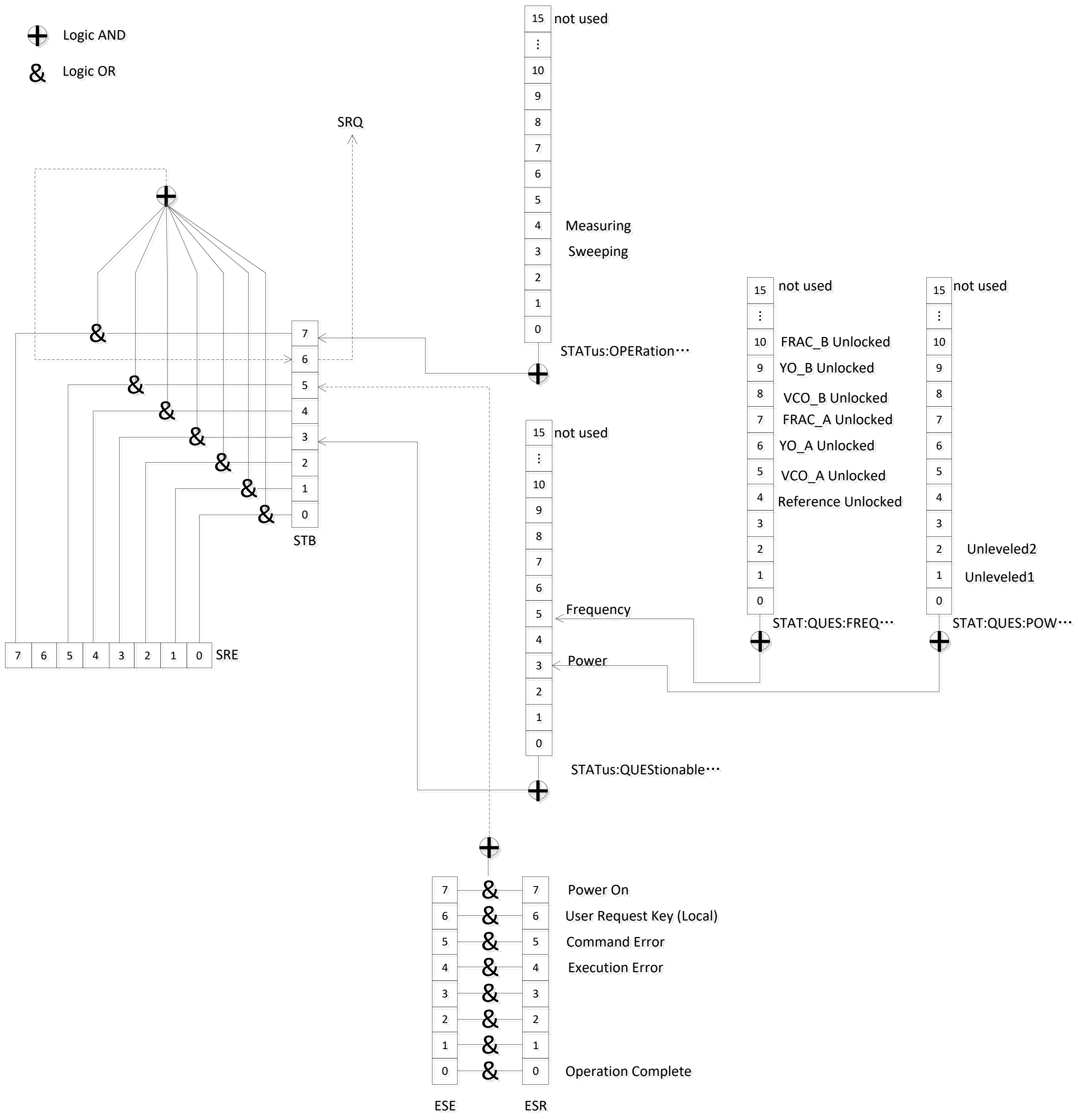


Figure 2.3 Hierarchical structure of the status register

State registers are described by classification below:

1) STB, SRE

The status byte (STB) register and its related mask register - service request enable register (SRE), comprise the top register of the status reporting system. STB saves the general working state of the instrument by collecting the information of lower registers.

2) ESR, SCPI status register

STB receives information from the following registers:

➢ The value of the event status register (ESR) and the event status enable (ESE) mask register.

➢ SCPI status register includes: STATus:OPERation and STATus:QUEStionable registers (SCPI definition),
which contain the specific operation information of the instrument. All SCPI status registers have the same internal structure
(please refer to Section 2.1.5.2 "Structure of SCPI Status Register"
of the program control manual).

3) Output buffer

It stores the messages returned by the network analyzer to the master. It does not belong to the status report system, but determines the MAV position value of the STB.

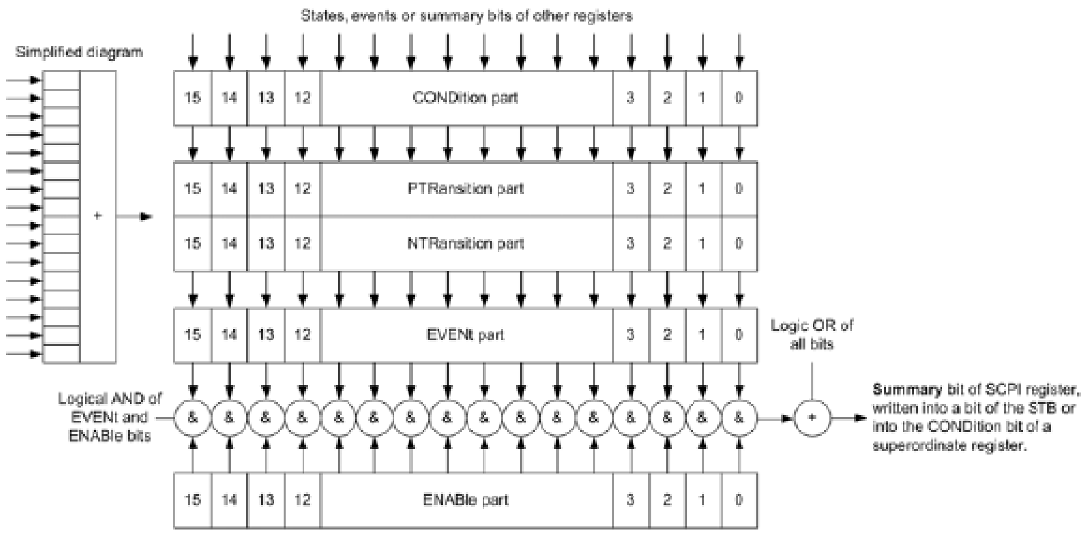
Tips

SRE, ESE

SRE may be used as the enable part of STB. Similarly, ESE may be used as the enable part of ESR.

2.1.5.2 SCPI Status Register Structure

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 state and valid for all five parts of the register. If Bit15 is set to 0, the value of the register is positive


States, events or summary bits of other registers
Simplified diagram
+
15
14
13
12
CONDition part
3
2
1
0
PTRansition part
NTRansition part
EVENt part
Logic OR of all bits
Logical AND of EVENt and ENABle bits
&
+
Summary bit of SCPI register, written into a bit of the STB or into the CONDition bit of a superordinate register.
ENABle part

integer data.

Fig.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/negative transition register**

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

The positive transition register is similar to the conversion filter. When a bit of the condition register is transformed from 0 to 1, relevant PTR bit determines whether the event bit is set to 1, as shown below:

-- PTR bit = 1: the event bit is set.

-- PTR bit = 0: the event bit is not set.

The positive transition register is readable and writable. It reads but not clearing the value.

The negative transition register is similar to the conversion filter. When a bit of the condition register is transformed from 1 to 0, relevant NTR bit determines whether the event bit is set to 1, as shown below:

-- NTR bit = 1: the event bit is set.

-- NTR bit = 0: the event bit is not set.

The positive transition register is readable and writable. It reads but not clearing the 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 network analyzer 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 summary data.

- enable bit = 1: the related event bit acts on the summary data.

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

➢ **Summary bit**

The summary bit of each register consists of the event and the enable part. The result enters the condition part of the upper register. the network analyzer automatically generates the summary bit for each register so that events can cause different levels of service requests.

2.1.5.3 State Register Description

The state 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 not empty Set to this bit to 1 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 message can be queried. Such method effectively reduces errors in program control.
3	Data and bit of question status register Modify the bit to 1 or 0 only when the event bit of question status register and the related enable bit are set to 1. This bit represents a question status of the instrument. Specific instrument status information can be obtained by querying the question status register of the status register.
4	MAV bit (message available) Set this bit to 1 if the output queue information is readable. This bit is used when the controller queries instrument information.
5	ESB bit Summary bit of the event status register. Set this bit to 1 or 0 when one of the bits in the event status register is set and the corresponding bit in the event status enable register is enabled. The bit of 1 indicates a serious error in the instrument. The specific error message can be found by querying the event status register.
6	MSS bit (master status summary bit)

	Set this bit to 1 if the instrument triggers a service request.
7	Summary bit of operation status register Modify this bit to 1 or 0 when the event bit of the operation status register and the corresponding enable bit are set to 1. This bit 1 indicates that the instrument has performed an operation, the type of which can be obtained by querying the operation status register.

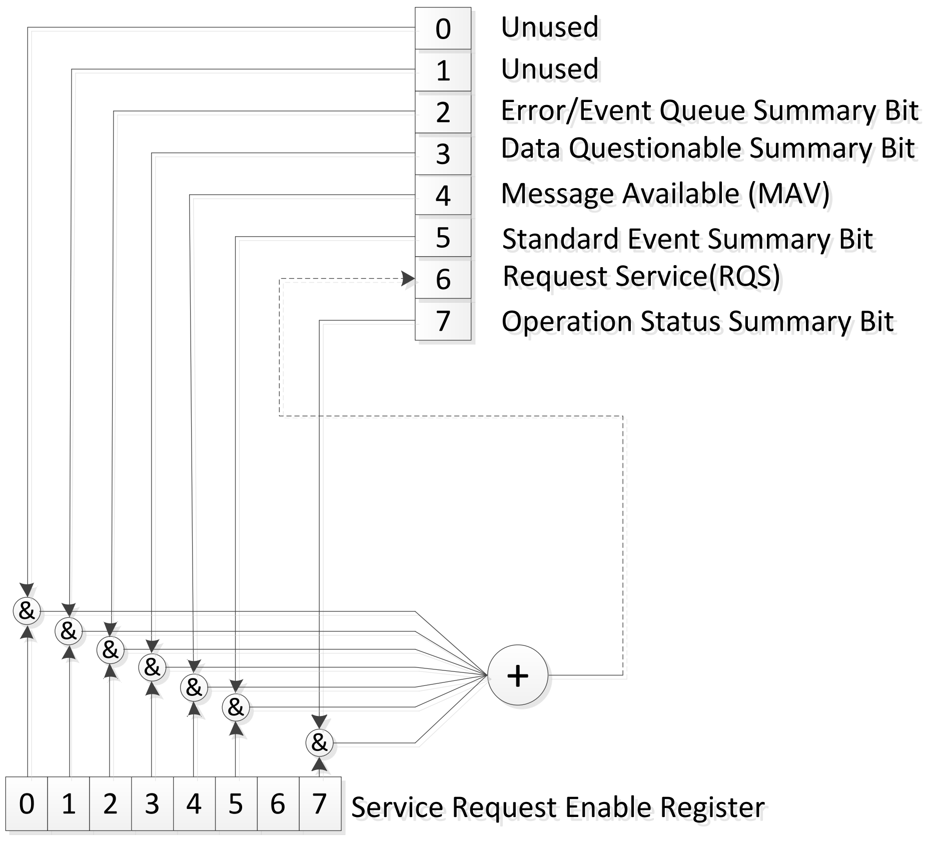

0 Unused
1 Unused
2 Error/Event Queue Summary Bit
3 Data Questionable Summary Bit
4 Message Available (MAV)
5 Standard Event Summary Bit
6 Request Service(RQS)
7 Operation Status Summary Bit
&
+
0 1 2 3 4 5 6 7
Service Request Enable Register

Figure 2.5 Logical structure diagram of status word register and service request enable register

2) Event status register (ESR) and event 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. For details, refer to Table 2.7 Description of event state byte:

Table 2.7 Description of event state byte

Bit	**Meaning**
0	Operation completed

	Set this bit to 1 when the previous commands have been executed and the command *OPC has been received.
1	Not used.
2	Query error (not supported) Set this bit to 1 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	Errors generated in the instrument (not supported) Set this bit to 1 when there is an instrument error. Error code range: -300 - -399, or positive error code. Specific error message can be found in relevant information in the error queue.
4	Execution error Set this bit to 1 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 this bit to 1 when the syntax of the command received is incorrect. Error code range: -100 - -200. Specific error message can be found in relevant information in the error queue.
6	User request Set this bit to 1 when the local key is pressed.
7	Power ON Set this bit to 1 when the instrument is powered on.

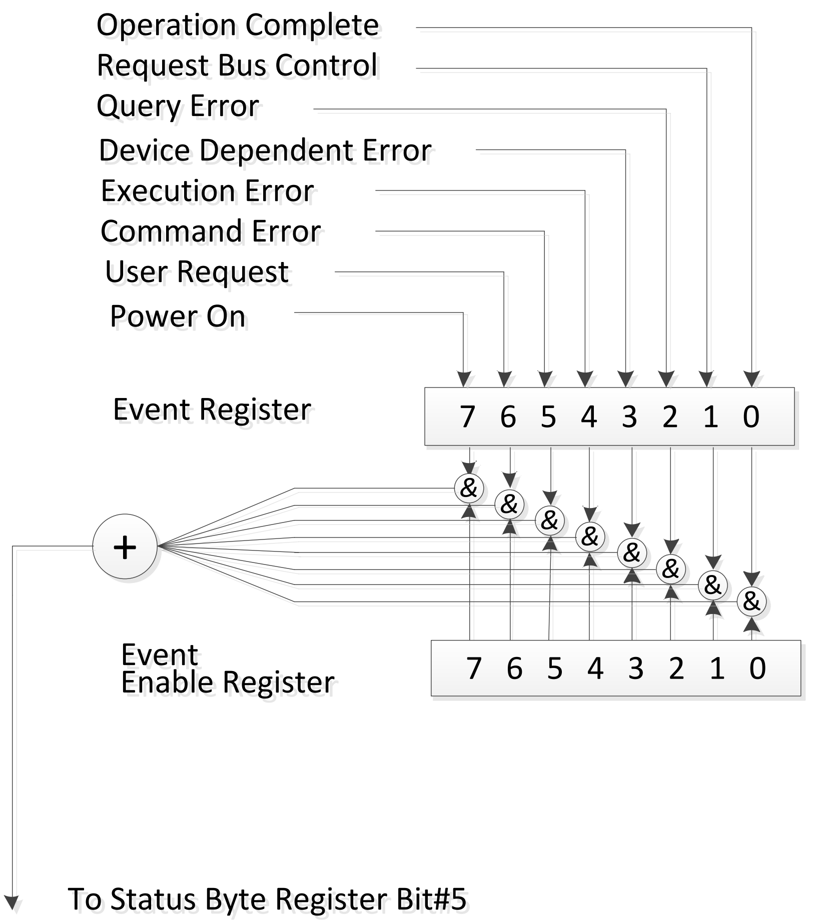


Figure 2.6 Logic structure of event status register (ESR) and event status enable register (ESE)

3) Status: question 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 Status: question register description

Bit	Meaning
0-2	Not used.
3	Set this bit to 1 when the local power setting is wrong.
4	Not used and always 0
5	Set this bit to 1 in case of frequency error of the LO or reference frequency error of any active path.
6	Not used.

7	Not used and always 0.
8	Set this bit to 1 (not supported at present) when the instrument is not calibrated (the prompt "Not calibrated " is displayed on the interface).
9	Set this bit to 1 in case of self-test error.
10-14	Not used
15	The bit is always 0.

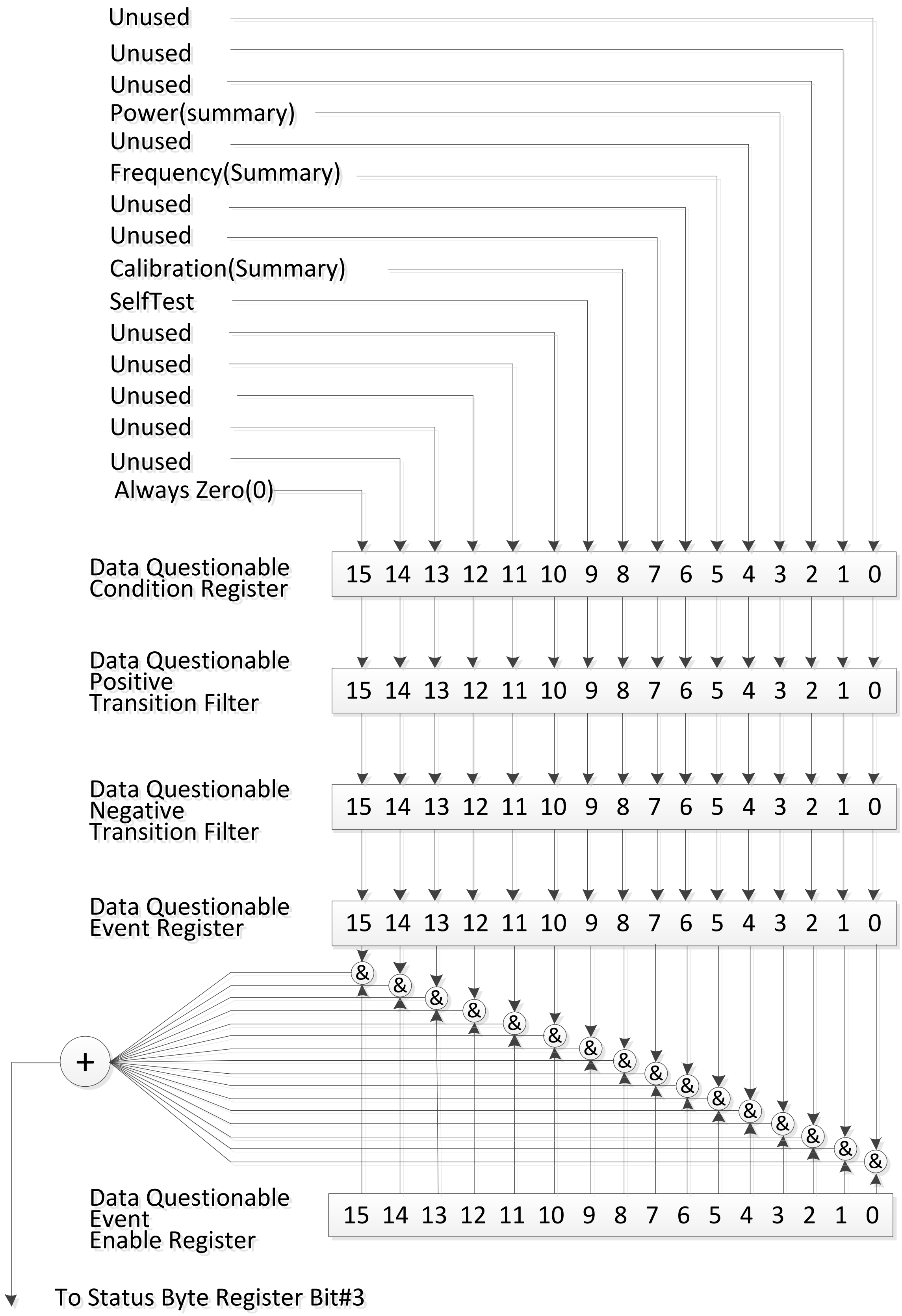


Figure 2.7 Logical structure diagram of question register and its enable register

Tips

Query register

Status: the question 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 question 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) Frequency question status register

The register stores the local oscillator and reference frequency information. Each active path corresponds to a separate frequency register. The register value may be read by using the command STATus:QUEStionable:FREQuency:CONDition? or STATus:QUEStionable: FREQuency [:EVENt]? : Read register value.

The register is described in Table 2.9 below.

Table 2.9 Description of frequency question status register

Bit	Meaning
0	If the bit Synth.Unlocked is 1, it indicates that the synthesizer loses lock.
1	10MHz reference loses lock. If this bit is 1, it indicates losing lock of 10MHz reference signal.
2	10MHz reference loses lock. If this bit is 1, it indicates losing lock of 10MHz reference signal.
3	Not used and always 0.
4	Reference loop loss of lock. If this bit is 1, it indicates losing lock of reference loop.
5	Sampling loop loss of lock. (VCO loop loss of lock). If this bit is 1, it indicates losing lock of channel A sampling loop (VCO loop losing lock)
6	YO loop loss of lock. If this bit is 1, it indicates AYO loop losing lock.
7	Decimal loop losing lock. If this bit is 1, it indicates losing lock of channel A decimal loop.
8	Sampling loop loss of lock. (VCO loop loss of lock). If this bit is 1, it indicates losing lock of channel B sampling loop (VCO loop losing lock)
9	YO loop loss of lock. If this bit is 1, it indicates bYO loop losing lock.

10	Decimal loop losing lock. If this bit is 1, it indicates losing lock of channel B decimal loop.
11-14	Not used.
15	Always 0

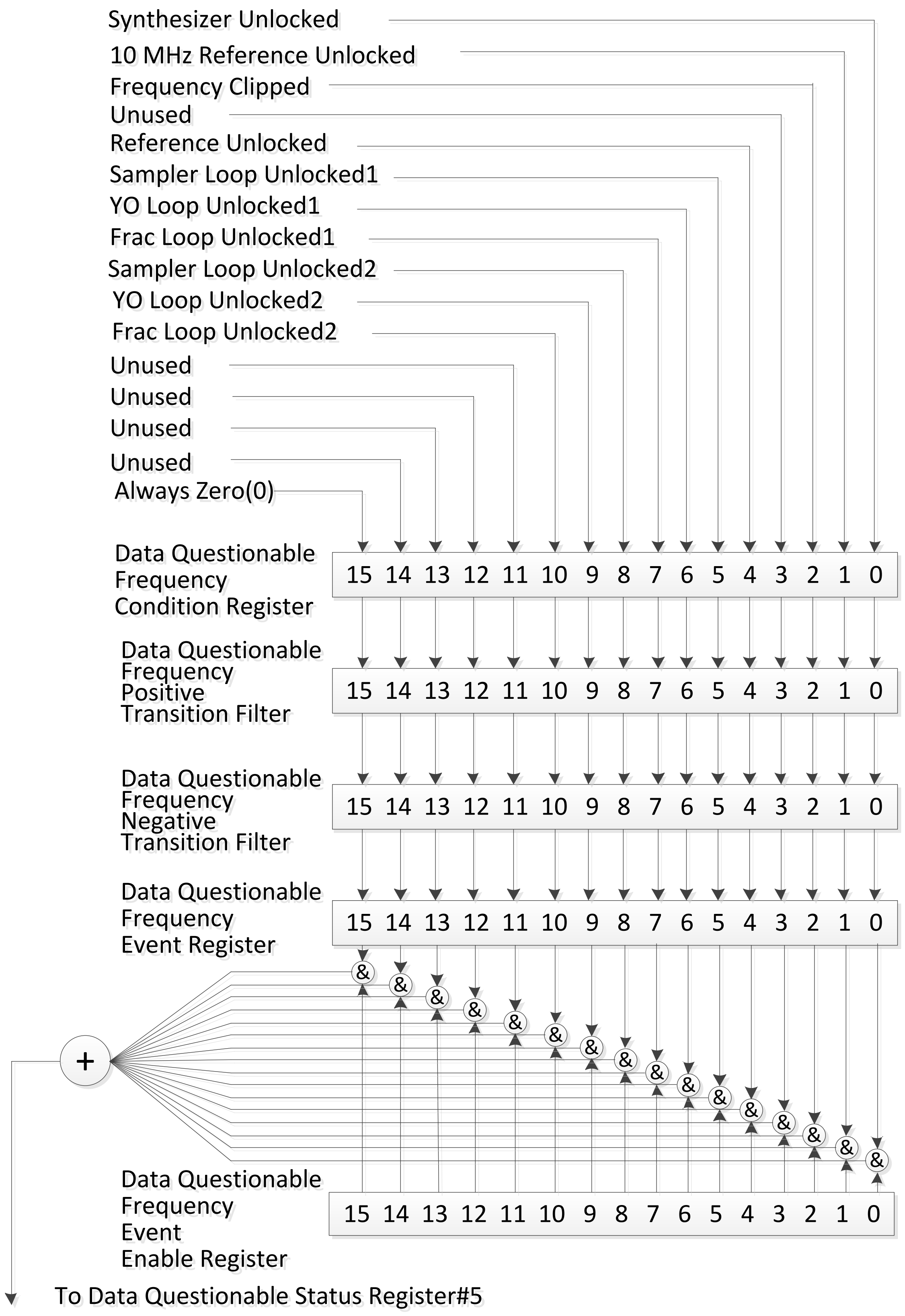


Figure 2.8 Logical structure diagram of frequency question register and its enable register

5) Power question status register

This register contains information about power unfixed amplitude 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]?: Read register value.

The register is described in Table 2.10 below.

Table 2.10 Power question status register description

Bit	**Meaning**
0	Not used
1	If the amplitude is unstable and the bit is 1, the power ALC loop of channel A is out of lock and the power is inaccurate.
2	If the amplitude is unstable and the bit is 1, the power ALC loop of channel B is out of lock and the power is inaccurate.
3-14	Not used.
15	Always 0

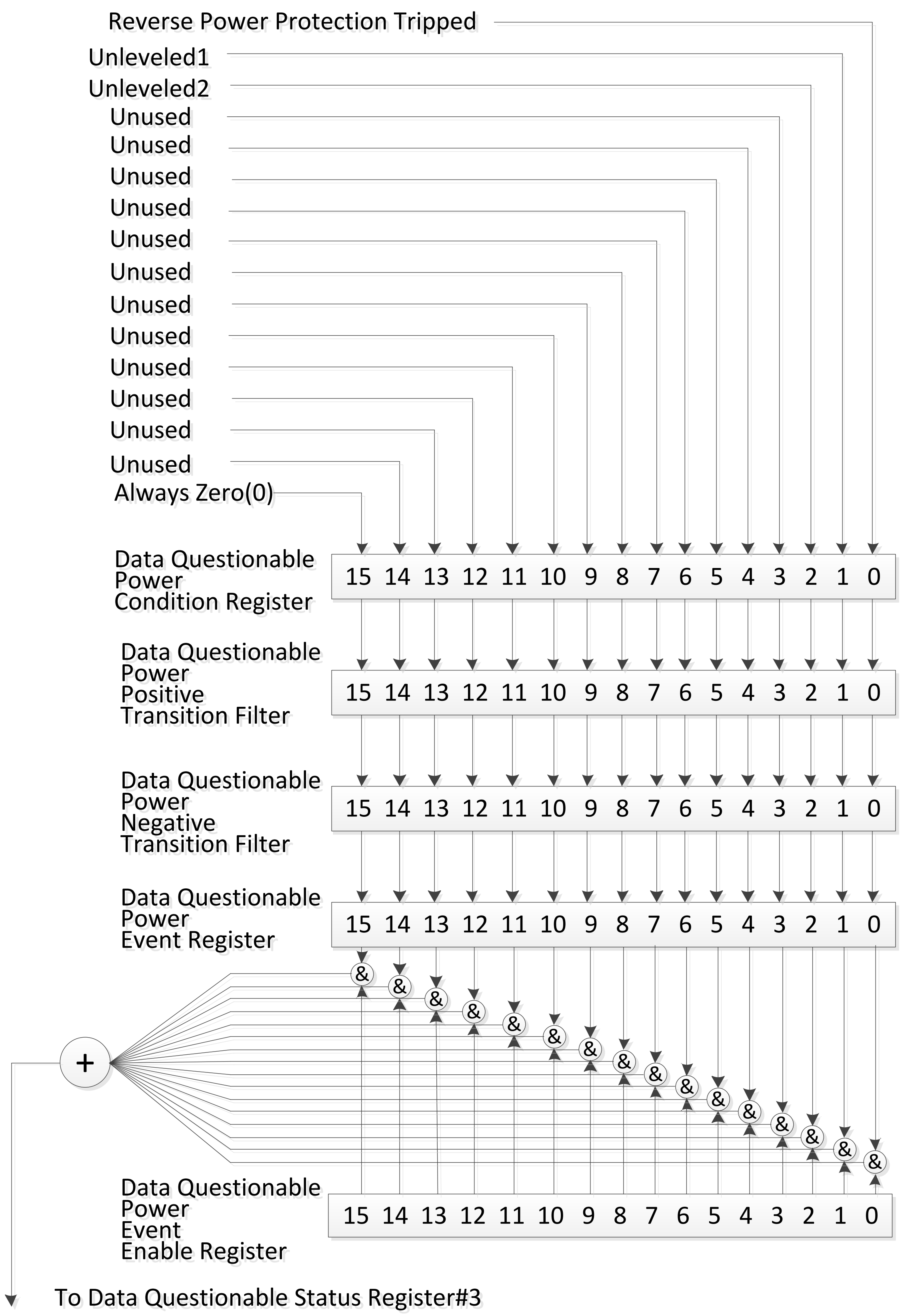

Reverse Power Protection Tripped
Unleveled1
Unleveled2
Unused
Unused
Unused
Unused
Unused
Unused
Unused
Unused
Unused
Unused
Unused
Unused
Always Zero(0)
Data Questionable Power Condition Register
15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0
Data Questionable Power Positive Transition Filter
15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0
Data Questionable Power Negative Transition Filter
15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0
Data Questionable Power Event Register
15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0
&
+
Data Questionable Power Event Enable Register
15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0
To Data Questionable Status Register#3

Figure 2.9 Logical structure diagram of power question register and its enable register

6) Operation state register

Operation state register contains information about the current operation of the instrument and information about the previously executed operations. The operation register value can be read by the command "STATus:OPERation:CONDition?" or "STATus:OPERation[:EVENt]?" to read the operation register value. The register is described in Table 2.11.

Table 2.11 Operation state register description

Bit	Description
0	I/Q calibration in progress. If this bit is 1, it indicates that the I/Q calibration is in progress. (not supported at present)
1	"Hold" state. If this bit is 1, it indicates that the instrument is in its current state. (not supported at present)
2	Not used, and this bit is 0.
3	Sweeping. If this bit is 1, it indicates that the instrument is sweeping. (There is a time difference of 100ms between the time when the sweep state is enabled and when this bit is set)
4	Testing. If this bit is 1, it indicates that the instrument is detecting errors.
5	Wait for triggering. If this bit is 1, it indicates that the signal source is in Wait for Trigger state. (not supported at present)
6-8	Not used. Set to 0.
9	If this bit is 1, it indicates that the signal generator is performing zeroing of DCFM/DCΦM. (not supported at present)
10	Baseband is busy. If this bit is 1, it indicates that the baseband generator is interacting or processing data. Sum of basebands. (not supported at present)
11	Sweep calculation. If this bit is 1, it indicates that the signal generator is sweeping and calculating. (not supported at present)
12-14	Not used. Always 0.
15	Always 0.

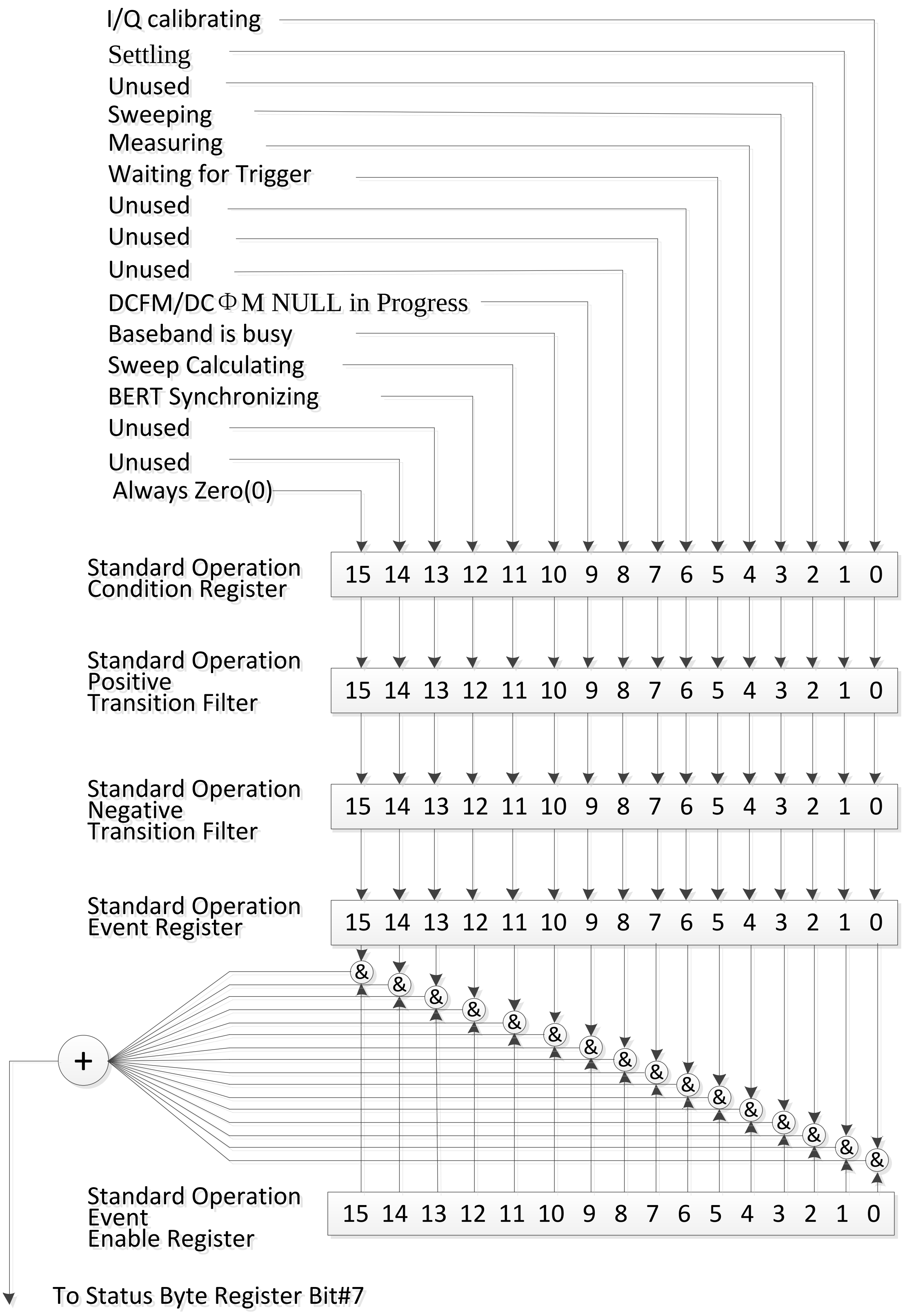

I/Q calibrating
Settling
Unused
Sweeping
Measuring
Waiting for Trigger
Unused
Unused
Unused
DCFM/DCΦM NULL in Progress
Baseband is busy
Sweep Calculating
BERT Synchronizing
Unused
Unused
Always Zero(0)
Standard Operation Condition Register
Standard Operation Positive Transition Filter
Standard Operation Negative Transition Filter
Standard Operation Event Register
Standard Operation Event Enable Register
15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0
&
+
To Status Byte Register Bit#7

Figure 2.10 Logical structure diagram of operation register and its enable register

2.1.5.4 Application of State Reporting System

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

1) Service request (SRQ) initiated by the instrument;
2) 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.
3) Program command to query the status of specific instruments;
4) 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: use the command *OPC to generate SRQ at the end of the sweep.

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.

After instrument setting, the instrument generates a SRQ.

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.

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) Query instrument status

➢ Command set

The content of the state register can be queried or set with the following command set:

:CONDition?

Query the value of the condition register of the state register. The return format is ＜NR1＞. The range is from 0 to 32767. After the query, the value of the condition register remains unchanged.

:ENABle <NRf>|<non-decimal number>

Query or set the event enable register of the state register, and the highest bit (bit15) is always 0.

[:EVENt?]

Query the event register of the state register, and clear the register after the query.

:NTRansition <NRf>|<non-decimal number>

Query or set the negative transition filter of the state register, and the highest bit is always 0.

:PTRansition <NRf>|<non-decimal number>。

Query or set the positive transition filter of the state register, and the highest bit is always 0.

The status registers supported in this case are

:STATus:QUEStionable

:STATus:QUEStionable:FREQuency

:STATus:QUEStionable:POWer

For example:

You can use :CONDition? to query the question status register STATus:QUEStionable

:STATus:QUEStionable:CONDition?

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

- Command *ESR?, *IDN?, *IST?, *STB? 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 register being queried is usually in decimal format and is detected by the controller program. For more details on why SRQ is generated, parallel query is usually done after SRQ.

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:

数据位	7	6	5	4	3	2	1	0
权 重	128	64	32	16	8	4	2	1

Figure 2.11 Relationship between the bit and the weight

➢ Transition Filter

Refer to Section 9.2 of SCPI-99 for a description of the transition filter. A brief description is given below.

1) Positive transition (PTR): It is used to detect the change of a bit value of a condition register from 0 to 1, and set the corresponding bit of the event register to 1.
2) Negative transition (NTR): It is used to detect the change of a bit value of a condition register from 1 to 0, and set the corresponding bit of the event register to 1.
3) Positive or negative transition: It is used to detect the change of a bit value of the condition register from 0 to 1 or from 1 to 0, and set the corresponding bit of the event register to 1.
4) Clearing the positive and negative transition registers will disable the detection of changes in the condition register.

4) 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 state 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 network analyzer.

2.1.5.5 Reset state reporting system

Commands and events for the reset state 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 2.12 Reset status reporting system

Event / Funct	Power On/Off (Powered	DCL, SDC (Instrument	*RST or SYSTem:	STATus: PRESet	*CLS

	state cleared)		**cleared, instrument selected to be cleared)**	**PRESet**		
	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 question 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 before changing settings**
When setting the instrument remotely, first initialize the instrument status (for example, send "*RST"), and then implement the required status setting.

2) **Command sequence**
In general, setting commands and query commands should be sent separately. Otherwise, the returned value of query commands will change with the current order of instrument operation.

3) **Fault response**
Service requests can only be initiated by the network analyzer. The controller program in the test system should guide the network analyzer to initiate service request actively when there is an error, and then enter corresponding interrupt service program for processing.

4) **Error queue**
Each time the controller program processes a service request, it should query the error queue of the network analyzer instead of the state register for a more precise cause of the error. Especially in the test phase of the controller program, the error queue should be frequently queried to obtain the error command sent by the controller to the network analyzer.

2.2 Instrument Program Port and Configuration

2.2.1 LAN

SICL-LAN is used to control 1466 series signal generator in Local Area Network (LAN).

Notice

Application of the connector at the master control port of front-panel USB

Type-A connector on the front panel is the connector at the master control port of USB. In 1466 series signal generator, the port is used to connect the flash disk of USB 2.0 interface to realize the upgrade of resident software of the instrument. It may also be connected to the USB keyboard and mouse to control the signal generator. The port should not be used for program control of the instrument.

2.2.1.1 Connection Establishment

Connect the 1466 series vector generator and external controller (computer) to the LAN with network cable, as shown in Figure 2.13:

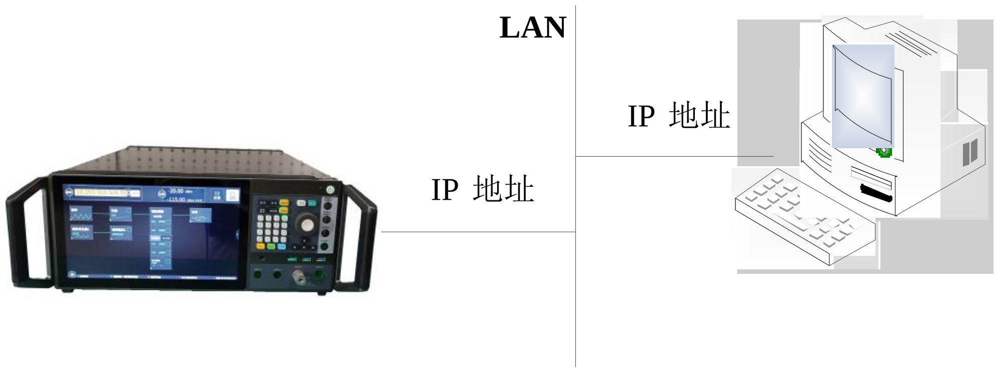
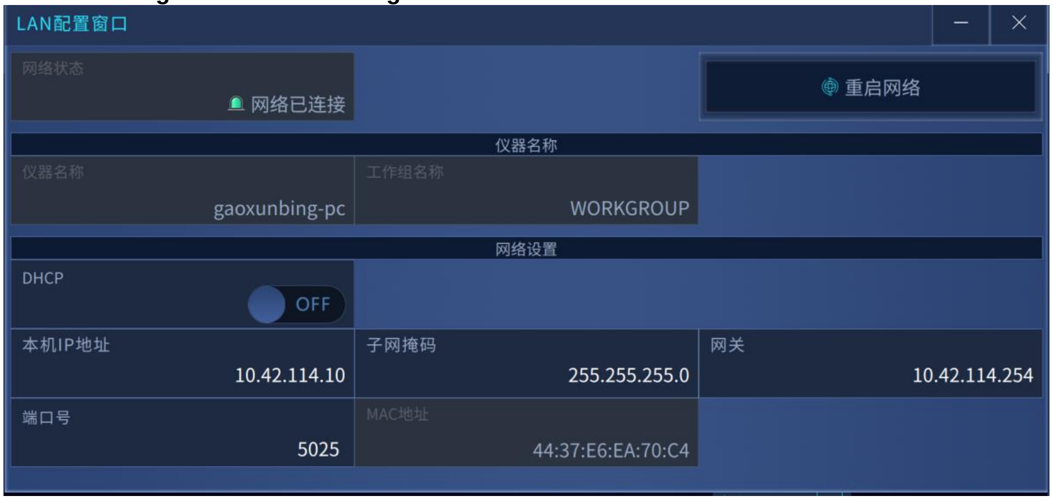


Figure 2.13 LAN interface connection

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 2.14. Set "IP address", "Subnet mask" and "Default gateway" to the subnet where the master controller is located.


LAN配置窗口

网络状态: 网络已连接

重启网络

仪器名称

仪器名称 gaoxunbing-pc	工作组名称 WORKGROUP	

网络设置

DHCP OFF		
本机IP地址 10.42.114.10	子网掩码 255.255.255.0	网关 10.42.114.254
端口号 5025	MAC地址 44:37:E6:EA:70:C4	

Figure 2.14 LAN interface setting

Notice

Ensure that the signal generator is physically connected normally via 10Base-T LAN or 100Base-T LAN cables

Since the signal generator only supports the establishment of a single LAN control system and the setting of static IP address instead of DHCP and host access through DNS and domain name server, users are not required to modify the subnet mask that is set to 255.255.255.0 by the instrument.

2.2.2 GPIB

2.2.2.1 Connection Establishment

Use GPIB cable to connect 1466 series signal generator with external controller (computer), as shown in Figure 2.15:

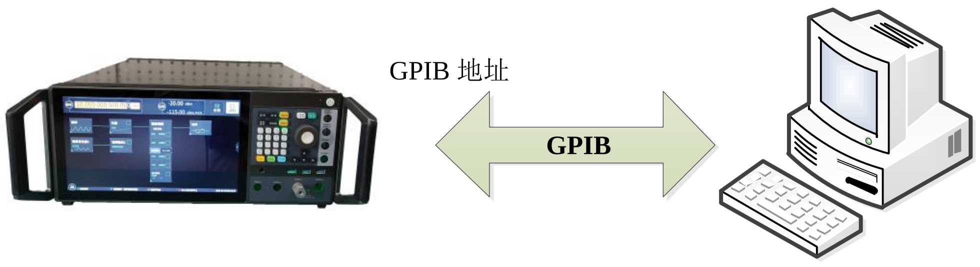
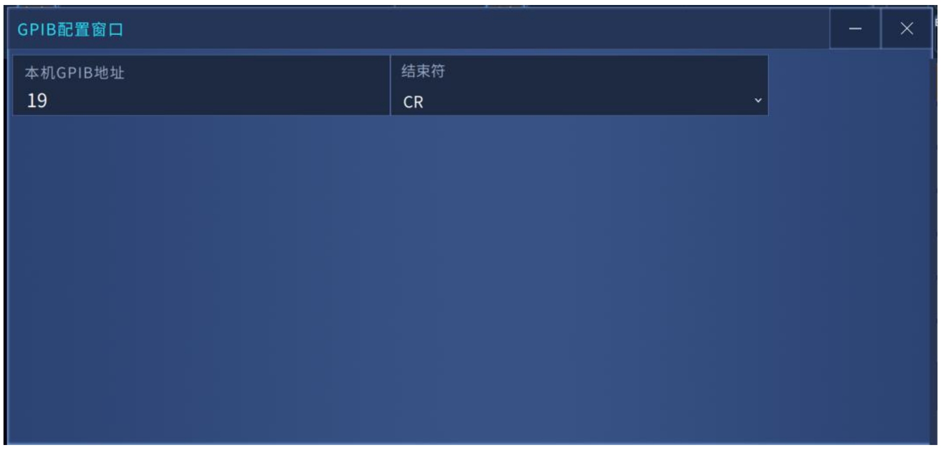


Figure 2.15 GPIB interface connection

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 Port] to go to the interface shown in Figure 2.16, and then use the number key on the front panel to modify in the GPIB address input box of the machine.



本机GPIB地址: 19

结束符: CR

Figure 2.16 GPIB interface setting

2.3 I/O library

2.3.1 Overview of I/O Library

I/O library is a pre-written software library for instruments, known as instrument driver. As a software between the computer and the network analyzer hardware, it consists of the function library, utility program, tool kit, etc. It is a combination of a series of software code modules and corresponds to operation of a plan, such as configuring the network analyzer, reading from the network analyzer, writing to the network analyzer and trigger the network analyzer, etc. Residing in the computer, it is the bridge and link between the computer and the network analyzer. By providing a high-level modular library for convenient programming, users no longer need to learn the complex low-level programming protocol for a specific instrument. Application of instrument driver is the key to develop test and measurement applications quickly.

Functionally, a universal instrument driver generally consists of five parts: functor, interactive developer interface, programmer interface, subprogram interface and I/O interface, as shown in Figure 2.17.

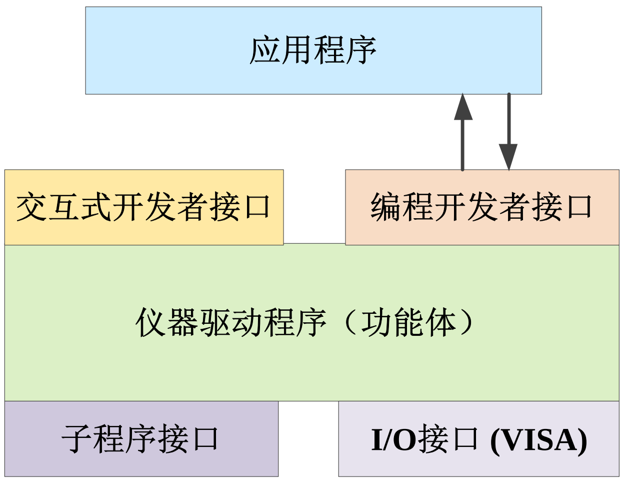


Figure 2.17 Structure model of instrument driver

The details are as follows:

1) Functor. As the main function part of the instrument driver, it may be understood as its framework program.
2) Interactive developer interface. Application development environment that supports instrument driver development is usually provided with graphical interactive developer interface for user convenience. For example, in Labwindows/CVI, the function panel is an interactive developer interface. In the function panel, each parameter of the instrument driver function is represented as a graphical control.
3) Programmer interface. It is a software interface for the application to call instrument driver function, such as dynamic link library file.dll of instrument driver in Windows system.
4) I/O interface. It completes the actual communication between the instrument driver and the instrument. The bus specific I/O software, such as GPIB and RS-232, or the common standard I/O software used across multiple buses, VISA I/O, may be used.
5) Subprogram interface. It is a software interface for the instrument driver to access other support libraries, such as databases, FFT functions, etc. The subprogram interface is used when the instrument driver needs to call other software modules, operating systems, program code libraries and analysis function libraries to complete its task.

2.3.2 Installation and configuration of I/O library

Along with the application in test field, it has gone through different development stages from traditional instrument to virtual instrument. In order to solve the interchangeability of instruments and reusability of test program in automatic test system, instrument driver has gone through different development processes. IVI (Interchangeable

Virtual Instruments) driver is relative popular and common at present. Based on IVI specification, it defines a new instrument programming interface, inserts the class driver and VPP architecture onto the VISA to make the test application and instrument hardware completely independent, adds such unique functions as instrument simulation, range sensing and status cache, improves the operation efficiency of the system, and realizes instrument exchange.

There are two types of IVI driver: IVI-C and IVI-COM, where the latter adopts the form of COM API based on the component object model (COM) of Microsoft, and the former adopts the form of C API based on ANSI C. Both types are designed according to the instrument class defined in the IVI specification and have the same application development environment, including Visual Studio, Visual Basic, Agilent VEE, LabVIEW, CVI/LabWindows, etc.

Two types of driver should be provided at present to meet the needs of different users in different development environments. IVI driver of the signal generator is developed with Nimbus Driver Studio and directly generates IVI-COM and IVI-C driver and program installation package. For specific installation and configuration, please refer to the attached documentation of the control card and I/O library you selected.

IVI driver after installation are divided into IVI inherent function group and instrument function group (basic function group and extended function group). Specific function classification, functions and attributes are shown in the help document of the driver.

Tips

Port configuration and IO library installation

Before using the computer to control the signal generator, please make sure you have the necessary ports and I/O libraries installed and configured correctly.

Tips

Use of I/O library

The driver function panel, help document and driver function examples will be installed automatically when installing the attached IVI-COM/C driver installation package, so as to facilitate users to develop integrated program functions.

3. Program Control Commands

3.1 Description of Commands

This section 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 1466.
- Other instructions.

The sections of common commands and instrument subsystem commands first list the order of command items to make convenient for users to query.

3.2 Common Commands

Common commands are used to control instrument state registers, state 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. IEEE488.2 common command is interpreted and explained below:

Tips

Command use:

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

If a command is used only for setting or querying, or to start an event, the command description will be explained separately.

*CLS

Function description: clear the status. Set the status byte (STB), standard event register (ESR)

and the event part of problem operation register to zero. This command does not change the values of the mask and transition registers, but clears the output buffer.

Setting format: *CLS

Example: *CLS Clear the instrument status

Description: for setting only.

*ESE <Value>

Function description: set or query standard event status enable register. 0 disable. 1 enable.

Setting format: *ESE <value>

Query format: *ESE?

Parameter description:

<Value> Integer value, the binary weighted sum of the individual bits, and the bit mapping is shown in Table 3.1

Range: [0, 255].

Example: *ESE 60 enables the corresponding bits 4+8+16+32, i.e., bits 2, 3, 4, 5.

*ESR?

Function description: Read the value of the event status register and clear the register. Refer to Table 3.1

Query format: *ESR?

Returned value: Integer value, the binary weighted sum of the individual bits, and the bit mapping is shown in Table 3.1

Range: [0, 255].

Description: For query only.

Table 3.1 Standard event bit mapping

Bit	Value	Description
0	1	Operation completed
1	2	Unused
2	4	Query error
3	8	Device-related errors
4	16	Execution error
5	32	Command error
6	64	Local key is pressed
7	128	Start up

*IDN?

Function description: Return the instrument mark.

Query format: *IDN?

Returned value: <ID> "manufacturer, <instrument model>, <serial number>, <firmware version>". For example: Ceyear Technologies,1466,2017008,1.0.0

Example: *IDN?

Description: For query only.

*OPC

Function description: This command is used to set or query bit 0 in the standard event register of the signal generator. When the query returns value 1, it indicates that all received commands have been completed.

Setting format: *OPC

Query format: *OPC?

Example: *OPC? Return 1 if the waiting operation completes, otherwise it waits.

*RCL <Value>

Function description: This command is used to call the instrument status from a register inside the specified signal generator.

Setting format: *RCL <value>

Parameter description: Integer value

Range: [0, 99].

Example: *RCL 1

Description: for setting only.

*RST

Function description: This command is used to complete the signal generator reset function, namely, resetting the instrument to the default state of the manufacture.

Setting format: *RST

Example: *RST resets the signal generator to the default state of the manufacture

Description: for setting only.

*SAV <Value>

Function description: this command is used to store the current status of the signal generator in the specified internal register.

Setting format: *SAV <value>

Parameter description: range [0, 99].

Example: *SAV 2

Description: for setting only.

*SRE <Value>

Function description: This command sets or queries the value of the service request enable register in the signal generator. 0 for disable, and 1 for enable.

Setting format: *SRE <value>

Query format: *SRE?

Parameter description: Integer value, the binary weighted sum of individual bits.

Range [0,63], [128,191]; The bit mapping is shown in Table 3.2; If the 6th bit is set to 1, the instrument will automatically ignore the value of the 6th bit, and the value of the 6th bit is always 0 when querying the returned value.

Example: *SRE 188 sets the value of digits 2, 3, 4, 5, 7 (4+8+16+32+128) to 1.

Table 3.2 Service request register bit mapping

Bit	**Value**	**Description**
0	1	Unused
1	2	Device information

2	4	Error/event queue
3	8	Question state
4	16	Information reception
5	32	Event state bit
6	64	Must be 0
7	128	Operation state

*STB?

Function description: this command is used to query the status byte register in the status reporting system. Refer to Table 3.3 for bit mapping.

Query format: *STB?

Returned value: Integer value, the binary weighted sum of individual bits. Refer to Table 3.3 for bit mapping.

Range: [0, 255].

Example: *STB?

Description: For query only.

Table 3.3 State word

Bit	Value	Description
0	1	Unused
1	2	Device information
2	4	Error/event queue
3	8	Question state
4	16	Information reception
5	32	Event state bit
6	64	Service request
7	128	Operation state

*TRG

Function description: This command is used when the signal generator selects the trigger key as the trigger source to trigger the action and all events waiting to be triggered.

Setting format: *TRG

Example: *TRG

Description: for setting only.

*TST?

Function description: this command is used to set the signal generator for a self-test and return the result of self-test. 0 means that the self-test is passed, and 1 means that there is an error in the self-test. This command will start instrument self-test. Because there are many self-test items, it will take a long time to perform a complete self-test; You must pay attention to the timeout settings. It is recommended to set the timeout to at least 180s.

Query format: *TST?

Returned value: 0: the self-test is passed;
1: the self-test fails.

Example: *TST? Start the self-test and query the test results

Description: For query only.

*WAI

Function description: This command sets the signal generator not to execute any other commands until the current command is completed.

Setting format: *WAI

Example: *WAI command is completed

Description: for setting only.

3.3 Instrument Subsystem Command

This section details the subsystem commands of 1466 series signal generator.

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:ALL[:STATe] <State>

Function description: This command is used to set the RF switch of all channels. It is used for setting only and does not support query. When the signal source has only one channel, this command is equivalent to ":OUTPut[1][:STATe]".

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

Parameter description:

<State> Boolean data, with the values listed below:

ON | 1: All channel RF outputs

OFF|0: All channels RF OFF.

Example: :OUTPut:ALL 1 Set the RF output of all channels of the signal generator to ON.

Reset state: 0

Key path: None

:OUTPut[1]|2:BLANking[:STATe] <State>

Function description: This command sets the state of RF blanking. When blanking is on, if the signal generator is in a point frequency state, the RF output signal will be turned off during frequency switching. If the signal generator is in sweep state, the RF output signal will be turned off during frequency band switching and retrace.

Setting format: :OUTPut[1]|2:BLANking[:STATe] ON|OFF|1|0

Query format: :OUTPut[1]|2:BLANking[:STATe]?

Parameter description:

<State> Boolean data, with the values listed below:

ON | 1: Blanking on,

OFF | 0: Blanking off.

Example: :OUTPut:BLANking 1 enables blanking function of signal generator channel A.

Reset state: 1

Key path: [PWR] -> [LopCtrl] -> [OutBlank ON OFF]

:OUTPut:MODulation:ALL[:STATe] <State>

Function description: This command is used to set the modulation state of all channels. The command only supports settings and does not support query. If no modulation is turned on, the modulation ON/OFF cannot be set to ON. If the modulation is set to ON at this time, a prompt message of "Please select the modulation type" will appear. When the signal generator has only one channel, this command is equivalent to ":OUTPut:MODulation:ALL[:STATe]"

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

Parameter description:

<State> Boolean data, with the values listed below:

ON | 1: Modulation ON,

OFF | 0: modulation OFF.

Example: :OUTPut:MODulation:ALL 1 Set the modulation of all channels to ON.

Reset state: 0

Key path: None

:OUTPut[1]|2:MODulation[:STATe] <State>

Function description: This command sets the modulation output status of each channel of the signal generator. If no modulation of this channel is turned on, the modulation ON/OFF cannot be set to ON. If the modulation is set to ON at this time, a prompt message of "Please select the modulation type" will appear.

Setting format: :OUTPut[1]|2:MODulation[:STATe] ON|OFF|1|0

Query format: :OUTPut[1]|2:MODulation[:STATe]?

Parameter description:

<State> Boolean data, with the values listed below:

ON | 1: RF output,

OFF | 0: RF OFF.

Example: :OUTPut:MODulation 1 Set the signal generator channel A modulation output to ON.

:OUTPut2:MODulation 1 Set the signal generator channel B modulation output to ON.

Reset state: 0

Key path: 【Modulation on/off】

:OUTPut[1]|2[:STATe] <State>

Function description: This command is used to set the RF output of each channel of the signal generator to ON/OFF.

Setting format: :OUTPut[1]|2[:STATe] ON|OFF|1|0

Query format: :OUTPut[1]|2[:STATe]?

Parameter description:

<State> Boolean data, with the values listed below:

ON | 1: RF output,

OFF | 0: RF OFF.

Example: :OUTPut 1 Set RF output of the signal generator channel A to ON.

:OUTPut2 1 Set the signal generator channel B RF output to ON.

Reset state: 0

Key path: [RF 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[1]|2]:FREQuency:CENTer <FreqCenter>

Function description: This command is used to set the center frequency of the step sweep.

Setting format: [:SOURce[1]|2]:FREQuency:CENTer <val>

Query format: [:SOURce[1]|2]:FREQuency:CENTer?

Parameter description:

<FreqCenter> Step sweep center frequency value

Model	Range
1466C	[6kHz ~45GHz]
1466D	[6kHz~20GHz]
1466E	[6kHz ~33GHz]

1466G [6kHz ~ 45GHz].

1466H [6kHz ~ 53GHz].

1466L	[6kHz ~45GHz]
1466N	[6kHz ~90GHz]

1466P [6kHz ~ 53GHz].

Example: :FREQuency:CENTer 2GHz Set the channel A step sweep center frequency to 2GHz.

Key path: [Sweep] - > [Step Sweep] - > [Center Frequency]

[:SOURce[1]|2]:FREQuency[:CW|FIXed] <Frequency>

Function description: This command is used to set the output frequency of the signal generator in continuous wave mode. Please refer to the command"[:SOURce[1]|2]:FREQuency:MODE" for setting of other frequency generation modes.

Setting format: [:SOURce[1]|2]:FREQuency:CW <val>

Query format: [:SOURce[1]|2]:FREQuency:CW?

Parameter description:

<Frequency> output frequency in continuous wave mode.

Model	Range
1466C	[6kHz ~45GHz]
1466D	[6kHz~20GHz]
1466E	[6kHz ~33GHz]

1466G [6kHz ~ 45GHz].

1466H [6kHz ~ 53GHz].

1466L	[6kHz ~45GHz]
1466N	[6kHz ~90GHz]

1466P [6kHz ~ 53GHz].

Example: :FREQuency 10GHz set the point frequency of the signal generator channel A to 10GHz. ,

SOURce2:FREQuency 10GHz set the point frequency of the signal generator channel B to 10GHz.

Reset state: Start frequency + (stop frequency - start frequency) / 2

Key path: [Frequency] -- > [Continuous Wave]

[:SOURce[1]|2]:FREQuency[:CW|FIXed]:AUTO <state>

Function description: This command sets the frequency following ON/OFF. When the frequency following is ON, the continuous wave frequency output by the signal generator will change with the editing of the corresponding frequency value in the internal user flatness list of the instrument.

Setting format: [:SOURce[1]|2]:FREQuency[:CW|FIXed]:AUTO ON|OFF|1|0

Query format: [:SOURce[1]|2]:FREQuency[:CW|FIXed]:AUTO?

Parameter description:

<State> Boolean data, with the values listed below:

ON | 1: Frequency following ON

OFF | 0: Frequency following OFF.

Example: :FREQuency: AUTO 1 Signal generator frequency following ON.

Reset state: 0

Key path: [Power]—> [User Calibration Compensation]—>[Select File]—> [Edit Calibration Compensation File]—> [Frequency Following ON/OFF]

-

[:SOURce[1]|2]:FREQuency:MODE <Mode>

Function description: Setting the frequency generation mode of the signal generator

Setting format: [:SOURce[1]|2]:FREQuency:MODE FIXed|CW|STEP|LIST|ANALog

Query format: [:SOURce[1]|2]:FREQuency:MODE?

Parameter description:

<Mode> discrete data, frequency generation mode. Values are taken as follows:

FIXed|CW	the meaning for setting of the two discrete parameters is the same in this signal generator, that is, when the signal generator is controlled to output continuous wave (point frequency) signal, this mode will stop the frequency sweep signal currently output by the instrument. When the parameter value is set to FIXed or CW, the query returns value FIX. For setting of point frequency, refer to command "[:SOURce[1]|2]:FREQuency[:CW|FIXed] ".
STEP	this parameter is used to set the current frequency generation to step sweep mode.

LIST	Set the frequency generation mode to list; To select the list sweep mode, you need to select a list file; If the list file is not selected, the signal generator will generate an execution error and keep the original frequency generation mode unchanged. At least one sweep point is stored in the list before the signal generator can start the sweep.
ANALog	Set the frequency generation mode to ramp sweep; This parameter is valid when the ramp sweep option is installed, otherwise, a "parameter not allowed" error will be generated.

Example: :FREQuency:MODE LIST set the signal generator channel A to list sweep mode.

Reset state: FIXed

Key path: [Sweep]—>[Sweep Mode] —>[Frequency Generation Mode]

[:SOURce[1]|2]:FREQuency:MULTiplier <FreqMult>

Function description: This command sets the frequency multiplier for the source frequency. When the frequency multiplier is set to a value greater than 1, the multiplier indicator "Multiplier" will be displayed at right of frequency editing area. At this time, the displayed frequency value = RF output frequency value * multiplier factor, but the real frequency output is still the frequency before multiplying the multiplier factor. When the frequency multiplier is set to 1, the indicator will disappear.

Setting format: [:SOURce[1]|2]:FREQuency:MULTiplier <val>

Query format: [:SOURce[1]|2]:FREQuency:MULTiplier?

Parameter description:

<FreqMult> Integer data type, multiplier factor.
Range: [1, 72].

Example: :FREQuency:MULTiplier 8 The frequency multiplier of channel A of the signal generator is 8.
:SOURce2:FREQuency:MULTiplier 2 The frequency multiplier of channel B of the signal generator is 2.

Reset state: 1

Key path: [Frequency] -> [Base Config]->[Freq Mul]

[:SOURce[1]|2]:FREQuency:OFFSet <FreqOffs>

Function description: This command sets the frequency offset value. When the frequency offset is not set to zero, the offset indicator "Offset" will be displayed at right of the frequency editing box, and the displayed value becomes the frequency after adding the offset. At this time, the displayed frequency value = RF output frequency value * frequency multiplier + frequency offset, but the real frequency output is still the frequency before multiplying the frequency multiplier and adding the frequency offset. When the frequency offset is set to zero, the indicator will disappear.

Setting format: [:SOURce[1]|2]:FREQuency:OFFSet <val>

Query format: [:SOURce[1]|2]:FREQuency:OFFSet?

Parameter description:

<FreqOffs> frequency offset.
Range: [-325GHz～ +325GHz].

Example: :FREQuency:OFFSetr 10GHz the frequency offset of the signal generator channel A is 10GHz.

Reset state: 0Hz

Key path: [Frequency] -> [Freq Settings] -> [Freq Offset]

[:SOURce[1]|2]:FREQuency:REFerence <FreqRef>

Function description: This command is used to set frequency reference value, and the set value may be used normally when frequency reference is set to ON state. Please refer to the command“[:SOURce[1]|2]:FREQuency:REFerence:STATe”. The frequency reference value will be subtracted from any continuous wave output signal set at this time. For example, if the current continuous wave output frequency is 1GHz and the frequency reference is set to 1GHz, the displayed continuous wave output frequency will be based on the frequency reference 0Hz. Therefore, 0 Hz will be displayed in the frequency display area, and the actual output frequency of the signal generator is 1GHz. If the continuous wave frequency is set to 1MHZ, 1MHZ will be displayed in the frequency display area, and the actual output frequency is 1.001GHz.

Setting format: [:SOURce[1]|2]:FREQuency:REFerence <val>

Query format: [:SOURce[1]|2]:FREQuency:REFerence?

Parameter description:

<FreqRef> frequency reference.
[0Hz,50GHz]

Example: :FREQuency:REFerence 10GHz. This example shows that the frequency reference of the signal generator channel A is set to 10GHz.

Reset state: 0Hz

Key path: [Frequency] -> [Freq Settings] -> [Freq Ref]

[:SOURce[1]|2]:FREQuency:REFerence:SET

Function description: This command sets the frequency reference value to 0.

When this command is set, the frequency reference of the signal generator will be changed to 0. For the frequency reference setting command, refer to "[:SOURce[1]|2]:FREQuency:REFerence ".

Setting format: [:SOURce[1]|2]:FREQuency:REFerence:SET

Example: :FREQuency:REFerence:SET The signal generator frequency reference is set to 0.

Key path: for setting only.

[:SOURce[1]|2]:FREQuency:REFerence:STATe <State>

Function description: This command is used to set the frequency reference to ON/OFF state. When the frequency reference is set to ON state and the continuous wave frequency of the signal generator is changed, the frequency reference indicator "Reference" will be displayed at right of the frequency editing box. The frequency value displayed in the frequency display area is based on the frequency reference. Please refer to [:SOURce]:FREQuency:REFerence“[:SOURce[1]|2]:FREQuency:REFerence” for setting of frequency reference; when it is set to OFF state, the frequency value displayed in the frequency display area is the actual continuous wave frequency of the signal generator. [:SOURce]:FREQuency:REFerence <Freq

Setting format: [:SOURce[1]|2]:FREQuency:REFerence:STATe ON|OFF|1|0

Query format: [:SOURce[1]|2]:FREQuency:REFerence:STATe?

Parameter description:

<State> Boolean data, with the values listed below:

ON | 1: Frequency reference ON,

OFF | 0: frequency reference OFF.

Example: :SOURce2:FREQuency:REFerence:STATe 1 the frequency reference of the signal generator channel B is set to ON state.

Reset state: 0

Key path: [Frequency] -- >[Freq Settings] -- >[Freq Ref ON/OFF]

[:SOURce[1]|2]:FREQuency:SPAN <FreqSpan>

Function description: This command sets the frequency sweep span of step sweep.

Setting format: [:SOURce[1]|2]:FREQuency:SPAN <val>

Query format: [:SOURce[1]|2]:FREQuency:SPAN?

Parameter description: If the span is negative, it indicates that the start frequency is greater than the stop frequency

<FreqSpan> Width of step sweep frequency:

Model	Range

1466C	[6kHz ~45GHz]
1466D	[6kHz~20GHz]
1466E	[6kHz ~33GHz]
1466G [6kHz ~ 45GHz].	
1466H	[-53GkHz ~53GHz]
1466L [6kHz ~ 53GHz].	
1466N	[6kHz ~90GHz]
1466P [6kHz ~ 53GHz].	

The specific sweep span range is also related to the setting of center frequency, the minimum frequency of the current model is ≤ center frequency - frequency span/2; And the center frequency + frequency span/2 is ≤ the maximum frequency of the current model.

Example: [:SOURce[1]|2]:FREQuency:SPAN 10GHz Set the sweep span of channel A step sweep to 10GHz.

Key path: [Sweep]—>[Step Sweep]—>[Frequency Span]

[:SOURce[1]|2]:FREQuency:STARt <StartFreq>

Function description: This command sets the start frequency of the instrument step sweep or ramp sweep; When the set frequency generation mode is ramp, the start frequency of the ramp sweep set or queried by this command; When the frequency generation mode is non-ramp, the start frequency of the step sweep. For commands on frequency generation mode, refer to "[:SOURce[1]|2]:FREQuency:MODE". For the setting of stop frequency of step or ramp sweep, refer to "[:SOURce[1]|2]:FREQuency:STOP".

Setting format: [:SOURce[1]|2]:FREQuency:STARt <val>

Query format: [:SOURce[1]|2]:FREQuency:STARt?

Parameter description:

<StartFreq> sweep start frequency.

Model	Step sweep range	Ramp sweep range
1466C	[6kHz~13GHz]	[50MHz~Stop Frequency]
1466D	[6kHz~20GHz]	[50MHz~Stop Frequency]
1466E	[6kHz~33GHz]	[50MHz~Stop Frequency]
1466G	[6kHz~45GHz]	[50MHz~Stop Frequency]
1466H	[6kHz~53GHz]	[50MHz~Stop Frequency]

1466L	[6kHz~53GHz]	[50MHz~Stop Frequency]
1466N	[6kHz~90GHz]	[50MHz~Stop Frequency]
1466P	[6kHz~110GHz]	[50MHz~Stop Frequency]

Example: :FREQuency:STARt 1MHz set the step sweep start frequency of channel A to 1MHz.

Reset state: 100MHz

Key path: [Sweep] -- >[Step Sweep] -- >[Freq Start]
[Swp] -- >[Slope Sweep] -- >[Freq Start]

[:SOURce[1]|2]:FREQuency:STEP[:INCRement] <FreqStep>

Function description: Setting the power step value. After setting, when the frequency step is set to ON and the continuous wave frequency of the signal generator is set, refer to command "[:SOURce[1]|2]:FREQuency[:CW|FIXed]". When using UP|DOWN to change the frequency, the variation of frequency will change according to the currently set step.

Setting format: [:SOURce[1]|2]:FREQuency:STEP[:INCRement] <val>

Query format: [:SOURce[1]|2]:FREQuency:STEP[:INCRement]?

Parameter description:

<FreqStep> frequency step, its value is as follows.
[0.001Hz, the current model can be set to the maximum frequency - the current model can be set the minimum frequency]

Example: :FREQuency: STEP 1MHz The channel A frequency step of signal generator is 1MHz.

Reset state: 100MHz

Key path: [Frequency]—> [Frequency Settings]-> [Frequency Step]

[:SOURce[1]|2]:FREQuency:STEP:STATe <State>

Function description: This command sets the frequency step ON/OFF state. When it is set to ON, the frequency settings can refer to "[:SOURce[1]|2]:FREQuency[:CW|FIXed]". By setting the parameters UP and DWON, the frequency can be increased or decreased by the set frequency step value.

Setting format: [:SOURce[1]|2]:FREQuency:STEP:STATe ON|OFF|1|0

Query format: [:SOURce[1]|2]:FREQuency:STEP:STATe?

Parameter description:

<State> Boolean data, with the values listed below:
ON | 1: Frequency step ON:

OFF | 0: Frequency step OFF.

Example: :FREQuency:STEP:STATe 1 Set the frequency step of signal generator channel A to ON.

Reset state: 0

Key path: [Frequency]—> [Frequency Settings]—>[Frequency Step ON/OFF]

[:SOURce[1]|2]:FREQuency:STOP <StopFreq>

Function description: This command sets the stop frequency of the instrument step sweep or ramp sweep; When the frequency generation mode is set as ramp, the stop frequency of the ramp sweep will be set or queried by this command; When the frequency generation mode is set as non-ramp, the stop frequency of the step sweep will be set or queried. For commands on frequency generation mode, refer to "[:SOURce[1]|2]:FREQuency:MODE". For the setting of start frequency of step sweep, refer to command "[:SOURce[1]|2]:FREQuency:STARt".

Setting format: [:SOURce[1]|2]:FREQuency:STOP <val>

Query format: [:SOURce[1]|2]:FREQuency:STOP?

Parameter description:

<StopFreq> sweep stop frequency.

Model	Step sweep range	Ramp sweep range
1466C	[6kHz~13GHz]	[Start Frequency~13GHz]
1466D	[6kHz~20GHz]	[Start Frequency~20GHz]
1466E	[6kHz~33GHz]	[Start Frequency~33GHz]
1466G	[6kHz~45GHz]	[Start Frequency~45GHz]
1466H	[6kHz~53GHz]	[Start Frequency~53GHz]
1466L	[6kHz~53GHz]	[Start Frequency~53GHz]
1466N	[6kHz~90GHz]	[Start Frequency~90GHz]
1466P	[6kHz~110GHz]	[Start Frequency~110GHz]

Example: :FREQuency:STOP 100MHz Set the stop frequency of channel A step sweep to 100MHz.

Reset state: It depends on the model. If the model is 20GHz, it will be 20GHz.

Key path: [Swp] -- >[Step Sweep] -- >[Freq Stop]
[Swp] -- >[Slope Sweep] -- >[Freq Stop]

3.3.3 POWer Subsystem

The power subsystem command is used to control the common functions of RF output signal power level.

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

[:SOURce[1]|2]:POWer:ALC:BANDwidth|BWIDth <AlcBandWidth>

Function description: This command is used to set the bandwidth of the ALC (automatic leveling control) loop. It is applicable to the bandwidth setting of ALC loop in different states when the signal generator outputs different frequency bands. The user may select four states: AUTO, HIGH, MEDium and LOW.

Note:

1. When the internal baseband of the instrument is opened, the bandwidth selected is invalid, and the appropriate bandwidth should be selected by the baseband.

Setting format: [:SOURce[1]|2]:POWer:ALC:BANDwidth|BWIDth AUTO|HIGH|MEDium|LOW

Query format: [:SOURce[1]|2]:POWer:ALC:BANDwidth|BWIDth?

Parameter description:

<AlcBandWidth > discrete data. The values of ALC loop bandwidth are as follows:

AUTO	:Auto;
HIGH	: 10kHz;
MEDium	: 1kHz;
LOW	: 100Hz.

Example: :SOURce2:POWer:ALC:BANDwidth HIGH

Set the ALC bandwidth of channel 2 to high-speed.

Reset state: AUTO

Key path: [Power]—>[ALC Bandwidth]—>[Bandwidth Selection]

[:SOURce[1]|2]:POWer:ALC:LEVel <AlcLevel>

Function description: This command is used to set the ALC level value when the attenuator is set to manual. For Manual/Auto mode selection of the attenuator, refer to command "[:SOURce[1]|2]:POWer:ATTenuation:AUTO".

Setting format: [:SOURce[1]|2]:POWer:ALC:LEVel <value>

Query format: [:SOURce[1]|2]:POWer:ALC:LEVel?

Parameter description:

<AlcLevel> ALC level.

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

Example: :POWer:ALC:LEVel 5dBm Set the ALC power level of channel A to 5dBm.

Reset state: 0dBm

Key path: [Amplitude] -- > [Attenuation] -- >[ALC Power]

[:SOURce[1]|2]:POWer:ALC:SEARch <Mode>

Function description: This command is used to activate or deactivate the automatic power search inside the signal generator when the ALC loop is open. The power search will make the power stabilize the signal generator on the output power selected by the user when the ALC loop is disconnected, and maintain the driving state of the internal modulator. For ALC loop state, refer to command "[:SOURce[1]|2]:POWer:ALC[:STATe]".

Setting format: [:SOURce[1]|2]:POWer:ALC:SEARch ON|OFF|1|0|ONCE

Query format: [:SOURce[1]|2]:POWer:ALC:SEARch?

Parameter description:

<Mode> discrete data. The values of automatic power search state are as follows:

OFF | 0: this command is used to stop the automatic power search. The search mode is manual.

ON | 1: the power is searched automatically with the change of RF output power or frequency. The search mode is automatic.

ONCE :perform a power search at the current RF output frequency.

Executing the search does not change the ALC loop search method. When querying, return the current loop search method.

Example: :POWer:ALC:SEARch 1 the channel A power search is in automatic state.

Reset state: OFF

Key path: [Amplitude] —> [ALC Loop] —> [Search Style: Auto/manual]/[Perform search]

[:SOURce[1]|2]:POWer:ALC:SOURce <Mode>

Function description: This command allows the user to select the ALC power stabilization mode applied by the signal generator according to the appropriate situation, including internal and external modes. When setting the external mode, "External Detection" will be displayed on the rightmost side of the power edit box on the main interface.

Setting format: [:SOURce[1]|2]:POWer:ALC:SOURce INTernal|EXTernal

Query format: [:SOURce[1]|2]:POWer:ALC:SOURce?

Parameter description:

<State > discrete data. Values are as follows in the Power Level Control mode:

INTernal: the power stabilization mode is internal

EXTernal: Power stabilization mode is external

Example: :POWer:ALC:SOURce INT Set the stabilization mode of channel A to Internal.

Reset state: INTernal

Key path: [Amplitude] -> [Level Control] -> [Level Control]

[:SOURce[1]|2]:POWer:ALC:SOURce:EXTernal:COUPling <value>

Function description: This command is used to set the coupling coefficient of external detection. When the power stabilization mode is external, this command is used to set the coupling factor to be used for external stabilization. For power stabilization mode, refer to command "[:SOURce[1]|2]:POWer:ALC:SOURce".

Setting format: [:SOURce[1]|2]:POWer:ALC:SOURce:EXTernal:COUPling <value>

Query format: [:SOURce[1]|2]:POWer:ALC:SOURce:EXTernal:COUPling?

Parameter description:

<value> coupling coefficient of external detection.
Range:[-90dB, +90dB].

Example: :POWer:ALC:SOURce:EXTernal:COUPling 16dB Set the coupling factor for channel A external stable amplitude power is 16dB.

Reset state: 16.00dB

Key path: [Amplitude] -- >[Level Control] -- >[Ext Detector Couple]

[:SOURce[1]|2]:POWer:ALC[:STATe] <State>

Function description: This command is used to open or close the ALC loop. The main function of ALC loop is to correct power drift and keep the output power level of signal generator unchanged with time and temperature. When setting to Open Loop, "Open Loop" will be displayed on the rightmost side of the power edit box on the main interface.

Setting format: [:SOURce[1]|2]:POWer:ALC[:STATe] ON|OFF|1|0

Query format: [:SOURce[1]|2]:POWer:ALC[:STATe]?

Parameter description:

<State> Boolean data, with the values listed below:
ON | 1: ALC loop closed,
OFF | 0: ALC loop opened.

Example: :POWer: ALC 1 Set the ALC loop of channel A to Close

Reset state: 1

Key path: [Power] -- >[Loop Control] -- >[ALC Loop State: ALC-ON/ALC-OFF]

[:SOURce[1]|2]:POWer:ATTenuation <Atten>

Function description: This command is used to set the power attenuation value of the mechanical attenuator of the signal generator. The command setting value can only work when the attenuator is kept in the manual state. Otherwise, the original value shall be kept unchanged. For Auto/Manual attenuation settings, refer to command "[:SOURce[1]|2]:POWer:ATTenuation:AUTO".
The minimum attenuation step set by this command is 10dB, that is, the user can only set the attenuation value as 0dB, 10dB, 20dB, with 10dB as the step value. When the set step value is not 10dB, it will be automatically set to an integer no greater than the set value. After setting the attenuation value, the output power of the signal generator is the current ALC power minus the attenuation value currently set.

Setting format: [:SOURce[1]|2]:POWer:ATTenuation <value>

Query format: [:SOURce[1]|2]:POWer:ATTenuation?

Parameter description:

<Atten> Power attenuation value, which is divided into three categories and related to the attenuators used.

Range: 130dB attenuator: [0dB，130dB]。
120dB attenuator: [0dB, 120dB].
90dB attenuator: [0dB，90dB]。
50dB attenuator: [0dB，50dB]

Example: :POWer: ATTenuation 10dB Set the attenuation value of channel A to 10dB.

Reset state: 130dB|120dB|90dB|50dB

Key path: [Power] -- >[Attenuation Control] -- >[Attenuation]

[:SOURce[1]|2]:POWer:ATTenuation:AUTO <State>

Function description: This command is used to set the control state of the internal programmable step attenuator: automatic or manual mode. In automatic mode, the signal generator will automatically set the value of the power attenuator according to the current output power. In manual mode, the power attenuation of the current attenuator will not change with the power output level.

Setting format: [:SOURce[1]|2]:POWer:ATTenuation:AUTO ON|OFF|1|0

Query format: [:SOURce[1]|2]:POWer:ATTenuation:AUTO?

Parameter description:

<State> Boolean data, with the values listed below:

ON|1: Auto attenuation
OFF | 0: manual attenuation.

Example: :POWer:ATTenuation:AUTO 0 the channel A attenuator is set to manual state.

Reset state: 1

Key path: [Power] -- >[Attenuation Control] -- >[Attenuation Coupling: Auto/Manual]

[:SOURce[1]|2]:POWer:CENTer <Ampl>

Function description: This command sets the center power during the signal generator power sweep. The center power and power sweep span will change the start and stop power of the power sweep, where the start power = center power - power sweep span/2 and the stop power = center power + power sweep span/2.

Setting format: [:SOURce[1]|2]:POWer:CENTer <value>

Query format: [:SOURce[1]|2]:POWer:CENTer?

Parameter description:

<Ampl> power value, as shown below:

Range: [-150dBm, 30dBm], this value range is also limited by the starting and ending power.

Example: :POWer: CENTer 20dBm Set the channel A power sweep center power value to 20dBm.

Reset state: -135dBm

Key path: None

[:SOURce[1]|2]:POWer[:LEVel][:IMMediate][:AMPLitude] <Ampl>

Function description: This command is used to set the power level of the signal generator.

Setting format: [:SOURce[1]|2]:POWer[:LEVel][:IMMediate][:AMPLitude] <value>

Query format: [:SOURce[1]|2]:POWer[:LEVel][:IMMediate][:AMPLitude]?

Parameter description:

<Ampl> power level value.

Range: [-(Attenuator max.+20), +30dBm].

Example: :POWer 0dBm Set the power level of channel A to 0dBm.

SOURce2: POWer 0dBm Set the power level of channel B to 0dBm.

Reset state: -130dB|-120dB|-90dB|-50dB, - (Attenuator max.).

Key path: [PWR] -> [PowSet] -> [PWR]

[:SOURce[1]|2]:POWer[:LEVel][:IMMediate]:OFFSet <PowOffset>

Function description: The command is used to set the actual output power offset value of the signal generator. When the value is not zero, "Offset" will be displayed on left of the power editing box, and the displayed power value is the actual output power plus the power offset. The power offset value will change the displayed power value instead of the actual output power of the signal generator.

Setting format: [:SOURce[1]|2]:POWer[:LEVel][:IMMediate]:OFFSet <value>

Query format: [:SOURce[1]|2]:POWer[:LEVel][:IMMediate]:OFFSet?

Parameter description:

<PowOffset> power offset value.

Range:[-100dB, +100dB].

Example: :POWer:OFFS -10dB the power offset is --10dB.

Reset state: 0dB

Key path: [Power] -> [Power Setting] -> [Power Offset]

[:SOURce[1]|2]:POWer:PEP?

Function description: This command is used to query the PEP value corresponding to the current power. It is only used for querying.

Query format: [:SOURce[1]|2]:POWer:PEP?

Example: :POWer:PEP? Query the PEP value of channel A.

Reset state: None

Key path: None

[:SOURce[1]|2]:POWer:PROTection[:STATe] <State>

Function description: When the loop is Open, you can set the state of the power search output. For power loop ON?OFF state, refer to command "[:SOURce[1]|2]:POWer:ALC[:STATe]".

Setting format: [:SOURce[1]|2]:POWer:PROTection <State>

Query format: [:SOURce[1]|2]:POWer:PROTection?

Parameter description:

<State> Boolean data, with the values listed below:

ON | 1: Min. search output

OFF | 0: Normal search output

Example: :POWer: PROTection ON Search output is the minimum.

Reset state: 1

Key path: [PWR] -> [LopCtrl] -> [Search Output]

[:SOURce[1]|2]:POWer:REFerence <PowRef>

Function description: When the power reference is set to ON state, the power reference value may be set. For power reference ON/OFF state, refer to command "[:SOURce[1]|2]:POWer:REFerence:STATe". When the power reference is set to ON state, the indicator "Reference" will be displayed on left of the power editing box, and the displayed power value = actual output power - power reference value.
For example, when the current continuous wave output power is 1dBm, if the power reference is set to 1dBm, the displayed continuous wave output power will be based on the power reference. Therefore, 0dBm will be displayed in the power display area, and the actual output frequency of the signal generator is still 1dBm.

Setting format: [:SOURce[1]|2]:POWer:REFerence <value>

Query format: [:SOURce[1]|2]:POWer:REFerence?

Parameter description:

<PowRef> power reference value.
Range:[-150dBm, +30dBm].

Example: :POWer: REFerence -10dBm Set the power reference of channel A to -10dBm.

Reset state: 0dBm

Key path: [Power] -- >[Power Setting] -- >[Power Ref]

[:SOURce[1]|2]:POWer:REFerence:STATe <State>

Function description: This command is used to set the power reference to ON/OFF state. When the power reference is set to ON state and the power reference value is not zero, the power level of the signal generator is changed, the power value displayed in the power editing box is based on the power reference. Please refer to the command "[:SOURce[1]|2]:POWer:REFerence" for setting of power reference. When the power reference is set to OFF state, the power value displayed in the power display area is the actual continuous wave output power of the signal generator. [:SOURce]:POWer:REFerence <PowRef> 1 [:SOURce]:POWer:REFerence <PowRef> 1

Setting format: [:SOURce[1]|2]:POWer:REFerence:STATe ON|OFF|1|0

Query format: [:SOURce[1]|2]:POWer:REFerence:STATe?

Parameter description:

<State> Boolean data, with the values listed below:
ON | 1: Power reference ON,
OFF | 0: power reference OFF.

Example: :POWer:REFerence:STATe 1 Set channel A power reference to be ON.

Reset state: 0

Key path: [Power] -- >[Power Setting] -- >[Power Ref ON/OFF]

[:SOURce[1]|2]:POWer:SPAN <PowSpan>

Function description: This command sets the power sweep span during the power sweep of the signal generator. The power sweep span and the center power will change the start and stop power of the power sweep, where the start power = center power - power sweep span/2 and the stop power = center power + power sweep span/2.

Setting format: [:SOURce[1]|2]:POWer:SPAN <PowSpan>

Query format: [:SOURce[1]|2]:POWer:SPAN?

Parameter description:

<PowSpanl> power value, as shown below:
Range: [-(Attenuator max. +50), (Attenuator max.+50)]; The negative value indicates that the start power is less than the stop power, and this value is limited by the start power and the stop power.

Example: :POWer: SPAN 10dB Set the power sweep span to 10dB.

Reset state: 0dB

Key path: None

Remarks: Commands are valid when the power sweep function option is installed in the instrument (S16)

[:SOURce[1]|2]:POWer:STARt <value>

Function description: This command sets the start power at the time of the power sweep of the signal generator

Setting format: [:SOURce[1]|2]:POWer:STARt <value>

Query format: [:SOURce[1]|2]:POWer:STARt?

Parameter description:

<value> power value, as shown below:
Range: [-(Attenuator max.+20), 30dBm]

Example: :POWer: STARt -10dBm Set the start power of power sweep to -10dBm.

Reset state: 0dB

Key path: [Power]—>[Power Sweep]->[Start Power]

Remarks: Commands are valid when the power sweep function option is installed in the instrument (S16)

[:SOURce[1]|2]:POWer:STEP <PowStep>

Function description: It is used to set the power step value.

Setting format: [:SOURce[1]|2]:POWer:STEP <value>

Query format: [:SOURce[1]|2]:POWer:STEP?

Parameter description:

<PowStep> power reference value.
Range:[0.01dB, 20dB].

Example: :POWer:STEP 1dB the power step is 1dB.

Reset state: 0.10dB

Key path: [PWR] -> [PowSet] -> [PWR Step]

[:SOURce[1]|2]:POWer:STEP:STATe <State>

Function description: This command sets the power step ON/OFF state. When it is set to ON, the power settings are as specified in "[:SOURce[1]|2]:POWer[:LEVel][:IMMediate][:AMPLitude]". By setting the parameters UP and DWON, the power can be increased or decreased according to the set power step value.

Setting format: [:SOURce[1]|2]:POWer:STEP:STATe ON|OFF|1|0

Query format: [:SOURce[1]|2]:POWer:STEP:STATe?

Parameter description:

<State> Boolean data, with the values listed below:
ON | 1: Power step ON,
OFF|0: Power step OFF.

Example: :FREQuency:STEP:STATe 1 Set the frequency step of signal generator channel A to ON.

Reset state: 0

Key path: [Power]—>[Power Settings]->[Power Step ON/OFF]

[:SOURce[1]|2]:POWer:STOP <PowSpan >

Function description: This command sets the stop power at the time of the power sweep of the signal generator

Setting format: [:SOURce[1]|2]:POWer:STOP <value><pow_unit>

Query format: [:SOURce[1]|2]:POWer:STOP?

Parameter description:

<PowSpanl> Stop power value of power sweep, as shown below:

Range: [-(Attenuator max.+20), 30dBm]

Example: :POWer:STOP 10dBm Set the stop power of power sweep to 10dBm.

Reset state: 0dBm

Key path: [Power]—>[Power Sweep]->[Termination Power]

Remarks: Commands are valid when the power sweep function option is installed in the instrument (S16)

[:SOURce[1]|2]:POWer:SWEep[:STATe] <State >

Function description: This command is used to set the power sweep of the signal generator to ON/OFF state.

Setting format: [:SOURce[1]|2]:POWer:SWEep ON|OFF|1|0

Query format: [:SOURce[1]|2]:POWer:SWEep?

Parameter description:

<State> Boolean data, with the values listed below:

ON|1: power sweep ON

OFF|0: power sweep OFF

Example: POWer:SWEep ON set power sweep to ON.

Reset state: 0

Key path: [Power] -- >[Power Sweep] -- >[Power Sweep ON/OFF]

Remarks: Commands are valid when the power sweep function option is installed in the instrument (S16)

[:SOURce[1]|2]:POWer:USER:ENABle <State >

Function description: This command is used to set the power limit of the signal generator to ON/OFF state.

Setting format: [:SOURce[1]|2]:POWer:USER:ENABle ON|OFF|1|0

Query format: [:SOURce[1]|2]:POWer:USER:ENABle?

Parameter description:

<State> Boolean data, with the values listed below:

ON|1: Power limit ON

OFF|0: Power limit OFF

Example: POWer:USER:ENABle ON Set the power limit to ON.

Reset state: 0

Key path: [PWR] -> [PowSet] -> [PWR Limit]

[:SOURce[1]|2]:POWer:USER:MAX <Value>

Function description: This command sets the maximum output power of the signal generator. When the power limit is ON, the command is valid. For power limit command, refer to "[:SOURce[1]|2]:POWer:USER:ENABle".

Setting format: [:SOURce[1]|2]:POWer:USER:MAX <Value>

Query format: [:SOURce[1]|2]:POWer:USER:MAX?

Parameter description:

<Value> The maximum output power value, as shown below:

[-(Attenuator max.+20), 30dBm]

Example: POWer:USER:MAX 10dBm Set the maximum output power to 10dBm.

Reset state: 30dBm

Key path: [PWR] -> [PowSet] -> [Max. Ouput PWR]

3.3.4 LIST Subsystem

This subsystem command is used to set the list sweep function of the RF output signal. The subsystem commands and parameters are as follows:

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

[:SOURce[1]|2]:LIST:CATalog?

Function description: This command is used to query the list file name under the default list storage path "/home/ceyear/SgData/user/List".

Query format: STEP[:SOURce[1]|2]:LIST:CATalog?

Description of returned value: Return all file names with the extension ".lst" under the default path, and separate by ";".

Example: :LIST:CATalog?

Key path: None

Reset state: None

Description: For query only.

[:SOURce[1]|2]:LIST:DWELl<Val>{,{Val}}

Function description: This command is used to set the dwell time for each sweep point in the current list. If the user needs to set different dwell time, the corresponding dwell time must be entered for each point in the list. It is just required to enter the dwell time parameter value of the list sweep point in turn, separated by commas. If the number A of parameters input by the user is less than the current number B of list points, only modify the dwell time of the first A points in the list; If the number A of input parameters is greater than the number B of list points, you can insert A-B sweep points in the list, with the frequency and power of the newly inserted sweep points being default values and the dwell time of all sweep points being the entered dwell time. The maximum number of parameters entered is 65,535, and the minimum number is 1. Before using this command, you need to make sure that list sweep file has been selected. For the file selection command, refer to "[:SOURce[1]|2]:LIST[:WAVeform]". If the list sweep file is not selected, select the default file "DefaultList.lst" and modify the data in the file.

Setting format: [:SOURce[1]|2]:LIST:DWELl <val>{,{val}}

Query format: [:SOURce[1]|2]:LIST:DWELl?

Parameter description:

<Val> dwell time of list sweep point.
Range:[1ms, 60s].

Example: :LIST:DWELl 30ms, 20ms Set the dwell time of the first point and the second point in the list to 30ms and 20ms respectively.

Key path: [Sweep] —> [List Sweep] —> [Edit List File…] —> [Dwell Time]

[:SOURce[1]|2]:LIST:DWELl:ALL <Val>

Function description: This command sets the unified dwell time in the current list, and when the type of list sweep global dwell time is global "[:SOURce[1]|2]:LIST:DWELl:TYPE", the dwell time set by this command will be valid.

Setting format: [:SOURce[1]|2]:LIST:DWELl:ALL <val>

Query format: [:SOURce[1]|2]:LIST:DWELl:ALL?

Parameter description:

<Val> dwell time of list sweep point.
Range:[1ms, 60s].

Example: :LIST:DWELl:ALL 30ms Set the list global dwell time to 30ms.

Key path: [Sweep] -- >[List Sweep] -- >[Dwell Time]

[:SOURce[1]|2]:LIST:DWELl:TYPE <Mode>

Function description: This command sets the type of list sweep dwell time, which can be set into two types: Global and List. List means that the dwell time used by the current sweep file is the dwell time for each point set in the current list file. Global means that the dwell time in the current sweep list file is invalid, and the dwell time for each point uses the set external time.

Setting format: [:SOURce[1]|2]:LIST:DWELl:TYPE GLOBal|LIST

Query format: [:SOURce[1]|2]:LIST:DWELl:TYPE?

Parameter description:

<Direc> discrete data. Dwell time type:

GLOBal The dwell time for each point in the list is the globally set dwell time.

LIST The dwell time for each point in the list is the dwell time set in the list.

Example: LIST:DWELl:TYPE GLOBal Set the list dwell time type to Global.

Reset state: LIST

Key path: [Sweep]—> [List Sweep]—>[Global Dwell Time List Global]

[:SOURce[1]|2]:LIST:FREQuency <Val>{,{Val}}

Function description: This command is used to set the frequency value of each sweep point in the current list. If the user needs to set a different frequency value, it must assign a corresponding frequency value to each frequency point in the list. It is just required to enter the frequency value of the list sweep point in turn, separated by commas. If the number of parameters input by the user is less than the current number B of list points, only modify the dwell time of the first A points in the list; If the number A of input parameters is greater than the number B of list points, you can insert A-B sweep points in the list, with the frequency and dwell time of the newly inserted sweep points being default values and the frequency of all sweep points being the entered frequency value. The maximum number of parameters entered is 65,535, and the minimum number is 1. Before using this command, you need to make sure that list sweep file has been selected. For the file selection command, refer to "[:SOURce[1]|2]:LIST[:WAVeform]". If the list sweep file is not selected, select the default file "DefaultList.lst" and modify the data in the file.

Setting format: [:SOURce[1]|2]:LIST:FREQuency <val>{,{val}}

Query format: [:SOURce[1]|2]:LIST:FREQuency?

Parameter description:

<Val> list sweep point frequency.

Model	Range
1466C	[6kHz ~45GHz]
1466D	[6kHz~20GHz]
1466E	[6kHz ~33GHz]
1466G	[6kHz ~ 45GHz].
1466H	[6kHz ~ 53GHz].
1466L	[6kHz ~45GHz]
1466N	[6kHz ~90GHz]
1466P	[6kHz ~ 53GHz].

Example: :LIST:FREQuency 300MHz, 1GHz, 500MHz set the continuous wave frequency in the list as 300MHz, 1GHz and 500MHz successively.

Key path: [Sweep] -- >[List Sweep] -- >[Edit List File...] - > [Frequency]

[:SOURce[1]|2]:LIST:FREQuency:POINts?

Function description: This command queries the number of sweep points in the currently selected list sweep file. If the list sweep file is not selected for the current list sweep, or if the specified list sweep file cannot be opened, return the value 0.

Query format: [:SOURce[1]|2]:LIST:FREQuency:POINts?

Example: :LIST:FREQuency:POINts?

Description: For query only.

[:SOURce[1]|2]:LIST:INDex:STARt <Val>

Function description: This command sets the sweep start index of the list, namely, the start play point of the current list sweep. The point before the sweep start index in the list will not be played. If the set sweep start index point is greater than the total number of points in the list, the sweep start index will be set to the total number of points - 1. If the set sweep start index point is greater than the sweep stop index, the sweep stop index value will be modified to be equal to the currently set sweep start index.

Setting format: [:SOURce[1]|2]:LIST:INDex:STARt <val>

Query format: [:SOURce[1]|2]:LIST:INDex:STARt?

Parameter description:

<Val> List play start index.
Range: [0, number of list points - 1].

Example: :LIST:INDex:STARt 2 Set the list start play index to 2

Key path: [Sweep]—> [List Sweep]—> [Sweep Start Index]

[:SOURce[1]|2]:LIST:INDex:STOP <Val>

Function description: This command sets the sweep stop index of the list, namely, the play stop point of the current list sweep. The point after the sweep stop index in the list will not be played. If the set sweep start index point is greater than the total number of points in the list, the sweep start index will be set to the total number of points - 1. If the set sweep stop index point is less than the sweep start index, the sweep start index value will be modified to be equal to the currently set sweep stop index.

Setting format: [:SOURce[1]|2]:LIST:INDex:STOP <val>

Query format: [:SOURce[1]|2]:LIST:INDex:STOP?

Parameter description:

<Val> List play start index.
Range: [0, maximum number of points - 1].

Example: :LIST:INDex:STARt 4 Set the list play stop index to 4

Key path: [Sweep]—>[List Sweep]—> [Sweep Stop Index]

[:SOURce[1]|2]:LIST:PLAY[:INDex]?

Function description: This command is used to query the index value of the current list sweep play. When the sweep point dwell time is short, the value queried by the programmed control shall be delayed to the current play index value.

Query format: [:SOURce[1]|2]:LIST:PLAY[:INDex]?

Parameter description:

Example: :LIST:PLAY:INDex?

Key path: [Sweep]—> [List Sweep]—>[Current Play Index]

[:SOURce[1]|2]:LIST:POWer <Val>{,{Val}}

Function description: This command is used to set the power value of each sweep point in the current list. If the user needs to set a different power value, it must assign a corresponding power value to each sweep point in the list. It is just required to enter the power value of the list sweep point in turn, separated by commas. If the number of parameters entered by the user is less than the current number B of points in the list, modify the power value of the first A points in the list; If the number of parameters A entered is greater than the number B of list points, insert A-B sweep points in the list, with the frequency and dwell time of the newly inserted sweep points being default values and the power of all sweep points being the entered power value. The maximum number of parameters entered is 65,535, and the minimum number is 1. Before using this command, you need to make sure that list sweep file has been selected. For the file selection command, refer to "[:SOURce[1]|2]:LIST[:WAVeform]". If the list sweep file is not selected, select the default file "DefaultList.lst" and modify the data in the file. The "DefaultList.lst" file data stores the original data. When this file is created for the first time, its data is empty.

Setting format: [:SOURce[1]|2]:LIST:POWer <val>{,{val}}

Query format: [:SOURce[1]|2]:LIST:POWer?

Parameter description:

<Val> Power value of list sweep point, floating point type, and number of parameters is variable.

Range [- (attenuator max.+30), +35dBm].

Example: :LIST:POWer 1dBm, 0.2dBm, 1.3dBm, 2.5dBm, 3.6dBm

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

Key path: [Sweep] -- >[List Sweep] -- >[Edit List File...] - > [Power]

[:SOURce[1]|2]:LIST:POWer:POINts?

Function description: This command queries the number of sweep points in the currently selected list sweep file. If the list sweep file is not selected for the current list sweep, or if the specified list sweep file cannot be opened, return the value 0.

Query format: [:SOURce[1]|2]:LIST:POWer:POINts?

Example: :LIST:POWer:POINts?

Description: For query only.

[:SOURce[1]|2]:LIST:RETRace <State>

Function description: Set the sweep to ON/OFF state. After the completion of a single list sweep, check whether the output frequency of the signal generator is kept at the first point or the last point in the list. This command can only be used in the single sweep mode. Same as "[:SOURce[1]|2]:SWEep:RETRace "

Setting format: [:SOURce[1]|2]:LIST:RETRace ON|OFF|1|0

Query format: [:SOURce[1]|2]:LIST:RETRace?

Parameter description:

<State> Boolean data, with the values listed below:

ON | 1: When the sweep is set to ON state, the output frequency is kept at the first frequency point in the list after sweep.

OFF | 0: When the sweep is set to OFF state, the output frequency is kept at the last frequency point in the list after sweep.

Example: :LIST:RETRace 0 when the sweep is set to OFF state, after the completion of a single list sweep, the continuous wave frequency output by the signal generator will reside at the last frequency point in the list.

Reset state: 0

Key path: [Sweep] -> [Sweep Mode]->[Sweep ON/OFF]

[:SOURce[1]|2]:LIST:TRIGger

Function description: When the list sweep trigger is set to trigger key, this command is used to trigger once, and each time the command is executed, the list sweep advances one sweep point. For sweep trigger command, refer to "[SOURce[1]|2]:LIST:TRIGger:SOURce".

Setting format: [:SOURce[1]|2]:LIST:TRIGger

Example: :LIST:TRIGger Trigger list sweep once

Reset state: None

Key path: [Sweep] -- >[List Sweep] -- >[Trig]

[:SOURce[1]|2]:LIST:TRIGger:SOURce <Source>

Function description: This command sets the trigger source type of list playing. The trigger source has three modes: Auto, Ext and Trigger Key.

Setting format: [:SOURce[1]|2]:LIST:TRIGger:SOURce IMMediate| EXTernal|KEY

Query format: [:SOURce[1]|2]:LIST:TRIGger:SOURce?

Parameter description:

<Source> discrete data. The values of list sweep trigger source are as follows:

IMMediate automatic; the trigger signal is always true, and the system will automatically trigger the next sweep point after completing a sweep point.

EXTernal external; the trigger signal is from the trigger input connector on the rear panel.

KEY Trigger key, the trigger signal comes from the trigger key on the front panel; A trigger is performed by a group whose trigger source is from GPIB at the time of programmed control, or when receiving the "*TRG" command.

Example: :LIST:TRIGger:SOURce KEY set the list sweep trigger source to trigger key.

Reset state: IMM

Key path: [Sweep] -- >[List Sweep] -- >[List Trig]

[:SOURce[1]|2]:LIST[:WAVeform] <name>

Function description: This command is used to select the list sweep file to be played when the signal generator is in list sweep state. The file extension is ".lst". When setting the list file, the specified file name cannot contain the path name and it can only use the file name selected for loading (the file name must contain the extension .lst). The specified file can be found in the folder in default path "SgData/usr/List". If the specified file does not exist, an error "file name not found" will appear. If the list sweep file is not currently selected during query, return "NULL". If a file is selected, return the name of the selected file.

Setting format: [:SOURce[1]|2]:LIST[:WAVeform] <name>

Query format: [:SOURce[1]|2]:LIST[:WAVeform]?

Parameter description:

<name> String type: The name of the list sweep file to be called must contain the file extension.

Example: :LIST:WAVeform "123.lst" Call the list sweep file named 123.lst.

Key path: [Sweep]-> [List Sweep]-> [Select List Sweep File]

[:SOURce]:LIST[:WAVeform]:DELete <fileName>

Function description: Delete the specified list file with the file extension ".lst". The specified file name shall not contain a path name, but shall be the name of the loaded file to be selected (the file name must contain the extension .lst), which can be searched in the default path "/home/ceyear/SgData/usr/List" folder. If the specified file does not exist, an error "file name not found" will appear.

Setting format: [:SOURce]:LIST[:WAVeform]:DELete <fileName>

Parameter description:

<fileName> string type, select the deleted list sweep file name. File extension must be contained.

Example: : LIST: WAVeform: DELete "test. Lst" Delete the "test. Lst" file under the path "/home/ceyear/SgData/usr/List"

Key path: None

[:SOURce]:LIST[:WAVeform]:DELete:ALL

Function description: Delete all list files under the path "/home/ceyear/SgData /usr/list ".

Setting format: [:SOURce]:LIST[:WAVeform]:DELete:ALL

Parameter description: None

Example: : LIST: WAVeform: DELete: ALL Delete all ". Lst" files under the path "/home/ceyear/SgData /usr/List".

Key path: None

[:SOURce]:LIST[:WAVeform]:EXPort <fileName>

Function description: Export the data of the currently selected list file to a file with the extension of csv. The parameter of this command is the exported file name, which only contains the file name and must contain the extension of ".csv"; The default storage path of the exported file is "/home/ceyear/SgData/user/list". If no file is selected in the current list sweep, a "file Name not found" "error will occur.

Setting format: [:SOURce]:LIST[:WAVeform]:EXPort “fileName”

Example: :LIST:WAVeform:EXPort “test.csv”

Key path: None

[:SOURce]:LIST[:WAVeform]:IMPort <fileName>

Function description: Load the specified file with the CSV extension into the currently selected list sweep file, and replace the data in the selected list sweep file with the data in the specified file. If no list sweep file is currently selected, generate a list sweep file with the same name as the imported file, copy the imported data to the generated list sweep file with the same name, and select the file. The parameter contains only the file name and must contain the extension ".csv"

Setting format: [:SOURce]:LIST[:WAVeform]:IMPort “fileName”

Example: :LIST:WAVeform:IMPort “text.csv” Load the “text.csv” file under/home/ceyear/SgData/user/List.

Key path: None

[:SOURce[1]|2]:LIST[:WAVeform]:POINts?

Function description: If the signal generator is in the list sweep state and the list sweep file is selected, query the number of sweep points in the list sweep file.

Query format: [:SOURce[1]|2]:LIST[:WAVeform]:POINts?

Example: :LIST:WAVeform:POINts? Query the number of sweep points in the listed sweep file.

Key path: None

[:SOURce[1]|2]:LIST[:WAVeform]:RESet

Function description: Delete the currently selected list sweep file. There is no list file selected currently after this command is executed. for setting only.

Setting format: [:SOURce[1]|2]:LIST[:WAVeform]:RESet

Parameter description: None

Example: :LIST:WAVeform:RESet

Key path: [Sweep]-> [List sweep]-> [Reset List File]

[:SOURce[1]|2]:LIST[:WAVeform]:STORe <FileName>

Function description: When the signal generator is in the list sweep state, this command will be used to save the list sweep file being played currently as a file with another name and extension of ".lst". When saving a list file, the specified file name shall not contain a pathname, but shall be the name of the file to be saved (the filename must contain the extension of .lst), and the file will be generated in the default "SgData/usr/list" folder. If the specified file already exists, delete the original file and generate a new file.

Setting format: [:SOURce[1]|2]:LIST[:WAVeform]:STORe “FileName”

Parameter description:

< FileName> string type: the name of the sweep file to be saved must contain the file extension.

Example: :LIST:WAVeform:STORe “123.lst” Save the current list data to the 123.ls file.

Key path: [Sweep]-> [List Sweep]-> [Edit List File]-> [Save As]

3.3.5 LFOutput Subsystem

The LFOutput subsystem is used to configure the low frequency output function. The command of this subsystem is valid only when the low frequency output/function generator (S14) option is installed in the instrument. Otherwise, the error message of "undefined command" error will appear.

3.3 Instrument Subsystem Command

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

[:SOURce[1]|2]:LFOutput:AMPLitude <Ampl>

Function description: This command is used to set the amplitude of the low frequency output signal.

Setting format: [:SOURce[1]|2]:LFOutput:AMPLitude <val>

Query format: [:SOURce[1]|2]:LFOutput:AMPLitude?

Parameter description:

<Ampl> LF output signal amplitude.

Range:[1mv, 8v]. The maximum (Z) of this value is also related to the absolute value of the DC offset (P), Z/2+|P|<=5.5v

Example: : LFOutput: AMPLitude 1 V set the LF output signal amplitude to 1V.

Reset state: 1mv

Key path: [LF Out] -- > [LF Out] -- > [LF Ampl]

[:SOURce[1]|2]:LFOutput[:FUNCtion]:FREQuency

Function description: This command is used to set the output frequency of the function generator. It should be noted that when the low frequency waveform is selected as sweep sine or dual sine, this command is used to set the start frequency of the sweep sine or the dual sine frequency 1. Please refer to "[:SOURce[1]|2]:LFOutput[:FUNCtion]:SHAPe" for low frequency waveform selection command. When the function generator frequency of channel A is set, the frequency of function generator 1 is actually set; and when the function generator frequency of channel B is set, the frequency of function generator 2 is actually set. This command is equivalent to "[:SOURce[1]|2]:FUNCtion:FREQuency".

Setting format: [:SOURce[1]|2]:LFOutput[:FUNCtion]:FREQuency <val>

Query format: [:SOURce[1]|2]:LFOutput[:FUNCtion]:FREQuency?

Parameter description:

<Frequency> frequency of function generator.

Range:

Sine: [0.010Hz, 10MHz]

Square wave, triangle wave and ramp wave: [0.010Hz, 1MHz].

Sweep sine start frequency: [0.001Hz, 9.99MHz 999999]. When 10 MHz is input, the software will automatically set the sweep sine start frequency to 9.99999999MHz without error message.

Dual sine frequency 1: [0.001Hz, 10 MHz].

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

Reset state: 400Hz

Key path:

- ➢ [Function Generator 1|2] — > [Data Source] — > [Frequency]
- ➢ [Function Generator 1|2] — > [Data Source] — > [Waveform Selection Sweep Sine]-> [Sweep Sine Stop Frequency]
- ➢ [Function Generator 1|2] — > [Data Source] — > [Waveform Selection Dual Sine]-> [Dual Sine Frequency 2]

[:SOURce[1]|2]:LFOutput[:FUNCtion]:FREQuency:ALTernate

Function description: This command is used to set the sweep sine stop frequency or the dual sine frequency 2 when the low frequency waveform is selected as sweep sine or dual sine. Please refer to "[:SOURce[1]|2]:LFOutput[:FUNCtion]:SHAPe" for low frequency waveform selection command. If the low frequency waveform is not a sweep sine or dual sine, the value of the dual sine frequency 2 is modified by default. When the function generator for channel A is set, function generator 1 is actually set. When the function generator for channel B is set, function generator 2 is actually set. This command is equivalent to ' [:SOURce[1]|2]:FUNCtion:FREQuency:ALTernate'.

Setting format: [:SOURce[1]|2]:LFOutput[:FUNCtion]:FREQuency:ALTernate <val>

Query format: [:SOURce[1]|2]:LFOutput[:FUNCtion]:FREQuency:ALTernate?

Parameter description:

<Frequency> Dual sine frequency 2.

Range: [-325GHz～ +325GHz].

The stop frequency of the sweep sine or

Range: [-325GHz～ +325GHz]. If the sweep sine stop frequency is set to 0.001Hz, the software will automatically set the sweep sine stop frequency to 0.002Hz without error message.

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

Reset state: 400Hz

Key path:

- [Function Generator 1|2] — > [Data Source] — > [Waveform Selection Sweep Sine]-> [Sweep Sine Stop Frequency]
- [Function Generator 1|2] — > [Data Source] — > [Waveform Selection Dual Sine]-> [Dual Sine Frequency 2]

[:SOURce[1]|2]:LFOutput[:FUNCtion]:FREQuency:ALTernate:AMPLitude:PERCent

Function description: This command is used to set the amplitude of the second tone as a percentage of the amplitude of the total output signal in a dual sine waveform when the function generator output waveform is dual sine; For example, if the second tone accounts for 20% of the total waveform power, it means the first tone accounts for 80% of the total power output. When the function generator for channel A is set, function generator 1 is actually set. When the function generator for channel B is set, function generator 2 is actually set. This command is equivalent to '[:SOURce[1]|2]:FUNCtion:FREQuency:AMPLitude:PERCent'.

Setting format:

[:SOURce[1]|2]:LFOutput:FREQuency:ALTernate:AMPLitude:PERCent <val>

Query format:

[:SOURce[1]|2]:LFOutput:FREQuency:ALTernate:AMPLitude:PERCent?

Parameter description:

<Percent> Amplitude percent of the dual sine frequency 2.
Range: [0, 100].

Example: :LFOutput:FREQuency:ALTernate:AMPLitude:PERCent 20
Set the second waveform of the dual sine to account for 20% of the total signal power output.

Reset state: 50%

Key path: [Function Generator 1|2] — > [Data Source] — > [Waveform Selection Dual Sine]-> [Dual Sine Frequency 2 Amplitude Percent]

[:SOURce[1]|2]:LFOutput[:FUNCtion]:SHAPe

Function description: This command is used to set the output waveform of the function generator. When the function generator of channel A is set, function generator 1 is actually set. When the function generator of channel B is set, function generator 2 is actually set. This command is equivalent to '[:SOURce[1]|2]:FUNCtion:SHAPe'. When the waveform is set to ramp or noise, the waveform will be automatically set to a certain ramp or noise, depending on the ramp type or noise type set last time. The default is an up ramp or white noise.

Setting format: [:SOURce[1]|2]:LFOutput[:FUNCtion]:SHAPe SINE|SQUare|TRIangle|
RAMP|NOISe|SWEPtsine|DUALsine

Query format: [:SOURce[1]|2]:LFOutput[:FUNCtion]:SHAPe?

Parameter description:

<Mode> discrete data. The values of LF signal output waveform are as follows:

SINE	sine wave,
SQUare	Square wave,
TRIangle	triangle wave,
RAMP	ramp wave,
NOISE	Noise,
SWEPtsine	sweep sine
DUALsine	Dual sine

Example: :LFOutput:SHAPe TRIangl set the waveform of function generator to triangle.

Reset state: SINE

Key path: [Function Generator 1|2] — > [Data Source] — > [Waveform Selection]

[:SOURce[1]|2]:LFOutput[:FUNCtion]:SHAPe:NOISe

Function description: This command is used to set the noise type when the function generator waveform is selected as Noise. Please refer to "[:SOURce[1]|2]:LFOutput[:FUNCtion]:SHAPe" for function generator waveform selection command.When the function generator for channel A is set, function generator 1 is actually set. When the function generator for channel B is set, function generator 2 is actually set. This command is equivalent to "[: SOURce [1] | 2]: FUNCtion: SHAPe: NOISe "

Setting format: [:SOURce[1]|2]:LFOutput[:FUNCtion]:SHAPe:NOISe GAUSsian |UNIForm

Query format: [:SOURce[1]|2]:LFOutput[:FUNCtion]:SHAPe:NOISe?

Parameter description:

<Mode> discrete data. The values of noise type of noise generator 1|2 are as follows:

GAUSsian	Gaussian noise
UNIForm	white noise.

Example: :LFOutput:SHAPe:NOISe GAUSsian set the noise type of function generator 1 to Gaussian.

Reset state: GAUS

Key path: [Function Generator 1|2] — > [Data Source] — > [Waveform Selection]

[:SOURce[1]|2]:LFOutput[:FUNCtion]:SHAPe:RAMP

Function description: This command is used to set the signal output type when the function generator waveform is of ramp type, including up and down. Please refer to [:SOURce[1]|2]:LFOutput[:FUNCtion]:SHAPe for function generator waveform selection command. When the function generator for channel A is set, function generator 1 is actually set. When the function generator for channel B is set, function generator 2 is actually set. This command is equivalent to " [: SOURce [1] | 2]: FUNCtion: SHAPe: RAMP "

Setting format: [:SOURce[1]|2]:LFOutput[:FUNCtion]:SHAPe:RAMP POSitive|NEGative

Query format: [:SOURce[1]|2]:LFOutput[:FUNCtion]:SHAPe:RAMP?

Parameter description:

<Mode> discrete data. The values of ramp signal type are as follows:

POSitive	up
NEGative	down.

Example: : LFOutput: SHAPe: RAMP NEGative Function generator ramp waveform is of down type.

Reset state: POS

Key path: [Function Generator 1|2] — > [Data Source] — > [Waveform Selection]

[:SOURce[1]|2]:LFOutput[:FUNCtion]:SWEep:TIME

Function description: This command is used to set the sweep time for a sweep sine when the function generator output waveform is a sweep sine. For the function generator waveform selection command, please refer to the " [: SOURce [1] | 2]: LFOutput [: FUNCtion]: SHAPe" . When the function generator of channel A is set, the function generator 1 is actually set; when the function generator of channel B is set, the function generator 2 is actually set.This command is equivalent to " [: SOURce [1] | 2]: FUNCtion: SWEep: TIME "

Setting format: [:SOURce[1]|2]:LFOutput[:FUNCtion]:SWEep:TIME <val>

Query format: [:SOURce[1]|2]:LFOutput[:FUNCtion]:SWEep:TIME?

Parameter description:

<Time> Sweep time of sweep sine.
Range:[10us, 2s].

Example: : LFOutput: SWEep: TIME 200ms The sweep time of sweep sine is 200ms.

Reset state: 10ms

Key path: [Function Generator 1|2] — > [Data Source] — > [Waveform Selection Sweep Sine] — > [Sweep Sine Sweep Time]

[:SOURce[1]|2]:LFOutput:OFFSet <val>

Function description: This command is used to set the DC offset of the low frequency output signal.

Setting format: [:SOURce[1]|2]:LFOutput:OFFSet <val>

Query format: [:SOURce[1]|2]:LFOutput:OFFSet?

Parameter description:

<val> LF DC offset, with the values listed below:

Range: [-5.4995v, 5.4995v], the maximum of this value (P) is related to the amplitude (Z), Z/2+|P|<=5.5v

Example: :LFOutput:OFFSet 3mv set the LF DC offset to 3mv.

Reset state: 0

Key path: [LF Out 1|2] — > [LF Out] — > [DC Offset]

[:SOURce[1]|2]:LFOutput:SOURce <Mode>

Function description: This command is used to set the signal source of the low frequency output signal.

Setting format: [:SOURce[1]|2]:LFOutput:SOURce FUNCtion|EXTernal[1]|2|AM|FMPM

Query format: [:SOURce[1]|2]:LFOutput:SOURce?

Parameter description:

<Mode> discrete variable, with the values listed below:

FUNction:	Function generator
EXTernal [1]:	External 1
EXTernal2:	External 2
AM:	AM modulation source
FMPM:	FM/PM modulation sources

Example: : LFOutput: SOURce EXT Set the low frequency source to external 1.

Reset state: FUNCtion

Key path: [LF Output 1|2] — > [LF Out] — > [Modulation Source Selection]

[:SOURce[1]|2]:LFOutput:STATe <State>

Function description: This command is used to set ON/OFF state of the LF output signal.

Setting format: [:SOURce[1]|2]:LFOutput:STATe ON|OFF|1|0

Query format: [:SOURce[1]|2]:LFOutput:STATe?

Parameter description:

<State> Boolean data, with the values listed below:

ON | 1: When the LF output is ON, the LF signal output is turned on.

OFF | 0: when the LF output is OFF, the LF signal output is turned off.

Example: :LFOutput:STATe OFF the LF signal output is turned off.

Reset state: 0

Key path: [LF Output 1|2] — > [LF Out] — > [LF Out ON OFF]

3.3.6 SWEep Subsystem

This subsystem command is used to control the sweep function of RF output signal. The subsystem commands and parameters are as follows:

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

[:SOURce[1]|2]:SWEep[:FREQuency]:DWELl <DwellTime>

Function description: This command is used to set the dwell time of the step sweep. The dwell time refers to the time suspended in the process of sweep at the current step frequency point. The dwell time set by the user works under the mode that the trigger source of the step sweep is selected as automatic. Please refer to "[:SOURce[1]|2]:SWEep[:FREQuency]:TRIGger:SOURce" for setting of trigger source.

Setting format: [:SOURce[1]|2]:SWEep[:FREQuency]:DWELl <value>

Query format: [:SOURce[1]|2]:SWEep[:FREQuency]:DWELl?

Parameter description:

<Val> float point type, step sweep dwell time.
Range:[1ms, 60s].

Example: :SWEep:DWELl 1s set the dwell time of all channel A step sweep points to 1s.

Reset state: 10.000ms

Key path: [Swp]—>[Step Sweep]—>[Dwell Time]

[:SOURce[1]|2]:SWEep[:FREQuency]:MODE <Mode>

Function description: This command is used to set or query the sweep mode, including continuous and single. When the frequency generation mode is list sweep, step sweep and ramp sweep, the set sweep mode is valid. When the frequency generation mode is continuous wave, the sweep mode can be set but is invalid .

Setting format: [:SOURce[1]|2]:SWEep:MODE CONTinuous|SINGle

Query format: [:SOURce[1]|2]:SWEep:MODE?

Parameter description:

<Mode> discrete data; with values taken as follows:

CONTinuous: Continuous; when the sweep is set to ON state, the sweep signal is output continuously.

SINGle: Single; when the sweep is set to ON stat, sweep and output all the sweep frequency points, and the sweep output is finished.

Example: :SWEep:MODE CONTinuous set the sweep mode of channel A to be continuous.

Reset state: CONTinuous

Key path: [Sweep] -- > [Sweep Mode] -- > [Sweep Mode]

[:SOURce[1]|2]:SWEep [:FREQuency]:SHAPe <Mode>

Function description: This command is used to set the shape of step sweep, including triangle and ramp.

Setting format: [:SOURce[1]|2]:SWEep[:FREQuency]:SHAPe TRIangle|RAMP

Query format: [:SOURce[1]|2]:SWEep[:FREQuency]:SHAPe?

Parameter description:

<Mode> discrete data. The values of step sweep direction are as follows:

TRIangle	triangle,
RAMP	

Example: :SWEep:SHAPe RAMP Set the shape of step sweep to ramp.

Reset state: RAMP

Key path: [Sweep]- > [Step Sweep] - > [Sweep Shape]

[:SOURce[1]|2]:SWEep[:FREQuency]:SINGle:RESet

Function description: When the sweep mode is set to single, the current sweep is stopped after the command is executed in the sweep process, and the next point is set as the start frequency point to restart the sweep.

Setting format: [:SOURce[1]|2]:SWEep[:FREQuency]:SINGle:RESet

Parameter description: No parameters

Example: :SWEep:SINGle:RESet

Reset state: None

Key path: [Sweep]- > [Sweep Mode] - > [Reset Single Sweep]

[:SOURce[1]|2]:SWEep[:FREQuency]:SPACing <Mode>

Function description: This command is used to set the step mode of the step sweep.

Setting format: [:SOURce[1]|2]:SWEep[:FREQuency]:SPACing LINear|LOGarithmic

Query format: [:SOURce[1]|2]:SWEep[:FREQuency]:SPACing?

Parameter description:

<Mode> discrete data. The values of step sweep are as follows::

LINear：	Linear,
LOGarithm:	Logarithm.

Example: : SWEep: SPACing LOGarithmic Set the channel A step sweep mode to logarithmic.

Reset state: LINear

Key path: [Sweep] -> [Step Sweep] -> [Step mode]

[:SOURce[1]|2]:SWEep[:FREQuency]:STARt:TRIGger

Function description: When the start sweep trigger is set to the trigger key, this command is used to perform one trigger, set only.

Setting format: [:SOURce[1]|2]:SWEep[:FREQuency]:STARt:TRIGger

Parameter description: No parameters

Example: :SWEep:STARt:TRIGger

Reset state: None

Key path: [Sweep]- > [Sweep Mode] - > [Start Sweep Trigger Key] - > [Trigger]

[:SOURce[1]|2]:SWEep[:FREQuency]:STARt:TRIGger:SOURce <Mode >

Function description: Set the initial sweep trigger type. This command is used to set the start sweep trigger type for three sweep modes: list, step, and ramp.

Setting format: [:SOURce[1]|2]:SWEep[:FREQuency]:STARt:TRIGger:SOURce IMMediate|EXTernal|KEY

Query format: [:SOURce[1]|2]:SWEep[:FREQuency]:STARt:TRIGge:SOURce?

Parameter description:

<Mode> discrete data; with values taken as follows:

IMMediate: Auto; the trigger signal is always true. When the sweep is set to ON state, the system will sweep automatically.

####

EXTernal: External; the trigger signal is from the trigger input connector on the rear panel.

KEY: Trigger key; the trigger signal is from the trigger key on the front panel or programmed control command

Example: : SWEep: STARt: TRIGger: SOURce EXTernal Set the channel A start sweep trigger mode to external.

Reset state: IMMediate

Key path: [Sweep] -> [Mode] -> [Trig Style of Start]

[:SOURce[1]|2]:SWEep[:FREQuency]:STEP[:LINear] <FreqStep>

Function description: This command is used to set the frequency step value when the step sweep mode is linear. After the user determines the start frequency and stop frequency of step sweep and setting the step value, the step sweep will start according to the step value. Please refer to "[:SOURce[1]|2]:FREQuency:STARt", "[:SOURce[1]|2]:FREQuency:STOP " for the start frequency and stop frequency commands of step sweep.

Setting format: [:SOURce[1]|2]:SWEep[:FREQuency]:STEP[:LINear] <value>

Query format: [:SOURce[1]|2]:SWEep[:FREQuency]:STEP[:LINear]?

Parameter description:

<FreqStep> Step sweep frequency step value range [0.001Hz, Max Frequency Value]

Example: : SWEep: STEP 1MHz Set the step value of the step sweep to 1MHz.

Reset state: 10MHz

Key path: [Sweep] -> [Step Sweep] -> [Step mode]

[:SOURce[1]|2]:SWEep[:FREQuency]:STEP:LOGarithmic <val>

Function description: This command is used to set the frequency step value when the step sweep mode is logarithmic. After the user determines the start frequency and stop frequency of step sweep and setting the step value, the step sweep will start according to the step value. Please refer to “[:SOURce[1]|2]:FREQuency:STARt”, “[:SOURce[1]|2]:FREQuency:STOP ” for the start frequency and stop frequency commands of step sweep.

Setting format: [:SOURce[1]|2]:SWEep[:FREQuency]:STEP:LOGarithmic <value>

Query format: [:SOURce[1]|2]:SWEep[:FREQuency]:STEP:LOGarithmic?

Parameter description:

<FreqStep> Step sweep frequency step value range
[0.01, 100]

Example: : SWEep: STEP: LOGarithmic 50 Set the step sweep to start at the step value of 50%.

Reset state: 1

Key path: [Sweep] -> [Step Sweep] -> [Step mode]

[:SOURce[1]|2]:SWEep[:FREQuency]:STEP:TYPE <Mode>

Function description: Set the step mode of step sweep.

Setting format: [:SOURce[1]|2]:SWEep[:FREQuency]:STEP:TYPE
LINear|LOGarithm

Query format: [:SOURce[1]|2]:SWEep[:FREQuency]:STEP:TYPE?

Parameter description:

<Mode> discrete data; with values taken as follows:

LINear：	Linear.
LOGarithm:	Logarithm.

Example: : SWEep: STEP: TYPE LINear Set the channel A step sweep step mode to linear.

Reset state: LINear

Key path: [Sweep] -> [Step Sweep] -> [Step Mode]

[:SOURce[1]|2]:SWEep[:FREQuency]:TRIGger

Function description: When the step trigger in the step sweep is a trigger key, this command will execute one trigger. Each time the trigger is executed, the sweep frequency changes once (only setting).

Setting format: [:SOURce[1]|2]:SWEep[:FREQuency]:TRIGger

Parameter description: No parameters

Example: : SWEep: TRIGger Trigger once.

Reset state: None

Key path: [Sweep] -- > [Step Sweep] -- > [Step Trig Trigger Key]-- >[Trigger]

[:SOURce[1]|2]:SWEep[:FREQuency]:TRIGger:SOURce <Mode>

Function description: This command is used to set the trigger source to start step sweep. The trigger source has three modes: Auto, Ext and Key. This execution is used to set the trigger mode of each frequency point.

Setting format: [:SOURce[1]|2]:SWEep[:FREQuency]:TRIGger:SOURce IMMediate|EXTernal|KEY

Query format: [:SOURce[1]|2]:SWEep[:FREQuency]:TRIGger:SOURce?

Parameter description:

<Mode> discrete data. The values of step sweep trigger source are as follows:

IMMediate automatic, trigger signal is always true, when a sweep is completed,

the system will automatically trigger the next sweep after completing a sweep.

EXTernal external; the trigger signal is from the trigger input connector on the rear panel.

KEY |trigger key; the trigger signal is from the trigger key on the front panel, or programmed control command.

Example: : SWEep: TRIGger: SOURce KEY Set the channel A step sweep trigger mode to trigger key.

Reset state: IMMediate

Key path: [Sweep] -- > [Step Sweep] -- > [Step Trig]

[:SOURce[1]|2]:SWEep:POWer:DWELl <DwellTime>

Function description: This command is used to set the dwell time of the power sweep. The dwell time refers to the time suspended in the process of sweep at the current step frequency point. The dwell time set by the user works under the mode that the trigger source of the step sweep is selected as automatic. Please refer to "[:SOURce[1]|2]:SWEep[:FREQuency]:TRIGger:SOURce" for setting of trigger source.

Setting format: [:SOURce[1]|2]:SWEep:POWer:DWELl <value>

Query format: [:SOURce[1]|2]:SWEep:POWer:DWELl?

Parameter description:

<Val> step sweep dwell time.

Range:[1ms, 60s].

Example: :SWEep:POWer:DWELl 1s Set the dwell time for all channel A power sweep points to 1 s.

Reset state: 10.000ms

Key path: [Power]- > [Power sweep] - > [Dwell Time]

Remarks: Commands are valid when the power sweep function option is installed in the instrument (S16)

[:SOURce[1]|2]:SWEep:POWer:MODE <Mode>

Function description: Set the sweep mode to power sweep.

Setting format: [:SOURce[1]|2]:SWEep:POWer:MODE CONTinuous|SINGle

Query format: [:SOURce[1]|2]:SWEep:POWer:MODE?

Parameter description:

<Mode> discrete data; with values taken as follows:

CONTinuous: Continuous; when the sweep is set to ON state, the power sweep is output continuously.

SINGle: Single; When the sweep is set to ON state, all the sweep power points will be swept and outputted before the sweep output is finished.

Example: : SWEep: POWer: MODE CONTinuous Set the channel A power sweep mode to continuous.

Reset state: CONTinuous

Key path: [Power]-> [Power Sweep]-> [Sweep Mode]

Remarks: Commands are valid when the power sweep function option is installed in the instrument (S16)

[:SOURce[1]|2]:SWEep:POWer:PLAY[:POWer]?

Function description: Query the current sweeping power value during power sweep.

Query format: [:SOURce[1]|2]:SWEep:POWer:PLAY[:POWer]?

Example: :SWEep:POWer:PLAY? Query the current power sweep value of channel A.

Key path: [Power]—>[Power Sweep]->[Current Power]

Description: For query only.

[:SOURce[1]|2]:SWEep:POWer:SINGle:RESet

Function description: When the power sweep mode is single, stop the current power sweep after executing this command during the sweep process, set the initial power point as the next power point, and restart the sweep.

Setting format: [:SOURce[1]|2]:SWEep:POWer:SINGle:RESet

Parameter description: No parameters

Example: :SWEep:POEWer:SINGle:RESet

Reset state: None

Key path: [Power]—>[Power Sweep]—>[Restart Single Sweep]

Remarks: Commands are valid when the power sweep function option is installed in the instrument (S16)

[:SOURce[1]|2]:SWEep:POWer:SPACing <Mode>

Function description: This command is used to set the step mode of power sweep,

Setting format: [:SOURce[1]|2]:SWEep:POWer:SPACing LINear|LOGarithmic

Query format: [:SOURce[1]|2]:SWEep:POWer:SPACing?

Parameter description:

<Mode> discrete data. The step mode of the power sweep is taken the following values:

LINear： Linear,

LOGarithm: Logarithm.

Example: :SWEep:POWer:SPACing LOGarithmic

Set the channel A power sweep step mode to logarithmic.

Reset state: LINear

Key path: [Power]—>[Power Sweep]—>[Step Mode]

Notice: **Power sweep does not support logarithmic mode yet.**

Remarks: Commands are valid when the power sweep function option is installed in the instrument (S16)

[:SOURce[1]|2]:SWEep:POWer:STARt:TRIGger

Function description: When the start sweep trigger of the power sweep is set as the trigger key, this command will execute a trigger to start the power sweep (only setting).

Setting format: [:SOURce[1]|2]:SWEep:POWer:STARt:TRIGger

Parameter description: No parameters

Example: :SWEep:POWer:STARt:TRIGger

Reset state: None

Key path: [Power]—>[Power Sweep]—>[Start Sweep Trigger Trigger Key]—>[Trigger]

Remarks: Commands are valid when the power sweep function option is installed in the instrument (S16)

[:SOURce[1]|2]:SWEep:POWer:STARt:TRIGger:SOURce <Mode>

Function description: Set the start sweep trigger type of power sweep, including automatic, external, trigger key.

Setting format: [:SOURce[1]|2]:SWEep:POWer:STARt:TRIGger:SOURce IMMediate
|EXTernal|KEY

Query format: [:SOURce[1]|2]:SWEep:POWer:STARt:TRIGger:SOURce?

Parameter description:

<Mode> discrete data; with values taken as follows:

IMMediate: Auto; the trigger signal is always true. When the sweep is set to ON state, the system will start power sweep automatically.

EXTernal: External; the trigger signal is from the trigger input connector on the rear panel,
The power sweep is not started until an external signal is received.

KEY: Trigger key; the trigger signal is from the trigger key on the front panel or programmed control command.

Example: :SWEep:POWer:STARt:TRIGger:SOURce EXTernal
Set the start sweep trigger of channel A as external.

Reset state: IMMdediate

Key path: [Power]—>[Power Sweep]—>[Start Sweep Trigger]

Remarks: Commands are valid when the power sweep function option is installed in the instrument (S16)

[:SOURce[1]|2]:SWEep:POWer:STEP[:LINear] <val>

Function description: This command sets the power step value when the power sweep step mode is linear. After the step value is set, the sweep will be performed according to the step value during step sweep.

Setting format: [:SOURce[1]|2]:SWEep:POWer:STEP[:LINear] <value>

Query format: [:SOURce[1]|2]:SWEep:POWer:STEP[:LINear]?

Parameter description:

<val > Floating point type, value range
[0.01dB,170dB]

Example: :SWEep:POWer:STEP 2dB
This command sets the step value to 2dB when the power sweep is linear.

Reset state: 1dB

Key path: [Power]—>[Power Sweep]—>[Step Mode Linear] —>[Step Value]

Remarks: Commands are valid when the power sweep function option is installed in the instrument (S16)

[:SOURce[1]|2]:SWEep:POWer:STEP:LOGarithmic <val>

Function description: This command sets the power step value when the power sweep mode is logarithmic. After the power step value is set, the power sweep will sweep according to the step value

Setting format: [:SOURce[1]|2]:SWEep:POWer:STEP:LOGarithmic <value>

Query format: [:SOURce[1]|2]:SWEep:POWer:STEP:LOGarithmic?

Parameter description:

<val> Floating point type, value range [0.01, 100]

Example: :SWEep:POWer:STEP:LOGarithmic 5 |

Set the step 5% in logarithmic mode of power sweep.

Reset state: 10

Key path: [Power]—>[Power Sweep]—>[Step Mode Logarithmic] —>[Step Value]

Remarks: Commands are valid when the power sweep function option is installed in the instrument (S16)

[:SOURce[1]|2]:SWEep:POWer:TRIGger

Function description: When the power sweep step trigger is the trigger key, this command executes a trigger, and the power is swept to the next power point (only setting).

Setting format: [:SOURce[1]|2]:SWEep:POWer:TRIGger

Parameter description: No parameters

Example: :SWEep:POWer:TRIGger

Trigger once.

Reset state: None

Key path: [Power]—>[Power Sweep]—>[Power Step Trigger Trigger Key]—>[Trigger]

Remarks: Commands are valid when the power sweep function option is installed in the instrument (S16)

[:SOURce[1]|2]:SWEep:POWer:TRIGger:SOURce <Mode>

Function description: This command sets the trigger source of power sweep. The trigger source has three modes: Auto, Ext and Trigger Key.

Setting format: [:SOURce[1]|2]:SWEep:POWer:TRIGger:SOURce IMMediate|EXTernal|KEY

Query format: [:SOURce[1]|2]:SWEep:POWer:TRIGger:SOURce?

Parameter description:

<Mode> discrete data. The values of step sweep trigger source are as follows:

IMMediate

Automatic, the trigger signal is always true, and the power sweep automatically jumps to the next power point.

EXTernal external; the trigger signal is from the trigger input connector on the rear panel. The power sweep jumps to the next power point only after an external signal is received.

KEY |trigger key; the trigger signal is from the trigger key on the front panel or programmed control command,

After the trigger signal is received, the power sweep will jump to the next power point.

Example: :SWEep:POWer:TRIGger:SOURce KEY

Set the trigger mode of channel power sweep to trigger key.

Reset state: IMMediate

Key path: [Power]—>[Power Sweep]—>[Power Step Trigger]

Remarks: Commands are valid when the power sweep function option is installed in the instrument (S16)

[:SOURce[1]|2]:SWEep:RETRace <State>

Function description: Set the sweep to ON/OFF state. After the completion of a single sweep, whether the output frequency of the signal generator is kept at the first point or the last point, this command can only be used in the single sweep mode. This command is the same as " [:SOURce[1]|2]:LIST:RETRace(?) ".

Setting format: [:SOURce[1]|2]:SWEep:RETRace ON|OFF|1|0

Query format: [:SOURce[1]|2]:SWEep:RETRace?

Parameter description:

<State> Boolean data, with the values listed below:

ON | 1: When the sweep is set to ON state, the output frequency is kept at the first frequency point after sweep.

OFF | 0: When the sweep is set to OFF state, the output frequency is kept at the last frequency point after sweep.

Example: : SWEep:RETRace 0 when the channel A sweep is set to OFF state, after the completion of a single sweep, the continuous wave frequency output by the signal generator will reside at the last frequency point.

Reset state: 0

Key path: [Sweep] -> [Sweep Mode]->[Sweep ON/OFF]

[:SOURce[1]|2]:SWEep:TIME <value>

Function description: This command sets the sweep time of the slope sweep. When the sweep time is set to manual, this command setting is valid. For the manual and automatic commands used to set the sweep time, refer to "[:SOURce[1]|2]:SWEep:TIME:AUTO".

Setting format: [:SOURce[1]|2]:SWEep:TIME <value>

Query format: [:SOURce[1]|2]:SWEep:TIME?

Parameter description:

<Val> Floating point type data, sweep time.
Range: [10ms, 100s], specifically related to the sweep start and stop frequencies set.

Example: :SWEep:TIME 1s

Set the slope sweep time of channel A to 1s.

Reset state: 10.000ms

Key path: [Sweep]—>[Slope Sweep]—>[Sweep Time Setting Manual]->[Sweep Time]

Remarks: The command is valid when the instrument is equipped with the analog sweeping function option (S15)

[:SOURce[1]|2]:SWEep:TIME:AUTO <state>

Function description: This command sets the sweep time type for slope sweep.

Setting format: [:SOURce[1]|2]:SWEep:TIME:AUTO <state>

Query format: [:SOURce[1]|2]:SWEep:TIME:AUTO?

Parameter description:

<state> Boolean type, sweep time type.
ON | 1: Auto
OFF | 0: Manual

Example: :SWEep:TIME:AUTO OFF

Set the slope sweep time of channel A to manual.

Reset state: 1

Key path: [Sweep]—>[Slope Sweep]—>[Sweep Time Setting]

Remarks: The command is valid when the instrument is equipped with the analog sweeping function option (S15)

3.3.7 PULM Subsystem

This subsystem command is used to control the pulse modulation function of the RF output signal, and it is valid only after the pulse modulation function (S12) option is installed in the instrument, otherwise an error message of "undefined command" will be generated.

The following commands are used to set the pulse modulation mode, including:

[:SOURce[1]|2]:PULM:INTernal:DELay <DelayTime>

Function description: This command is used to set the pulse delay of pulse modulation. The maximum value to be set for pulse delay actually depends on the pulse period currently set by the user. In addition, it should be noted that only when automatic, square, dual pulse and trigger are selected as pulse source will the setting of pulse delay work, and when selecting the trigger mode, there is inherent pulse delay of 100ns. Please refer to“[:SOURce[1]|2]:PULM:INTernal:PERiod”, “[:SOURce[1]|2]:PULM:SOURce” for relevant commands.

Setting format: [:SOURce[1]|2]:PULM:INTernal:DELay <val>

Query format: [:SOURce[1]|2]:PULM:INTernal:DELay?

Parameter description:

<DelayTime> floating point type, pulse delay time for pulse modulation.

Range: non-triggered mode: [0ns, period value],

Trigger mode: [100ns，100.000000000s]。

Example: :PULM:INTernal:DELay 1ms set the pulse delay of channel A to 1ms

Reset state: 0s

Key path: [Pulse Modulation]—> [Pulse Modulation]—>[Pulse Source Auto|Square Wave|Double Pulse]—>[Delay]

[:SOURce[1]|2]:PULM:INTernal:DELay:STEP <DelayTime>

Function description: This command sets the step value of each pulse delay. After this value is set, refer to"[:SOURce[1]|2]:PULM:INTernal:DELay" command for the setting of pulse delay time, and use UP|DOWN to change the pulse delay time which will vary according to the currently set step value.

Setting format: [:SOURce[1]|2]:PULM:INTernal:DELay:STEP <val>

Query format: [:SOURce[1]|2]:PULM:INTernal:DELay:STEP?

Parameter description:

<DelayTime> pulse delay time step for pulse modulation.

Range: [0s,1ms],

Example: :PULM:INTernal:DELay:STEP 1ms set the pulse delay step of channel A to 1ms

Reset state: 10ns

Key path: None

[:SOURce[1]|2]:PULM:INTernal:FREQuency <Frequency>

Function description: This command is used to set the pulse modulation frequency. For pulse source commands, refer to"[:SOURce[1]|2]:PULM:SOURce".

Setting format: [:SOURce[1]|2]:PULM:INTernal:FREQuency <val>

Query format: [:SOURce[1]|2]:PULM:INTernal:FREQuency?

Parameter description:

<Frequency> pulse modulation frequency.

Range: With narrow pulse option: [0.01Hz, 25MHz].

Without narrow pulse option: [0.01Hz, 8.333333333MHz].

Example: :PULM:INTernal:FREQuency 1MHz set the pulse frequency of channel A to 1MHz.

Reset: 1kHz

Key path: [Pulse Modulation]—>[Pulse Modulation]—>[Repetition Frequency]

[:SOURce[1]|2]:PULM:INTernal:FREQuency:STEP < val >

Function description: This command sets the pulse modulation repetition frequency step. After this value is set, refer to the "[:SOURce[1]|2]:PULM:INTernal:FREQuency "command for the setting of pulse modulation repetition frequency, and use UP|DOWN to change the repetition frequency value which will vary according to the currently set step value.

Setting format: [:SOURce[1]|2]:PULM:INTernal:FREQuency:STEP <val>

Query format: [:SOURce[1]|2]:PULM:INTernal:FREQuency:STEP?

Parameter description:

< val > Pulse modulation repetition frequency step.

Range:[0, 25MHz]

Example: :PULM:INTernal:FREQuency:STEP 10kHz set the pulse frequency of channel A to 10kHz.

Reset: 1kHz

Key path: None

[:SOURce[1]|2]:PULM:INTernal:JITTered:MODE <Mode>

Function description: This command is used to set the jittered mode of pulse modulation period, including uniform and Gaussian. In uniform mode, the pulse period changes within the range of 0-10%. In Gaussian mode, the pulse period changes within the range of 0-10% in the way of Gaussian distribution.

Setting format: [:SOURce[1]|2]:PULM:INTernal:JITTered:MODE UNIForm|GAUSsian

Query format: [:SOURce[1]|2]:PULM:INTernal:JITTered:MODE?

Parameter description:

<Mode> discrete data. The values of jittered mode of pulse modulation are as follows:

UNIForm	: Uniform
GAUSsian	: Gaussian.

Example: :SOURce2:PULM:INTernal:JITTered:MODE GAUS sets the pulse jitter mode of channel B to Gaussian.

Reset state: UNIForm

Key path: [Pulse Modulation]—>[Pulse Modulation]—>[Pulse Source Jitter]—>[Jitter Mode]

[:SOURce[1]|2]:PULM:INTernal:JITTered:PERCent <Percent>

Function description: This command is used to set the jittered percent of pulse modulation signal. When jittered is selected as the pulse source, the pulse period changes while the pulse width remains unchanged.

Setting format: [:SOURce[1]|2]:PULM:INTernal:JITTered:PERCent <val>

Query format: [:SOURce[1]|2]:PULM:INTernal:JITTered:PERCent?

Parameter description:

<Percent> jittered percent of pulse modulation.

Range: [2, 100].

Example: :SOURce2:PULM:INTernal:JITTered:PERCent 5 set the pulse jittered percent of channel B is 5%.

Reset state: 10

Key path: [Pulse Modulation]—>[Pulse Modulation]—>[Pulse Source Jitter]—>[Jitter Percentage]

[:SOURce[1]|2]:PULM:INTernal:PERiod <Period>

Function description: This command is used to set the period of the pulse signal generated within the signal generator. If the set period is less than or equal to the current pulse width, the pulse width will be automatically adjusted to be less than the pulse period. For the pulse width command, refer to "[:SOURce[1]|2]:PULM:INTernal:PWIDth".

Setting format: [:SOURce[1]|2]:PULM:INTernal:PERiod <value>

Query format: [:SOURce[1]|2]:PULM:INTernal:PERiod?

Parameter description:

<Percent> pulse period.
Range: With narrow pulse option: [40ns, 100.000000000s].
Without narrow pulse option: [120ns, 100.000000000s].

Example: :PULM:INTernal:PERiod 10ms set the pulse signal period of channel A to 10ms.

Reset state: 1.000000ms

Key path: [Pulse Modulation] -- > [Pulse Modulation] -- > [Period]

[:SOURce[1]|2]:PULM:INTernal:PERiod:STEP[:INCRement] <Period>

Function description: This command sets the step value of each pulse period. After this value is set, refer to "[:SOURce[1]|2]:PULM:INTernal:PERiod" command for the setting of pulse period, and use UP|DOWN to change the pulse period which will vary according to the currently set step value.

Setting format: [:SOURce[1]|2]:PULM:INTernal:PERiod:STEP <value>

Query format: [:SOURce[1]|2]:PULM:INTernal:PERiod:STEP?

Parameter description:

<Percent> pulse period step.
Range:[0ns, 100.000000000s].

Example: :PULM:INTernal:PERiod:STEP 10ms means that the period step of the pulse signal from channel A is set to 10ms.

Reset state: 10ns

Key path: None

[:SOURce[1]|2]:PULM:INTernal:PTRain:DATA <PlsWidth>,<PlsPerd> { PlsWidth , PlsPerd …}

Function description: When the pulse source for pulse modulation is in multi-pulse mode, this command is used to set its pulse train. The parameter of the pulse train are: pulse width and pulse period. Maximum 1024 pulse trains are supported. Pulse width and period should be set in pairs. Otherwise, the command parameter is invalid. When this command is executed, it will first determine the currently second flush string file. If no file is selected, a default file named DefaultPtrain will be automatically created in the SgData/user/Train/ folder, and the file will be selected to set the incoming parameters; When the pulse train file is selected, modify the data in the specified file; The modification takes effect immediately after being saved to the file. Paired pulse width values in the incoming parameters shall be smaller than the period value, otherwise a prompt of "numerical data error" will be generated.

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

Parameter description:

<PlsWidth> Pulse width of pulse train.
Range: With narrow pulse option: [40ns, 99.999999980s].
Without narrow pulse option: [120ns, 99.999999980s].

< PlsPerd> Pulse period of pulse train.
Range: With narrow pulse option: [40ns, 100.000000000s].
Without narrow pulse option: [120ns, 100.000000000s].

Example: :PULM:INTernal:PTRain:DATA 100us,1ms,200us,2ms
Set pulse trains with pulse width of 100us and period of 1ms as well as pulse width of 200us and period of 2ms.

Key path: [Pulse Modulation]—>[Pulse Modulation]—>[Pulse Source Pulse Train]—>[Edit File]

Description: for setting only.

[:SOURce[1]|2]:PULM:INTernal:PTRain:DELete <Index>

Function description: This command is used to delete any index point in the pulse train list of the signal generator. If the index to be deleted exceeds the range of list points, the deletion is invalid. For this reason, users may query the current list points before deletion. If the number of points is queried to be 1, in order to effectively delete the pulse train list, the index value should be set to zero. Please refer to "[:SOURce[1]|2]:PULM:INTernal:PTRain:POINts" for pulse train points.

Setting format: [:SOURce[1]|2]:PULM:INTernal:PTRain:DELete <Index>

Parameter description:

<Index> pulse train list index.

Range: [0,1023].

Example: :PULM:INTernal: PTRain:DELete 1 delete the point with index number 1 in the current pulse train list.

Key path: [Pulse Modulation]—>[Pulse Modulation]—>[Pulse Source Pulse Train]—>[Edit Pulse Train]—>[Delete]

]

Description: for setting only.

[:SOURce[1]|2]:PULM:INTernal:PTRain:POINts?

Function description: This command is used to query the current pulse train points. Zero indicates that the current list is empty, and non-zero indicates the actual number of points in the current list. It should be noted that if the user manually operates the instrument interface, the index in the pulse train list starts from zero, that is, if the number of list points is queried to be 1, the index number actually displayed in the list is 0. If the pulse train file is not selected, the instrument will receive an error prompt of "execution error" when receiving this command; A timeout error will occur on the upper computer.

Query format: [:SOURce[1]|2]:PULM:INTernal:PTRain:POINts?

Returned value:

<Num> integer data, pulse train list points returned.

Range: [0,1024].

Example: :PULM:INTernal:PTRain:POINts? query the current pulse train list points.

Reset state: 0

Key path: None

Description: For query only.

[:SOURce[1]|2]:PULM:INTernal:PTRain:PRESet

Function description: This command is used to delete all points in the pulse train list. Before executing this command, it is necessary to first select a pulse train file, otherwise an error prompt of "execution error" will be generated; If the current list is empty, there will be no action. Users may query the current list points before deletion. Please refer to“[:SOURce[1]|2]:PULM:INTernal:PTRain:POINts”for pulse train points.

Setting format: [:SOURce[1]|2]:PULM:INTernal:PTRain:PRESet

Example: [:SOURce[1]| 2]:PULM:INTernal:PTRain:PRESet delete all points in the pulse train list.

Key path: [Pulse Modulation]—>[Pulse Modulation]—>[Pulse Source Pulse Train]—>[Edit Pulse Train]—>[Delete]
]

Description: for setting only.

[:SOURce[1]|2]:PULM:INTernal:PTRain:SELect <FileName>

Function description: Choose to load a pulse train file. If the file does not exist, create an empty file with the parameter of the pulse file name selected to load. The file name cannot include a path, but only the loaded file name. The default file path is "/home/ceyear/SgData/user/Train/".

Setting format: [:SOURce[1]|2]:PULM:INTernal:PTRain:SELect “FileName”

Parameter description: String type

Example: :PULM:INTernal:PTRain:SELect “Test.tra”. Choose to load SgData/user/Train/Test.tra pulse train file.

Key path: [Pulse Modulation]—>[Pulse Modulation]—>[Pulse Source Pulse Train]—>[Select Pulse Train File]

Description: for setting only.

[:SOURce[1]|2]:PULM:INTernal:PTRain:STORe <FileName>

Function description: Save the current pulse train to a file, with the parameter being the file name and there is no need to specify a file path. The default path is "/home/ceyear/SgData/user/Train/".

Setting format: [:SOURce[1]|2]:PULM:INTernal:PTRain:STORe “FileName”

Parameter description: String type

Example: :PULM: Internal: PTRain: STORe "Test. tra". Save the current pulse train data to the file Test. tra.

Key path: [Pulse Modulation]—>[Pulse Train]—>[Select Pulse Train File]—>[Edit File]—>[Save As]

Description: for setting only.

[:SOURce[1]|2]:PULM:INTernal:PWIDth <PWidth>

Function description: This command is used to set the pulse width of the pulse signal generated within the signal generator. If the pulse width value set is greater than or equal to the current pulse period, the pulse period will be automatically adjusted to a value greater than the current pulse period. In addition, if the pulse width set is less than 1us, it is recommended to perform the power search function. When the pulse source is the pulse stagger mode, the pulse width in the stagger list is the uniform value. It is also necessary to change the pulse width in the stagger list through this command.

Setting format: [:SOURce[1]|2]:PULM:INTernal:PWIDth <val>

Query format: [:SOURce[1]|2]:PULM:INTernal:PWIDth?

Parameter description:

<Pwidth> float type data, pulse signal width.
Range: With narrow pulse option: [40ns, 99.999999980s].
Without narrow pulse option: [120ns, 99.999999980s].

Example: :PULM:INTernal:PWIDth 10us set the pulse width of channel A to 10us.

Reset state: 50.000us

Key path: [Pulse Modulation]—>[Pulse Modulation]—>[Pulse Width]

[:SOURce[1]|2]:PULM:INTernal:PWIDth:STEP <Pwidth>

Function description: This command sets the step value for each pulse width. After this value is set, refer to"[:SOURce[1]|2]:PULM:INTernal:PWIDth"command for the setting of pulse width, and use UP|DOWN to change the pulse width which will vary according to the currently set step value.

Setting format: [:SOURce[1]|2]:PULM:INTernal:PWIDth:STEP <val>

Query format: [:SOURce[1]|2]:PULM:INTernal:PWIDth:STEP?

Parameter description:

<Pwidth> float type data, pulse signal width step.
Range:[0ns, 99.990s].

Example: :PULM:INTernal:PWIDth:STEP 10us set the pulse width step of channel A to 10us.

Reset state: 10ns

Key path: None

[:SOURce[1]|2]:PULM:INTernal:SLIDing:MODE <Mode>

Function description: When sliding is selected as the pulse source, this command is used to set the pulse sliding mode.

Setting format: [:SOURce[1]|2]:PULM:INTernal:SLIDing:MODE POSitive|NEGative|TRIangle

Query format: [:SOURce[1]|2]:PULM:INTernal:SLIDing:MODE?

Parameter description:

<Mode> discrete data. The sliding mode takes the following values,

POSitive	sliding mode is positve
NEGative	sliding mode is negative
TRIangle	sliding mode is triangular

Example: :PULM:INTernal:SLIDing:MODE POSitive Set channel A sliding mode to positive

Reset: POSitive

Key path: [Pulse Modulation]—>[Pulse Modulation]—>[Pulse Source Sliding]—>[Sliding Mode]

[:SOURce[1]|2]:PULM:INTernal:SLIDing:POINts <Num>

Function description: When sliding is selected as the pulse source, this command is used to set the pulse sliding points.

Setting format: [:SOURce[1]|2]:PULM:INTernal:SLIDing:POINts <Num>

Query format: [:SOURce[1]|2]:PULM:INTernal:SLIDing:POINts?

Parameter description:

<StepTime> shaping data, sliding counts.
Range:
Non triangular sliding mode: [2, 1024].
Triangular sliding mode: [3.1024]

Example: :PULM:INTernal:SLIDing:POINts 10 set the number of pulse sliding points to 10.

Reset: 1024

Key path: [Pulse Modulation] -- > [Pulse Modulation] -- > [Pulse Source Sliding] -- > [Sliding Counts]

[:SOURce[1]|2]:PULM:INTernal:SLIDing:STEP <StepTime>

Function description: When sliding is selected as the pulse source, this command is used to set the pulse sliding step. The pulse signal is based on the current pulse period and increases sequentially with the set sliding step value. For example, if the number of step points is 1024, the current pulse period is 1ms and the sliding step value is set to 100us, the pulse period will change from 1ms to 103.4ms.

Setting format: [:SOURce[1]|2]:PULM:INTernal:SLIDing:STEP <val>

Query format: [:SOURce[1]|2]:PULM:INTernal:SLIDing:STEP?

Parameter description:

<StepTime> float type data, sliding step.
Range:
Non triangular sliding mode: [0s, (100s - period)/(sliding counts -1)].
Triangular sliding mode: [0s, (100s - period)/(sliding counts -1)/2]

Example: PULM:INTernal:SLIDing:STEP 100us set the pulse sliding step of channel A to 100us.

Reset: 100ns

Key path: [Pulse Modulation]—>[Pulse Modulation]—>[Pulse Source Sliding]—>[Sliding Step]

[:SOURce[1]|2]:PULM:INTernal:STAGger:DATA <PlsWidth>,<PlsPerd>,{PlsPerd, …}

Function description: When the pulse source for pulse modulation of the signal generator is in stagger mode, this command is used to set its pulse stagger data (i.e. stagger file data). When no stagger file is selected, this command will automatically select the defaultStagger.sta file (if DefaultStagger.sta does not exist, create the file first and then select it), and modify the data in the file. The pulse width of all pulses in the file is PlsWidth, and the period is the incoming PlsPerd. The maximum number of periods is 1024.

Setting format: [:SOURce[1]|2]:PULM:INTernal:STAGger:DATA <pls_width>,<pls_period>{,pls_period…}

Parameter description:

<PlsWidth> float type, pulse train pulse width, only the first data in the parameter is the pulse width value.

Range:

With narrow pulse options [20ns, 99.9999999980s].

Without narrow pulse options [100ns, 99.9999999980s]

<PlsPerd> floati type, pulse train pulse period, parameters from the second data to the last data are period data.

Range:

With narrow pulse option: [40ns, 100.000000000s].

Without narrow pulse option [120ns, 100.00000000 s].

Example: :PULM:INTernal:STAGger:DATA 100us,1ms,200us,2ms

Set a pulse stagger string with a pulse width of 100us and a period of 1ms, 200us, and 2ms.

Key path: [Pulse Modulation]—>[Pulse Modulation]—>[Pulse Source Stagger]—>[Edit File]

Description: for setting only.

[:SOURce[1]|2]:PULM:INTernal:STAGger:DELete <Index>

Function description: This command is used to delete any index in the pulse stagger list of the signal generator. If the index number to be deleted exceeds the range of list points, the deletion is invalid. For this reason, users may query the current stagger list points before deletion. It should be noted that if the number of stagger points is queried to be 1, users shall set the index value to zero for effective deletion. For the stagger list points command, refer to“[:SOURce[1]|2]:PULM:INTernal:STAGger:POINts ”. If no file is selected for the current stagger, an error message of "execution error" will be generated.

Setting format: [:SOURce[1]|2]:PULM:INTernal:STAGger:DELete <num>

Parameter description:

<Index> pulse stagger list index.
Range: [0,1023].

Example: : PULM:INTernal:STAGger:DELete 1 Delete the stagger point with index number 1 in the current pulse stagger list.

Reset state: 0

Key path: [Pulse Modulation]—>[Pulse Modulation]—>[Stagger]—>[Edit File...]—>
[Delete Current Point]

Description: for setting only.

[:SOURce[1]|2]:PULM:INTernal:STAGger:INSert <Index>,<PlsPerd>

Function description: This command is used to modify the period of existing pulse stagger points, or to insert a new stagger point at the end of the stagger. The user writes parameters in the order of index number and pulse period, where the index starts from 0, and specifies the modified stagger point. When the index is greater than or equal to the stagger point, data is added at the end of the stagger. Therefore, when using this command, users can first query the current number of stagger points. In addition, the pulse width is a unified value in the stagger list. For related commands, please refer to: "[:SOURce[1]|2]:PULM:INTernal:STAGger:POINts", "[:SOURce[1]|2]:PULM:INTernal:PWIDth".

Setting format: [:SOURce[1]|2]:PULM:INTernal:STAGger:INSert <index>,<pls_period>

Parameter description:

<Index> integer data, stagger list point index.
Range: [0,1023].

<PlsPerd> float data, pulse period.
Range:
With narrow pulse option: [20ns, 100s].
Without narrow pulse option [120ns, 100s]

Example: :PULM:INTernal:STAGger:INSert 1,1ms
Modify the data period with index number 1 in the current stagger list to 1ms.

Key path: None

Description: for setting only.

Note: If the index set exceeds the number of index points in the current list, the instrument will automatically modify the index to the current point value and insert data at the modified index position.

[:SOURce[1]|2]:PULM:INTernal:STAGger:POINts?

Function description: This command is used to query the current pulse stagger points. Zero indicates that there are no stagger points in the current list. Non-zero indicates the actual points in the current list. It should be noted that if the user manually operates the instrument interface, the index in the stagger list starts from zero, that is, if the number of list points is queried to be 1, the index number actually displayed in the stagger list is 0. If the stagger file is not selected, the instrument will receive an error prompt of "execution error" when receiving this command; A timeout error will be generated on the upper computer.

Query format: [:SOURce[1]|2]:PULM:INTernal:STAGger:POINts?

Returned value:

<Num>	integer data, stagger list points returned. Range: [0,1024].

Example: :PULM:INTernal:STAGger:POINts?
Query the current stagger list points.

Reset state: 0

Key path: None

Description: For query only.

[:SOURce[1]|2]:PULM:INTernal:STAGger:PRESet

Function description: This command clears all points in the pulse stagger list. Before executing this command, it is necessary to first select a stagger file, otherwise an error prompt of "execution error" will be generated; If there are no stagger points in the current list, the operation is invalid. Therefore, you can first query the stagger points in the current list before deleting it. For the command to query the number of points in the stagger list, refer to "[:SOURce[1]|2]:PULM:INTernal:STAGger:POINts".

Setting format: [:SOURce[1]|2]:PULM:INTernal:STAGger:PRESet

Example: :PULM:INTernal:STAGger:PRESet delete all points in the stagger list.

Key path: [Pulse Modulation]—>[Pulse Modulation]—>[Pulse Source Stagger]—>[Edit File...]—>
[Delete]

Description: for setting only.

[:SOURce[1]|2]:PULM:INTernal:STAGger:SELect <FileName>

Function description: Choose to load a pulse stagger file. If the file does not exist, create an empty file, with the parameter being the name of the file selected to be loaded, which cannot include a path. The default path is

"/home/ceyear/SgData/user/Stagger/".

Setting format: [:SOURce[1]|2]:PULM:INTernal:STAGger:SELect "FileName"

Parameter description: String type

Example: : PULM: Internal: STAGGer: SELECT "text. sta" Select to load the text. sta stagger file.

Key path: [Pulse Modulation]—>[Pulse Modulation]—>[Pulse Source Stagger]—>[Select Stagger File]

Description: for setting only.

[:SOURce[1]|2]:PULM:INTernal:STAGger:STORe <FileName>

Function description: Save the current pulse stagger to the file. Save the current pulse train to the file, with the parameter being the file name and there is no need to specify a file path. The default path is "/home/ceyear/SgData/user/Stagger/"

Setting format: [:SOURce[1]|2]:PULM:INTernal:STAGger:STORe "FileName"

Parameter description: String type

Example: : PULM: Internal: STAGGer: STORe "text. sta" Save the stagger data to the /home/ceyear/SgData/user/Stagger/text.sta file.

Key path: None

Description: for setting only.

[:SOURce[1]|2]:PULM:SOURce <Mode>

Function description: The command is used to set the pulse source mode of pulse modulation, including: external, scalar, auto, square wave, D-pulse, pulse train, gated control, trigger, stagger, jitter and sliding. In the scalar mode, it is not allowed to change the related pulse parameters, and the signal generator will automatically output pulse signal with pulse width of 18 microsecond and period of 36 microsecond.

Setting format: [:SOURce[1]|2]:PULM:SOURce EXTernal|SCALar|INTernal|SQUare |DOUBlet|PTRain|GATed|TRIGgered| STAGger | JITTered | SLIDing

Query format: [:SOURce[1]|2]:PULM:SOURce?

Parameter description:

<Mode> discrete data. The values of pulse source mode are as follows:

EXTernal	The pulse source is external.
SCALar	The pulse source is scalar, with 27.8kHz square wave output.
INTernal	The pulse source is internal.
SQUare	The pulse source is square.
DOUBlet	The pulse source is doublet.

PTRain	The pulse source is the pulse train.
GATed	The pulse source is gated control.
TRIGgered	Activate the internal pulse automatic trigger mode, in which the period is that of the external synchronization pulse and the pulse width is that set by the machine.
STAGger	The pulse source is stagger.
JITTered	The pulse source is jittered.
SLIDing	The pulse source is sliding.

Example: :PULM:SOURce SQUare set the pulse source to square mode.

Reset state: INTernal

Key path: [Pulse Modulation]—>[Pulse Modulation]—>[Pulse Source]

[:SOURce[1]|2]:PULM:SOURce:EXTernal <Mode>

Function description: Set and select the external type, with two external pulses: External 1 and External 2.

Setting format: [:SOURce[1]|2]:PULM:SOURce:EXTernal EXTernal[1]|2

Query format: [:SOURce[1]|2]:PULM:SOURce:EXTernal?

Parameter description:

<Mode> discrete data. The values of pulse source mode are as follows:

EXTernal[1]:	The pulse source employs External 1 externally.
EXTernal2:	The pulse source employs External 2 externally.

Example: :PULM:SOURce:EXTernal EXTernal2 Set to use External 2.

Reset state: EXTernal[1]

Key path: [Pulse Modulation]—>[Pulse Modulation]—>[Pulse Source External]—>[External Selection]

[:SOURce[1]|2]:PULM:SOURce:EXTernal[1]|2:IMPedance <Mode>

Function description: Set the external impedance value when the pulse source is external.

Setting format: [:SOURce[1]|2]:PULM:SOURce:EXTernal[1]|2:IMPedance OHM50|OHM1K

Query format: [:SOURce[1]|2]:PULM:SOURce:EXTernal[1]|2:IMPedance?

Parameter description:

<Mode> discrete data. Impedance:

OHM50:	The external impedance is 50 Ω.
OHM1K	The external impedance is 1k Ω.

Example: :PULM:SOURce:EXTernal2 OHM1K Set the impedance of External 2 to 1k Ω.

Reset state: OHM50

Key path: [Pulse Modulation]—>[Pulse Modulation]—>[Pulse Source External]—>[External 1 | 2 Impedance]

[:SOURce[1]|2]:PULM:STATe <State>

Function description: This command is used to set whether the pulse modulation signal of the signal generator is output.

Setting format: [:SOURce[1]|2]:PULM:STATe ON|OFF|1|0

Query format: [:SOURce[1]|2]:PULM:STATe?

Parameter description:

<State> Boolean data, with the values listed below:

ON | 1: Pulse modulation ON,

OFF | 0: Pulse modulation OFF.

Example: :PULM:STATe 1 the pulse modulation state is ON.

Reset: 0

Key path: [Pulse Modulation]—>[Pulse Modulation]**—>** [Pulse Modulation ON/OFF]

3.3.8 AMPLitude MODulation Subsystem

The AMPLitude MODulation subsystem is used to configure the amplitude modulation function. The subsystem command is only valid after installing the analog modulation function (S11) option in the instrument. Otherwise, an error prompt of "undefined command" will be generated.

The following commands are used to set the amplitude modulation (AM) mode, including:

[:SOURce[1]|[2]:AM[1]|2[:DEPTh]:EXPonential <val>

Function description: When the AM type is exponential, set the AM depth of the AM signal of AM Path 1 or Path 2 with dB as the unit. For amplitude modulation type commands, refer to:“[:SOURce[1]|[2]:AM:TYPE”.

Setting format: [:SOURce[1]|[2]:AM[1]|2[:DEPTh]:EXPonential <val>

Query format: [:SOURce[1]|[2]:AM[1]|2[:DEPTh]:EXPonential?

Parameter description:

<val> float type, AM depth (index).
Range:[0.00dB, 40.00dB].

Example: :AM2:DEPTh:EXPonential 10dB set the AM depth of AM path 2 of RF channel A to 10dB.

Reset state: 30dB

Key path: [AM1|2]—> [AM Type Index]—>[Modulation Depth]

[:SOURce[1]| 2]:AM[1]|2[:DEPTh][:LINear] <AmDepthLine>

Function description: This command is used to set the AM signal depth of AM Path 1 or Path 2 expressed as a percent. The set value only works when the AM type is linear. Please refer to the command“[:SOURce[1]|[2]:AM:TYPE”.

Setting format: [:SOURce[1]|[2]:AM[1]|2[:DEPTh][:LINear] <.val>

Query format: [:SOURce[1]|[2]:AM[1]|2[:DEPTh][:LINear]?

Parameter description:

<AmDepthExp> AM depth (linear).
Range: [0, 100].

Example: :AM:DEPTh 10 set the linear AM depth of AM path 1 of RF channel A to 10%

Reset state: 30

Key path: [AM1|2]—>[AM Type Linear]—>[Modulation Depth]

[:SOURce[1]|[2]:AM[:DEPTh]:STEP[:INCRement] <AmDepthStep>

Function description: This command sets the step value for each step of linear AM depth. This command does not distinguish between channels A and B, and A and B share this step value synchronously. After this value is set, refer to the ”[:SOURce[1]|[2]:AM[1]|2[:DEPTh][:LINear]”command for the setting of linear AM depth, and use UP|DOWN to change the linear AM depth which will vary according to the currently set step value.

Setting format: [:SOURce[1]|[2]:AM[:DEPTh]:STEP[:INCRement] <val>

Query format: [:SOURce[1]|[2]:AM[:DEPTh]:STEP[:INCRement]?

Parameter description:

<AmDepthStep> AM depth step (linear).

Range: [0, 100].

Example: :AM:DEPTh:STEP 5 set the linear AM depth step to 5%

Reset state: 0

Key path: None

[:SOURce[1]|2]:AM[1]|2:EXTernal[1]|2:COUPling <Mode>

Function description: This command sets the AM external input coupling mode, which is the same as [:SOURce]:EXTernal[1]|2[:SOURce]:COUPling“.

Setting format: [:SOURce[1]|2]:AM[1]|2:EXTernal[1]|2:COUPling DC|AC

Query format: [:SOURce[1]|2]:AM[1]|2:EXTernal[1]|2:COUPling?

Parameter description:

<Mode> discrete data. The values of AM external input coupling mode are as follows:

DC	DC coupling
AC	AC coupling

Example: :AM:EXTernal:COUPling AC set the external input coupling mode to AC coupling.

Reset state: DC

Key path: [AM1|2]—>[Data Source]—>[External]—>[Input Coupling Mode]

[:SOURce[1]|2]:AM[1]|2:INTernal:FREQuency <Frequency>

Function description: This command sets the internal modulation rate of the amplitude modulation channel of the signal generator; In addition, this command can also set the first tone when the AM waveform is a dual sine, and the AM waveform is selected as the start frequency of the sweep sine. For the command of AM signal output waveform, refer to “[:SOURce[1]|2]:AM[1]|2:INTernal:SHAPe ”.

Setting format: [:SOURce[1]|2]:AM[1]|2:INTernal:FREQuency <val>

Query format: [:SOURce[1]|2]:AM[1]|2:INTernal:FREQuency?

Parameter description:

<Frequency> AM rate.

Range: [0.01Hz～ 100kHz].

Sweep sine start frequency

Range: [0.001Hz, 9.99999999MHz]. When 10 MHz is input, the software will automatically set the sweep sine start frequency to 9.99999999MHz without error message.

Dual sine frequency 1

Range: [0.001Hz, 10MHz].

Example: :AM:INTernal:FREQuency 100kHz set the internal AM rate of Path 1 to 100kHz.

Reset state: 1kHz.

Key path: [AM1|2]—>[Modulation Rate]

[:SOURce[1]|2]:AM[1]|2:INTernal:FREQuency:ALTernate <Frequency>

Function description: This command sets the second tone when the AM waveform of the signal generator is selected as a dual sine, or the stop frequency when the AM waveform is selected as a sweep sine. For the command of AM signal output waveform, refer to "“[:SOURce[1]|2]:AM[1]|2:INTernal:SHAPe”.

Setting format: [:SOURce[1]|2]:AM[1]|2:INTernal:FREQuency:ALTernate <.val>

Query format: [:SOURce[1]|2]:AM[1]|2:INTernal:FREQuency:ALTernate?

Parameter description:

<Frequency> The frequency 2 when the AM waveform is dual sine or the stop frequency when the AM waveform is sweep sine.

Range: Dual sine [0.001Hz, 10.000000000 MHz].

Sweep sine [0.002Hz, 10.000000000MHz].

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

Key path: **[**AM1|2**]—**>[AM Waveform Sweep Sine]—>[Sweep Sine Stop Frequency],

[AM1|2**]—**>[AM Waveform Dual Sine]—>[Dual Sine Frequency 2].

[:SOURce[1]|2]:AM[1]|2:INTernal:FREQuency:ALTernate:AMPLitude:PERCent<Pert>

Function description: This command sets the amplitude of the second tone as a percentage of the total output signal amplitude in the dual sine waveform when the AM output waveform is dual sine; For example, if the second tone accounts for 20% of the total waveform power, then the first tone accounts for 80% of the total power output.

Setting format:
[:SOURce[1]|2]:AM[1]|2:INTernal:FREQuency:ALTernate:AMPLitude:
PERCent <val>

Query format: [:SOURce[1]|2]:AM[1]|2:INTernal:FREQuency:ALTernate:AMPLitude:
PERCent?

Parameter description:

<Pert> Amplitude percent of frequency 2 when the AM waveform is dual sine.
Range: [0, 100].

Example: :AM:INTernal:FREQuency:ALTernate:AMPLitude:PERCent 20
Set the second waveform of the dual sine to account for 20% of the total signal power output.

Reset state: 50

Key path: **[**AM|2**]—**>[AM Waveform Dual Sine]—>[Amplitude Percent of Dual

Sine Frequency 2]

[:SOURce[1]|2]:AM[1]|2:INTernal:FREQuency:STEP <Frequency>

Function description: This command sets the step value of the AM rate. After this value is set, refer to the ”[:SOURce[1]|2]:AM[1]|2:INTernal:FREQuency”command for the setting of AM rate, and use UP|DOWN to change the AM rate which will vary according to the currently set step value. This step value is invalid for waveforms with sweep sine and dual sine.

Setting format: [:SOURce[1]|2]:AM[1]|2:INTernal:FREQuency:STEP <val>

Query format: [:SOURce[1]|2]:AM[1]|2:INTernal:FREQuency:STEP?

Parameter description:

<Frequency> AM modulation rate step.
Range: [0Hz, 1MHz].

Example: :AM:INTernal:FREQuency:STEP 1kHz set the internal modulation rate step of channel 1 AM to 1kHz.

Reset state: 1kHz

Key path: None

[:SOURce[1]|2]:AM[1]|2:INTernal:SHAPe <Mode>

Function description: This command is used to set the output waveform of AM signal, including sine, square, triangle, ramp wave, noise, sweep frequency sine and dual sine. When the waveform is set to ramp or noise, the waveform will be automatically set to a certain ramp or noise, depending on the ramp type or noise type set last time. The default is an up ramp or white noise.

Setting format: [:SOURce[1]|2]:AM[1]|2:INTernal:SHAPe SINE|SQUare|TRIangle|RAMP|NOISe|SWEPtsine|DUALsine

Query format: [:SOURce[1]|2]:AM[1]|2:INTernal:SHAPe?

Parameter description:

<Mode> discrete data. The values of output waveform of AM signal are as follows:

SINE	Sine wave
SQUare	square wave
TRIangle	triangle wave
RAMP	ramp wave
NOISE	noise
SWEPtsine	Sweep sine
DUALsine	Dual sine

Example: :AM:INTernal:SHAP RAMP set the AM signal waveform of Path 1 to ramp.

Reset state: SINE

Key path: [AM1|2]—>[AM Waveform]

[:SOURce[1]|2]:AM[1]|2:INTernal:SHAPe:NOISe <Mode>

Function description: This command sets the signal output type when the AM waveform is noise, including white noise and Gaussian noise. For the AM signal output waveform command, refer to“[:SOURce[1]|2]:AM[1]|2:INTernal:SHAPe”.

Setting format: [:SOURce[1]|2]:AM[1]|2:INTernal:SHAPe:NOISe UNIForm|GAUSsian

Query format: [:SOURce[1]|2]:AM[1]|2:INTernal:SHAPe:NOISe?

Parameter description:

<Mode> discrete data. The values of signal output type when the AM waveform is noise are as follows:

UNIForm	white noise,
GAUSsian	Gaussian noise.

Example: :AM:INTernal:SHAPe:NOISe GAUS AM noise is Gaussian noise.

Reset state: UNIF

Key path: **[**AM|2**]—**>[AM Waveform]—>[Noise]

[:SOURce[1]|2]:AM[1]|2:INTernal:SHAPe:RAMP <Mode>

Function description: This command is used to set the signal output type when the AM waveform of Path 1 or Path 2 is ramp, including up and down. Please refer to “[:SOURce[1]|2]:AM[1]|2:INTernal:SHAPe” for the output waveform of AM signal.

Setting format: [:SOURce[1]|2]:AM[1]|2:INTernal:SHAPe:RAMP POSitive|NEGative

Query format: [:SOURce[1]|2]:AM[1]|2:INTernal:SHAPe:RAMP?

Parameter description:

<Mode> discrete data. The values of signal output type when the AM waveform is ramp are as follows:

POSitive	up
NEGative	down.

Example: :AM:INTernal:SHAPe:RAMP NEG the AM ramp signal is up.

Reset state: POSitive

Key path: [AM1|2]—>[AM Waveform]—>[Ramp Wave]

[:SOURce[1]|2]:AM[1]|2:INTernal:SWEep:TIME <Time>

Function description: This command sets the sweep time of the sweep sine when the AM output waveform is set to the sweep sine.

Setting format: [:SOURce[1]|2]:AM[1]|2:INTernal:SWEep:TIME <val>

Query format: [:SOURce[1]|2]:AM[1]|2:INTernal:SWEep:TIME?

Parameter description:

<Time> Sweep time when the AM output waveform is a sweep sine wave. Range:[10.000us, 2s].

Example: :AM:INTernal:SWEep:TIME 1s
The sweep sine of AM signal sweep sine is 1s.

Reset state: 10.000ms

Key path: **[**AM1|2**]—**> [AM Waveform Sweep Sine]—>[Sweep Sine Sweep Time]

[:SOURce[1]|2]:AM:MODE <Mode>

Function description: This command is used to set the AM mode. When DEEP mode is selected, the AM depth of the signal generator has a larger dynamic range than the modulation depth when the ALC loop is closed, and the AM index is better than the index in the data manual.When NORMal mode is selected, the AM index is the same as that in the data manual. Please refer to the data index of 1466 series signal generator.

Setting format: [:SOURce[1]|2]:AM:MODE DEEP|NORMal

Query format: [:SOURce[1]|2]:AM:MODE?

Parameter description:

<Mode> discrete data. The values of AM mode are as follows:

DEEP	AM Depth on,
NORMal	Deep AM OFF.

Example: :AM:MODE NORM AM Depth off.

Reset state: NORMal

Key path: [AM1|2]—>[Depth AM ON/OFF]

[:SOURce[1]|2]:AM[1]|2:SOURce <Mode>

Function description: This command is used to select the AM source, including: INTernal, FUNCtion and EXTernal[1]|2. When external mode is selected, it is required to connect the external AM signal to the AM input interface on the rear panel of the signal generator.

Setting format: [:SOURce[1]|2]:AM[1]|2:SOURce INTernal|FUNCtion|EXTernal[1]|2

Query format: [:SOURce[1]|2]:AM[1]|2:SOURce?

Parameter description:

<Mode> discrete data. The values of AM source are as follows:

INTernal	Internal modulation source.
FUNCtion	Function generator
EXTernal[1]	External 1
EXTernal2	External 2

Example: :AM:SOURce INT the AM source is internal.
Reset state: INTernal
Key path: [AM1|2]—> [AM Source Selection]

[:SOURce[1]|2]:AM[1]|2:STATe <Mode>

Function description: This command is used to set the AM Path 1 or Path 2 of the signal generator to ON/OFF state. Only when the path and modulation are both set to ON state can the AM signal be output.
Setting format: [:SOURce]:AM[1]|2:STATe ON|OFF|1|0
Query format: [:SOURce]:AM[1]|2:STATe?
Parameter description:
<State> Boolean data, with the values listed below:
ON | 1: Path output ON,
OFF | 0: Path output OFF.
Example: :AM:STATe 1 Path 1 output ON.
Reset state: 0
Key path: [AM1|2]—> [Amplitude Modulation]

[:SOURce[1]|2]:AM:TYPE <Mode>

Function description: This command is used to select the AM type of the signal generator: exponential or linear. When the user selects exponential AM, the AM depth will be in dB. When the user selects linear AM, the AM depth will be expressed as a percent.
Setting format: [:SOURce[1]|2]:AM:TYPE EXPonential|LINear
Query format: [:SOURce[1]|2]:AM:TYPE?
Parameter description:
<Mode> discrete data. The values of AM type are as follows:

EXPonential	exponential AM
LINear	linear AM

Example: :AM:TYPE EXP exponential AM
Reset state: LINear
Key path: [AM1|2]—>[AM Type Exponential Linear]

3.3.9 FREQuency MODulation Subsystem

The FREQuency MODulation subsystem is used to configure the FM function. The subsystem command is only valid after installing the analog modulation function (S11) option in the instrument. Otherwise, an error prompt of "undefined command" will be generated.

The following commands are used to select the frequency modulation (FM) mode, including:

[:SOURce[1]|2]:FM[1]|2[:DEViation] <Deviation>

Function description: This command is used to set the frequency deviation of FM Path 1 or Path 2 for the signal generator. It should be noted that different frequency bands should correspond to different frequency deviation ranges when setting frequency deviation.

Setting format: [:SOURce[1]|2]:FM[1]|2[:DEViation] <val>

Query format: [:SOURce[1]|2]:FM[1]|2[:DEViation]?

Parameter description:

<Deviation> the relationship between the current frequency and the frequency deviation is as follows:

Current frequency	FM offset
6kHz– 50MHz	1Hz – 20MHz
50MHz – 75MHz	1Hz – 0.078125MHz
75MHz – 125MHz	1Hz – 0.15625MHz
50MHz – 75MHz	1Hz – 0.078125MHz
50MHz – 75MHz	1Hz – 0.078125MHz
50MHz – 75MHz	1Hz – 0.078125MHz
50MHz – 75MHz	1Hz – 0.078125MHz
50MHz – 75MHz	1Hz – 0.078125MHz
50MHz – 75MHz	1Hz – 0.078125MHz
8GHz – 20GHz	1Hz – 20MHz
50MHz – 75MHz	1Hz – 0.078125MHz
50MHz – 75MHz	1Hz – 0.078125MHz
50MHz – 75MHz	1Hz – 0.078125MHz

Example: :FM:DEViation 500kHz set the frequency deviation of FM signal for

FM Path 1 to 500kHz.

Key path: [FM1|2]—>[FM Offset]

[:SOURce[1]|2]:FM[:DEViation]:STEP[:INCRement] <val>

Function description: This command sets the step value of the FM offset. After this value is set, refer to" [:SOURce[1]|2]:FM[1]|2:[DEViation]"command for the setting of FM offset, and use UP|DOWN to change the value of the FM offset which will vary according to the currently set step value.

Setting format: [:SOURce[1]|2]:FM[:DEViation]:STEP[:INCRement] <val>

Query format: [:SOURce[1]|2]:FM[:DEViation]:STEP[:INCRement]?

Parameter description:

< val > FM offset step (linear).

Range: [0, 160MHz].

Example: :FM: STEP 10MHz Set the FM offset step to 10MHz.

Reset state: 0

Key path: None

[:SOURce[1]|2]:FM[1]|2:EXTernal[1]|2:COUPling <Mode>

Function description: This command sets the FM external input coupling mode, which is equivalent to "[:SOURce]:EXTernal[1]|2[:SOURce]:COUPling".

Setting format: [:SOURce[1]|2]:FM[1]|2:EXTernal[1]|2:COUPling DC|AC

Query format: [:SOURce[1]|2]:FM[1]|2:EXTernal[1]|2:COUPling?

Parameter description:

<Mode> discrete data. The values of FM external input coupling mode are as follows:

DC	DC coupling
AC	AC coupling

Example: :FM:EXTernal:COUPling AC set the external input coupling mode to AC coupling.

Reset state: DC

Key path: [FM1|2]—>[Data Source]—>[External]—>[Input Coupling Mode]

[:SOURce[1]|2]:FM[1]|2:INTernal:FREQuency <Frequency>

Function description: This command is used to set the internal FM rate of FM Path 1 or Path 2 for the signal generator. It should be noted that the internal FM rate cannot be set when external is selected as the FM source. For relevant commands, please refer to"[:SOURce[1]|2]:FM[1]|2:SOURce". In addition, this command can also be used to set the first tone when the AM waveform is a dual sine and selected as the start frequency of the sweep sine. For AM signal output waveform commands, refer to"[:SOURce[1]|2]:FM[1]|2:INTernal:SHAPe"

Setting format: [:SOURce[1]|2]:FM[1]|2:INTernal:FREQuency <val>.

Query format: [:SOURce[1]|2]:FM[1]|2:INTernal:FREQuency?

Parameter description:

<Frequency> FM modulation rate.

Range:

Sine: [0.01Hz, 10MHz]

Square wave, triangle wave and ramp wave: [0.01Hz, 1MHz].

Sweep sine start frequency: [0.001Hz, 9.99MHz 999999]. When 10 MHz is input, the software will automatically set the sweep sine start frequency to 9.99999999MHz without error message.

Dual sine frequency 1: [0.001Hz, 10 MHz].

Example: :FM:INTernal:FREQuency 300kHz set the FM rate of FM Path 1 to 300kHz.

Reset state: 0.001MHz

Key path: [FM1|2]—>[Modulation Rate]

[:SOURce[1]|2]:FM[1]|2:INTernal:FREQuency:ALTernate <Frequency>

Function description: This command sets the second tone when the FM waveform of the signal generator is selected as dual sine, or the stop frequency of the FM waveform is selected as sweep sine. Refer to "[:SOURce[1]|2]:FM[1]|2:INTernal:SHAPe" for the modulation waveform command.

Setting format: [:SOURce[1]|2]:FM[1]|2:INTernal:FREQuency:ALTernate <.val>

Query format: [:SOURce[1]|2]:FM[1]|2:INTernal:FREQuency:ALTernate?

Parameter description:

<Frequency> Frequency 2 for AM waveform dual sine or stop frequency for AM waveform sweep sine.

Range: Dual sine [0.001Hz, 10.000000000 MHz].

Sweep sine [0.002Hz, 10.000000000MHz] .

Example: :FM:INTernal:FREQuency:ALTernate 500kHz

Set the frequency to 500kHz.

Key path: [FM1|2]—>[FM Waveform Sweep Sine]—>[Stop Frequency]

[FM1|2]—>[FM Waveform Dual Sine]—>[Frequency 2].

[:SOURce[1]|2]:FM[1]|2:INTernal:FREQuency:ALTernate:AMPLitude:PERCent<Pert>

Function description: This command sets the amplitude of the second tone as a percentage of the total output signal in the dual sine waveform when the modulation output waveform is dual sine. For example, the second tone accounts for 20% of the total waveform power, then the first tone accounts for 80% of the total power output.

Setting format:

[:SOURce[1]|2]:FM[1]|2:INTernal:FREQuency:ALTernate:AMPLitude:
PERCent <val>

Query format:

[:SOURce[1]|2]:FM[1]|2:INTernal:FREQuency:ALTernate:AMPLitude:
PERCent?

Parameter description:

<Pert> The percent of frequency 2 to amplitude when the modulation output waveform is dual sine.

Range: [0, 100].

Example: :FM:INTernal:FREQuency:ALTernate:AMPLitude:PERCent 20

The second waveform of dual sine accounts for 20% of the total signal output power.

Reset state: 50

Key path: [FM1|2]—>[FM Waveform Dual sine]—>[Dual Sine Frequency 2 Amplitude Percentage].

[:SOURce[1]|2]:FM[1]|2:INTernal:FREQuency:STEP <Frequency>

Function description: This command sets the step value of the modulation rate. After this value is set, refer to the "[:SOURce[1]|2]:FM[1]|2:INTernal:FREQuency"command for the setting of modulation rate, and use UP|DOWN to change the modulation rate which will vary according to the currently set step value. This step value is invalid for waveforms with sweep sine and dual sine.

Setting format: [:SOURce[1]|2]:FM[1]|2:INTernal:FREQuency:STEP <val>

Query format: [:SOURce[1]|2]:FM[1]|2:INTernal:FREQuency:STEP?

Parameter description:

<Frequency> Modulation modulation rate step.

Range: [-325GHz～ +325GHz]. When the modulation rate plus or minus the step value and it exceeds the modulation rate range,

it will be automatically limited to the maximum or minimum value of the modulation rate.

Example: :FM:INTernal:FREQuency:STEP 1kHz set the internal modulation rate step of channel 1 frequency modulation to 1kHz.

Reset state: 0.001Hz

Key path: None

[:SOURce[1]|2]:FM[1]|2:INTernal:SHAPe <Mode>

Function description: This command is used to set the FM signal output waveform of Path 1 or Path 2, including sine, square, triangle and ramp. When the waveform is set to ramp or noise, the waveform will be automatically set to a certain ramp or noise, depending on the ramp type or noise type set last time. The default is an up ramp or white noise.

Setting format: [:SOURce[1]|2]:FM[1]|2:INTernal:SHAPe SINE|SQUare|TRIangle|RAMP|NOISe|SWEPtsine|DUALsine

Query format: [:SOURce[1]|2]:FM[1]|2:INTernal:SHAPe?

Parameter description:

<Mode> discrete data. The values of FM signal output waveform are as follows:

SINE	Sine wave
SQUare	square wave
TRIangle	triangle wave
RAMP	ramp wave
NOISE	noise
SWEPtsine	Sweep sine
DUALsine	Dual sine

Example: :FM2:INTernal:SHAP RAMP set the FM signal waveform of path 2 of channel 1 to ramp.

Reset state: SINE

Key path: [FM|2]—>[FM Waveform]

[:SOURce[1]|2]:FM[1]|2:INTernal:SHAPe:NOISe <Mode>

Function description: This command sets the signal output type when the modulation waveform is noise, including: white noise and Gaussian noise. For the FM selection command, refer to "[:SOURce[1]|2]:FM[1]|2:INTernal:SHAPe".

Setting format: [:SOURce[1]|2]:FM[1]|2:INTernal:SHAPe:NOISe UNIForm|GAUSsian

Query format: [:SOURce[1]|2]:FM[1]|2:INTernal:SHAPe:NOISe?

Parameter description:

<Mode> discrete data. The values of signal output type when FM waveform is noise are as follows:

UNIForm	white noise,
GAUSsian	Gaussian noise.

Example: :modulation:INTernal:SHAPe:NOISe GAUS modulation noise is Gaussian noise.

Reset state: UNIForm

Key path: [FM1|2]—>[FM Noise]

[:SOURce[1]|2]:FM[1]|2:INTernal:SHAPe:RAMP <Mode>

Function description: This command is used to set the type of ramp when the FM waveform of Path 1 or Path 2 is ramp, including up and down. Please refer to the command"[:SOURce[1]|2]:FM[1]|2:INTernal:SHAPe".

Setting format: [:SOURce[1]|2]:FM[1]|2:INTernal:SHAPe:RAMP POSitive|NEGative

Query format: [:SOURce[1]|2]:FM[1]|2:INTernal:SHAPe:RAMP?

Parameter description:

<Mode> discrete data. The values of signal output type when the FM waveform is ramp are as follows:

POSitive	up
NEGative	down.

Example: :FM:INTernal:SHAPe:RAMP NEG set the type of ramp for FM path 1 to down.

Reset state: POSitive

Key path: [FM1|2]—>[FM]—>[Ramp]

[:SOURce[1]|2]:FM[1]|2:INTernal:SWEep:TIME <Time>

Function description: This command sets the sweep time of sweep sine when the FM output waveform is set to sweep sine by the command.

Setting format: [:SOURce[1]|2]:FM[1]|2:INTernal:SWEep:TIME <val>

Query format: [:SOURce[1]|2]:FM[1]|2:INTernal:SWEep:TIME?

Parameter description:

<Time> The sweep time when the FM output waveform is set to sweep sine. Range:[10.000us, 2s].

Example: :FM:INTernal:SWEep:TIME 1s

The sweep time of FM signal sweep sine is 1s.

Reset state: 10.000us

Key path: [FM]—> [FM Waveform Sweep Sine]—>[Sweep Time]

[:SOURce[1]|2]:FM[1]|2:SOURce <Mode>

Function description: This command sets the modulation source, including: INTernal, FUNCtion, EXTernal[1]|2. When external mode is selected, it is required to connect the external FM signal to the FM input interface on the rear panel of the signal generator.

Setting format: [:SOURce[1]|2]:FM[1]|2:SOURce INTernal|FUNCtion|EXTernal[1]|2

Query format: [:SOURce[1]|2]:FM[1]|2:SOURce?

Parameter description:

<Mode> discrete data. The values of FM source are as follows:

INTernal	Internal modulation source.
FUNCtion	Function generator
EXTernal[1]	External 1
EXTernal2	External 2

Example: :FM:SOURce INT the FM source is internal.

Reset state: INTernal

Key path: [FM1|2]—>[Modulation Source Selection]

[:SOURce[1]|2]:FM[1]|2:STATe <Mode>

Function description: This command is used to set the FM Path 1 or Path 2 of the signal generator to ON/OFF state. Only when the frequency modulation and modulation are both set to ON state can the FM signal be output. For modulation switch commands, refer to ":OUTPut[1]|2:MODulation[:STATe]"

Setting format: [:SOURce[1]|2]:FM[1]|2:STATe ON|OFF|1|0

Query format: [:SOURce[1]|2]:FM[1]|2:STATe?

Parameter description:

<State> Boolean data, with the values listed below:

ON | 1: Path output ON,

OFF | 0: Path output OFF.

Example: :FM:STATe 1 The FM output of channel 1 FM path 1 is ON.

Reset state: 0

Key path: [FM1|2]—> [FM]

3.3.10 PHASe MODulation Subsystem

The PHASe MODulation subsystem is used to configure the PM function. The command of this subsystem is valid only after the analog modulation function (S11) option is installed in the instrument, otherwise, an error message of "Undefined Command" will be generated.

The following commands are used to select the phase modulation (PM) mode, including:

[:SOURce[1]|2]:PM:BANDwidth|BWIDth

Function description: This command selects the PM signal bandwidth of the signal generator. Values are discrete variables: 100kHz, 1MHz, 10MHz

Setting format: [:SOURce[1]|2]:PM:BANDwidth|BWIDth 100KHZ|1MHZ|10MHZ

Query format: [:SOURce[1]|2]:PM:BANDwidth|BWIDth?

Parameter description:

<Mode> discrete data. The values of phase modulation bandwidth are as follows:

100KHZ: normal

1MHZ: broadband

10MHZ: low noise

Example: :PM:BWIDth 1MHz The PM bandwidth is set to 1MHz.

Reset state: 100KHZ

Key path: [PM]—>[PM1|2]—>[PM Bandwidth]

[:SOURce[1]|2]:PM[1]|2[:DEViation] <Deviation>

Function description: This command is used to set the phase deviation of PM Path 1 or Path 2 for the signal generator. It should be noted that different frequency bands should correspond to different phase deviation ranges when setting phase deviation.

Setting format: [:SOURce]:PM[1]|2:DEViation <val>

Query format: [:SOURce]:PM[1]|2:DEViation?

Parameter description:

<Deviation> the relationship between the phase deviation range and the PM bandwidth is as follows:

Current Frequency

PM Offset (Bandwidth 100kHz)

PM Offset (Bandwidth 1MHz)

PM Offset (Bandwidth 10MHz)

6kHz– 50MHz 0 – 20.00rad 0 – 20.00rad 0 – 20.00rad

50MHz – 75MHz 0 – 0.078125rad 0 – 0.007812rad 0 – 0.000781rad 75MHz – 125MHz 0 – 0.15625rad 0 – 0.015625rad 0 – 0.001562rad

125MHz – 250MHz 0 – 0.3125rad 0 – 0.03125rad 0 – 0.003125rad

250MHz – 500MHz 0 – 0.625rad 0 – 0.0625rad 0 – 0.00625rad

500MHz – 1GHz 0 – 1.25rad 0 – 0.125rad 0 – 0.0125rad
1GHz – 2GHz 0 – 2.50rad 0 – 0.25rad 0 – 0.025rad
2GHz – 4GHz 0 – 5.00rad 0 – 0.50rad 0 – 0.05rad
4GHz – 8GHz 0 – 10.00rad 0 – 1.00rad 0 – 0.10rad
8GHz – 20GHz 0 – 20.00rad 0 – 2.00rad 0 –0.20rad
20GHz – 40GHz 0 – 40.00rad 0 – 4.00rad 0 – 0.40rad
40GHz – 72GHz 0 – 80.00rad 0 – 8.00rad 0 – 0.80rad
72GHz – 110GHz 0 – 160.00rad 0 – 16.00rad 0 – 1.60rad

Example: :PM2:DEViation 3rad set the PM phase deviation of PM Path 2 of path 1 to 3rad.

Reset state: 0.001rad

Key path: [PM]—>[PM1|2]—>[PM Offset]

[:SOURce[1]|2]:PM[:DEViation]:STEP[:INCRement] <val>

Function description: This command sets the step value of the PM offset. After this value is set, refer to the ” [:SOURce[1]|2]:PM[1]|2 [:DEViation]” command for the setting of PM offset, and use UP|DOWN to change the PM offset value which will vary according to the currently set step value.

Setting format: [:SOURce]:PM[:DEViation]:STEP[:INCRement] <val>

Query format: [:SOURce]:PM[:DEViation]:STEP[:INCRement]?

Parameter description:

< val > PM offset step (linear).

Range: [0, 160rad]. When the PM offset plus or minus the step value and it exceeds the PM offset range, it will be automatically limited to the maximum or minimum value.

Example: :PM:STEP 10rad This example Set the PM offset step to 10rad.

Reset state: 0

Key path: None

[:SOURce[1]|2]:PM[1]|2:EXTernal[1]|2:COUPling <Mode>

Function description: This command sets the PM external input coupling mode, which is the same as “[:SOURce]:EXTernal[1]|2[:SOURce]:COUPling”.

Setting format: [:SOURce[1]|2]:PM[1]|2:EXTernal[1]|2:COUPling DC|AC

Query format: [:SOURce[1]|2]:PM[1]|2:EXTernal[1]|2:COUPling?

Parameter description:

<Mode> discrete data. The values of PM external input coupling mode are as follows:

DC	DC coupling
AC	AC coupling

Example: :PM:EXTernal:COUPling AC set the external input coupling mode to AC coupling.

Reset state: DC

Key path: [PM]—>[PM1|2]—> [Data Source]—>[External]—>[Input Coupling Mode]

[:SOURce[1]|2]:PM[1]|2:INTernal:FREQuency <Frequency>

Function description: This command is used to set the internal PM rate of PM Path 1 or Path 2 for the signal generator. It should be noted that the internal PM rate cannot be set when external is selected as the PM source. For related commands, refer to“[:SOURce[1]|2]:PM[1]|2:SOURce”. In addition, this command can also be used to set the first tone when the AM waveform is a dual sine and selected as the start frequency of the sweep sine. For AM signal output waveform commands, refer to“[:SOURce[1]|2]:PM[1]|2:INTernal:SHAPe”

Setting format: [:SOURce]:PM[1]|2:INTernal:FREQuency <val>

Query format: [:SOURce]:PM[1]|2:INTernal:FREQuency?

Parameter description:

<Frequency> PM modulation rate

Range:

Sine: [0.01Hz, 10MHz]

Square wave, triangle wave and ramp wave: [0.01Hz, 1MHz].

Sweep sine start frequency: [0.001Hz, 9.99MHz 999999]. When 10 MHz is input, the software will automatically set the sweep sine start frequency to 9.99999999MHz without error message.

Dual sine frequency 1: [0.001Hz, 10 MHz].

Example: :PM:INTernal:FREQuency 300kHz set the PM rate of Path 1 to 300kHz.

Reset state: 1kHz

Key path: [PM]—>[PM1|2]—>[Modulation Rate]

[PM]—>[PM1|2]—>[PM Waveform Sweep Sine]1466—>[Sweep Sine Start Frequency]

[PM]—>[PM1|2]—>[PM Waveform Dual Sine]—>[Dual Sine Frequency 1]

[:SOURce[1]|2]:PM[1]|2:INTernal:FREQuency:ALTernate <Frequency>

Function description: This command sets the second tone when the PM waveform of the signal generator is selected as dual sine, or the stop frequency of the PM waveform is selected as sweep sine. For the FM waveform command, refer to “[:SOURce[1]|2]:PM[1]|2:INTernal:SHAPe”.

Setting format: [:SOURce[1]|2]:PM[1]|2:INTernal:FREQuency:ALTernate <.val>

Query format: "[:SOURce[1]|2]:PM[1]|2:INTernal:SHAPe".[:SOURce[1]|2]:PM[1]|2:INTernal:FREQuency:ALTernate?

Parameter description:

<Frequency> The frequency range corresponding to the FM waveform is as follows:

Sweep sine: [0.002Hz, 10.000000000MHz],

Dual sine: [0.001Hz, 10.000000000MHz],

Example: :PM:INTernal:FREQuency:ALTernate 500kHz

Set the frequency to 500kHz.

Key path: [PM]—>[PM1|2]—>[Modulation Waveform Sweep Sine]—> [Sweep Sine Stop Frequency]

[PM]—>[PM1|2]—>[Modulation Waveform Dual Sine]—> [Dual Sine Frequency 2].

[:SOURce[1]|2]:PM[1]|2:INTernal:FREQuency:ALTernate:AMPLitude:PERCent<Pert>

Function description: This command sets the amplitude of the second tone as a percentage of the total output signal in the dual sine waveform when the PM output waveform is dual sine. For example, the second tone accounts for 20% of the total waveform power, then the first tone accounts for 80% of the total power output.

Setting format:

[:SOURce[1]|2]:PM[1]|2:INTernal:FREQuency:ALTernate:AMPLitude:PERCent <val>

Query format:

[:SOURce[1]|2]:PM[1]|2:INTernal:FREQuency:ALTernate:AMPLitude:PERCent?

Parameter description:

<Pert> The percent of frequency 2 to amplitude when the PM output waveform is dual sine.

Range: [0, 100].

Example: :PM:INTernal:FREQuency:ALTernate:AMPLitude:PERCent 20

The second waveform of dual sine accounts for 20% of the total signal output power.

Reset state: 50

Key path: [PM]—>[PM1|2]—> [PM Waveform Dual Sine]—>[Dual Sine Frequency 2 Amplitudes Percent].

].

[:SOURce[1]|2]:PM[1]|2:INTernal:SHAPe <Mode>

Function description: This command is used to set the output waveform of PM path 1 or 2, including sine, square, triangle, ramp, noise, sweep frequency sine and double and the like sine. When the waveform is set to ramp or noise, the waveform will be automatically set to a certain ramp or noise, depending on the ramp type or noise type set last time. The default is an up ramp or white noise.

Setting format: [:SOURce[1]|2]:PM[1]|2:INTernal:SHAPe SINE|SQUare|TRIangle|RAMP|NOISe|SWEPtsine|DUALsine

Query format: [:SOURce[1]|2]:PM[1]|2:INTernal:SHAPe?

Parameter description:

<Mode> discrete data. The values of output waveform of PM signal are as follows:

SINE	Sine wave
SQUare	square wave,
TRIangle	triangle wave
RAMP	ramp wave
NOISE	noise
SWEPtsine	Sweep sine
DUALsine	Dual sine

Example: :PM2:INTernal:SHAP RAMP set the PM signal waveform of Path 2 to ramp.

Reset state: SINE

Key path: [PM]—>[PM1|2]—>[PM Waveform]

[:SOURce[1]|2]:PM[1]|2:INTernal:SHAPe:NOISe <Mode>

Function description: This command sets the signal output type when the PM waveform is noise, including: white noise and Gaussian noise. For the PM waveform selection command, refer to "[:SOURce[1]|2]:PM[1]|2:INTernal:SHAPe".

Setting format: [:SOURce[1]|2]:PM[1]|2:INTernal:SHAPe:NOISe UNIForm|GAUSsian

Query format: [:SOURce[1]|2]:PM[1]|2:INTernal:SHAPe:NOISe?

Parameter description:

<Mode> discrete data. The values of signal output type when FM waveform is noise are as follows:

UNIForm	white noise,
GAUSsian	Gaussian noise.

Example: :PM:INTernal:SHAPe:NOISe GAUS PM noise is Gaussian noise.

Reset state: UNIForm

Key path: [PM]—>[PM1|2]—>[FM Waveform]—>[Noise]

[:SOURce[1]|2]:PM[1]|2:INTernal:SHAPe:RAMP <Mode>

Function description: This command is used to set the direction of ramp when the waveform of PM Path 1 or Path 2 is ramp, including up and down. For PM waveform commands, refer to "[:SOURce[1]|2]:PM[1]|2:INTernal:SHAPe".

Setting format: [:SOURce[1]|2]:PM[1]|2:INTernal:SHAPe:RAMP POSitive|NEGative

Query format: [:SOURce[1]|2]:PM[1]|2:INTernal:SHAPe:RAMP?

Parameter description:

<Mode> discrete data. The values of signal output type when the PM waveform is ramp are as follows:

POSitive	up
NEGative	down.

Example: :PM:INTernal:SHAPe:RAMP NEG set the ramp for PM Path 1 to down.

Reset state: POSitive

Key path: [PM]—>[PM1|2]—>[PM Waveform]—>[Ramp Wave]

[:SOURce[1]|2]:PM[1]|2:INTernal:SWEep:TIME <Time>

Function description: This command sets the sweep time of sweep sine when the PM output waveform is set to sweep sine by the command.

Setting format: [:SOURce[1]|2]:FM[1]|2:INTernal:SWEep:TIME <val>

Query format: [:SOURce[1]|2]:FM[1]|2:INTernal:SWEep:TIME?

Parameter description:

<Time> Sweep time when the PM output waveform is a sweep sine. Range:[10.000us, 2s].

Example: :FM:INTernal:SWEep:TIME 1s

The sweep time of PM signal sweep sine is 1s.

Reset state: 10.000us

Key path: [PM]—>[PM1|2]—>[PM Waveform Sweep Sine]—> [Sweep Sine Sweep Time]

[:SOURce[1]|2]:PM[1]|2:SOURce <Mode>

Function description: This command is used to select the PM source. including: INTernal, FUNCtion and EXTernal[1]|2. When external mode is selected, it is required to connect the external PM signal to the FM/PM input interface on the rear panel of the signal generator.

Setting format: [:SOURce[1]|2]:PM:SOURce INTernal|FUNCtion|EXTernal[1]|2

Query format: [:SOURce[1]|2]:PM:SOURce?

Parameter description:

<Mode> discrete data. The values of PM source are as follows:

INTernal	Internal modulation source.
FUNCtion	Function generator
EXTernal[1]	External 1
EXTernal2	External 2

Example: :PM:SOURce INT the PM source is internal

Reset: INTernal

Key path: [PM]—>[PM1|2]—>[Modulation Source Selection]

[:SOURce[1]|2]:PM[1]|2:STATe <Mode>

Function description: This command is used to set the PM Path 1 or Path 2 of the signal generator to ON/OFF state. Only when the phase modulation and modulation are both set to ON state can the PM signal be output. For modulation switch commands, refer to ":OUTPut[1]|2:MODulation[:STATe]"

Setting format: [:SOURce[1]|2]:PM[1]|2:STATe ON|OFF|1|0

Query format: [:SOURce[1]|2]:PM[1]|2:STATe?

Parameter description:

<State> Boolean data, with the values listed below:

ON | 1: Path output ON,

OFF | 0: Path output OFF.

Example: :PM:STATe 1 The modulation of channel A PM path 1 is ON.

Reset state: 0

Key path: [PM]—>[PM1|2]—>[PM]

3.3.11 Digital MODulation Subsystem

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

[:SOURce[1]|2]:DM:ATTenuation<Val>

Function description: This command sets the gain adjustment of the modulated I/Q signal, and there are six modes available to users: AUTO, DBM6, DBM3, DB0, DB3, and DB6.

Setting format: [:SOURce[1]|2]:DM:ATTenuation <IqGain>

Query format: [:SOURce[1]|2]:DM:ATTenuation?

Parameter description:

<Val> discrete data. The gain value of I/Q modulation is as follows:

AUTO	Auto
DBM6	-6.00dB
DBM3	-3.00dB
DB0	0.00dB
DB3	3.00dB
DB6	6.00dB

Example: :DM:ATTenuation DBM3 The gain value of the I/Q modulator is -3.00dB.

Reset state: AUTO

Key path: [I/Q Modulation]—>[I/Q Setting]—>[Gain Adjustment]

[:SOURce[1]|2]:DM:CORRection

Function description: This command is used to correct the IQ modulation at current frequency point. for setting only.

Setting format: [:SOURce[1]|2]:DM:CORRection.

Parameter description: No parameters

Example: :DM:CORRection Correct the IQ modulation at current frequency point.

Key path: [I/Q Modulation]—>[I/Q Setting]—>[Correct IQ Modulation at current Frequency Point]

[:SOURce[1]|2]:DM:IQADjustment:GAIN <Gain>

Function description: When I/Q input adjustment is ON, set the gain of signal I of the signal generator relative to signal Q. Refer to"[:SOURce[1]|2]:DM:IQADjustment[:STATe]" for the I/Q input adjustment ON/OFF command.

Setting format: [:SOURce[1]|2]:DM:IQADjustment:GAIN <val>

Query format: [:SOURce[1]|2]:DM:IQADjustment:GAIN?

Parameter description:

<Gain> signal I/Q gain balance.
Range:[-4.00dB, +4.00dB].

Example: :DM:IQADjustment:GAIN 0dB
Set I signal gain balance relative to Q as 0 dB.

Reset state: 0dB

Key path: **[I/Q] —>** [I/Q Input Adj] —> [Gain Balance]

[:SOURce[1]|2]:DM:IQADjustment:IOFFset <Offset>

Function description: When I/Q input adjustment is ON, set the offset of Path I of the signal generator. The parameter set is expressed as a percent, with the maximum value corresponding to 1.5V DC. This parameter is used to suppress the carrier leakage signal. After the user completes other adjustments, such as orthogonality adjustment and modulator attenuation, etc., the carrier leakage will increase. Therefore, after completing other adjustments, it is still necessary to adjust the DC offset.

Setting format: [:SOURce[1]|2]:DM:IQADjustment:IOFFset <val>

Query format: [:SOURce[1]|2]:DM:IQADjustment:IOFFset?

Parameter description:

<Offset> signal I offset in I/Q
Range: [-50, +50].

Example: :DM:IQADjustment:IOFfset 30
Set I offset to 30%.

Reset state: 0dB

Key path: **[I/Q] —>** [I/Q Input Adj] —> [I Offset]

[:SOURce[1]|2]:DM:IQADjustment:OUTPut:GAIN <Gain >

Function description: This command is used to set the gain balance of I/Q output adjustment. When I/Q output adjustment is ON, the command works. Please refer to"[:SOURce[1]|2]:DM:IQADjustment:OUTPut[:STATe]".

Setting format: [:SOURce[1]|2]:DM:IQADjustment:OUTPut:GAIN <val>.

Query format: [:SOURce[1]|2]:DM:IQADjustment:OUTPut:GAIN?

Parameter description:

<Gain> I/Q output adjustment gain balance.
Range:[-4dB, 4dB].

Example: :DM:IQADjustment:OUTPut:GAIN 2dB To set I/Q output gain balance to 2dB.

Reset state: 0dB

Key path: [I/Q] —> [I/Q Output Adj] -> [Gain Balance]

[:SOURce[1]|2]:DM:IQADjustment:OUTPut:IOFFset <offset >

Function description: This command is used to set the I offset of I/Q output adjustment. When I/Q output adjustment is ON, the command works. Please refer to“[:SOURce[1]|2]:DM:IQADjustment:OUTPut[:STATe]”.

Setting format: [:SOURce[1]|2]:DM:IQADjustment:OUTPut:IOFFset <val>.

Query format: [:SOURce[1]|2]:DM:IQADjustment:OUTPut: IOFFset?

Parameter description:

<offset> I offset of I/Q output adjustment
Range: [-50, 50].

Example: :DM:IQADjustment:OUTPut:IOFFset 50 set I offset of I/Q output to 50%.

Reset state: 0

Key path: [I/Q] —> [I/Q Output Adj] -> [I Offset]

[:SOURce[1]|2]:DM:IQADjustment:OUTPut:QOFFset <offset >

Function description: This command is used to set the Q offset I/Q output adjustment. When I/Q output adjustment is ON, the command works. Please refer to“[:SOURce[1]|2]:DM:IQADjustment:OUTPut[:STATe]”.

Setting format: [:SOURce[1]|2]:DM:IQADjustment:OUTPut:QOFFset <val>.

Query format: [:SOURce[1]|2] :DM:IQADjustment:OUTPut:QOFFset?

Parameter description:

<offset> Q offset of I/Q output adjustment
Range: 0 [-50, 50].

Example: :DM:IQADjustment:OUTPut:QOFFset 50 set Q offset of I/Q output to 50.

Reset state: 0

Key path: [I/Q] —> [I/Q Output Adj] -> [Q Offset]

[:SOURce[1]|2]:DM:IQADjustment:OUTPut:SKEW <skew >

Function description: This command is used to set the orthogonal offset I/Q output adjustment. When I/Q output adjustment is ON, the command works. Please refer to“[:SOURce[1]|2]:DM:IQADjustment:OUTPut[:STATe]”.

Setting format: [:SOURce[1]|2]:DM:IQADjustment:OUTPut:SKEW <val>.

Query format: [:SOURce[1]|2]:DM:IQADjustment:OUTPut:SKEW?

Parameter description:

<skew> orthogonal offset of I/Q output adjustment
Range:[- 10deg, 10deg].

Example: :DM:IQADjustment:OUTPut:SKEW 1deg To set I/Q output orthogonality offset to 1deg.

Reset state: 0deg

Key path: [I/Q] —> [I/Q Output Adj] -> [Orthogonal Offset]

[:SOURce[1]|2]:DM:IQADjustment:OUTPut[:STATe] <State>

Function description: This command is used to set I/Q input adjustment to ON/OFF state.

Setting format: [:SOURce[1]|2]:DM:IQADjustment:OUTPut[:STATe] ON|OFF|1|0

Query format: [:SOURce[1]|2]:DM:IQADjustment:OUTPut [:STATe]?

Parameter description:

<State> Boolean data, with the values listed below:
ON | 1: I/Q output adjustment ON,
OFF | 0: I/Q output adjustment OFF.

Example: DM:IQADjustment:OUTPut 1 I/Q output adjustment ON

Reset state: 0

Key path: [I/Q Modulation]**—>**[I/Q Output Adj]->[I/Q Output Adj ON OFF]

[:SOURce[1]|2]:DM:IQADjustment:QOFFset <Offset>

Function description: This command is used to set the offset of Path Q of the signal generator. The parameter set is expressed as a percent, with the maximum value corresponding to 1.5V DC. This parameter is used to suppress the carrier leakage signal. After the user completes other adjustments, such as orthogonality adjustment and modulator attenuation, etc., the carrier leakage will increase. Therefore, after completing other adjustments, it is still necessary to adjust the DC offset.

Setting format: [:SOURce[1]|2]:DM:IQADjustment:QOFFset <val>

Query format: [:SOURce[1]|2]:DM:IQADjustment:QOFFset?

Parameter description:

<Offset> signal Q offset in I/Q
Range: [-50, +50].

Example: :DM:IQADjustment:QOFFset 30
Set Q offset to 30%.

Reset state: 0

Key path: [I/Q] **—>** [I/Q Input Adj] —> [Q Offset]

[:SOURce[1]|2]:DM:IQADjustment:QSKew <Offset>

Function description: When I/Q input/output adjustment is ON, this command is used to adjust the phase angle between vector I and vector Q by increasing or decreasing the phase angle of I or Q.

Setting format: [:SOURce[1]|2]:DM:IQADjustment:QSKew <val>

Query format: [:SOURce[1]|2]:DM:IQADjustment:QSKew?

Parameter description:

<Offset> orthogonal offset of I/Q input adjustment.
Range:[-10.00deg, +10.00deg].

Example: :DM:IQADjustment:QSKew 10deg
Set the orthogonal offset of I/Q input adjustment to 10deg.

Reset state: 0deg

Key path: **[I/Q] —**> [I/Q Input Adj] —> [Orthogonal offset]

[:SOURce[1]|2]:DM:IQADjustment[:STATe] <State>

Function description: This command is used to set the I/Q input adjustment enable to ON/OFF state. After this function is enabled, I/Q input adjustment parameters, such as gain balance, orthogonal offset, I offset and Q offset, will be added to the adjustment circuit. When this function is disabled, the above parameters will not be used, but the modulator gain will not be affected by the I/Q adjustment state. For related commands, refer to "[:SOURce[1]|2]:DM:ATTenuation".

Setting format: [:SOURce[1]|2]:DM:IQADjustment[:STATe] ON|OFF|1|0

Query format: [:SOURce[1]|2]:DM:IQADjustment[:STATe] ?

Parameter description:

<State> Boolean data, with the values listed below:
ON | 1: I/Q adjustment ON,
OFF | 0: I/Q adjustment OFF.

Example: :DM:IQADjustment 1 enable I/Q input adjustment function.

Reset state: 0

Key path: [I/Q Modulation]**—**>[I/Q Input Adj]—>[I/Q Input Adj ON OFF]

[:SOURce[1]|2]:DM:IQEXchange:STATe <State>

Function description: This command sets the I/Q exchange ON/OFF, and when the ON/OFF state is set to ON, the two input IQ signals will cross.

Setting format: [:SOURce[1]|2]:DM:IQEXChange:STATe ON|OFF|1|0

Query format: [:SOURce[1]|2]:DM:IQEXChange:STATe?

Parameter description:

<State> Boolean data, with the values listed below:
ON | 1: I/Q exchange ON/OFF state is set to ON.

OFF | 0: exchange ON/OFF state is set to OFF.

Example: :DM:IQEXChange:STATe 1 I/Q exchange ON

Reset state: 0

Key path: [I/Q Modulation]—>[I/Q Setting]->[I/Q Exchange ON OFF]

[:SOURce[1]|2]:DM:OUTPut:AMPLitude<Amp>

Function description: This command sets the value of the I/Q output amplitude.

Setting format: [:SOURce[1]|2]:DM:OUTPut:AMPLitude <Val>

Query format: [:SOURce[1]|2]:DM:OUTPut:AMPLitude?

Parameter description:

<Amp> represents the I/Q output amplitude.
Value range: [0V, 1V]

Example: :DM:OUTP:AMP 1.000 I/Q output amplitude is 1.000V.

Reset state: 1.000V

Key path: [I/Q Modulation]—>[I/Q Output]->[I/Q Output Amplitude]

[:SOURce[1]|2]:DM:OUTPut:ICOFfset <Offset>

Function description: This command sets the common mode voltage (CMV) of the I/Q output.

Setting format: [:SOURce[1]|2]:DM:OUTPut:ICOFFset <Val>

Query format: [:SOURce[1]|2]:DM:OUTPut:ICOFFset?

Parameter description:

<Offset> represents the I-way common mode voltage of the I/Q output.
Value range: [0V, 1V]

Example: :DM:OUTP:ICOFFset 0 The I-way common mode voltage of the I/Q output is 0.

Reset state: 0V

Key path: [I/Q Modulation]—>[I/Q Output]->[I-way CMV]

[:SOURce[1]|2]:DM:OUTPut:QCOFfset<Offset>

Function description: This command sets the Q-way common mode voltage of the I/Q output.

Setting format: [:SOURce[1]|2]:DM:OUTPut:QCOFFset <Val>

Query format: [:SOURce[1]|2]:DM:OUTPut:QCOFFset?

Parameter description:

<Offset> represents the Q-way common mode voltage of the I/Q output.
Value range: [0V, 1V]

Example: :DM:OUTP:QCOFFset 0 The Q-way common mode voltage of the I/Q output is 0.

Reset state: 0V

Key path: [I/Q Modulation]—>[I/Q Output]->[Q-way CMV]

[:SOURce[1]|2]:DM:OUTPut[:STATe] <State>

Function description: This command enables I/Q output ON/OFF state.

Setting format: [:SOURce[1]|2]:DM:OUTPut:STATe ON|OFF|1|0

Query format: [:SOURce[1]|2]:DM:OUTPut:STATe?

Parameter description:

<State> Boolean data, with the values listed below:

ON | 1: I/Q output ON

OFF | 0: I/Q output OFF.

Example: :DM:OUTP:STATe 1 I/Q input OFF.

Reset state: 0

Key path: [I/Q Modulation]—>[I/Q Output]->[I/Q Output ON OFF]

[:SOURce[1]:2]:DM:SOURce <Mode>

Function description: This command is used to select the I/Q modulation source for the signal generator to enter the IQ modulator. Users may select EXTernal or INTernal.

Setting format: [:SOURce[1]|2]:DM:SOURce EXTernal | INTernal

Query format: [:SOURce[1]|2]:DM:SOURce?

Parameter description:

<Mode> discrete data. Values are taken as follows:

EXTernal : external 50ohm impedance matched I/Q signal input,

INTernal : internal I/Q signal input I/Q modulator

Example: : DM:SOURce EXT select the I/Q modulation source to be external.

Reset state: INTernal

Key path: [I/Q Modulation]—>[I/Q Settings]—>[Data Source Selection]

[:SOURce[1]|2]:DM:STATe <State>

Function description: This command is used to enable internal I/Q modulator to ON/OFF state.

Setting format: [:SOURce[1]|2]:DM:STATe ON|OFF|1|0

Query format: [:SOURce[1]|2]:DM:STATe?

Parameter description:

<State> Boolean data, with the values listed below:

ON | 1: I/Q modulation ON,

OFF | 0: I/Q modulation OFF.

Example: :DM:STATe 1 turn on I/Q modulator.

Reset state: 0

Key path: [I/Q Modulation]—>[I/Q Setting]->[I/Q Modulation ON OFF]

3.3.12 MEMory Subsystem

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

:MEMory:CATalog:ALL?

Function description: This command is used to query user-defined FSK modulation files. It will return all folder or file name information in the /home/ceyear/SgData/user/ file, with "," spacing between different file names.

Query format: :MEMory:CATalog:ALL?

Return data: <Used Size> Size of all files in /home/ceyear/SgData/user/ path, in bytes

<Remaining Size> Remaining disk size, in bytes

<File Information> File name, file type, file size; Used between different files

Segmentation. The file types are divided into ASCii, BIN, and DIR (folder).

. The file size is calculated in bytes.

Example: :MEMory:CATalog:ALL?.

Key path: None

Description: For query only.

:MEMory:CATalog:FSK?

Function description: This command is used to query user-defined FSK modulation files. It will return a list of all file names with the suffix ".fsk" in the /home/ceyear/SgData/user/Fsk file, with "," spacing between different file names.

Query format: :MEMory:CATalog:FSK?

Example: :MEMory:CATalog:FSK?

Key path: None

Description: For query only.

:MEMory:CATalog:IQ?

Function description: This command is used to query user-defined IQ files. It will return a list of all file names with the suffix ".iqm" in the /home/ceyear/SgData/user/IqMap/ file, with ";" spacing between different file names.

Query format: :MEMory:CATalog:IQ?

Example: :MEMory:CATalog:IQ?

Key path: None

Description: For query only.

:MEMory:CATalog:LIST?

Function description: This command is used to query user-defined list sweep files. It will return a list of all file names with the suffix ".lst" in the /home/ceyear/SgData/user/List/ file, with "," spacing between different file names.

Query format: :MEMory:CATalog:LIST?

Example: :MEMory:CATalog:LIST?

Key path: None

Description: For query only.

:MEMory:CATalog:MTONe?

Function description: This command is used to query user-defined multi-tone files. It will return a list of all file names with the suffix ".ton" in the /home/ceyear/SgData/user//Mtone/ file, with ";" spacing between different file names.

Query format: :MEMory:CATalog:MTONe?

Example: :MEMory:CATalog:MTONe?

Key path: None

Description: For query only.

:MEMory:CATalog:PTRain?

Function description: This command is used to query user-defined pulse train files. It will return a list of all file names with the suffix ".tra" in the /home/ceyear/SgData/user/Train/ file, with ";" spacing between different file names.

Query format: :MEMory:CATalog:PTRain?

Example: :MEMory:CATalog:PTRain?

Key path: None

Description: For query only.

:MEMory:CATalog:SEQ?

Function description: This command is used to query user-defined sequence files. It will return a list of all file names with the suffix ".seg" in the /home/ceyear/SgData/user/Wav/ file, with ";" spacing between different file names.

Query format: :MEMory:CATalog:SEQ?

Example: :MEMory:CATalog:SEQ?

Key path: None

Description: For query only.

: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. It should be noted that the absolute path must be attached to the file name. The extensions of two files can be different.

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 "/home/ceyear/SgData/user/seq1.dat","/home/ceyear/SgData/data/user/seq2.dat" Copy the data from seq1.dat to seq2.dat.

Key path: **[File]** —> [Copy]

Description: for setting only.

:MEMory:DATA <FileName>,<#AB><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>, <#AB><data_block>

Query format: :MEMory:DATA？ <file_name>

Parameter description:

<FileName> Character string type. Name of arbitrary wave file saved in the signal generator specified by users should not

It includes the path, and the default path is in the "/home/ceyear/SgData/user/Wav/" folder.

<#AB> # is a fixed format that represents the beginning of the data, where A represents the number of digits in the data length B, and B represents the data length. For example, #210, where 2 represents a total of 2 bytes, 10 represents the total number of bytes, and the<DataBlock> data block after 10 represents 10 bytes of data.

<DataBlock> data block to be transmitted.

Example: :MEMory:DATA "arb1.dat", #41024 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.

Notice: Query is not supported at present

:MEMory:DATA:APPend <FileName>,<#AB><DataBlock>

Function description: This command downloads the upper computer data to the instrument and saves the data in an existing file (specified by FileName). It can only transfer data with a byte count of below 1000000000 bytes, that is, data with a total length of less than 10 bits. If the file specified by "FileName" does not exist, the data cannot be saved. This command does not need to specify a file path, and it defaults to the file in "/home/ceyear/SgData/user/Wav/".

Setting format: :MEMory:DATA:APPend <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> # is a fixed format, where # represents the beginning of the data, A represents the length of this data, and B represents the length of data.

For example, #210, where 2 represents a total of 2 bytes,

10 represents the total number of bytes, and <DataBlock> following 10 represents 10 bytes of data.

<DataBlock> data block to be transmitted.

Example: :MEMory:DATA:APPend “arb1.dat”, #15jklas
Represents sending 5 bytes of data to the signal generator and appending the data to the file arb1.dat.

Description: for setting only.

:MEMory:DELete:FSK

Function description: This command deletes all user-defined FSK files in the signal generator, that is, files with the extension ".fsk" in the /home/ceyear/SgData/user/Fsk/ folder.

Setting format: :MEMory:DELete:FSK

Parameter description: No parameters

Example: :MEMory:DELete:FSK Delete all .fsk files in /home/ceyear/SgData/user/Fsk/ folder.

Key path: None

Description: for setting only.

:MEMory:DELete:IQ

Function description: This command deletes all user-defined IQ files in the signal generator, that is, files with the extension ".iqm" in the /home/ceyear/SgData/user/IqMap/ folder.

Setting format: :MEMory:DELete:IQ

Parameter description: No parameters

Example: :MEMory:DELete:IQ Delete all .iqm files in /home/ceyear/SgData/user/IqMap / folder.

Key path: None

Description: for setting only.

:MEMory:DELete:LIST

Function description: This command deletes all user-defined list sweep files in the signal generator, that is, files with the extension ".lst" in the /home/ceyear/SgData/user/List/ folder.

Setting format: :MEMory:DELete:LIST

Parameter description: No parameters

Example: :MEMory:DELete:LIST Delete all .lst files in /home/ceyear/SgData/user/List/ folder

Key path: None

Description: for setting only.

:MEMory:DELete:MTONe

Function description: This command deletes all user-defined multi-tone files in the signal generator, that is, files with the extension ".ton" in the /home/ceyear/SgData/user/Mtone/ folder.

Setting format: :MEMory:DELete:MTONe

Parameter description: No parameters

Example: :MEMory:DELete:MTONeDelete all .ton files in /home/ceyear/SgData/user/Mtone/ folder

Key path: None

Description: for setting only.

:MEMory:DELete[:NAME] <FileName>

Function description: This command deletes user files from the signal generator. The parameters can include a path name. If the parameter is only a file name, the default path will be found based on the file suffix, and the specified file will be searched for deletion. If the file does not exist, an error will be generated. The root directory of the absolute path is "/SgData".

Setting format: :MEMory:DELete[:NAME] <FileName>

Parameter description:

<FileName> Character string type. User file saved in the signal generator.

Example: :MEMory:DELete:NAME “/SgData/user/test.txt”
Delete text.txt files in /home/ceyear/SgData/user/ file

Key path: None

Description: for setting only.

:MEMory:DELete:SEQ

Function description: This command deletes all user-defined sequence files in the signal generator, that is, files with the extension ".seq" in the /home/ceyear/SgData/user/Wav/ folder.

Setting format: :MEMory:DELete:SEQ

Parameter description: No parameters

Example: :MEMory:DELete:SEQ Delete all .seq files in /home/ceyear/SgData/user/Wav/ folder

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. The root directory of the absolute path is "/SgData".

Setting format: :MEMory:MOVE <Sourfile_name>，<Desfile_name>

Parameter description:

<FileName> Character string type. File name saved in the signal generator.

Example: :MEMory:MOVE “/SgData/user/text1.txt”,” /SgData/user/text2.txt”
Rename file text1.txt to text2.txt

Description: for setting only. 5025
5

3.3.13 ROSCillator Subsystem

The oscillator subsystem command is used to realize functions of the signal generator related to time base.

[:SOURce]:ROSCillator[:ADJust]: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.

Setting format: [:SOURce]:ROSCillator[:ADJust]:REFerence <val>

Query format: [:SOURce]:ROSCillator[:ADJust]:REFerence?

Parameter description:

<Val> internal calibration parameter.
Range: [0, 65535].

Example: :ROSCillator:REFerence 30000 Adjust the internal reference accuracy value to 30000.

Reset state: 3000

Key path: [System]—>[Basic Settings]—>[Reference Settings]—>[Internal Reference Accuracy Adjustment]

[:SOURce]:ROSCillator:DEFaults

Function description: Restore the internal reference accuracy adjustment of the configuration to the factory default value.

Setting format: [:SOURce]:ROSCillator:DEFaults

Parameter description: No parameters

Example: :ROSCillator:DEFaults Restore the configuration parameters set by reference to the factory default values.

Key path: [System]—>[Basic Settings]—>[Reference Settings]—>[Restore Factory Default Values]

[:SOURce]:ROSCillator:FREQuency:EXTernal <Val>

Function description: This command is used to set the external reference frequency. When the reference is manually selected as "[:SOURce]:ROSCillator:SOURce:AUTO" and the reference is manually set as "[:SOURce]:ROSCillator:SOURce", this command is valid.

Setting format: [:SOURce]:ROSCillator:FREQuency:EXTernal <val>

Query format: [:SOURce]:ROSCillator:FREQuency:EXTernal?

Parameter description:

<Val>	External reference frequency data. Range: [1MHz～ 100MHz].

Example: :ROSCillator:FREQuency:EXTernal 10MHz Set the external reference frequency to 10MHz.

Reset state: 10MHz

Key path: [System]—>[Basic Settings]—>[Reference Settings]—>[External Reference Frequency]

[:SOURce]:ROSCillator:SOURce <Val>

Function description: This command is used to set the reference source as external or internal. When the reference selection is set to manual, this command will be valid. When the reference selection is set to automatic, this command will be set. Since the instrument will automatically detect the reference source, it will be automatically selected, and there is a situation where this command is invalid.

Setting format: [:SOURce]:ROSCillator:SOURce INTernal|EXTernal

Query format: [:SOURce]:ROSCillator:SOURce?

Parameter description:

<Val>	Reference manual settings. INTernal : The reference source is internal. EXTernal: The reference source is external.

Example: :ROSCillator:SOURce EXT Manually set the reference to external.

Reset state: INTernal

Key path: [System]—>[Basic Settings]—>[Reference Settings]—>[Reference Manual Selection]

[:SOURce]:ROSCillator:SOURce:AUTO <Val>

Function description: This command is used to set the reference selection as Automatic or Manual. When the reference selection is set as automatic, the signal generator automatically detects the input of an external 10MHz reference signal. If an external reference signal input is detected, the reference source automatically switches to external. If there is no external reference signal, the internal reference is used. When the reference selection is set to manual, the user can specify whether the reference source is external or internal.

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

Query format: [:SOURce]:ROSCillator:SOURce:AUTO?

Parameter description:

<Val>	Reference selection. ON|1: Reference selection is set to automatic. OFF|0: Reference selection is set to manual.

Example: :ROSCillator:SOURce:AUTO OFF
Set the reference selection to manual.

Reset state: 1

Key path: [System]—>[Basic Settings]—>[Reference Settings]—>[Reference Selection]

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[:CPU]:INFormation:CCOunt:PON?

Function description: Query the accumulative startup times of the instrument

Query format: :DIAGnostic[:CPU]:INFormation:CCOunt:PON?

Example: :DIAGnostic:CPU: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[:CPU]:INFormation:OPTions?

Function description: This command queries the installed option information of the instrument

Query format: :DIAGnostic[:CPU]:INFormation:OPTions?

Example: :DIAGnostic:CPU:INFormation:OPTions? This example shows the option information for querying the installed signal generator.

Description: For query only.

:DIAGnostic[:CPU]:INFormation:OTIMe?

Function description: This command queries the cumulative startup time of the instrument, in hours

Query format: :DIAGnostic[:CPU]:INFormation:OTIMe?

Example: :DIAGnostic:CPU: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[:SELF]: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. The instrument reset and status call GPIB addresses remain unchanged.

Setting format: :SYSTem:COMMunicate:GPIB[:SELF]:ADDRess <val>

Query format: :SYSTem:COMMunicate:GPIB[:SELF]:ADDRess?

Parameter description:

<Address> integer data, GPIB address.
Range: [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:GTLocal
Switch the signal generator from remote control state to local state.

Description: for setting only.

:SYSTem:COMMunicate:LAN:ADDRess|IP <Address>

Function description: This command is used to set IP address of the signal generator. The parameters are expressed in dotted decimal notation. The instrument reset and status call IP address remain unchanged.

Setting format: :SYSTem:COMMunicate:LAN:ADDRess|IP <ipstring>

Query format: :SYSTem:COMMunicate:LAN:ADDRess|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:DGATeway|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. The instrument reset and state call gateway remain unchanged.

Setting format: :SYSTem:COMMunicate:LAN:DGATeway|GATeway <ipstring>

Query format: :SYSTem:COMMunicate:LAN:DGATeway|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:COMMunicate:LAN:HNAMe|HOSTname <HostName>

Function description: This command is used to set and query the host name of the instrument. To set a host name, the new host name shall not take effect immediately. It is necessary to restart the instrument for the new host name to be effective.

Setting format: :SYSTem:COMMunicate:LAN:HNAMe|HOSTname <HostMane>

Query format: :SYSTem:COMMunicate:LAN:HNAMe|HOSTname?

Parameter description:

<HostMane> with a maximum length of 15 characters and letters, numbers, and ! @ # $% ^ ') allowed
(. - _{ } ~.

Example: SYSTem:COMMunicate:LAN:HOSTname “41-PC” Set the signal generator host name to 41-PC.

Reset state: None

Key path: [System]**—**>[LAN] **—**>[Local Name]

Notice: Setting is not supported at present

:SYSTem:COMMunicate:LAN:MAC?

Function description: This command is set to query the MAC address of the instrument.

Query format: :SYSTem:COMMunicate:LAN:MAC?

Parameter description

Example: :SYSTem:COMMunicate:LAN:MAC?
Query the MAC address of the signal generator.

Key path: None

Description: For query only.

:SYSTem:COMMunicate:LAN:PORT <num>

Function description: This command sets the port number of the network controlled by the signal generator program. After setting by this command, the new port number takes effect immediately, and the original network program connection established through the port number becomes invalid. The connection needs to be re-established using the new port number; The instrument reset and status call port numbers remain unchanged.

Setting format: :SYSTem:COMMunicate:LAN:PORT <num>

Query format: :SYSTem:COMMunicate:LAN:PORT?

Parameter description

<num> Shaping type, with a range of values as follows: [1024.65535]

Example: :SYSTem:COMMunicate:LAN:PORT 5001
Set the port number of the signal generator programmed control to 5001.

Reset state: Port number previously set

Key path: [System] —> [LAN Port] —> [Port number]

:SYSTem:COMMunicate:LAN:RESTart

Function description: This command sets the signal generator to restart the network.

Setting format: :SYSTem:COMMunicate:LAN:RESTart

Example: :SYSTem:COMMunicate:LAN:RESTart
Set the signal generator network to restart.

Key path: None

:SYSTem:COMMunicate:LAN:SMASk|SUBNet <Address>

Function description: This command is used to set the subnet mask address of the signal generator. The parameters are expressed in dotted decimal notation; The instrument reset and status call and subnet mask remain unchanged.

Setting format: :SYSTem:COMMunicate:LAN:SMASk|SUBNet <ipstring>

Query format: :SYSTem:COMMunicate:LAN:SMASk|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: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 cannot be switched in real time, but when the command is used for query, the returned value is the language value after 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 (not supported yet)

Example: SYSTem:DEVice:LANGuage ENGLish set the interface of the instrument to English.

Reset state: Keep the current language.

Key path: **[System]** —> [Basic Settings] —> [Language/LANG]

Description: English is not supported at present

:SYSTem:ERRor:CLEar

Function description: This command clears the signal generator error queue. After clearing, the user will query the information: "+0, No ERROR".

Setting format: :SYSTem:ERRor:CLEar

Example: :SYSTem:ERRor:CLEar Clear signal generator error queue

Description: for setting only.

:SYSTem:ERRor:COUNt?

Function description: This command queries the number of errors recorded in the signal generator error queue.

Query format: :SYSTem:ERRor:COUNt?

Returned value: Number of returned errors

Example: :SYSTem:ERRor:COUNt? Query the number of error messages from the signal generator

Description: For query only.

: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:HELP:HEADers?

Function description: This command queries the list of programmed control commands for the signal generator and returns all programmed control strings supported by the instrument in its current mode. The format of the returned data is as follows:

#nNddd……ddd<LF>

Data newline character indicating the end of the data block.

Data length (i.e., the number of bytes of d)

Number of bits of data length (i.e., the quantity of N)

The marker for the start of the data block.

The format of the data block is # 510331… In < LF>, n=5, N=10331, …, indicating that the returned data is a supported programmable instruction string.

Query format: :SYSTem:HELP:HEADers?

Example: :SYSTem:HELP:HEADers? This example represents querying all supported programmed control commands for the signal generator

Description: For query only.

Notice: The data block returned by this instruction is large and requires reading N+1 bytes until a newline character is read. Otherwise, the returned data will be stored in the network buffer. The next time the data is read, the unread bytes will be read first.

: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: manufacturer
USER: user
LAST: 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]

:SYSTem:PRESet[:USER]:SAVE

Function description: This command is used to save the current state of the user, and is valid when the reset type is user.

Setting format: :SYSTem:PRESet[:USER]:SAVE

Example: :SYSTem:PRESet:SAVE Save user state.

Reset state: None

Key path: [System] ->[Reset State] ->[Reset Type Selection User] ->[Save User State]

Description: for setting only.

:SYSTem:SHUTdown

Function description: This command sets the remote shutdown of the signal generator instrument.

Setting format: :SYSTem:SHUTdown

Example: :SYSTem:SHUTdown Shut down the signal generator.

Key path: None

Description: for setting only.

:SYSTem:VERSion?

Function description: This command queries the SCPI version number used by the instrument. The return form is YYYY.X, where YYYY denotes the year and X denotes the version number. 1999.0 is returned for this instrument.

Query format: :SYSTem:VERSion?

Example: :SYSTem:VERSion?

Key path: None

Description: For query only.

3.3.15 ARB Subsystem

The ARB subsystem is used to configure arbitrary wave function. The subsystem command is valid only when the arbitrary wave (S01) option in installed the instrument, otherwise an error prompt of "Undefined command" will be generated.

The following commands set the relevant configurations for arbitrary wave, including:

3.3 Instrument Subsystem Command

[:SOURce]:RADio:ARB:IMAP[1]|3:TYPE MARKer|TRIGger

Function description: This instruction is used to set the interface type in arbitrary wave interface mapping. When the interface name is "Marker 1/Trigger A" and the channel is Channel A or the interface name is "Marker 3/Trigger B" and the channel is Channel B, the corresponding interface type is set.

Setting format: [:SOURce]:RADio:ARB:IMAP[1]|3:TYPE MARKer|TRIGger

Query format: [:SOURce]:RADio:ARB:IMAP[1]|3:TYPE?

Parameter description:

< num >	Discrete data,	
	MARKer:	Marker
	TRIGger:	Trigger

Example: :RADio:ARB:IMAP3:TYPE TRIGger

Set the interface name to "Marker 3/Trigger B" as the trigger interface for channel B

Reset state: MARKer

Key path: [Baseband] ->[Arbitrary Wave] ->[Interface Mapping] ->[Interface Type]

[:SOURce]:RADio:ARB:IMAP[1]|2|3|4:USED <TYPE>

Function description: This command sets which channel Marker 1/Trigger A, Maker2, Marker 3/Trigger B, and Marker 4 are used for, respectively. This instruction is valid when dual channels are used, and can only be used for channel A and cannot be changed when a single channel is used.

Setting format: [:SOURce]:RADio:ARB:IMAP[1]|2|3|4:USED ACHannel|BCHannel

Query format: [:SOURce]:RADio:ARB:IMAP[1]|2|3|4:USED?

Parameter description:

< num >	Discrete data,	
	ACHannel:	Channel A
	BCHannel:	Channel B

Example: :RADio:ARB:IMAP2:USED BCHannel Set marker 2 for channel B

Reset state: ACHannel

Key path: [Baseband] ->[Arbitrary Wave] ->[Interface Mapping] ->[Channel]

[:SOURce[1]|2]:RADio:ARB:MARKer[1]|2|3|4:DIVider <Val>

Function description: When the marker type in arbitrary wave is selected as pulse, this command is used to set the average value when setting pulse marker.

Setting format: [:SOURce[1]|2]:RADio:ARB:MARKer[1]|2|3|4:DIVider <val>

Query format: [:SOURce[1]|2]:RADio:ARB:MARKer[1]|2|3|4:DIVider?

Parameter description:

<val > Integer data, marked as the average value of the pulse.
Range: [2,1024].

Example: :RADio:ARB:MARKer:DIVider 100
Set the average value of the pulse marker type for Marker 1 to 100.

Reset state: 2

Key path: [Baseband] ->[Arbitrary Wave] ->[Marker Settings] ->[Marker Pulse] ->[Average Value]

[:SOURce[1]|2]:RADio:ARB:MARKer[1]|2|3|4:OFFTime <Val>

Function description: This command sets the number of samples for the OFF time when the marker type is selected as a fixed ON/OFF ratio.

Setting format: [:SOURce[1]|2]:RADio:ARB:MARKer[1]|2|3|4:OFFTime <Val>.

Query format: [:SOURce[1]|2]:RADio:ARB:MARKer[1]|2|3|4:OFFTime?

Parameter description:

< Val > Integer data, the number of OFF time samples when the marker type is fixed ON/OFF ratio.
Range: [1, 16777215]

Example: :RADio:ARB:MARKer:OFFTime 100
When the Marker 1 type is set to fixed ON/OFF ratio, the number of OFF time samples is 100.

Reset state: 1

Key path: [Baseband] ->[Arbitrary Wave] ->[Marker Settings] ->[Marker Fixed ON/OFF Ratio] ->[Number of Off Time Samples]

[:SOURce[1]|2]:RADio:ARB:MARKer[1]|2|3|4:ONTime <Val>

Function description: This command sets the number of samples for the ON time when the marker type is selected as a fixed ON/OFF ratio.

Setting format: [:SOURce[1]|2]:RADio:ARB:MARKer[1]|2|3|4:ONTime <Val>.

Query format: [:SOURce[1]|2]:RADio:ARB:MARKer[1]|2|3|4:ONTime?

Parameter description:

< Val > Integer data, the number of ON time samples when the marker type is fixed ON/OFF ratio
Range: [1, 16777215]

Example: :RADio:ARB:MARKer:ONTime 1
When the Marker 1 type is set to a fixed ON/OFF ratio, the number of ON time samples is 1.

Reset state: 1

Key path: [Baseband] ->[Arbitrary Wave] ->[Marker Settings] ->[Marker Fixed ON/OFF Ratio] ->[Number of ON Time Samples]

[:SOURce[1]|2]:RADio:ARB:MARKer[1]|2|3|4:PATTern <Val>,<BitCount>

Function description: When the marker type is selected as a predefined style, this command sets the value of the predefined style, which can be set according to different decimal systems, but shall always return in hexadecimal (# H) when querying. The instruction contains two parameters, the first parameter Val is unsigned 64 bit integer data, supporting binary (# B), decimal, and hexadecimal (# H) data, representing the set PATTern value, which is displayed in binary form in the instrument software interface; The second parameter, BitCount, is integer signed data, which represents the number of bits set when the data is displayed in binary format. If the number of bits set for displaying Val as binary data does not match the number of bits specified in BitCount, automatic truncation or zeroing will be performed, with the former starting from the low position, and the latter performed at the high position.

Setting format: [:SOURce[1]|2]:RADio:ARB:MARKer[1]|2|3|4:PATTern <Val>,<BitCount>.

Query format: [:SOURce[1]|2]:RADio:ARB:MARKer[1]|2|3|4:PATTern?

Parameter description:

< Val> Unsigned integer data, with the highest bit processed as 1 when set to a negative number.

<BitCount> Integer data, value range: [1,64]

Example: Example of hexadecimal data: RADio: ARB: MARKer2: PATT # HF, 4
Example of binary data: RADio: ARB: MARKer2: PATT # B1111,4
Example of decimal data: Raio: ARB: Marker2: PATT 15,4
When Marker 2 is set as a predefined style, the value is 1111 (binary), and returns to # HF, 4 when queried

Reset state: 10(binary)

Key path: [Baseband] ->[Arbitrary Wave] ->[Marker Settings] ->[Marker Predefined Style ->[Predefined Style]

[:SOURce[1]|2]:RADio:ARB:MARKer[1]|2|3|4:[:TYPE] <Mode>

Function description: This command selects the marker type of arbitrary wave in the signal generator, including five types: UNCHanged, RESTart, PULSe, PATTern, and RATio.

Setting format: [:SOURce[1]|2]:RADio:ARB:MARKer[1]|2|3|4:TYPE UNCHanged|RESTart | PULSe | PATTern | RATio

Query format: [:SOURce[1]|2]:RADio:ARB:MARKer[1]|2|3|4:TYPE?

Parameter description:

<Mode> discrete data. The type of digital modulation marker, with the values

listed below:

UNCHanged: unchanged,

RESTart: start

PULSe :Pulse

PATTern: predefined pattern,

RATio: fixed ON/OFF ratio.

Example: :RADio:ARB:MARKer PULSe The marker type pulse of arbitrary wave.

Reset state: UNCHanged

Key path: [Baseband] ->[Arbitrary Wave] ->[Marker Settings] ->[Marker]

[:SOURce[1]|2]:RADio:ARB:MCARrier:CARRier[1]|2--512:CONFlict?

Function description: This command is used to query whether the specified carrier and all preceding carriers have overlapping conflicts in the frequency domain. If there are conflicts, return to 1. If there are no conflicts, return to 0. The carrier is specified by the suffix after CARRier, and the first carrier (corresponding to the sequence number 0 in the multi carrier interface) does not conflict, so it always returns to 0. When the specified carrier does not exist, an error of "Command suffix out of range" is generated.

Query format:

[:SOURce[1]|2]:RADio:ARB:MCARrier:CARRier[1]|2--512:CONFlict?

Example: RADio:ARB:MCARrier:CARRier2:CONFlict? Check if there is a conflict with the second carrier.

Reset state: 0

Key path: [Baseband] ->[Arbitrary Wave] ->[Basic Configuration] ->[Create Multi Carrier] ->[Multi Carrier List] ->[Identification]

Description: For query only.

Remarks: The instruction is valid when the multi carrier signal generation function option is installed on the instrument (S08)

[:SOURce[1]|2]:RADio:ARB:MCARrier:CARRier:COUNt <Num>

Function description: This command is used to set the number of carriers for multiple carriers. After this value is modified, the number of carriers in the multi-carrier list will also be modified.

Setting format: [:SOURce[1]|2]:RADio:ARB:MCARrier:CARRier:COUNt <Num>

Query format: [:SOURce[1]|2]:RADio:ARB:MCARrier:CARRier:COUNt?

Parameter description:

<Num> Integer data, number of multiple carriers

Range: [1,512]

Example: RADio:ARB:MCARrier:CARRier:COUNt 3 Set the number of multiple carriers to 3.

Reset state: 1

Key path: [Baseband] ->[Arbitrary Wave] ->[Basic Configuration] ->[Create Multiple Carriers] ->[General] ->[Number of Carriers]

Remarks: The instruction is valid when the multi carrier signal generation function option is installed on the instrument (S08)

[:SOURce[1]|2]:RADio:ARB:MCARrier:CARRier[1]|2--512:DELay <Val>

Function description: This command is used to set the delay for the specified multi carrier. The carrier is specified by the suffix after CARRIER, and an error prompt of "Command suffix out of range" is generated when the specified carrier does not exist.

Setting format: [:SOURce[1]|2]:RADio:ARB:MCARrier:CARRier[1]|2--512:DELay <Val>

Query format: [:SOURce[1]|2]:RADio:ARB:MCARrier:CARRier[1]|2--512:DELay?

Parameter description:

<Val> Floating point data, delay time set
Range: [0,1s]

Example: RADio:ARB:MCARrier:CARRier:DELay 1ms Set the delay of the first multi carrier to 1ms.

Reset state: 0

Key path: [Baseband] ->[Arbitrary Wave] ->[Basic Configuration] ->[Create Multi Carrier] ->[Multi Carrier List] ->[Delay]

Remarks: The instruction is valid when the multi carrier signal generation function option is installed on the instrument (S08)

[:SOURce[1]|2]:RADio:ARB:MCARrier:CARRier[1]|2--512:FREQuency <Val>

Function description: This command is used to set the carrier frequency offset of the specified multiple carriers. When the carrier frequency interval mode is custom interval, this command is valid; If the carrier spacing mode is equal spacing, the instruction is invalid. At this time, sending the instruction will generate an error prompt of "Execution error". For details of the carrier compartment mode command, please refer tin [:SOURce[1]|2]:RADio:ARB:MCARrier:CARRier:MODE"; The carrier is specified by the suffix after CARRIER, and when the specified carrier does not exist, an error prompt of "Command suffix out of range" will be generated.

Setting format:

[:SOURce[1]|2]:RADio:ARB:MCARrier:CARRier[1]|2--512:FREQuency <Val>

Query format:

[:SOURce[1]|2]:RADio:ARB:MCARrier:CARRier[1]|2--512:FREQuency?

Parameter description:

<Val> Floating point data, carrier frequency offset set

Scope: The range of multi carrier spacing is related to the options, as follows:

Standard Configuration: [-100MHz, 100MHz].

Option H31: [-250MHz, 250MHz]. Option H31: [-250MHz, 250MHz].

Option H31: [-250MHz, 250MHz]. Option H31: [-250MHz, 250MHz].

Option H31: [-250MHz, 250MHz]. Option H31: [-250MHz, 250MHz].

Example: RADio:ARB:MCARrier:CARRier:FREQuency 1MHz Set the frequency offset of the first multi-carrier to 1MHz.

Reset state: 0

Key path: [Baseband] ->[Arbitrary Wave] ->[Basic Configuration] ->[Create Multi Carrier] ->[Multi Carrier List] ->[Frequency Offset]

Remarks: The instruction is valid when the multi carrier signal generation function option is installed on the instrument (S08)

[:SOURce[1]|2]:RADio:ARB:MCARrier:CARRier:MODE <Mode>

Function description: This command is used to set the spacing mode for multi-carriers. The spacing mode is divided into equal spacing and custom spacing. When the spacing mode is equal spacing, the frequency offset of each carrier in the multi-carrier list cannot be set. All carrier frequency offsets can only be set through carrier spacing. For carrier spacing instructions, please refer to"[:SOURce[1]|2]:RADio:ARB:MCARrier:CARRier:SPACing".

Setting format: [:SOURce[1]|2]:RADio:ARB:MCARrier:CARRier:MODE EQUidistant|ARBitrary

Query format: [:SOURce[1]|2]:RADio:ARB:MCARrier:CARRier:MODE?

Parameter description:

<Mode> Enumeration type

EQUidistant: Equivalent distance

ARBitrary: Arbitrary distance

Example: RADio:ARB:MCARrier:CARRier:MODE ARB Set the distance mode of multiple carriers to arbitrary distance.

Reset state: EQUidistant

Key path: [Baseband] ->[Arbitrary Wave] ->[Basic Configuration] ->[Create Multiple Carriers] ->[General] ->[Distance Mode]

Remarks: The instruction is valid when the multi carrier signal generation function option is installed on the instrument (S08)

[:SOURce[1]|2]:RADio:ARB:MCARrier:CARRier[1]|2--512:PHASe <val >

Function description: This command is used to set the phase of the specified multi carrier. The carrier is specified by the suffix after CARRIER, and an error prompt of "Command suffix out of range" is generated when the specified carrier does not exist.

Setting format: [:SOURce[1]|2]:RADio:ARB:MCARrier:CARRier[1]|2--512:PHASe <Val>

Query format: [:SOURce[1]|2]:RADio:ARB:MCARrier:CARRier[1]|2--512:PHASe?

Parameter description:

<Val> Floating point data, phase set
Range:[0.00deg, 359.99deg]

Example: RADio:ARB:MCARrier:CARRier:PHASe 60deg Set the phase of the first multi carrier to 60deg.

Reset state: 0.00

Key path: [Baseband] ->[Arbitrary Wave] ->[Basic Configuration] ->[Create Multi Carrier] ->[Multi Carrier List] ->[Phase]

Remarks: The instruction is valid when the multi carrier signal generation function option is installed on the instrument (S08)

[:SOURce[1]|2]:RADio:ARB:MCARrier:CARRier[1]|2--512:POWer <val >

Function description: This command is used to set the power gain value of the specified multi carrier; The carrier is specified by the suffix after CARRier. When the specified carrier does not exist, an error prompt of "command suffix out of range" is generated.

Setting format: [:SOURce[1]|2]:RADio:ARB:MCARrier:CARRier[1]|2--512:POWer <Val>

Query format: [:SOURce[1]|2]:RADio:ARB:MCARrier:CARRier[1]|2--512:POWer?

Parameter description:

<Val> Floating point data, carrier power gain value set
Range:[-80dB, 0dB]

Example: RADio:ARB:MCARrier:CARRier:POWer -20dB Set the gain of the first multi carrier to -20dB.

Reset state: 0dB

Key path: [Baseband] ->[Arbitrary Wave] ->[Basic Configuration] ->[Create Multi Carrier] ->[Multi Carrier List] ->[Gain]

Remarks: The instruction is valid when the multi carrier signal generation function option is installed on the instrument (S08)

[:SOURce[1]|2]:RADio:ARB:MCARrier:CARRier:SPACing <val >

Function description: This command is used to set the carrier spacing for multiple carriers. When the carrier frequency spacing is equal, this command is valid. The carrier spacing command is detailed in "[:SOURce[1]|2]:RADio:ARB:MCARrier:CARRier:MODE".

Setting format: [:SOURce[1]|2]:RADio:ARB:MCARrier:CARRier:SPACing <Val>

Query format: [:SOURce[1]|2]:RADio:ARB:MCARrier:CARRier:SPACing?

Parameter description:

<Val> Floating point data, carrier spacing set
Range: [0Hz, maximum modulation bandwidth/(number of carriers -1) Hz]

Example: RADio:ARB:MCARrier:CARRier:SPACing 1MHz Set the carrier spacing to 1MHz.

Reset state: 0Hz

Key path: [Baseband] ->[Arbitrary Wave] ->[Basic Configuration] ->[Create Multiple Carriers] ->[General] ->[Carrier Spacing]

Remarks: The instruction is valid when the multi carrier signal generation function option is installed on the instrument (S08)

[:SOURce[1]|2]:RADio:ARB:MCARrier:CARRier[1]|2--512:STATe <State >

Function description: This command is used to set the state of the specified multi carrier. The carrier is specified by the suffix after CARRIER, and an error prompt of "Command suffix out of range" is generated when the specified carrier does not exist.

Setting format: [:SOURce[1]|2]:RADio:ARB:MCARrier:CARRier[1]|2--512:STATe ON|OFF|1|0

Query format: [:SOURce[1]|2]:RADio:ARB:MCARrier:CARRier[1]|2--512:STATe?

Parameter description:

<State> Enumerated data, state of multiple carriers set
ON|1: ON state
OFF|0: OFF state

Example: RADio:ARB:MCARrier:CARRier:STATe ON Set the state of the first multi carrier to ON.

Reset state: 0

Key path: [Baseband] ->[Arbitrary Wave] ->[Basic Configuration] ->[Create Multi Carrier] ->[Multi Carrier List] ->[State]

Remarks: The instruction is valid when the multi carrier signal generation function option is installed on the instrument (S08)

[:SOURce[1]|2]:RADio:ARB:MCARrier:CARRier[1]|2--512:WAVeform <FileName >

Function description: This command is used to set the file name for the s pecified multi carrier. The parameter cannot include a path name. The default path is the file under the "/home/ceyear/SgData/user/W av/" path. The parameter only includes the file name (with an exte nsion) and the file is in the ". seg" format. When the specified file does not exist, an error message "File name not found" is gener ated. The carrier is specified by the suffix after CARRIER, and an error prompt of "Command suffix out of range" is generated when the specified carrier does not exist.

Setting format: [:SOURce[1]|2]:RADio:ARB:MCARrier:CARRier[1]|2--512:WAVefo rm "FileName"

Query format: [:SOURce[1]|2]:RADio:ARB:MCARrier:CARRier[1]|2--512:WAVefor m?

Parameter description:

<FileName> string type data

Example: RADio:ARB:MCARrier:CARRier:WAVeform "arbMccwOutput.seg" Set the modulation file for the first multi carrier load to "arbMccwOutp ut. seg".

Reset state: ArbMccw.seg

Key path: [Baseband] ->[Arbitrary Wave] ->[Basic Configuration] ->[Create Mul ti Carrier] ->[Multi Carrier List] ->[File Name]

Remarks: The instruction is valid when the multi carrier signal generation fun ction option is installed on the instrument (S08)

[:SOURce[1]|2]:RADio:ARB:MCARrier:CATalog?

Function description: This command is used to query the configuration file of the currently saved multi-carrier. The command will return the file name of the existing multi-carrier configuration file in the "/home/ceyear/SgData/user/McarrierConfig/" path, with different file names divided with ";".

Query format: [:SOURce[1]|2]:RADio:ARB:MCARrier:CATalog?

Example: RADio:ARB:MCARrier:CATlog? Query the list of saved multi-carrier configuration files under the path "home/ceyear/SgData/user/Mcarri erConfig/".

Key path: None

Description: For query only.

Remarks: The command is valid when the instrument is installed with multiple carrier signals to generate function options (S08)

[:SOURce[1]|2]:RADio:ARB:MCARrier:CONFigure:LOAD <FileName >

Function description: This command is used to load a multi-carrier configuration file for multi-carrier settings. The parameter of this command is the name of the multi-carrier configuration file, which must include the extension ". mcarcfg" and the parameter does not include a path name.

The default path is the file under the "/home/ceyear/SgData/user/McarrierConfig/" path. If the specified file does not exist, an error message "File name not found" will be generated.

Setting format: [:SOURce[1]|2]:RADio:ARB:MCARrier:CONFigure:LOAD “FileName”

Parameter description:

< FileName> String type data, the selected file name

Example: RADio:ARB:MCARrier:CONFigure:LOAD “test.mcarcfg”

Load the text.mcarcfg multi carrier configuration file for /home/ceyear/SgData/user/McarrierConfig/.

Reset state: None

Key path: [Baseband] ->[Arbitrary Wave] ->[Basic Configuration] ->[Create Multiple Carriers] ->[General] ->[Load Multiple Carriers]

wave]

Description: for setting only.

Remarks: The instruction is valid when the multi carrier signal generation function option is installed on the instrument (S08)

[:SOURce[1]|2]:RADio:ARB:MCARrier:CONFigure:STORe <FileName >

Function description: This command is used to save the current multi carrier configuration as a file. The parameter of this command is the file name to store for the multi carrier configuration, which must include the extension ". mcarcfg" and the parameter does not include a path name. The default save path is "/home/ceyear/SgData/user/McarrierConfig/"; If a file with the same name exists, delete the original file and recreate it.

Setting format: [:SOURce[1]|2]:RADio:ARB:MCARrier:CONFigure:STORe “FileName”

Parameter description:

<FileName> String data, file name to store for configuration files

Example: RADio:ARB:MCARrier:CONFigure:STORe ”test.mcarcfg” Store the current configuration file as a text.mcarcfg file in the path /home/ceyear/SgData/user/McarrierConfig/.

Reset state: None

Key path: [Baseband] ->[Arbitrary Wave] ->[Basic Configuration] ->[Create Multiple Carriers] ->[General] ->[Store Multiple Carriers]

Description: for setting only.

Remarks: The instruction is valid when the multi carrier signal generation function option is installed on the instrument (S08)

[:SOURce[1]|2]:RADio:ARB:MCARrier:EDIT:CARRier:DELay:STARt <val >

Function description: This command is used to set the delay start value in the carrier auto fill configuration. After the command is executed, the multi carrier list data will not take effect immediately, but can only take effect after using the application configuration. The application configuration commands are detailed in "[:SOURce[1]|2]:RADio:ARB:MCARrier:EDIT:CARRier:EXECute".

Setting format:

[:SOURce[1]|2]:RADio:ARB:MCARrier:EDIT:CARRier:DELay:STARt <Val>

Query format:

[:SOURce[1]|2]:RADio:ARB:MCARrier:EDIT:CARRier:DELay:STARt?

Parameter description:

<Val> Floating point data, delay start value set

Range: [0,1s]

Example: RADio:ARB:MCARrier:EDIT:CARRier:DELay:STARt 1ms Set the delay start value to 1ms.

Reset state: 0s

Key path: [Baseband] ->[Arbitrary Wave] ->[Basic Configuration] ->[Create Multiple Carriers] ->[Multi Carrier List] ->[Carrier Table Auto Fill] ->[Delay Start Value]

Remarks: The instruction is valid when the multi carrier signal generation function option is installed on the instrument (S08)

[:SOURce[1]|2]:RADio:ARB:MCARrier:EDIT:CARRier:DELay:STEP <val >

Function description: This command is used to set the delay step value in the carrier auto fill configuration. After the command is executed, the multi carrier list data will not take effect immediately, but can only take effect after using the application configuration. The application configuration commands are detailed in "[:SOURce[1]|2]:RADio:ARB:MCARrier:EDIT:CARRier:EXECute".

Setting format:

[:SOURce[1]|2]:RADio:ARB:MCARrier:EDIT:CARRier:DELay:STEP <Val>

Query format:

[:SOURce[1]|2]:RADio:ARB:MCARrier:EDIT:CARRier:DELay:STEP?

Parameter description:

<Val> Floating point data, delay step value set

Range:[- 1s, 1s]

Example: RADio:ARB:MCARrier:EDIT:CARRier:DELay:STEP 1ms Set the delay step to 1ms.

Reset state: 0s

Key path: [Baseband] ->[Arbitrary Wave] ->[Basic Configuration] ->[Create Multiple Carriers] ->[Multi Carrier List] ->[Carrier Table Auto Fill] ->[Delay Step Value]

Remarks: The instruction is valid when the multi carrier signal generation function option is installed on the instrument (S08)

[:SOURce[1]|2]:RADio:ARB:MCARrier:EDIT:CARRier:EXECute

Function description: This command takes effect all configurations of carrier auto fill and applies the configuration to the current multi carrier list.

Setting format: [:SOURce[1]|2]:RADio:ARB:MCARrier:EDIT:CARRier:EXECute

Example: RADio:ARB:MCARrier:EDIT:CARRier:EXECute Apply the current auto fill configuration.

Reset state: None

Key path: [Baseband] ->[Arbitrary Wave] ->[Basic Configuration] ->[Create Multiple Carriers] ->[Multi Carrier List] ->[Carrier Table Auto Fill] ->[Application Configuration]

Description: for setting only.

Remarks: The instruction is valid when the multi carrier signal generation function option is installed on the instrument (S08)

[:SOURce[1]|2]:RADio:ARB:MCARrier:EDIT:CARRier:PHASe:STARt <val >

Function description: This command is used to set the phase start value in the carrier auto fill configuration. After the command is executed, the multi carrier list data will not take effect immediately, but can only take effect after using the application configuration. The application configuration commands are detailed in "[:SOURce[1]|2]:RADio:ARB:MCARrier:EDIT:CARRier:EXECute".

Setting format:

[:SOURce[1]|2]:RADio:ARB:MCARrier:EDIT:CARRier:PHASe:STARt <Val>

Query format:

[:SOURce[1]|2]:RADio:ARB:MCARrier:EDIT:CARRier:PHASe:STARt?

Parameter description:

<Val> Floating point data, phase start value set

Range:[0deg, 360deg]

Example: RADio:ARB:MCARrier:EDIT:CARRier:PHASe:STARt 30deg Set the phase start value to 30deg.

Reset state: 0

Key path: [Baseband] ->[Arbitrary Wave] ->[Basic Configuration] ->[Create Multiple Carriers] ->[Multi Carrier List] ->[Carrier Table Auto Fill] ->[Phase Start Value]

Remarks: The instruction is valid when the multi carrier signal generation function option is installed on the instrument (S08)

[:SOURce[1]|2]:RADio:ARB:MCARrier:EDIT:CARRier:PHASe:STEP <val >

Function description: This command is used to set the phase step value in the carrier auto fill configuration. After the command is executed, the multi carrier list data will not take effect immediately, but can only take effect after using the application configuration. The application configuration commands are detailed in "[:SOURce[1]|2]:RADio:ARB:MCARrier:EDIT:CARRier:EXECute".

Setting format:

[:SOURce[1]|2]:RADio:ARB:MCARrier:EDIT:CARRier:PHASe:STEP <Val>

Query format:

[:SOURce[1]|2]:RADio:ARB:MCARrier:EDIT:CARRier:PHASe:STEP?

Parameter description:

<Val> Floating point data, phase step value set

Range:[- 359.99deg, 359.99deg]

Example: RADio:ARB:MCARrier:EDIT:CARRier:PHASe:STEP 30deg Set the phase step value to 30deg.

Reset state: 0

Key path: [Baseband] ->[Arbitrary Wave] ->[Basic Configuration] ->[Create Multiple Carriers] ->[Multi Carrier List] ->[Carrier Table Auto Fill] ->[Phase Step Value]

Remarks: The instruction is valid when the multi carrier signal generation function option is installed on the instrument (S08)

[:SOURce[1]|2]:RADio:ARB:MCARrier:EDIT:CARRier:POWer:STARt <val >

Function description: This command is used to set the gain start value in the carrier auto fill configuration. After the command is executed, the multi carrier list data will not take effect immediately, but can only take effect after using the application configuration. The application configuration commands are detailed in "[:SOURce[1]|2]:RADio:ARB:MCARrier:EDIT:CARRier:EXECute".

Setting format:

[:SOURce[1]|2]:RADio:ARB:MCARrier:EDIT:CARRier:POWer:STARt <Val>

Query format:

[:SOURce[1]|2]:RADio:ARB:MCARrier:EDIT:CARRier:POWer:STARt?

Parameter description:

<Val> Floating point data, gain start value set

Range:[-80dB, 0dB]

Example: RADio:ARB:MCARrier:EDIT:CARRier:POWer:STARt -5dB Set the gain start value to -5dB.

Reset state: 0dB

Key path: [Baseband] ->[Arbitrary Wave] ->[Basic Configuration] ->[Create Multiple Carriers] ->[Multi Carrier List] ->[Carrier Table Auto Fill] ->[Gain Start Value]

Remarks: The instruction is valid when the multi carrier signal generation function option is installed on the instrument (S08)

[:SOURce[1]|2]:RADio:ARB:MCARrier:EDIT:CARRier:POWer:STEP <val >

Function description: This command is used to set the gain step value in the carrier auto fill configuration. After the command is executed, the multi carrier list data will not take effect immediately, but can only take effect after using the application configuration. The application configuration commands are detailed in "[:SOURce[1]|2]:RADio:ARB:MCARrier:EDIT:CARRier:EXECute".

Setting format:

[:SOURce[1]|2]:RADio:ARB:MCARrier:EDIT:CARRier:POWer:STEP <Val>

Query format:

[:SOURce[1]|2]:RADio:ARB:MCARrier:EDIT:CARRier:POWer:STEP?

Parameter description:

<Val> Floating point data, gain step value set

Range:[-80dB, 80dB]

Example: RADio:ARB:MCARrier:EDIT:CARRier:POWer:STEP -10dB Set the gain step value to -10dB.

Reset state: 0dB

Key path: [Baseband] ->[Arbitrary Wave] ->[Basic Configuration] ->[Create Multiple Carriers] ->[Multi Carrier List] ->[Carrier Table Auto Fill] ->[Gain Step Value]

Remarks: The instruction is valid when the multi carrier signal generation function option is installed on the instrument (S08)

[:SOURce[1]|2]:RADio:ARB:MCARrier:EDIT:CARRier:STARt <val >

Function description: This command is used to set the start index in the carrier auto fill configuration.

Setting format: [:SOURce[1]|2]:RADio:ARB:MCARrier:EDIT:CARRier:STARt <Val>

Query format: [:SOURce[1]|2]:RADio:ARB:MCARrier:EDIT:CARRier:STARt?

Parameter description:

<Val> Integer data, start index set
Range: [0, current number of carriers -1]

Example: RADio:ARB:MCARrier:EDIT:CARRier:STARt 0 Set the start index to 0.

Reset state: 0

Key path: [Baseband] ->[Arbitrary Wave] ->[Basic Configuration] ->[Create Multiple Carriers] ->[Multi Carrier List] ->[Carrier Table Auto Fill] ->[Start Index]

Remarks: The instruction is valid when the multi carrier signal generation function option is installed on the instrument (S08)

[:SOURce[1]|2]:RADio:ARB:MCARrier:EDIT:CARRier:STATe <State>

Function description: This command is used to set the carrier state in carrier auto fill. After the command is executed, the multi carrier list data will not take effect immediately, but can only take effect after using the application configuration. The application configuration commands are detailed in "[:SOURce[1]|2]:RADio:ARB:MCARrier:EDIT:CARRier:EXECute".

Setting format: [:SOURce[1]|2]:RADio:ARB:MCARrier:EDIT:CARRier:STATe ON|OFF|1|0

Query format: [:SOURce[1]|2]:RADio:ARB:MCARrier:EDIT:CARRier:STATe?

Parameter description:

<State> Enumeration type, carrier state set
ON|1 :ON
OFF|0 :OFF

Example: RADio:ARB:MCARrier:EDIT:CARRier:STATe 1 Set the carrier state to ON.

Reset state: 0

Key path: [Baseband] ->[Arbitrary Wave] ->[Basic Configuration] ->[Create Multiple Carriers] ->[Multi Carrier List] ->[Carrier Table Auto Fill] ->[Carrier State]

Remarks: The instruction is valid when the multi carrier signal generation function option is installed on the instrument (S08)

[:SOURce[1]|2]:RADio:ARB:MCARrier:EDIT:CARRier:STOP <val >

Function description: This command is used to set the end index in carrier auto fill.

Setting format: [:SOURce[1]|2]:RADio:ARB:MCARrier:EDIT:CARRier:STOP <Val>

Query format: [:SOURce[1]|2]:RADio:ARB:MCARrier:EDIT:CARRier:STOP?

Parameter description:

<Val> Integer data, carrier frequency set
Range: [0, current number of carriers -1]

Example: RADio:ARB:MCARrier:EDIT:CARRier:STOP 1 Set the end index to 1.

Reset state: 0

Key path: [Baseband] ->[Arbitrary Wave] ->[Basic Configuration] ->[Create Multiple Carriers] ->[Multi Carrier List] ->[Carrier Table Auto Fill] ->[End Index]

Remarks: The instruction is valid when the multi carrier signal generation function option is installed on the instrument (S08)

[:SOURce[1]|2]:RADio:ARB:MCARrier:EDIT:CARRier:WAVeform <val >

Function description: This command is used to set the selection waveform file in carrier auto fill. The parameter is a string type parameter, indicating the file name of the waveform segment to be selected. The file name must include the extension ". seg", and the file name does not need to include a path. The default path is "/home/ceyear/SgData/user/Wav/". If the specified file does not exist, an error message "File name not found" will be generated. After the command is executed, the multi carrier list data will not take effect immediately, but can only take effect after using the application configuration. The application configuration commands are detailed in "[:SOURce[1]|2]:RADio:ARB:MCARrier:EDIT:CARRier:EXECute".

Setting format: [:SOURce[1]|2]:RADio:ARB:MCARrier:EDIT:CARRier:WAVeform "FileName"

Query format: [:SOURce[1]|2]:RADio:ARB:MCARrier:EDIT:CARRier:WAVeform?

Parameter description:

<Val> String type, waveform file name selected

Example: RADio:ARB:MCARrier:EDIT:CARRier:WAVeform "test.seg" Set the selected waveform file to test.seg under /home/ceyear/SgData/user/Wav/.

Reset state: ArbMccw.seg

Key path: [Baseband] ->[Arbitrary Wave] ->[Basic Configuration] ->[Create Multiple Carriers] ->[Multi Carrier List] ->[Carrier Table Auto Fill] ->[Select Waveform File]

Remarks: The instruction is valid when the multi carrier signal generation function option is installed on the instrument (S08)

[:SOURce[1]|2]:RADio:ARB:MCARrier:PERiod <val >

Function description: This command is used to set the multi carrier signal period mode to the custom signal period. For commands on setting period mode, please refer to "[:SOURce[1]|2]:RADio:ARB:MCARrier:PERiod:MODE".

Setting format: [:SOURce[1]|2]:RADio:ARB:MCARrier:PERiod <Val>

Query format: [:SOURce[1]|2]:RADio:ARB:MCARrier:PERiod?

Parameter description:

<Val> Floating point data, signal period value set
Range: [0,1000000000s]

Example: RADio:ARB:MCARrier:PERiod 1ms Set the signal period under the custom period mode to 1ms.

Reset state: 0s

Key path: [Baseband] ->[Arbitrary Wave] ->[Basic Configuration] ->[Create Multiple Carriers] ->[General] ->[Signal Period Mode Custom] ->[Signal Period]

Remarks: The instruction is valid when the multi carrier signal generation function option is installed on the instrument (S08)

[:SOURce[1]|2]:RADio:ARB:MCARrier:PERiod:MODE <Mode >

Function description: This command is used for multi carrier signal period modes.

Setting format: [:SOURce[1]|2]:RADio:ARB:MCARrier:PERiod:MODE LONG|SHORt|USER|LCM

Query format: [:SOURce[1]|2]:RADio:ARB:MCARrier:PERiod:MODE?

Parameter description:

<Mode> Enumeration type data
LONG: Longest carrier period
SHORt: Shortest carrier period
USER: User defined
LCM: Least common multiple

Example: RADio:ARB:MCARrier:PERiod:MODE SHORt Set the signal period mode to the shortest carrier period.

Reset state: LONG

Key path: [Baseband] ->[Arbitrary Wave] ->[Basic Configuration] ->[Create Multiple Carriers] ->[General] ->[Signal Period Mode]

Remarks: The instruction is valid when the multi carrier signal generation function option is installed on the instrument (S08)

[:SOURce[1]|2]:RADio:ARB:MCARrier:POWer:REFerence <Mode >

Function description: This command is used to set the power reference mode.

Setting format: [:SOURce[1]|2]:RADio:ARB:MCARrier:POWer:REFerence RMS|PEAK

Query format: [:SOURce[1]|2]:RADio:ARB:MCARrier:POWer:REFerence?

Parameter description:

<Mode> Enumeration type data
RMS: Root mean square
PEAK :peak value

Example: RADio:ARB:MCARrier:POWer:REFerence PEAK Set power reference to peak value

Reset state: RMS

Key path: [Baseband]—>[Arbitrary Wave]—>[Basic Configuration]—>[Create Multiple Carriers]—>[General]—>[Power Reference]

Remarks: The instruction is valid when the multi carrier signal generation function option is installed on the instrument (S08)

[:SOURce[1]|2]:RADio:ARB:MCARrier:RESet

Function description: This command resets the multi carrier configuration to the default values.

Setting format: [:SOURce[1]|2]:RADio:ARB:MCARrier:RESet

Example: RADio:ARB:MCARrier:RESet Reset multi carrier to default values.

Reset state: None

Key path: [Baseband]—>[Arbitrary Wave]—>[Basic Configuration]—>[Create Multiple Carriers]—>[General]—>[Reset to Default Values]

Description: for setting only.

Remarks: The instruction is valid when the multi carrier signal generation function option is installed on the instrument (S08)

[:SOURce[1]|2]:RADio:ARB:MCARrier:WAVeform:CLOad <FileName>

Function description: This command is used to create an arbitrary wave file based on the current multi-carrier configuration and load the output. The generated arbitrary wave file name is specified by the parameter, and only the file name with the extension ". seg" is included in the parameter. There is no need to set the path, and the default path is /home/ceyear/SgData/user/Wav/. If a file with the same name exists in the path, the original file will be deleted and the file will be recreated.

Setting format: [:SOURce[1]|2]:RADio:ARB:MCARrier:WAVeform:CLOad "FileName"

Parameter description:

<FileName> String type data, file name set

Example: RADio:ARB:MCARrier:WAVeform:CLOad "test.seg" Create an arbitrary wave file named test.seg based on the current multi carrier configuration and load it for playback.

Reset state: None

Key path: [Baseband]—>[Arbitrary Wave]—>[Basic Configuration]—>[Create Multiple Carriers]—>[General]—>[Create and Load Output File]

Description: for setting only.

Remarks: The instruction is valid when the multi carrier signal generation function option is installed on the instrument (S08)

[:SOURce[1]|2]:RADio:ARB:MCARrier:WAVeform:CREate <FileName >

Function description: This command is used to create an arbitrary wave file based on the current multi carrier configuration. The generated arbitrary wave file name is specified by the parameter, and only the file name with the extension ". seg" is included in the parameter. There is no need to set the path, and the default path is /home/ceyear/SgData/user/Wav/. If a file with the same name exists in the path, the original file will be deleted and the file will be recreated.

Setting format: [:SOURce[1]|2]:RADio:ARB:MCARrier:WAVeform:CREate "FileName"

Parameter description:

<FileName> String type data, file name generated

Example: RADio:ARB:MCARrier:WAVeform:CREate "test.seg" Generate an arbitrary wave file named test.seg based on the current multi carrier settings.

Reset state: None

Key path: [Baseband]—>[Arbitrary Wave]—>[Basic Configuration]—>[Create Multiple Carriers]—>[General]—>[Create Output File]

Description: for setting only.

Remarks: The instruction is valid when the multi carrier signal generation function option is installed on the instrument (S08)

[:SOURce[1]|2]:RADio:ARB:SCLock:RATE <ClockRate>

Function description: This command is used to set the arbitrary wave signal clock frequency.

Setting format: [:SOURce[1]|2]:RADio:ARB:SCLock:RATE <val><freq unit>

Query format: [:SOURce[1]|2]:RADio:ARB:SCLock:RATE?

Parameter description:

<ClockRate> Set the arbitrary wave clock frequency.
Range: [-325GHz～ +325GHz].

Example: :RADio:ARB:SCLock:RATE 50MHz the arbitrary wave clock frequency is 50MHz.
SOURce2:RADio:ARB:SCLock:RATE 50Mhz the arbitrary wave clock frequency is 50MHz.

Reset state: 100MHz

Key path: [Baseband]—>[Arbitrary Wave]—>[Basic Configuration]—>[Clock Frequency]

[:SOURce[1]|2]:RADio:ARB:SEQuence:CLOad <FileName>

Function description: Create and generate a sequence file based on the current sequence configuration file, and load it into arbitrary wave; If the current sequence configuration list is empty, a prompt of "execution error" will be generated. The parameter is a string type parameter used to specify the generated sequence file name, which can be a file name with an absolute path or just a file name, and the root directory of the absolute path is/SgData/; If it does not include an absolute path, then create the file in the default path "/home/ceyear/SgData/user/War/" folder.

Setting format: [:SOURce[1]|2]:RADio:ARB:SEQuence:CLOad “filename”

Parameter description: String type, file name to be selected

Example: :RADio:ARB:SEQuence:CLOad “/SgData/user/test.seq” Create a sequence file named test.seq in the /home/ceyear/SgData/user file and load it into arbitrary wave.

Reset state: None

Key path: [Baseband]—>[Arbitrary Wave]—>[Basic Configuration]—>[Create Sequence]—>[Create and Load Sequence File]

Remarks: The command is valid when generating options for instrument installation sequence files (S07)

[:SOURce[1]|2]:RADio:ARB:SEQuence:CLOCk <Mode>

Function description: This command sets the clock type used by the signal generator when creating sequence files. Users can choose from three modes: CURRent, HIGH, and CUSTom.

Setting format: [:SOURce[1]|2]:RADio:ARB:SEQuence:CLOCk CURRent|HIGH|CUSTom

Query format: [:SOURce[1]|2]:RADio:ARB:SEQuence:CLOCk?

Parameter description:

<Mode> discrete data. The values of sampling clock type in arbitrary mode are as follows:

CURrent: The sequence file is generated at the sampling rate of each waveform segment;

HIGH: The sequence file is generated at the highest sampling rate among all waveform segments;

CUSTom: The sequence file is generated at the set clock frequency.

Example: :RADio:ARB:SEQuence:CLOCk HIGH

Set the clock type to be the highest when generating sequence files.

Reset state: CURRent

Key path: [Baseband]—>[Arbitrary Wave]—>[Basic Configuration]—>[Create Sequence]—>[Clock/Marker]—>[Clock Type]

Clock type]

Remarks: The command is valid when generating options for instrument installation sequence files (S07)

[:SOURce[1]|2]:RADio:ARB:SEQuence:CLOCk:RATE <ClockRate>

Function description: When creating a sequence, set the clock type to the custom clock frequency.

Setting format: [:SOURce[1]|2]:RADio:ARB:SEQuence:CLOCk:RATE <val>

Query format: [:SOURce[1]|2]:RADio:ARB:SEQuence:CLOCk:RATE?

Parameter description:

<val> Clock frequency.

Range: [-325GHz～ +325GHz].

Example: [:SOURce]:RADio:ARB:SEQuence:CLOCk:RATE 100MHz

Set the clock frequency for creating the sequence to 100MHz.

Reset state: 100MHz

Key path: [Baseband]—>[Arbitrary Wave]—>[Basic Configuration]—>[Create Sequence]—>[Clock/Marker]—>[Clock Type]

[Clock Type Custom]—>[Clock Frequency]

Remarks: The command is valid when generating options for instrument installation sequence files (S07)

[:SOURce[1]|2]:RADio:ARB:SEQuence:CONFigure:APPend <WaveForm>

Function description: Append new waveform segment file information to the sequence configuration list for creating sequence files. for setting only. The parameter is a string type parameter used to specify the file name of the appended waveform segment, which can be a file name with an absolute path or just a file name, and the root directory of the absolute path is/SgData/; If it does not include an absolute path, the default path is "/home/ceyear/SgData/user/War/".

Setting format: [:SOURce[1]|2]:RADio:ARB:SEQuence:CONFigure:APPend “WaveForm”

Parameter description:

< WaveForm > String type, the new waveform segment file name to be included in the sequence

Example: :RADio:ARB:SEQuence:CONFigure:APPend “/SgDara/user/test.seg”

Append the waveform segment file /home/ceyear /SgDara/user/test.seg to the sequence configuration list

Reset state: None

Key path: [Baseband]—>[Arbitrary Wave]—>[Basic Configuration]—>[Create

Sequence]—>[General]—>[Add Waveform Segments]

Segment]

Remarks: The command is valid when generating options for instrument installation sequence files (S07)

[:SOURce[1]|2]:RADio:ARB:SEQuence:CONFigure:DELete <val>

Function description: Delete the waveform segment file information of the specified index in the sequence configuration list. for setting only.

Setting format: [:SOURce[1]|2]:RADio:ARB:SEQuence:CONFigure:DELete <val>

Parameter description:

< val > Shaping data, specifying the index of the deleted waveform segment in the sequence configuration, starting from 0,

and the maximum value range depends on the number of currently loaded waveform segments,

Range: [0, number of waveform segments -1].

Example: :RADio:ARB:SEQuence:CONFigure:DELete 0 Delete the information of the 0th waveform segment

Reset state: None

Key path: None

Remarks: The command is valid when generating options for instrument installation sequence files (S07)

[:SOURce[1]|2]:RADio:ARB:SEQuence:CONFigure:RESet

Function description: Delete all waveform segment information from the sequence configuration list and only set it.

Setting format: [:SOURce[1]|2]:RADio:ARB:SEQuence:CONFigure:RESet

Parameter description: None

Example: :RADio:ARB:SEQuence:CONFigure:RESet

Reset state: None

Key path: [Baseband]—>[Arbitrary Wave]—>[Basic Configuration]—>[Create Sequence]—>[General]—>[Delete All Waveform Segments]

Waveform segment]

Remarks: The command is valid when generating options for instrument installation sequence files (S07)

[:SOURce[1]|2]:RADio:ARB:SEQuence:CONFigure:SELect <FileName>

Function description: Select a sequence of configuration files, create a new configuration file with the currently specified file name when the file does not exist, and select it; The new configuration file created is empty. The parameter is a string type parameter used to specify the selected configuration file, which can be a file name with an absolute path or just a file name, and the root directory of the absolute path is /SgData/; If it does not include an absolute path, then the default path is "/home/ceyear/SgData/user/SquenceConfig/“

Setting format: [:SOURce[1]|2]:RADio:ARB:SEQuence:CONFigure:SELect “filename”

Parameter description: String type, sequence configuration file name to be selected

Example: :RADio:ARB:SEQuence:CONFigure:SELect “text.seqcfg” Select /home/ceyear/SgData/user/SquenceConfig/text.seqcfg sequence configuration file.

Reset state: None

Key path: [Baseband]—>[Arbitrary Wave]—>[Basic Configuration]—>[Create Sequence]—>[General]—>[Select Sequence]

Remarks: The command is valid when generating options for instrument installation sequence files (S07)

[:SOURce[1]|2]:RADio:ARB:SEQuence:CONFigure:STORe <FileName>

Function description: Store the configuration of the current sequence to the specified file. The parameter is a string type parameter used to specify the name of the stored configuration file, which can be a file name with an absolute path or just a file name, and the root directory of the absolute path is /SgData/; If it does not include an absolute path, then the default path is "/home/ceyear/SgData/user/SquenceConfig/".

Setting format: [:SOURce[1]|2]:RADio:ARB:SEQuence:CONFigure:STORe “filename”

Parameter description: String type, file name to be stored

Example: :RADio:ARB:SEQuence:CONFigure:STORe “text.seqcfg”

Reset state: None

Key path: [Baseband]—>[Arbitrary Wave]—>[Basic Configuration]->[Create Sequence]—>[General]—>[Store Sequence]

Remarks: The command is valid when generating options for instrument installation sequence files (S07)

[:SOURce[1]|2]:RADio:ARB:SEQuence:CREate <FileName>

Function description: Create and generate a sequence file based on the current sequence configuration; If the current sequence configuration list is empty, a prompt of "execution error" will be generated. The parameter is a string type parameter used to specify the sequence file name to be created, which can be a file name with an absolute path or just a file name, and the root directory of the absolute path is /SgData/; If it does not include an absolute path, the default path is "/home/ceyear/SgData/user/Wav/".

Setting format: [:SOURce[1]|2]:RADio:ARB:SEQuence:CREate “filename”

Parameter description: String type, file name to be selected

Example: :RADio:ARB:SEQuence:CREate “/SgData/user/text.seq” Create a sequence file named text.seq in the path /home/ceyear/SgData/user/.

Reset state: None

Key path: [Baseband]—>[Arbitrary Wave]—>[Basic Configuration]—>[Create Sequence]—>[General]—>[Create Sequence File]

Remarks: The command is valid when generating options for instrument installation sequence files (S07)

[:SOURce[1]|2]:RADio:ARB:SEQuence:INDex <num>

Function description: This command specifies the index number of the waveform segment in arbitrary wave sequence, with a start value of 0 for the index, and is valid when arbitrary wave file is selected with the extension ". seq".

Setting format: [:SOURce[1]|2]:RADio:ARB:SEQuence:INDex <num>

Query format: [:SOURce[1]|2]:RADio:ARB:SEQuence:INDex?

Parameter description:

< num > Shaping type, with a value range of [0,64], the maximum value depends on the current number of loaded waveform segments

Example: :RADio:ARB:SEQuence:INDex 1

Set the sequence number for sequence playback to 1

Reset state: 0

Key path: [Baseband]—>[Arbitrary Wave]—>[Playback Control]—>[Waveform Segment Number]

Remarks: The command is valid when generating options for instrument installation sequence files (S07)

[:SOURce[1]|2]:RADio:ARB:SEQuence:MARKer:ESEGment<Mode>

Function description: This command sets the start marker type for each waveform segment in the sequence.

Setting format: [:SOURce[1]|2]:RADio:ARB:SEQuence:MARKer:ESEGment OFF|MRK1|MRK2|MRK3|MRK4

Query format: [:SOURce[1]|2]:RADio:ARB:SEQuence:MARKer:ESEGment?

Parameter description:

< Val > Discrete type.

OFF: Ignored

MRK1: Marker 1

MRK2: Marker 2
MRK3: Marker 3
MRK4: Marker 4

Example: :RADio:ARB:SEQuence:MARKer:ESEGment MRK1 设置序列中波形段起始标记为标记 1 Set the start marker of the waveform segment in the sequence to Marker 1.

Reset state: OFF

Key path: [Baseband]—>[Arbitrary Wave]—>[Create Sequence]—>[Clock/Marker]—>[Waveform Segment Start Marker]

Remarks: The command is valid when generating options for instrument installation sequence files (S07)

[:SOURce[1]|2]:RADio:ARB:SEQuence:MARKer:MODE <Mode>

Function description: This command sets how to use the marker signal for each waveform segment in the sequence

Setting format: [:SOURce[1]|2]:RADio:ARB:SEQuence:MARKer:MODE IGNore|TAKE

Query format: [:SOURce[1]|2]:RADio:ARB:SEQuence:MARKer:MODE?

Parameter description:

<Mode> discrete data. Values are taken as follows:

IGNore: Ignored, the sequence turns off the marker signal of each waveform segment during playback.

TAKE: Effective, the sequence is played while using the marker signal of each waveform segment itself.

Example: :RADio:ARB:SEQuence:MARKer:MODE TAKE During sequence playback, the marker signals of each waveform segment in the sequence are played simultaneously.

Reset state: IGNore

Key path: [Baseband]—>[Arbitrary Wave]—>[Create Sequence]—>[Clock/Marker]—>[Waveform Segment Marker]

Remarks: The command is valid when generating options for instrument installation sequence files (S07)

[:SOURce[1]|2]:RADio:ARB:SEQuence:MARKer:STARt<Mode>

Function description: This command sets the mode of sequence start marker.

Setting format: [:SOURce[1]|2]:RADio:ARB:SEQuence:MARKer:STARt OFF|MRK1|MAK2|MRK3|MRK4

Query format: [:SOURce[1]|2]:RADio:ARB:SEQuence:MARKer:STARt?

Parameter description:

<Mode> discrete data. Values are taken as follows:

OFF: Ignored

MRK1: Marker 1
MRK2: Marker 2
MRK3: Marker 3
MRK4: Marker 4

Example: :RADio:ARB:SEQuence:MARKer:STARt MRK3 The sequence start marker is marker 3.

Reset state: OFF

Key path: [Baseband]—>[Arbitrary Wave]—>[Create Sequence]—>[Clock/Marker]—>[Sequence Start Marker]

Remarks: The command is valid when generating options for instrument installation sequence files (S07)

[:SOURce[1]|2]:RADio:ARB:SEQuence:NEXT[:SEGMent]

Function description: When any selected wave is a sequence and the playback mode is sequential, this command plays the next waveform segment in the sequence. When arbitrary wave file is selected with the extension ". seq", this command is valid.

Setting format: [:SOURce[1]|2]:RADio:ARB:SEQuence:NEXT[:SEGMent]

Parameter description: None

Example: :RADio:ARB:SEQuence:NEXT
Play the next waveform segment in the sequence.

Reset state: None

Key path: [Baseband]—>[Arbitrary Wave]—>[Play Control]—>[Playback Mode Sequential Playback]—>[Play Next Waveform Segment]

.

Remarks: The command is valid when generating options for instrument installation sequence files (S07)

[:SOURce[1]|2]:RADio:ARB:SEQuence[:TYPE] <MODE>

Function description: When arbitrary wave loaded is a sequence, this command sets the playback mode of arbitrary wave, including SAME, NEXT, and SEQuencer. When arbitrary wave file is selected with the extension ". seq", this command is valid.

Setting format: [:SOURce[1]|2]:RADio:ARB:SEQuence[:TYPE] SAME|NEXT|SEQuencer.

Query format: [:SOURce[1]|2]:RADio:ARB:SEQuence[:TYPE]?

Parameter description:

< Val >	Discrete type.
	SAME: Single playback
	NEXT: Sequential playback
	SEQuencer: Sequence playback

Example: :RADio:ARB:SEQuence SEQuencer Set the sequence playback mode to sequence playback

Reset state: SAME

Key path: [Baseband]—>[Arbitrary Wave]—>[Playback Control]—>[Playback Mode]

Remarks: The command is valid when generating options for instrument installation sequence files (S07)

[:SOURce[1]|2]:RADio:ARB[:STATe] <State>

Function description: This command sets the status switch for arbitrary wave. When it is ON, the selected arbitrary wave file is played. When it is OFF, no arbitrary wave data is played. When the arbitrary wave switch is set to ON, it is necessary to first select a file for "Load arbitrary wave", otherwise the arbitrary wave switch cannot be turned on.

Setting format: [:SOURce[1]|2]:RADio:ARB:STATe ON|OFF|1|0

Query format: [:SOURce[1]|2]:RADio:ARB:STATe?

Parameter description:

<State> Boolean data, with the values listed below:

ON | 1: Arbitrary wave ON,

OFF | 0: arbitrary wave OFF.

Example: :SOURce:RADio:ARB:STATe 1 Channel A arbitrary wave ON

:SOURce2:RADio:ARB:STATe 1 Channel B arbitrary wave ON

Reset state: 0

Key path: [Baseband]—>[Arbitrary Wave]—>[Basic Configuration]—>[Arbitrary Wave ON OFF]

[:SOURce[1]|2]:RADio:ARB:TRIGger:EXECute

Function description: When arbitrary wave trigger source is set as the trigger key, the command executes a trigger. for setting only.

Setting format: [:SOURce[1]|2]:RADio:ARB:TRIGger:EXECute

Example: :RADio:ARB:TRIGger:EXECute Execute a trigger

Reset state: None

Key path: [Baseband]—>[Arbitrary Wave]—>[Trigger Setting]—>[Trigger Source Trigger Key]—>[Trigger]

[:SOURce[1]|2]:RADio:ARB:TRIGger[:SOURce] <Mode>

Function description: This command is used to set the arbitrary wave trigger source.

Setting format: [:SOURce[1]|2]:RADio:ARB:TRIGger:SOURce KEY|EXTernal

Query format: [:SOURce[1]|2]:RADio:ARB:TRIGger:SOURce?

Parameter description:

<Mode> discrete data. Arbitrary wave trigger source,
Values are taken as follows:
KEY: trigger key,
EXTernal : external

Example: :RADio:ARB:TRIGger:SOURce KEY Set the trigger source as the trigger key.

Reset state: KEY

Key path: [Baseband]—>[Arbitrary Wave]—>[Trigger Setting]—>[Trigger Source]

[:SOURce[1]|2]:RADio:ARB:TRIGger[:SOURce]:EXTernal:DELay <Val>

Function description: This command is used to set the time of external trigger signal delay for the instrument to respond to the trigger signal when external is selected as trigger source of the signal generator arbitrary wave. The effective prerequisite for this setting is to set the external delay to ON state. For external delay switch state commands, refer to“[:SOURce[1]|2]:RADio:ARB:TRIGger[:SOURce]:EXTernal:DELay:STATe”.

Setting format: [:SOURce[1]|2]:RADio:ARB:TRIGger[:SOURce]:EXTernal:DELay <val>

Query format: [:SOURce[1]|2]:RADio:ARB:TRIGger[:SOURce]:EXTernal:DELay?

Parameter description:

<Val> delay time when external is selected as the baseband trigger source.
Range: [0, 10s].

Example: :RADio:ARB:TRIGger:SOURce:EXTernal:DELay 5
The external trigger delay is 5 bits.

Reset state: 0s

Key path: [Baseband]—>[Arbitrary Wave]—>[Trigger Setting]—>[Trigger Source External]—>
[Delay Type Time]—>[Delay Time]

[:SOURce[1]|2]:RADio:ARB:TRIGger[:SOURce]:EXTernal:DELay:SAMPle <Val>

Function description: This command is used to set the bit of external trigger signal delay for the instrument to respond to the trigger signal when external is selected as trigger source of the signal generator arbitrary wave. The effective prerequisite for this setting is to set the external delay to ON state. For external delay switch state commands, refer to“[:SOURce[1]|2]:RADio:ARB:TRIGger[:SOURce]:EXTernal:DELay:STATe”.

Setting format:

[:SOURce[1]|2]:RADio:ARB:TRIGger[:SOURce]:EXTernal:DELay:SAMPle <val>

Query format: [:SOURce[1]|2]:RADio:ARB:TRIGger[:SOURce]:EXTernal:DELay:SAMPle?

Parameter description:

<Val> delay time bit when external is selected as the baseband trigger source. Range: [0,25000000000].

Example: :RADio:ARB:TRIGger:SOURce:EXTernal:DELay:SAMPle 1024
The external trigger delay sample bit is 1024.

Reset state: 0

Key path: [Baseband]—>[Arbitrary Wave]—>[Trigger Setting]—>[Trigger Source External]—>
[Delay Type Bit]—>[Delay]

[:SOURce[1]|2]:RADio:ARB:TRIGger[:SOURce]:EXTernal:DELay:STATe <State>

Function description: This command is used to set the external trigger delay state when external is selected as trigger source of the signal generator arbitrary wave. When it is ON, the external delay set takes effect. For external delay time commands, refer to“[:SOURce[1]|2]:RADio:ARB:TRIGger[:SOURce]:EXTernal:DELay ”. “[:SOURce[1]|2]:RADio:ARB:TRIGger[:SOURce]:EXTernal:SAMPle”.

Setting format:
[:SOURce[1]|2]:RADio:ARB:TRIGger:SOURce:EXTernal:DELay:STATe ON|OFF|1|0

Query format:
[:SOURce[1]|2]:RADio:ARB:TRIGger:SOURce:EXTernal:DELay:STATe?

Parameter description:

<State> Boolean data. The values of delay state when external is selected as arbitrary wave trigger source are as follows:

ON | 1: Delay ON
OFF | 0: delay OFF.

Example: :RADio:ARB:TRIGger:SOURce:EXTernal:DELay:STATe 1 External trigger delay ON.

Reset state: 0

Key path: [Baseband]—>[Arbitrary Wave]—>[Trigger Setting]—>[Trigger Source External]—>[Delay]

[:SOURce[1]|2]:RADio:ARB:TRIGger[:SOURce]:EXTernal:SLOPe <Mode>

Function description: This command is used to set the trigger input signal slope of the instrument rear panel to high effective trigger baseband output or low effective trigger baseband output when external is selected as trigger source of the signal generator arbitrary wave.

Setting format: [:SOURce[1]|2]:RADio:ARB:TRIGger:SOURce:EXTernal:SLOPe POSitive|NEGative

Query format: [:SOURce[1]|2]:RADio:ARB:TRIGger:SOURce:EXTernal:SLOPe?

Parameter description:

<Mode> discrete data. The values of external trigger slope are as follows:

POSitive	: positive
NEGative	: negative

Example: :RADio:ARB:TRIGger:SOURce:EXTernal:SLOPe NEGative External trigger slope is low effective.

Reset state: POSitive

Key path: [Baseband]—>[Arbitrary Wave]—>[Trigger Setting]—>[Trigger Source External]—>[External Trigger Slope]

Negative Positive]

[:SOURce[1]|2]:RADio:ARB:TRIGger:TYPE <Mode>

Function description: This command is used to set the trigger mode when controlling the arbitrary wave playing. Users may select CONTinuous, SINGle, or GATE. The CONTinuous, SINGle, and GATE modes all have multiple states. Refer to"[:SOURce[1]|2]:RADio:ARB:TRIGger:TYPE:CONTinuous[:TYPE]", "[:SOURce[1]|2]:RADio:ARB:TRIGger:TYPE:SINGle", "[:SOURce[1]|2]:RADio:ARB:TRIGger:TYPE:GATE[:ACTive] ".

Setting format: [:SOURce[1]|2]:RADio:ARB:TRIGger:TYPE CONTinuous|SINGle|GATE

Query format: [:SOURce[1]|2]:RADio:ARB:TRIGger:TYPE?

Parameter description:

<Mode> discrete data. In arbitrary mode, the values of trigger mode for controlling waveform play are as follows:

CONTinuous	: Select continuous as trigger mode, and the instrument will repeat the waveform sequence after receiving an effective trigger event;
SINGle	: select single as trigger mode, and the instrument will play the waveform sequence once after receiving an effective trigger event;
GATE	: Select gate as trigger mode, and the waveform sequence will be played continuously within the valid period of the gate signal.

Example: :RADio: ARB:TRIGger: TYPE SING the trigger mode is single.

Reset state: CONTinuous

Key path: [Baseband]—>[Arbitrary Wave]—>[Trigger Setting]—>[Trigger Mode]

[:SOURce[1]|2]:RADio:ARB:TRIGger:TYPE:CONTinuous[:TYPE] <Mode>

Function description: This command is used to set the type for sequence file to respond to trigger signal in arbitrary wave continuous trigger mode. Users may select FREE, TRIGger or RESet. For trigger mode commands, refer to“[:SOURce[1]|2]:RADio:ARB:TRIGger:TYPE”.

Setting format: [:SOURce[1]|2]:RADio:ARB:TRIGger:TYPE:CONTinuous FREE|TRIGger|RESet

Query format: [:SOURce[1]|2]:RADio:ARB:TRIGger:TYPE:CONTinuous?

Parameter description:

<Mode> discrete data. The values of the type for sequence file to respond to trigger signal in arbitrary continuous mode

Values are taken as follows:

FREE: Select automatic mode, in which the sequence waveform data is automatically triggered to start playing after downloading,
and all triggering events are ignored during the playback process

TRIGger: Select the trigger mode, in which the system will not play the current arbitrary wave
until it receives a valid trigger event. After receiving a valid trigger event,
the system starts continuously playing the current arbitrary wave, and does not receive new content during playback.
trigger signal.

RESet: Select real-time mode, and the system will not play the arbitrary wave until it receives a valid trigger event;
After receiving a valid trigger event, the system starts playing the arbitrary wave signal continuously.
If a new trigger signal is received during the playback process,
the currently playing arbitrary wave signal will be suspended and the arbitrary wave signal will be replayed from scratch.

Example: :RADio:ARB:TRIGger:TYPE:CONTinuous TRIG

In the continuous trigger mode of the arbitrary wave, set the continuous trigger type to trigger.

Reset state: FREE

Key path: [Baseband]—>[Arbitrary Wave]—>[Trigger Setting]—>[Trigger Mode

Continuous]—>[Continuous Trigger Type]

Type]

[:SOURce[1]|2]:RADio:ARB:TRIGger:TYPE:GATE[:ACTive] <Mode>

Function description: This command is used to set the trigger type in the arbitrary wave gate control trigger mode. Users may select low effective or high effective mode. For trigger mode commands, refer to"[:SOURce[1]|2]:RADio:ARB:TRIGger:TYPE".

Setting format: [:SOURce[1]|2]:RADio:ARB:TRIGger:TYPE:GATE[:ACTive] LOW|HIGH

Query format: [:SOURce[1]|2]:RADio:ARB:TRIGger:TYPE:GATE[:ACTive]?

Parameter description:

<Mode> discrete data. The values of the type for sequence file to respond to trigger signal in arbitrary wave gate mode are as follows:

Values are taken as follows:

LOW	: low effective
HIGH	: high effective

Example: :RADio:ARB:TRIGger:TYPE:GATE:ACTive LOW

Set the type of trigger signal to be low effective in the arbitrary wave gate control trigger mode.

Reset state: LOW

Key path: [Baseband]—>[Arbitrary Wave]—>[Trigger Setting]—>[Trigger Mode Gating]—>[Gate Trigger Type]

Type]

[:SOURce[1]|2]:RADio:ARB:TRIGger:TYPE:SINGle <Mode>

Function description: This command is used to set the type for sequence file to respond to trigger signal in arbitrary single mode. Users may select IGNore, CACHe or RESet. For trigger mode commands, refer to"[:SOURce[1]|2]:RADio:ARB:TRIGger:TYPE".

Setting format: [:SOURce[1]|2]:RADio:ARB:TRIGger:TYPE:SINGle IGNore|CACHe|RESet

Query format: [:SOURce[1]|2]:RADio:ARB:TRIGger:TYPE:SINGle?

Parameter description:

<Mode> discrete data. The values of the type for sequence file to respond to trigger signal in arbitrary single mode are as follows:

Values are taken as follows:

IGNore: Select to ignore repeated triggering. Under this trigger type, after receiving the trigger signal, the instrument will play arbitrary wave file once and no longer receive the trigger signal during the playback process.

Send signal.

CACHe: Select the buffered repeat mode. Under this trigger type, after receiving the trigger signal,
the instrument will play an arbitrary wave file once. During the playback process, if a new trigger signal is received,
the new trigger signal will be cached. After completing this playback,
the instrument will respond to the cached trigger signal to play the next arbitrary wave file.

RESet: Select the real-time trigger mode, in which the instrument will play arbitrary wave file once after receiving the trigger signal.
During the playback process, if a new trigger signal is received,
the playback of arbitrary wave will be immediately stopped and then restarted.
.

Example: :RADio:ARB:TRIGger:TYPE:SINGle RESet

In any single trigger mode, set the trigger type of arbitrary wave response to real-time repeated triggering.

Reset state: IGNore

Key path: [Baseband]—>[Arbitrary Wave]—>[Trigger Setting]—>[Trigger Mode Single]—>[Single Trigger]
Type]

[:SOURce[1]|2]:RADio:ARB:TWAVeform:SINE:CLOad

Function description: Create a sine wave test waveform and load it into arbitrary wave. At this time, the created sine wave file name and storage path are both default values and cannot be set. The default generated file name is SINTESTWAVE.seg, which is stored in the /home/ceyear/SgData/user/Wav/ folder.

Setting format: [:SOURce[1]|2]:RADio:ARB:TWAVeform:SINE:CLOad

Example: :RADio:ARB:TWAVeform:SINE:CLOad

Create a sine wave test waveform and load it into arbitrary wave.

Reset state: None

Key path: [Baseband]**—**>[Arbitrary Wave]**—**>[Basic Configuration]**—**>[Test Waveform Type Sine Wave]**—**>
[Create Test Waveform]—>[Create and Load Output File]

[:SOURce[1]|2]:RADio:ARB:TWAVeform:SINE:CREate <filename>

Function description: Create a sine wave test waveform. The parameter is the file name of the sine wave waveform created as specified, which is of string type, and only contains the file name and does not include the path. The default storage path for the file is /home/ceyear/SgData/user/Wav/.

Setting format: [:SOURce[1]|2]:RADio:ARB:TWAVeform:SINE:CREate “fileName”

Parameter description: Parameter of string type

Example: :RADio:ARB:TWAVeform:SINE:CREate “text.seg”

Create a sine wave test waveform and name it text

Reset state: None

Key path: [Baseband]**—**>[Arbitrary Wave]**—**>[Basic Configuration]**—**>[Test Waveform Type Sine Wave]**—**>

[Create Test Waveform]—>[Create Output File]

[:SOURce[1]|2]:RADio:ARB:TWAVeform:SINE:FREQuency <val>

Function description: Set the frequency value for creating sine wave test waveforms

Setting format: [:SOURce[1]|2]:RADio:ARB:TWAVeform:SINE:FREQuency val

Query format: [:SOURce[1]|2]:RADio:ARB:TWAVeform:SINE:FREQuency?

Parameter description: Floating point type parameter

Range :[100Hz,625MHz]

Example: :RADio:ARB:TWAVeform:SINE:FREQuency 1MHz

Set the frequency value of the sine test waveform created to 1MHz

Reset state: 1kHz

Key path: [Baseband]**—**>[Arbitrary Wave]**—**>[Basic Configuration]**—**>[Test Waveform Type Sine Wave]**—**>

[Create Test Waveform]—>[Frequency]

[:SOURce[1]|2]:RADio:ARB:TWAVeform:SINE:PHASe <val>

Function description: Set the Q-path phase offset for creating sine wave test waveforms

Setting format: [:SOURce[1]|2]:RADio:ARB:TWAVeform:SINE:PHASe val

Setting format: [:SOURce[1]|2]:RADio:ARB:TWAVeform:SINE:PHASe?

Parameter description: Floating point type parameter

Range:[- 180deg, 180deg]

Example: :RADio:ARB:TWAVeform:SINE:PHASe 78deg

Set the Q-path phase offset of the sine test waveform created to 78deg

Reset state: 90

Key path: [Baseband]**—**>[Arbitrary Wave]**—**>[Basic Configuration]**—**>[Test

Waveform Type Sine Wave]**—**>
[Create Test Waveform]—>[Q-path Phase Offset]

[:SOURce[1]|2]:RADio:ARB:TWAVeform:SINE:SAMPles <val>

Function description: Set the number of sampling points per week for creating sine wave test waveforms

Setting format: [:SOURce[1]|2]:RADio:ARB:TWAVeform:SINE:SAMPles val

Setting format: [:SOURce[1]|2]:RADio:ARB:TWAVeform:SINE:SAMPles?

Parameter description: Shaping type parameter
Range [4, max (40000,2.5e10/sine test waveform frequency)]

Example: :RADio:ARB:TWAVeform:SINE:SAMPles 2000
Set the number of sampling points per week for the created sine test waveform to 2000

Reset state: 4

Key path: [Baseband]**—**>[Arbitrary Wave]**—**>[Basic Configuration]**—**>[Test Waveform Type Sine Wave]**—**>
[Create Test Waveform]—>[Number of Sampling Points per Week]

[:SOURce[1]|2]:RADio:ARB:TWAVeform:SQUare:AMPLitude<val>

Function description: Set the amplitude for creating square wave test waveforms.

Setting format: [:SOURce[1]|2]:RADio:ARB:TWAVeform:SQUare:AMPLitude val

Setting format: [:SOURce[1]|2]:RADio:ARB:TWAVeform:SQUare:AMPLitude?

Parameter description: Floating point type parameter, expressed in FS as a percentage of full scale
Range: [0,1- | Square wave test waveform DC offset |]

Example: :RADio:ARB:TWAVeform:SQUare:AMPLitude 0.5
Set the amplitude of 0.5 for creating square wave test waveforms

Reset state: 0.8

Key path: [Baseband]—>[Arbitrary Wave]—>[Basic Configuration]—>[Test Waveform Type Square Wave]—>
[Create Test Waveform]—>[Amplitude]

[:SOURce[1]|2]:RADio:ARB:TWAVeform:SQUare:CLOad

Function description: Create a square wave test waveform and load it into arbitrary wave. At this time, the created square wave file name and storage path are both default values and cannot be set. The default generated file name is SQUATESTWAVE.seg, which is stored in the /home/ceyear/SgData/user/Wav/ folder

Setting format: [:SOURce[1]|2]:RADio:ARB:TWAVeform:SQUare:CLOad

Example: :RADio:ARB:TWAVeform:SQUare:CLOad
Create a square wave test waveform and load it into arbitrary wave.

Reset state: None

Key path: [Baseband]—>[Arbitrary Wave]—>[Basic Configuration]—>[Test Waveform Type Square Wave]—>
[Create Test Waveform]—>[Create and Load Output File]

[:SOURce[1]|2]:RADio:ARB:TWAVeform:SQUare:CREate <filename>

Function description: Create a square wave test waveform. The parameter is the specified file name of the square wave waveform created, which is of string type, and only includes the file name and does not include the path. The default storage path for the file is /home/ceyear/SgData/user/Wav/

Setting format: [:SOURce[1]|2]:RADio:ARB:TWAVeform:SQUare:CREate “fileName”

Parameter description: Parameter of string type

Example: :RADio:ARB:TWAVeform:SQUare:CREate “text.seg”
Create a sine wave test waveform and name it test.seg

Reset state: None

Key path: [Baseband]—>[Arbitrary Wave]—>[Basic Configuration]—>[Test Waveform Type Square Wave]—>
[Create Test Waveform]—>[Create Output File]

[:SOURce[1]|2]:RADio:ARB:TWAVeform:SQUare:FREQuency <val>

Function description: Set the frequency value for creating square wave test waveforms

Setting format: [:SOURce[1]|2]:RADio:ARB:TWAVeform:SQUare:FREQuency val

Query format: [:SOURce[1]|2]:RADio:ARB:TWAVeform:SQUare:FREQuency?

Parameter description: Floating point type parameter
Range :[100Hz,625MHz]

Example: :RADio:ARB:TWAVeform:Square:FREQuency 1MHz
Set the frequency value of the square wave test waveform created to 1MHz

Reset state: 1kHz

Key path: [Baseband]—>[Arbitrary Wave]—>[Basic Configuration]—>[Test Waveform Type Square Wave]—>
[Create Test Waveform]—>[Frequency]

[:SOURce[1]|2]:RADio:ARB:TWAVeform:SQUare:OFFSet <val>

Function description: Set the DC offset for creating square wave test waveforms

Setting format: [:SOURce[1]|2]:RADio:ARB:TWAVeform:SQUare:OFFSet val

Query format: [:SOURce[1]|2]:RADio:ARB:TWAVeform:SQUare:OFFSet?

Parameter description: Floating point type parameter, expressed in FS as a percentage of full scale

Range: [| Square Wave Test Waveform Amplitude | -1,1- | Square Wave Test Waveform Amplitude |]

Example: :RADio:ARB:TWAVeform:SQUare:AMPLitude 0.1

Set the DC offset of 0.1 for creating square wave test waveforms

Reset state: 0

Key path: [Baseband]—>[Arbitrary Wave]—>[Basic Configuration]—>[Test Waveform Type Square Wave]—>

[Create Test Waveform]—>[DC Offset]

[:SOURce[1]|2]:RADio:ARB:TWAVeform:SQUare:SAMPles <val>

Function description: Set the number of sampling points for creating square wave test waveforms

Setting format: [:SOURce[1]|2]:RADio:ARB:TWAVeform:SQUare:SAMPles <val>

Query format: [:SOURce[1]|2]:RADio:ARB:TWAVeform:SQUare:SAMPles?

Parameter description: Shaping type parameter

Range: [4, max (40000, 2.5e10/square wave test waveform frequency)]

Example: :RADio:ARB:TWAVeform:SQUare:SAMPles 2000

Set the number of sampling points for creating square wave test waveforms to 2000

Reset state: 1000

Key path: [Baseband]—>[Arbitrary Wave]—>[Basic Configuration]—>[Test Waveform Type Square Wave]—>

[Create Test Waveform]—>[Number of Sampling Points per Week]

[:SOURce[1]|2]:RADio:ARB:TWAVeform:TYPE <Type>

Function description: Set test waveform type

Setting format: [:SOURce[1]|2]:RADio:ARB:TWAVeform:TYPE SINE|SQUare

Parameter description: Discrete type

SINE: sine wave

SQUare: Square wave

Example: :RADio:ARB:TWAVeform:TYPE SINE

Set the created test waveform type to sine wave

Reset state: SINE

Key path: [Baseband]—>[Arbitrary Wave]—>[Basic Configuration]—>[Test Waveform Type]

[:SOURce[1]|2]:RADio:ARB:WAVeform <FileName>

Function description: This command is used to select arbitrary wave file to load and play. The parameter is a string type, indicating the name of arbitrary wave file to be loaded, including the path. The default path is the "/home/ceyear/SgData/user/Wav/" folder

Setting format: [:SOURce[1]|2]:RADio:ARB:WAVeForm “<file_name>”

Query format: [:SOURce[1]|2]:RADio:ARB:WAVeForm?

Parameter description:

< file_name >	Arbitrary wave file name.

Example: :RADio:ARB:WAVeform “wavetest.bin”
Channel A loads an arbitrary wave file named wavetest.bin
SOURce2:RADio:ARB:WAVeform “wavetest2.bin”
Channel B loads an arbitrary wave file named wavetest2. bin

Key path: [Baseband]—>[Arbitrary Wave]—>[Basic Configuration]—>[Load Arbitrary Wave]

[:SOURce[1]|2]:RADio:ARB:WAVeform:RESet

Function description: This command is used to reset the current playback settings of arbitrary wave, delete the selected file of arbitrary wave, and set the arbitrary wave to OFF.

Setting format: [:SOURce[1]|2]:RADio:ARB:WAVeForm:RESet

Example: :RADio:ARB:WAVeform RESet Reset to default value

Key path: [Baseband]—>[Arbitrary Wave]—>[Basic Configuration]—>[Reset to Default Value]

Description: for setting only.

3.3.16 Custom subsystem

The CUSTom subsystem command is used to control the digital modulation function, and the subsystem commands are as follows:

[:SOURce[1]|2]:RADio:CUSTom:ALPHa <FilterAlpha>

Function description: This command is used to set the filter factor of Nyquist filter, Root Nyquist filter, Gaussian filter and rectangular filter. If the user changes the value, it will affect the bandwidth occupied by the baseband signal spectrum. Please refer to the command"[:SOURce[1]|2]:RADio:CUSTom:FILTer".

Setting format: [:SOURce[1]|2]:RADio:CUSTom:ALPHa <val>.

Query format: [:SOURce[1]|2]:RADio:CUSTom:ALPHa?

Parameter description:

<FilterAlpha> filter factor.

Range: [0, 1.000].

Example: :RADio:CUSTom:ALPHa 0.350 set the filter factor to 0.35.

Reset state: 0.350

Key path: [Baseband]—>[Digital Modulation]—>[Filter Settings]—>[Filter Factor α]

[:SOURce[1]|2]:RADio:CUSTom:DATA <Mode>

Function description: This command sets the data source for the digital modulation signal of the signal generator. Users can choose from 11 data sources: PN9, PN11, PN15, PN16, PN20, PN21, PN23, ZERO, ONE, PATTern, and PRAM.

Setting format: [:SOURce[1]|2]:RADio:CUSTom:DATA PN9|PN11|PN15 |PN16|PN20|PN21|PN23|ZERO|ONE|PATTern|PRAM

Query format: [:SOURce[1]|2]:RADio:CUSTom:DATA?

Parameter description:

<Mode> discrete data. Please refer to the setting command format for data source type of digital modulation signal.

Example: :RADio:CUSTom:DATA ZERO

The digital modulation data source is selected as all 0

Reset state: PN9

Key path: [Baseband]—>[Digital Modulation]—>[Data Source Configuration]—>[Data Source Selection]

[:SOURce[1]|2]:RADio:CUSTom:DATA:PATTern <Val>,<BitCount>

Function description: This command sets the value of the custom sequence when the data source is selected as a custom sequence. Settings can be made in different decimal systems, but shall always be returned in hexadecimal when queried. The command contains two parameters. The first parameter Val is unsigned 64-bit shaped data, supporting binary, decimal and hexadecimal data and indicating the set PATTern value, which is displayed in binary form in the instrument software interface. The second parameter, BitCount, is shaped signed data, indicating the number of bits when the setting data is displayed in binary form. If the number of bits set for displaying Val as binary data does not match the number of bits specified in BitCount, automatic truncation or zeroing will be performed, with the former starting from the low position, and the latter performed at the high position.

Setting format: [:SOURce[1]|2]:RADio:CUSTom:DATA:PATTern <Val>,<BitCount>

Query format: [:SOURce[1]|2]:RADio:CUSTom:DATA:PATTern?

Parameter description:

< Val> Unsigned integer data, with the highest bit processed as 1 when set to a negative number.

<BitCount> Integer data, value range: [1,64]

Example: :RADio:CUSTom:DATA:PATT #HF,4 Set the sequence segment data to 1111 (binary), which returns to # HF, 4 when queried.

Reset state: 0

Key path: [Baseband]—>[Digital Modulation]—>[Data Source Configuration]—> [Data Source Selection Custom Sequence]—>[Custom Sequence]

[:SOURce[1]|2]:RADio:CUSTom:DATA:PRAM <S>

Function description: When file stream is selected as the data source in the digital modulation of signal generator, this command is used to select the stream file that must be saved in /home/ceyear/SgData/user/DataSrc with the extension of ".src". The parameter is a string, containing only the specified file name and not the path name. When no file is selected, it returns to "NULL” when queried.

Setting format: [:SOURce[1]|2]:RADio:CUSTom:DATA:PRAM <S>

Query format: [:SOURce[1]|2]:RADio:CUSTom:DATA:PRAM?

Parameter description:

<S > the name of file stream selected contains the extension.

Example: :RADio:CUSTom:DATA:PRAM “Test.src”
Select stream file “Test.src”.

Key path: [Baseband]—>[Digital Modulation]—>[Data Source Configuration]—> [Data Source Selection File Stream] - [Select File]

[:SOURce[1]|2]:RADio:CUSTom:DATA:SCOunt <Num>

Function description: Set the number of sampling points for generating waveform files.

Setting format: [:SOURce[1]|2]:RADio:CUSTom:DATA:SCOunt <Num>

Query format: [:SOURce[1]|2]:RADio:CUSTom:DATA:SCOunt?

Parameter description:

<Num> Shaping data, number of sampling points for generating waveform files.

Range: [1281000000000], the value must be a multiple of 4. If the set value P is not a multiple of 4,

then it is automatically set to a value that is less than the integer multiple of the maximum 4 of P.

Example: :RADio:CUSTom:DATA:SCOUNT 50000 Set the number of sampling points for generating waveform segment files to 50000

Reset state: 40000

Key path: [Baseband]—>[Digital Modulation]—>[Data Source Configuration]—>[Number of Sampling Points]

Remarks: The command is valid when generating function options for instrument installation waveform segment files (S06)

[:SOURce[1]|2]:RADio:CUSTom:FILTer <Mode>

Function description: This command is used to select the type of digital modulation filter for the signal generator, including RNYQuist, NYQuist, GAUSsian, RECTangle and IS95. Among them, RECTangle is applicable for digital frequency modulation signals, such as FSK and MSK. Refer to “[:SOURce[1]|2]:RADio:CUSTom:MODulation[:TYPE]” command.

Setting format: [:SOURce[1]|2]:RADio:CUSTom:FILTer RNYQuist|NYQuist |GAUSsian| RECTangle|IS95

Query format: [:SOURce[1]|2]:RADio:CUSTom:FILTer?

Parameter description:

<Mode> discrete data. The values of digital modulation filter type are as follows:

RNYQuist	:Rnyquist filter
NYQuist	:Nyquist filter
GAUSsian	:Gaussian filter
RECTangle	:Rectangle filter

IS95: CDMA2000 baseband shaping filter.

Example: :RADio:CUSTom:FILTer RNYQuist

The digital modulation filter is Root-Nyquist filter.

Reset state: RNYQuist

Key path: [Baseband]—>[Digital Modulation]—>[Filter Settings]—>[Filter Selection]

[:SOURce[1]|2]:RADio:CUSTom:FILTer:PRESet

Function description: This command is used to restore the filter of digital modulation to the default filter setting, that is, the filter type is QPSK and the filtering factor is α 0.35.

Setting format: [:SOURce[1]|2]:RADio:CUSTom:FILTer:PRESet

Parameter description: No parameters

Example: :RADio:CUSTom:FILTer:PRESet

Restore the digital modulation filter to the default filter settings.

Key path: [Baseband]-> [Digital Modulation]-> [Filter Settings]-> [Restore Default Filter Dode]

[:SOURce]:RADio:CUSTom:IMAP[1]|3:TYPE MARKer|TRIGger

Function description: This instruction is used to set the interface type in digital modulation interface mapping. When the channel of interface "Marker 1/ Trigger A" belongs to channel A or the channel of interface "Marker 3/ Trigger B" belongs to channel B, the corresponding interface type is set.

Setting format: [:SOURce]:RADio:CUSTom:IMAP[1]|3:TYPE MARKer|TRIGger

Query format: [:SOURce]:RADio:CUSTom:IMAP[1]|3:TYPE?

Parameter description:

< num >	Discrete data,
	MARker: Marker
	TRIGger: Trigger

Example: :RADio: :CUSTom:IMAP3:TYPE TRIGger

Set Marker 3 as the trigger channel of channel B

Reset state: MARKer

Key path: [Baseband]-> [Digital Modulation]-> [Interface Mapping]-> [Interface Type]

[:SOURce]:RADio:CUSTom:IMAP[1]|2|3|4:USED <TYPE>

Function description: This command sets which channel the marker output is used for. When the instrument is dual-channel, this command is valid.

Setting format: [:SOURce]:RADio:CUSTom:IMAP[1]|2|3|4:USED ACHannel|BCHannel.

Query format: [:SOURce]:RADio:CUSTom:IMAP[1]|2|3|4:USED?

Parameter description:

< num >	Discrete data,
	ACHannel: Channel A
	BCHannel: Channel B

Example: :RADio:CUSTom:IMAP2:USED BCHannel

This setting meas that Marker 2 is used for channel B

Reset state: Single-channel: ACHannel

Dual-channel: Marker 1, Marker 2 ACHannel; Marker 3, Marker 4 BCHannel

Key path: [Baseband]-> [Digital Modulation]-> [Interface Mapping]-> [Channel]

[:SOURce[1]|2]:RADio:CUSTom:IQData <Ival>{,<Qval>…}

Function description: This command is used to download arbitrary data into the instrument in the way of I/Q data pair through the communication interface of the signal generator, and play data I and Q through the radio. This command can only to used to transmit string data, and the data is normalized data. The first data is I, the second data is Q, and the number of data I and Q is even. At most 5000 I/Q data pairs are supported.

Setting format: [:SOURce[1]|2]:RADio:CUSTom:IQData <val>{,<val>}

Parameter description:

<IVal>	string data type. Data I in I/Q data pair. Range: [-32767, 32767].
<QVal>	string data type. Data Q in I/Q data pair. Range: [-32767, 32767].

Example: :RADio:CUSTom:IQData 1024,32767,-13678,40
Send 4 normalized data I and Q to the signal generator: the first data is I, the second data is Q, and so on.

Description: for setting only.

Remarks: It is not supported at present.

[:SOURce[1]|2]:RADio:CUSTom:MARKer[1]|2|3|4:CUSTom <S>

Function description: When the marker type in digital modulation of the signal generator is selected as custom, this command is used to select a custom marker file, which must be saved under the file "/home/ceyear/SgData/user/Marker/"with the file extension ".mrk". The parameter is of string type, which is only used to specify the file name and does not contain the path. If no file is selected for the marker type file, the instruction will return a "NULL" string when querying at this time.

Setting format: [:SOURce[1]|2]:RADio:CUSTom:MARKer[1]|2|3|4:CUSTom <S>

Query format: [:SOURce[1]|2]:RADio:CUSTom:MARKer[1]|2|3|4:CUSTom?

Parameter description:

< s > String type, the selected custom marker file name, including the file extension.

Example: :RADio:CUSTom:MARKer:CUSTom “Test.mrk”
Select “Test.mrk” as the custom file of Marker 1.

Key path: [Baseband]-> [Digital Modulation]-> [Marker Setting]-> [
[Mkr 1|2|3|4 Type Custom] -> [Select Marker File]

[:SOURce[1]|2]:RADio:CUSTom:MARKer[1]|2|3|4:DIVider <Val>

Function description: When the marker type in digital modulation of the signal generator is selected as pulse, this command is used to set the average value of pulse marker.

Setting format: [:SOURce[1]|2]:RADio:CUSTom:MARKer[1]|2|3|4:DIVider <val>

Query format: [:SOURce[1]|2]:RADio:CUSTom:MARKer[1]|2|3|4:DIVider？

Parameter description:

<val > Integer data, marked as the average value of the pulse.
Range: [2,1024].

Example: :RADio:CUSTom:MARKer:DIVider 100
Set the average value of the pulse marker type for Marker 1 to 100.

Reset state: 2

Key path: [Baseband]-> [Digital Modulation]-> [Marker Setting]-> [
[Mkr 1|2|3|4 Type Pulse] —> [Average Value]

[:SOURce[1]|2]:RADio:CUSTom:MARKer[1]|2|3|4:OFFTime <Val>

Function description: This command sets the marker number of off time when the marker type is selected as fixed ON/OFF ratio.

Setting format: [:SOURce[1]|2]:RADio:CUSTom:MARKer[1]|2|3|4:OFFTime <Val>.

Query format: [:SOURce[1]|2]:RADio:CUSTom:MARKer[1]|2|3|4:OFFTime?

Parameter description:

< Val > Integer data, used to set the marker number of OFF time when the marker type is set as fixed ON/OFF ratio.
Range: [1, 16777215]

Example: :RADio:CUSTom:MARKer:OFFTime 100
Set the marker number of OFF time to 100 when the marker 1 type is selected as fixed ON/OFF ratio.

Reset state: 1

Key path: [Baseband]-> [Digital Modulation]-> [Marker Configuration]->
[Mkr 1|2|3|4 Type Fixed ON/OFF Ratio] —> [OFF Time Sampling Points]

[:SOURce[1]|2]:RADio:CUSTom:MARKer[1]|2|3|4:ONTime <Val>

Function description: This command sets the marker number of On time when the marker type is selected as fixed ON/OFF ratio.

Setting format: [:SOURce[1]|2]:RADio:CUSTom:MARKer[1]|2|3|4:ONTime <Val>.

Query format: [:SOURce[1]|2]:RADio:CUSTom:MARKer[1]|2|3|4:ONTime?

Parameter description:

< Val > Integer data, used to set the marker number of On time when the marker type is set as fixed ON/OFF ratio.
Range: [1, 16777215]

Example: :RADio:CUSTom:MARKer:ONTime 1
Set the marker number of On time to 1 when the marker 1 type is selected as fixed ON/OFF ratio.

Reset state: 1

Key path: [Baseband]-> [Digital Modulation]-> [Marker Configuration]->
[Mkr 1|2|3|4 Type Fixed ON/OFF Ratio] —> [On Time Sampling Points]

[:SOURce[1]|2]:RADio:CUSTom:MARKer[1]|2|3|4:PATTern <Val>,<BitCount>

Function description: This command sets the value of the predefined style when the tag type is selected as the predefined style. Settings can be made in different decimal systems, but shall always be returned in hexadecimal when queried. The command contains two parameters. The first parameter Val is unsigned 64-bit shaped data, supporting binary, decimal and hexadecimal data and indicating the set PATTern value, which is displayed in binary form in the instrument software interface. The second parameter, BitCount, is shaped signed data, indicating the number of bits when the setting data is displayed in binary form. If the number of bits set for displaying Val as binary data does not match the number of bits specified in BitCount, automatic truncation or zeroing will be performed, with the former starting from the low position, and the latter performed at the high position.

Setting format: [:SOURce[1]|2]:RADio:CUSTom:MARKer[1]|2|3|4:PATTern <Val>,<BitCount>.

Query format: [:SOURce[1]|2]:RADio:CUSTom:MARKer[1]|2|3|4:PATTern?

Parameter description:

< Val> Unsigned integer data, with the highest bit processed as 1 when set to a negative number.

<BitCount> Integer data, value range: [1,64]

Example: Hexadecimal data example: RADio:CUSTom:MARKer2:PATT #HF,4
Binary data example: RADio:CUSTom:MARKer2:PATT #B1111,4
Decimal data example: Raio:CUSTom:MARker2:PATT 15,4
When Marker 2 is set as a predefined style, the value is 1111 (binary), and returns to # HF, 4 when queried

Reset state: 10(binary)

Key path: [Baseband]-> [Digital Modulation]-> [Marker Configuration]-> [Mkr 1|2|3|4 Type Predefined Style] —> [Predefined Style]

[:SOURce[1]|2]:RADio:CUSTom:MARKer[1]|2|3|4:[:TYPE] <Mode>

Function description: This command selects the tag types of digital modulation of the signal generator, including four types: CUSTom, PULSe, PATTern and RATio.

Setting format: [:SOURce[1]|2]:RADio:CUSTom:MARKer[1]|2|3|4:TYPE CUSTom|PULSe
|PATTern| RATio

Query format: [:SOURce[1]|2]:RADio:CUSTom:MARKer[1]|2|3|4:TYPE?

Parameter description:

<Mode> discrete data. The type of digital modulation marker, with the values

listed below:

CUSTom - custom,

PULSe :Pulse

PATTern : Predefined style,

RATio : Fixed ON/OFF ratio.

Example: :RADio:CUSTom:MARKer PULSe

The Digital Modulation type is Pulse.

Reset state: PULSe

Key path: [Baseband]-> [Digital Modulation]-> [Marker Setting]-> [Mkr 1|2|3|4 Type]

[:SOURce[1]|2]:RADio:CUSTom:MODulation:APSK[16]|32:GAMMa <mode>

Function description: This command sets the gamma value when the modulation format is APSK. Users can choose from G2D3, G3D4, G4D5, G5D6, G8D9, G9D10 and CUSTom. When the modulation type is 16APSK, the selectable gamma values are G2D3, G3D4, G4D5, G5D6, G8D9, G9D10 and CUSTom; when the modulation type is 32APSk, the selectable gamma values are G3D4, G4D5, G5D6, G8D9, G9D10 and CUSTom. For instructions on setting APSK modulation type, refer to "[:SOURce[1]|2]:RADio:CUSTom:MODulation[:TYPE]"

Setting format: [:SOURce[1]|2]:RADio:CUSTom:MODulation:APSK16:GAMMa G2D3|G3D4|G4D5|G5D6|G8D9|G9D10|CUSTom

Query format: [:SOURce[1]|2]:RADio:CUSTom:MODulation:APSK16:GAMMa?

Parameter description:

<Mode> discrete data. Gamma value of APSK.

APSK16:		APSK32	
G2D3	: 3.15(DVB-S2 2/3)		
G3D4	: 2.85(DVB-S2 3/4)	G3D4	: 2.84(DVB-S2 3/4)
G4D5	: 2.75(DVB-S2 4/5)	G4D5	: 2.72(DVB-S2 4/5)
G5D6	: 2.70(DVB-S2 5/6)	G5D6	: 2.64(DVB-S2 5/6)
G8D9	: 2.60(DVB-S2 8/9)	G8D9	: 2.54(DVB-S2 8/9)
G9D10	: 2.57(DVB-S2 9/10)	G9D10:	2.53(DVB-S2 9/10)
CUSTom :	自定义	CUSTom:	Custom

Example: :RADio:CUSTom:MODulation:APSK16:GAMMa G2D3

APSK16 的伽马值为 G2D3.

Reset state: APSK16:G2D3 APSK32:G3D4

Key path: **[Baseband]—>[Digital Modulation] —>** [Select Modul Type] -> [Select Modul Type APSK]

->[Gamma Value]

[:SOURce[1]|2]:RADio:CUSTom:MODulation:APSK[16]|32:GAMMa:CUSTom[1]|2 <Val>

Function description: this command sets the modulation format as APSK, and the value of γ, γ1 or γ2 is customized when the gamma value is user-defined. When the modulation type is 16APSK, this command is used to set the user-defined gamma value, and the command format is [:SOURce[1]|2]:RADio:CUSTom:MODulation:APSK[16]:GAMMa:CUSTom[1]. CUSTom2 setting is not supported. When the modulation type is 32APSK, the command format is [:SOURce[1]|2]:RADio:CUSTom:MODulation:APSK32:GAMMa:CUSTom[1]|2. In this modulation type, you can set the value of γ2 through CUSTom2.

Setting format: [:SOURce[1]|2]:RADio:CUSTom:MODulation:APSK[16]|32:GAMMa:CUSTom <val>

Query format: [:SOURce[1]|2]:RADio:CUSTom:MODulation:APSK[16]|32:GAMMa:CUSTom?

Parameter description:

< Val> Custom gamma value of APSK.

Range: 16APSK : [1.00,10000.00]

32APSK: γ1:[1.00,10000.00]

γ2:[1.00,10000.00]

Example: :RADio:CUSTom:MODulation:APSK32:GAMMa:CUSTom2 2.5

APSK32 的 γ2 的值为 2.5.

Reset state: In the case of 16APSK: γ is 2.57, in the case of 32APSK, γ1 is 2.53, and y2 is 4.30

Key path: **[Baseband]—>[Digital Modulation]** —> [Select Modul format -> [Select Modul Type APSK]

->[Gamma Value Custom]

[:SOURce[1]|2]:RADio:CUSTom:MODulation:ASK[:DEPTh] <Val>

Function description: This command is used to set the modulation depth of ASK when the digital modulation type is ASK mode. The value works after ASK mode is selected as the modulation type. For modulation type commands of digital modulation, please refer to "[:SOURce[1]|2]:RADio:CUSTom:MODulation[:TYPE]".

Setting format: [:SOURce[1]|2]:RADio:CUSTom:MODulation:ASK:DEPTh <Val>

Query format: [:SOURce[1]|2]:RADio:CUSTom:MODulation:ASK:DEPTh?

Parameter description:

<val> modulation depth of ASK when the modulation type is ASK mode.

Range: [0, 100].

Example: :RADio:CUSTom:MODulation:ASK:DEPTh 30

The modulation depth of ASK is 30%.

Reset state: 100

Key path: **[Baseband]—>[Digital Modulation]** —> [Select Modul Type—> [Select Modul Type ASK]
—> [Modulation depth]

[:SOURce[1]|2]:RADio:CUSTom:MODulation:AQPSk[:ANGLe] <Val>

Function description: This command is used to set the modulation angle of AQPSk when the modulation type of digital modulation is AQPSk. The value works after AQPSk mode is selected as the modulation type. For modulation type commands of digital modulation, please refer to "[:SOURce[1]|2]:RADio:CUSTom:MODulation[:TYPE]".

Setting format: [:SOURce[1]|2]:RADio:CUSTom:MODulation:AQPSk:ANGLe <Val>

Query format: [:SOURce[1]|2]:RADio:CUSTom:MODulation:AQPSk:ANGLe?

Parameter description:

<Val> Modulation angle of AQPSk when the modulation type of floating point data is in the AQPSk mode.
Range:[0deg, 180.00deg]

Example: :RADio:CUSTom:MODulation:AQPSk:ANGLe 30deg
The modulation angle of AQPSk is 30 degrees.

Reset state: 0

Key path: **[Baseband]—>[Digital Modulation]** —> [Select Modul Type—> [Select Modul Type ASK]
—>[Modulation Angle]

[:SOURce[1]|2]:RADio:CUSTom:MODulation:FSK[:DEViation] <Dev>

Function description: This command is used to set the frequency offset of FSK when the digital modulation of is FSK mode. The value works after the user select FSK mode as the modulation type. For modulation type commands of the base, please refer to "[:SOURce[1]|2]:RADio:CUSTom:MODulation[:TYPE]".

Setting format: [:SOURce[1]|2]:RADio:CUSTom:MODulation:FSK[:DEViation] <val>

Query format: [:SOURce[1]|2]:RADio:CUSTom:MODulation:FSK[:DEViation]?

Parameter description:

<Dev> frequency deviation of FSK when the modulation type is FSK mode.
Range: [0 Hz, 2*Symbol Rate].

Example: :RADio:CUSTom:MODulation:FSK 1MHz
The frequency deviation of FSK is 1MHz.

Reset state: 0.4kHz

Key path: **[Baseband]—>[Digital Modulation]] —>** [Select Modul Type]—> [Select Modul Type 2FSK|4FSK|8FSK|16FSK|32FSK|64FSK]—> [Frequency Offset]

[:SOURce[1]|2]:RADio:CUSTom:MODulation:PRESet

Function description: This command is used to set the modulation type of digital modulation as the default, and the default modulation type is QPSK. For modulation type commands of digital modulation, please refer to "[:SOURce[1]|2]:RADio:CUSTom:MODulation[:TYPE]". This command is for setting only.

Setting format: [:SOURce[1]|2]:RADio:CUSTom:MODulation:PRESet

Parameter description: No parameters

Example: :RADio:CUSTom:MODulation:PRESet

Set the modulation tyope of digital modulation as QPSk, the default modulation type.

Key path: **[Baseband]—>[Digital Modulation] —>** [Select Modul Type—> [Restore Default Modul Type]

[:SOURce[1]|2]:RADio:CUSTom:MODulation[:TYPE] <Mode>

Function description: This command is used to set the radio modulation type.

Setting format: **[:SOURce[1]|2]:RADio:CUSTom:MODulation[:TYPE] ASK|BPSK|QPSK|QPSK45|QEDGe|AQPSk|OQPSk|P4QPsk|P2DBpsk|P4DQpsk|P8D8psk|8PSK|8PEDG|16QAM|16QEDG|32QAM|32QEDG|64QAM|128QAM|256QAM|512QAM|1024QAM|2048QAM|4096QAM|MSK|2FSK|4FSK|8FSK|16FSK|32FSK|64FSK|UFSK|16APSK|32APSK|UIQ**

Query format: [:SOURce[1]|2]:RADio:CUSTom:MODulation[:TYPE]?

Parameter description:

<Mode> discrete data. Please refer to the setting command format for digital modulation type.

Example: :RADio:CUSTom:MODulation 8PSK

The Digital Modulation type is 8PSK.

Reset state: QPSK

Key path: **[Baseband]—>[Digital Modulation] —>** [Select Modul Type] -> [Select Modul Type]

[:SOURce[1]|2]:RADio:CUSTom:MODulation:UFSK:DEViation[1]|2--16 <Dev>

Function description: This command sets each frequency offset value of the current custom FSK type. If the user needs to set different frequency offset values, the digital suffix after the DEViation indicates which offset value is set.

Setting format:

[:SOURce[1]|2]:RADio:CUSTom:MODulation:UFSK:DEViation[1]|2--16 <val>

Query format:

[:SOURce[1]|2]:RADio:CUSTom:MODulation:UFSK:DEViation[1]|2--16?

Parameter description:

<Dev> frequency offset of custom FSK when the modulation type is custom FSK mode.

Range: [-2*Symbol Rate, 2*Symbol Rate]. The parameter range is related to the symbol rate of the modulation, in Hz.

Example: :RADio:CUSTom:MODulation:UFSK:DEViation 1MHz

Custom FSK has a frequency modulation frequency offset of 1MHz.

Reset state: 0.4kHz

Key path: **[Baseband]—>[Digital Modulation]] —**> [Select Modul Type]—> [Select Modul Type Custom FSK]-> [Frequency Offset]

[:SOURce[1]|2]:RADio:CUSTom:MODulation:UFSK:TYPE <MODE>

Function description: This command sets the current custom FSK type,

Setting format: [:SOURce[1]|2]:RADio:CUSTom:MODulation:UFSK:TYPE 4FSK|8FSK|16FSK

Query format: [:SOURce[1]|2]:RADio:CUSTom:MODulation:UFSK:TYPE?

Parameter description:

<Mode> Discrete variable

Value: 4FSK|8FSK|16FSK

Example: :RADio:CUSTom:MODulation:UFSK:TYPE 8FSK

Reset state: 4FSK

Key path: **[Baseband] —>[Digital Modulation]] —**> [Select Modul Type] -> [Select Modul Type] **—**> [Custom FSK]

[:SOURce[1]|2]:RADio:CUSTom:MODulation:UIQ <S>

Function description: This command is used to set the user I/Q file to be loaded when the digital modulation type is user I/Q. Please refer to "[:SOURce[1]|2]:RADio:CUSTom:MODulation[:TYPE]" for radio modulation type commands. The command parameter is a file name containing the extension .iqm, without file path. The default path is /home/ceyear/SgData/user/IqMap/.

Setting format: [:SOURce[1]|2]:RADio:CUSTom:MODulation:UIQ <S>

Parameter description:

<S>   Selected user I/Q file name.

Example: :RADio:CUSTom:MODulation:UIQ “test.iqm”

Select test.iqm file.

Key path: **[Baseband]—>[Digital Modulation] —>** [Select Modul Type] —> [Select Modul Type Custom I/Q File] -> [Select I/Q File]

[:SOURce[1]|2]:RADio:CUSTom:SRATe <Val>

Function description: It is used to set the code element rate of the digital modulation of the signal generator, and the units of the code element are sps, ksps, Msps and Gsps.

Setting format: [:SOURce[1]|2]:RADio:CUSTom:SRATe <val>

Query format: [:SOURce[1]|2]:RADio:CUSTom:SRATe?

Parameter description:

<Val>   symbol rate for digital modulation, the value range is related to the modulation bandwidth, with the value range as follow:

Standard configuration: [50Sps,125MSps]

Optional H31|H41: [50Sps,312.5MSps]

Optional H31|H42: [50Sps,625MSps]

Optional H31|H43: [50Sps,1.25GSps]

Example: :RADio:CUSTom:SRATe 3Msps   the symbol rate is 3Msps.

Reset state: 24.300000ksps

Key path: [Baseband]->   [Digital   Modulation]->   [Data   Source Configuration]->[Symbol Rate]

[:SOURce[1]|2]:RADio:CUSTom[:STATe] <State>

Function description: This command is used to enable the digital modulation switch of the signal generator.

Setting format: [:SOURce[1]|2]:RADio:CUSTom:STATe ON|OFF|1|0

Query format: [:SOURce[1]|2]:RADio:CUSTom:STATe?

Parameter description:

<State>   Boolean data, with the values listed below:

ON | 1: Radio ON

OFF | 0: Baseband OFF.

Example: :RADio:CUSTom:STATe 1   Turn on digital modulation.

Reset state: 0

Key path: [Baseband]->   [Digital   Modulation]->   [Data   Source Configuration]->[Digital Modulation ON OFF]

[:SOURce[1]|2]:RADio:CUSTom:TRIGger:EXECute

Function description: When the trigger source of digital modulation is set as the trigger key, the instruction executes a trigger. for setting only.

Setting format: [:SOURce[1]|2]:RADio:CUSTom:TRIGger:EXECute

Example: :RADio:CUSTom:TRIGger:EXECute. Execute a trigger

Reset state: None

Key path: [Baseband]-> [Digital Modulation]-> [Trigger Mode]->[Trigger Source Trigger Key]-> [Trigger]

[:SOURce[1]|2]:RADio:CUSTom:TRIGger[:SOURce] <Mode>

Function description: It is used to set the baseband signal trigger source of the signal generator, including two modes, namely, KEY and EXT.

Setting format: [:SOURce[1]|2]:RADio:CUSTom:TRIGger:SOURce KEY| EXT

Query format: [:SOURce[1]|2]:RADio:CUSTom:TRIGger:SOURce?

Parameter description:

<Mode> discrete data. The values of radio signal trigger source are as follows:

KEY : the trigger source is the trigger key on the instrument front panel

EXT : the trigger source is the trigger input at the interface of the instrument rear panel.

Example: :RADio:CUSTom:TRIGger:SOURce KEY

The baseband signal trigger source is trigger key.

Reset state: KEY

Key path: [Baseband]-> [Digital Modulation]-> [Trigger Mode]->[Trigger Source]

[:SOURce[1]|2]:RADio:CUSTom:TRIGger[:SOURce]:EXTernal:DELay:SAMPle <Val>

Function description: This command is used to set the sampling points of external trigger signal delay for the baseband signal to respond to the trigger signal when external is selected as trigger source of the signal generator. The effective prerequisite for this setting is to set the external delay to ON state. Please refer to "[:SOURce[1]|2]:RADio:CUSTom:TRIGger[:SOURce]:EXTernal:DELay:STATe " for external delay ON/OFF status commands.

Setting format:
[:SOURce[1]|2]:RADio:CUSTom:TRIGger[:SOURce]:EXTernal:DELay
:SAMPLe <val>

Query format: [:SOURce[1]|2]:RADio:CUSTom:TRIGger[:SOURce]:EXTernal:DELay
:SAMPle?

Parameter description:

<Val> the quantity of delayed sampling points when external is selected as the baseband trigger source.

Range: [0,1048575]。

Example: :RADio:CUSTom:TRIGger:EXTernal:DELay:SAMPle 3000
The external trigger delay is 3000 bits.

Reset state: 0

Key path: [Baseband]-> [Digital Modulation]-> [Trigger Mode]->[Trigger Source Ext]—>
[Delay Type Bit]—>[Delay]

[:SOURce[1]|2]:RADio:CUSTom:TRIGger[:SOURce]:EXTernal:DELay:STATe <State>

Function description: This command is used to set the external trigger delay state when external is selected as trigger source of the signal generator. When it is ON, the external delay set takes effect.

Setting format:
[:SOURce[1]|2]:RADio:CUSTom:TRIGger[:SOURce]:EXTernal:DELay
:STATe ON|OFF|1|0

Query format:
[:SOURce[1]|2]:RADio:CUSTom:TRIGger[:SOURce]:EXTernal:DELay
:STATe?

Parameter description:

<State> Boolean data. The values of delay state when external is selected as the radio trigger source are as follows:
ON | 1: Delay ON
OFF | 0: delay OFF.

Example: :RADio:CUSTom:TRIGger:EXTernal:DELay:STATe 1 external trigger delay ON

Reset state: 0

Key path: [Baseband]-> [Digital Modulation]-> [Trigger Mode]->[Trigger Source Ext]—>
—>[Time Delay ON OFF]

[:SOURce[1]|2]:RADio:CUSTom:TRIGger[:SOURce]:EXTernal:DELay[:TIME] <Val>

Function description: This command is used to set the time of external trigger signal delay for the baseband signal to respond to the trigger signal when external is selected as trigger source of the signal generator. The effective prerequisite for this setting is to set the external delay to ON state. Please refer to "[:SOURce[1]|2]:RADio:CUSTom:TRIGger[:SOURce]:EXTernal:DELay:STATe " for external delay ON/OFF status commands.

Setting format:
[:SOURce[1]|2]:RADio:CUSTom:TRIGger[:SOURce]:EXTernal:DELay

[:TIME] <val>

Query format: [:SOURce[1]|2]:RADio:CUSTom:TRIGger[:SOURce]:EXTernal:DELay [TIME]?

Parameter description:

<Val> delay time when external is selected as the radio trigger source. Range: [0, 10s].

Example: :RADio:CUSTom:TRIGger:EXTernal:DELay 3000ms
The external trigger delay is 3000 ms.

Reset state: 0s

Key path: [Baseband]-> [Digital Modulation]-> [Trigger Mode]->[Trigger Source Ext]—>
[Delay Type Time]—>[Delay Time]

[:SOURce[1]|2]:RADio:CUSTom:TRIGger[:SOURce]:EXTernal:SLOPe <Mode>

Function description: This command is used to set the trigger input signal polarity of the instrument rear panel to high effective trigger radio output or low effective trigger radio output when external is selected as trigger source of the signal generator.

Setting format:
[:SOURce[1]|2]:RADio:CUSTom:TRIGger[:SOURce]:EXTernal:SLOPe POSitive|NEGative

Query format:
[:SOURce[1]|2]:RADio:C0USTom:TRIGger[:SOURce]:EXTernal:SLOPe?

Parameter description:

<Mode> discrete data. The values of external trigger polarity are as follows:

POSitive	: positive
NEGative	: negative

Example: :RADio:CUSTom:TRIGger:EXTernal:SLOPe NEGative the external trigger slope is low effective.

Reset state: POSitive

Key path: [Baseband]-> [Digital Modulation]-> [Trigger Mode]->[Trigger Source Ext]
—> [External Trigger Slope Negative Positive]

[:SOURce[1]|2]:RADio:CUSTom:TRIGger:TYPE <Mode>

Function description: It is used to set the trigger mode of the baseband signal that is used for data transmission control, including three modes, namely, continuous, single and gate.

Setting format: [:SOURce[1]|2]:RADio:CUSTom:TRIGger:TYPE CONTinuous|SINGle|GATE

Query format: [:SOURce[1]|2]:RADio:CUSTom:TRIGger:TYPE?

Parameter description:

<Mode> discrete data. The values of radio signal trigger mode are as follows:

CONTinuous	:	the baseband trigger mode is set to be continuous,
SINGle	:	the baseband trigger mode is set to be single,
GATE	:	the baseband trigger mode is set to be gate.

Example: :RADio:CUSTom:TRIGger:TYPE SING

The radio trigger mode is single.

Reset state: CONTinuous

Key path: [Baseband]-> [Digital Modulation]-> [Trigger Mode]->[Trigger Mode]

[:SOURce[1]|2]:RADio:CUSTom:TRIGger:TYPE:CONTinuous[:TYPE] <Mode>

Function description: This command is used to set the type for radio data to respond to trigger signal when continuous is selected as the radio trigger mode. Users may select automatic, trigger or real-time. Please refer to “[:SOURce[1]|2]:RADio:CUSTom:TRIGger:TYPE” for baseband signal trigger mode commands.

Setting format:

[:SOURce[1]|2]:RADio:CUSTom:TRIGger:TYPE:CONTinuous:TYPE FREE|TRIGger|RESet

Query format:

[:SOURce[1]|2]:RADio:CUSTom:TRIGger:TYPE:CONTinuous:TYPE?

Parameter description:

<Mode> discrete data. The values of trigger signal type when continuous is selected as the radio trigger mode are as follows:

FREE : Select the AUTO mode, and the digital sub-modulation signal automatically starts to play,
and all trigger events are ignored during the playing process.

TRIGger: Select the trigger mode, in which the system will not play the current arbitrary wave
Load the digital modulation signal before the event. After receiving a valid trigger event,
The system starts to play the digital modulation signal continuously. and does not receive new content during playback.
trigger signal.

RESet: Select real-time mode, and the system will not play the arbitrary wave until it receives a valid trigger event;
until a valid trigger event is received; the system starts

to continuously current play the digital modulation signal after receiving a valid trigger event. New trigger signal received during the playing process, will stop the digital modulation signal currently being played and play it again from the beginning.

Example: :RADio:CUSTom:TRIGger:TYPE:CONTinuous:TYPE FREE
Set the continuous trigger to continuous automatic mode.

Reset state: FREE

Key path: [Baseband]-> [Digital Modulation]-> [Trigger Mode]->[Trigger Mode Cont]
-> [Cont Trig Type]

[:SOURce[1]|2]:RADio:CUSTom:TRIGger:TYPE:GATE:ACTive <Mode>

Function description: This command is used to set the gate trigger mode in baseband trigger mode. Users may select low effective or high effective. When the external trigger signal is high or low, the baseband signal output will be triggered. Please refer to "[:SOURce[1]|2]:RADio:CUSTom:TRIGger:TYPE" for baseband signal trigger mode commands.

Setting format: [:SOURce[1]|2]:RADio:CUSTom:TRIGger:TYPE:GATE:ACTive HIGH|LOW

Query format: [:SOURce[1]|2]:RADio:CUSTom:TRIGger:TYPE:GATE:ACTive?

Parameter description:

<Mode> discrete data. The values of trigger signal type when gate is selected as the radio trigger mode are as follows:

HIGH	: high effective,
LOW	: low effective.

Example: :RADio:CUSTom:TRIGger:TYPE:GATE:ACTive HIGH
The digital modulation signal is output when the external trigger signal is high level.

Reset state: HIGH

Key path: [Baseband]-> [Digital Modulation]-> [Trigger Mode]->[Trigger Mode Gate]
-> [Gate Trigger Type]

[:SOURce[1]|2]:RADio:CUSTom:WAVeform:CREate <S>

Function description: When the necessary parameters in the digital modulation of the signal generator are configured, this command is used to set the file name of the generated waveform file. The default suffix of the waveform file is .seg, and the file is placed in the folder /home/ceyear/SgData/user/Wav/ by default, and the file path is obtained by referring to the default characteristics and the waveform file is generated. This command is for setting only.

Setting format: [:SOURce[1]|2]:RADio:CUSTom:WAVeform:CREate <S>

Parameter description:

<S > File name used by the generate waveform segment

Example: :RADio:CUSTom:WAVeform:CREate "Test.seg"

Set the file path of the generated waveform segment to /home/ceyear/SgData/user/Wav/Test.seg, and store the generated waveform segments in this file.

Key path: [Baseband]-> [Digital Modulation]-> [Data Sourse Configuration]->[Generate Waveform Segment]

Remarks: The command is valid when generating function options for instrument installation waveform segment files (S06)

3.3.17 MILimeter subsystem

The multi-tone subsystem is used to configure the multi-tone modulation function. The instruction of this subsystem will be valid only after the multi-tone modulation function option (S01) is installed in the instrument, otherwise an error prompt of "Undefined command" will be generated.

The Multitone subsystem command is used to control the multi-tone modulation function, and the subsystem command is as follows:

[:SOURce[1]|2]:RADio:MTONe:ARB:CFACtor < value>

Function description: This command sets the target peak-to-average ratio in multi-tone configuration. When the peak-to-average ratio optimization mode is set as the target, this command setting is valid. Please refer to "[:SOURce[1]|2]:RADio:MTONe:ARB:CFACtor:MODE " for the instruction of setting peak-to-average ratio optimization mode.

Setting format: [:SOURce[1]|2]:RADio:MTONe:ARB:CFACtor < value>

Query format: [:SOURce[1]|2]:RADio:MTONe:ARB:CFACtor?

Parameter description:

<value> Floating-point type parameter, target peak-to-average ratio
Range:[0dB, 30dB].

Example: :RADio:MTONe:ARB:CFACtor 15dB
Set the target peak-to-average ratio to 15dB.

Reset state:

Key path: [Baseband]-> [Multi-tone Modulation] -> [Base Config]-> [Peak-to-average Ratio Optimization Mode Target] ->
Target Peak-to-Average Ratio

Description: It is not supported at present.

[:SOURce[1]|2]:RADio:MTONe:ARB:CFACtor:MODE < Mode>

Function description: This command sets the target peak-to-average ratio optimization mode in multi-tone configuration.

Setting format: [:SOURce[1]|2]:RADio:MTONe:ARB:CFACtor:MODE <Mode >

Query format: [:SOURce[1]|2]:RADio:MTONe:ARB:CFACtor:MODE?

Parameter description:

<value> Discrete type parameter, target peak-to-average ratio optimization mode

OFF:	Off
CHIRp:	Quick
SLOW:	Objective (not supported at present)

Example: :RADio:MTONe:ARB:CFACtor:MODE CHIR
Set the peak-to-average ratio optimization mode to Fast

Key path: [Baseband]-> [Multi-tone Modulation] -> [Base Config]-> [Peak-to-average Ratio Optimization Mode]

[:SOURce[1]|2]:RADio:MTONe:ARB:SETup <FileName>

Function description: This command is used to select a multitone file and load it into the memory of the signal generator for play. It is only required to set the name of the multitone file instead of specifying the absolute path. The default path is "/home/ceyear/SgData/user/Mtone/".

Setting format: [:SOURce[1]|2]:RADio:MTONe:ARB:SETup <file_name>

Parameter description:

<FileName> Character string type, namely, the multiple-tone file name.

Example: :RADio:MTONe:ARB:SETup "mtone1.ton"
Load mtone1.ton file into the memory of the signal generator.

Key path: [Baseband]-> [Multi-tone Modulation] -> [Multiple-tone List Setup] -> [Load Multi-tone File]

Description: for setting only.

[:SOURce[1]|2]:RADio:MTONe:ARB:SETup:RESet

Function description: This command restores the tone list and tone configuration parameters to their default values. By default, the multi-tone modulation switch is off, the number of tones is 2, the initial phase is fixed, the phase relationship between tones is the same, the frequency interval is 1MHz, and the initial phase is 0.

Setting format: [:SOURce[1]|2]:RADio:MTONe:ARB:SETup:RESet

Example: :RADio:MTONe:ARB:SETup:RESet

Key path: [Baseband]-> [Multi-tone Modulation]-> [Base Config]-> [Restore Default Value]

Description: for setting only.

[:SOURce[1]|2]:RADio:MTONe:ARB:SETup:STORe <FileName>

Function description: This command saves the waveform data from current multitone list in the multitone file of the signal generator. The parameter of this instruction is a string type, excluding path name, and is only the name of the file saved. The default path is /home/ceyear/SgData/user/Mtone/.

Setting format: [:SOURce[1]|2]:RADio:MTONe:ARB:SETup:STORe <file_name>

Parameter description:

<FileName> Character string type, namely, the multiple-tone file name.

Example: :RADio:MTONe:ARB:SETup:STORe “mtone1.ton”

Save the multitone list in the multitone file mtone1.ton of the signal generator.

Key path: [Baseband] -> [Multiple-tone List Setup] -> [Save As]

Description: for setting only.

[:SOURce[1]|2]:RADio:MTONe:ARB:SETup:TABLe <FreqSpacing>,<NumTones>,<Pow>,<Phase>,<State>

Function description: This command is used to create and configure a multitone waveform. The parameters include <freq_offset>, <num_tones>, <pow>, <phase> and <state>. <Freq_offset> is subject to the bandwidth of the baseband and the number of tones in the multiple-tone list.

Setting format: [:SOURce[1]|2]:RADio:MTONe:ARB:SETup:TABLe <freq_spacing>,<num_tones>,{<pow>,<phase>,<state>…}

Parameter description: Parameter of string type

<FreqSpacing> Frequency spacing between multiple tones.
Range: [-100MHz, 100MHz].

<NumTones>number of tones.
Range: [2, 512].

<Pow> Power gain.
Range: [-100dB,0dB]。

<Phase> initial phase.
Range: [0deg, 359.99deg].

<State> state. Boolean data; the values are taken as follows:
1: ON
0: OFF.

Example: :RADio:MTONe:ARB:SETup:TABLe “1000000, 3, -10, 90, 0, -20, 0, 1, -30, 45, 1”
This example shows that the multitone modulation frequency spacing is set to 1MHz, and there are 3 tones. The power gain of the first tone is 10dB, the phase is 90deg, and the state is OFF. The power gain of the second tone is 20dB, the phase is 0deg, and the state is ON. The power gain of the third tone is 30dB, the phase is 45deg, and the state is ON.

Reset state: 1MHz，2，0dB，0deg，1

Key path: None

Description: for setting only.

[:SOURce[1]|2]:RADio:MTONe:ARB:SETup:TABLe:FSPacing <FreqSpacing>

Function description: This command sets the frequency interval between multi-tone tones. When modifying this value, the frequency offset value of multi-tone list will be modified at the same time.

Setting format: [:SOURce]:RADio:MTONe:ARB:SETup:TABLe:FSPacing <val><freq unit>

Query format: [:SOURce]:RADio:MTONe:ARB:SETup:TABLe:FSPacing?

Parameter description:

<FreqSpacing> Frequency spacing between multiple tones.
Range: The range of multi-tone interval is related to the option, as follows
Standard Configuration: [-100MHz, 100MHz].
Option H31: [-250MHz, 250MHz]. Option H41: 1MHz[100Hz, 500MHz].
Option H31: [-250MHz, 250MHz]. Option H41: 1MHz[100Hz, 500MHz].
Option H31: [-250MHz, 250MHz]. Option H41: 1MHz[100Hz, 500MHz].

Example: :RADio:MTONe:ARB:SETup:TABLe:FSPacing 200kHz
Set the frequency spacing of multitone list to 200kHz.

Reset state: 1MHz

Key path: [Baseband]-> [Multi-tone Modulation]-> [Base Config]-> [Freq Interval]

[:SOURce[1]|2]:RADio:MTONe:ARB:SETup:TABLe:NTONes <NumTones>

Function description: This command sets the number of tones in the multi-tone modulation list. When this value is modified, the number of multi-tone points in the multi-tone will be modified at the same time.

Setting format: [:SOURce[1]|2]:RADio:MTONe:ARB:SETup:TABLe:NTONes <val>

Query format: [:SOURce[1]|2]:RADio:MTONe:ARB:SETup:TABLe:NTONes?

Parameter description:

<NumTones> number of tones in multitone modulation list.

Range: [2, 65535].

Example: :RADio:MTONe:ARB:SETup:TABLe:NTONes 3

Set the number of tones in multitone list to 3.

Reset state: 2

Key path: [Baseband]-> [Multi-tone Modulation]-> [Base Config]-> [Number of Tones]

[:SOURce[1]|2]:RADio:MTONe:ARB:SETup:TABLe:PHASe <val>

Function description: Set the initial phase of multi-tone, modify the value of initial phase and modify the phase value of existing sequence in multi-tone list. When the initial multi-tone phase is fixed, this command is valid. For the initial multi-tone phase setting, please refer to "[:SOURce[1]|2]:RADio:MTONe:ARB:SETup:TABLe:PHASe:INITialize".

Setting format: [:SOURce[1]|2]:RADio:MTONe:ARB:SETup:TABLe:PHASe <val>

Query format: [:SOURce[1]|2]:RADio:MTONe:ARB:SETup:TABLe:PHASe?

Parameter description:

<Val>val> Float data type

Range: [0,359.99]

Example: :RADio:MTONe:ARB:SETup:TABLe:PHASe 10

Set the initial phase of the multi-tone list to 10 deg.

Reset state: 0

Key path: [Baseband]-> [Multi-tone Modulation]-> [Base Config]-> [Initial Phase]

[:SOURce[1]|2]:RADio:MTONe:ARB:SETup:TABLe:PHASe:INITialize <Mode>

Function description: To initialize the initial phase mode in the multiple-tone modulation list, including two modes, namely, random and fixed. When it is set to fixed, all tone phase positions in the multiple-tone list will be set to the fixed value of 0 degrees; when it is set to random, such phase positions will be set with different random values based on the random seed settings. Please refer to"[:SOURce[1]|2]:RADio:MTONe:ARB:SETup:TABLe:PHASe:INITialize:SEED"for relationship between tone phases in multitone modulation.

Setting format:

[:SOURce[1]|2]:RADio:MTONe:ARB:SETup:TABLe:PHASe:INITialize RANDom|FIXed

Query format:

[:SOURce[1]|2]:RADio:MTONe:ARB:SETup:TABLe:PHASe:INITialize?

Parameter description:

<Mode> discrete data. The values of initial phase mode in multitone modulation list are as follows:

RANDom	: set all tones to a random value.
FIXed	: set all tones to a fixed value.

Example: :RADio:MTONe:ARB:SETup:TABLe:PHASe:INITialize FIX

Set the tone phase in multitone list to a fixed value.

Reset state: FIXed

Key path: [Baseband]-> [Multi-tone Modulation]-> [Base Config]-> [Initial Phase]

[:SOURce[1]|2]:RADio:MTONe:ARB:SETup:TABLe:PHASe:INITialize:SEED <Mode>

Function description: This command is used to set the relationship between tone phases in multitone modulation, including random and fixed.

Setting format:

[:SOURce[1]|2]:RADio:MTONe:ARB:SETup:TABLe:PHASe:INITialize:S
E
ED RANDom|FIXed

Query format:

[:SOURce[1]|1]:RADio:MTONe:ARB:SETup:TABLe:PHASe:INITialize:S
E
ED?

Parameter description:

<Mode> discrete data. The values of relationship between tone phases in multitone modulation are as follows:

RANDom	: the relationship between tone phases is random
FIXed	: the relationship between tone phases is fixed.

Example: :RADio:MTONe:ARB:SETup:TABLe:PHASe:INITialize:SEED FIX set the relationship between tone phases to fixed.

Reset state: FIX

Key path: [Baseband]-> [Multi-tone Modulation]-> [Base Config]-> [Phase

Relationship between Multitones]

[:SOURce[1]|2]:RADio:MTONe:ARB:SETup:TABLe:ROW <RowIndex>,<Pow>,<Phase>,<State>

Function description: This command is used to modify the multitone parameters in a row of the multitone modulation list, including <row_index>, <pow>, <phase> and <state>. If users need to modify the whole multitone list, please refer to the command "[:SOURce[1]|2]:RADio:MTONe:ARB:SETup:TABLe".

Setting format: [:SOURce[1]|2]:RADio:MTONe:ARB:SETup:TABLe:ROW <row_index>,<pow>,<phase>,<state>

Query format: [:SOURce[1]|2]:RADio:MTONe:ARB:SETup:TABLe:ROW? <row_Index>

Parameter description:

<RowIndex>	row index in multitone list. Range: [0, 65534].
<Pow>	Power gain. Range:[-100dB, 0dB].
<Phase>	initial phase. Range:[0deg, 359.99deg].
<State>	state. Boolean data; the values are taken as follows: 1: ON 0: OFF.

Example: :RADio:MTONe:ARB:SETup:TABLe:ROW 2,-10,40,0 Set the power gain in the third line of the multi-tone list to -10dB, the phase to 40deg, the status to OFF.

:RADio:MTONe:ARB:SETup:TABLe:ROW? 2 query the index 2, that is, the information in the third row of the multitone list includes frequency offset, attenuation gain, phase and state.

Reset state: None

Key path: None

[:SOURce[1]|2]:RADio:MTONe:ARB[:STATe] <State>

Function description: This command sets the multi-tone ON/OFF state of the signal generator.

Setting format: [:SOURce[1]|2]:RADio:MTONe:ARB:STATe ON|OFF|1|0

Query format: [:SOURce[1]|2]:RADio:MTONe:ARB:STATe?

Parameter description:

<State> Boolean data. The values of multitone modulation state are as follows:

ON | 1: Multitone modulation ON

OFF | 0: multitone modulation OFF.

Example: :RADio:MTONe:ARB:STATe 1 multitone modulation ON.

Reset state: 0

Key path: [Baseband]-> [Multi-tone Modulation]-> [Base Config]-> [multitone Modulation Setting: ON OFF]

3.3.18 PWM Subsystem

The PWM subsystem is used to configure the intra-pulse modulation function. The instruction of this subsystem will be valid only after the intra-pulse modulation function option (S03) is installed in the instrument, otherwise an error prompt of "Undefined command" will be generated.

The PWM subsystem command sets the functions of intra-pulse modulation, and the subsystem commands are as follows:

[:SOURce[1]|2]:PWM:BARKer:TYPE < Mode>

Function description: This command sets the symbol type of Barker code when the intra-pulse modulation type is Barker code. The value can be 2, 3, 4, 5, 7, 11, or 13.

Setting format: [:SOURce[1]|2]:PWM:BARKer:TYPE 2|3|4|5|7|11|13

Query format: [:SOURce[1]|2]:PWM:BARKer:TYPE?

Parameter description:

<Mode> Discrete type parameter, Barker symbol type

The value can be 2, 3, 4, 5, 7, 11, or 13.

Example: :PWM:BARKer:TYPE 7

Set the Barker symbol type to 7

Reset state: 2

Key path: [Baseband]**—**>[Intra-pulse Modulation] **—**>[Intra-pulse Modulation Type Barker Code|**—**> [Barker Code]

[:SOURce[1]|2]:PWM:BWIDth <value>

Function description: This command sets the modulation bandwidth of three types of intra-pulse modulation: linear frequency modulation, triangular frequency modulation, and cosine quartic frequency modulation.

Setting format: [:SOURce[1]|2]:PWM:BWIDth <value>

Query format: [:SOURce[1]|2]:PWM:BWIDth?

Parameter description:

<value> float parameter, the set modulation bandwidth value

Range: The value range of intra-pulse modulation bandwidth is related to the options, as follows:

Standard Configuration: [-100MHz, 100MHz].

Option H31: [-250MHz, 250MHz]. Option H41: [-100MHz, 100MHz].

Option H31: [-250MHz, 250MHz]. Option H41: [-100MHz, 100MHz].

Option H31: [-250MHz, 250MHz]. Option H41: [-100MHz, 100MHz].

Example: :PWM:BWIDth 50MHz

Set the Intra-pulse modulation bandwidth to 50 MHz.

Reset state: 3.84MHz

Key path: [Baseband]—>[Intra-pulse Modulation] —>[Intra-pulse Modulation Type linear FM| Triangular FM|Cosine Fourth-law FM] —>[Modulation BW]

[:SOURce[1]|2]:PWM:DIRection < Mode>

Function description: This command sets the frequency modulation direction when the intra-pulse modulation type is linear frequency modulation (LFM), triangular frequency modulation (triangular FM), or cosine fourth-law frequency modulation (Cosine Fourth-law FM), which is related to the currently selected intra-pulse modulation type; when the intra-pulse modulation type is LFM, it is set to the frequency modulation direction of linear frequency modulation; When the intra-pulse modulation is triangular FM, the frequency modulation direction of triangular frequency modulation is set, and when the intra-pulse modulation is Cosine Fourth-law FM, the frequency modulation direction of cosine quartic frequency modulation is set. When the intra-pulse modulation type is Barker Code or PM Code, the frequency modulation direction of LFM is set. For instructions for setting intra-pulse modulation types, refer to "[:SOURce[1]|2]:PWM:TYPE".

Setting format: [:SOURce[1]|2]:PWM:DIRection POSitive|NEGative

Query format: [:SOURce[1]|2]:PWM:DIRection?

Parameter description:

<Mode> Discrete type parameter, FM direction

POSitive : Increase (Linear FM)
Upper triangle (Triangular FM)
Rise (Cosine Fourth-law FM)

NEGative : Reduce (Linear FM)
Lower triangle (Triangular FM)
Fall (Cosine Fourth-law FM)

Example: :PWM:DIRection POSitive
Set the intra-pulse FM direction to Increase | Up Triangule | Rise.

Reset state: NEGative

Key path: [Baseband]—>[Intra-pulse Modulation] —>[Intra-pulse Modulation Type linear FM| Triangular FM|Cosine Fourth-law FM] —>[Modulation Direction]

[:SOURce[1]|2]:PWM:FSK[:DEViation] < Val>

Function description: When the intra-pulse modulation type is FSK, this command sets or queries the frequency offset of FSK. For instructions for setting intra-pulse modulation types, refer to "[:SOURce[1]|2]:PWM:TYPE".

Setting format: [:SOURce[1]|2]:PWM:FSK[:DEViation] <Val>

Query format: [:SOURce[1]|2]:PWM:FSK[:DEViation]?

Parameter description:

<value> Floating point data, set the FM frequency offset value
Range: [0Hz, Max. Modulation Bandwidth]

Example: :PWM:FSK:DEViation 100MHz Set the frequency offset of FSK to 100MHz

Reset state: 400Hz

Key path: [Baseband]—>[Intra-pulse Modulation] —>[Intra-pulse Modulation Type FSK|—> [Frequency Offset]

[:SOURce[1]|2]:PWM:LENGth <value>

Function description: This command sets the sequence length when the intra-pulse modulation is phase modulation code or FSK modulation. For instructions for setting intra-pulse modulation types, refer to "[:SOURce[1]|2]:PWM:TYPE".

Setting format: [:SOURce[1]|2]:PWM:LENGth

Query format: [:SOURce[1]|2]:PWM:LENGth?

Parameter description:

<Value> Integer parameter, sequence length
Value range: [1, 100]

Example: :PWM:LENGth 3
Set the sequence length of the PM Code to 3.

Reset state: 1

Key path: [Baseband]—>[Intra-pulse Modulation] —>[Intra-pulse Modulation Type PM Code|—> [Sequence Length]

[:SOURce[1]|2]:PWM:PPCM:TYPE < Mode>

Function description: This command sets the type of PM Code when the intra-pulse modulation type is PM Code. The values are P1|P2|P3|P4|FRANk|BPSK|QPSK.

Setting format: [:SOURce[1]|2]:PWM:PPCM:TYPE P1|P2|P3|P4|FRANk|BPSK|QPSK

Query format: [:SOURce[1]|2]:PWM:PPCM:TYPE?

Parameter description:

<Mode> Discrete type parameter, PM code type

P1	: P1 multi-phase code
P2	: P1 multi-phase code
P3	: P1 multi-phase code
P4	: P4 multi-phase
FRANk	: Frank code
BPSK	: BPSK modulation
QPSK	: QPSK modulation

Example: :PWM:PPCM:TYPE FRANk

Set the FM code type of the multi-phase code to FRANk

Reset state: P1

Key path: [Baseband]—>[Intra-pulse Modulation] —>[Intra-pulse Modulation Type PM Code|—> [PM Code Type]

[:SOURce[1]|2]:PWM:STATe <State>

Function description: This command sets the intrapulse modulation ON/OFF of the signal generator.

Setting format: [:SOURce[1]|2]:PWM:STATe ON|OFF|1|0

Query format: [:SOURce[1]|2]:PWM:STATe?

Parameter description:

<State> Boolean data. The values of intrapulse modulation ON/OFF state are as follows:

ON | 1: Intra-pulse modulation ON

OFF | 0: intrapulse modulation OFF.

Example: :PWM:STATe 1 Turn on intra-pulse modulation.

Reset state: 0

Key path: [Baseband]—>[Intra-pulse Modulation] —>[Intra-pulse Modulation ON OFF]

[:SOURce[1]|2]:PWM:SYMBols?

Function description: This command inquires the number of symbols when the pulse modulation is phase modulation code or FSK. For instructions for setting intra-pulse modulation types, refer to "[:SOURce[1]|2]:PWM:TYPE".

Query format: [:SOURce[1]|2]:PWM:SYMBols?

Example: :PWM:SYMBols?

Query the number of symbols of PM code.

Key path: [Baseband]**—**>[Intra-pulse Modulation] **—**>[Intra-pulse Modulation FM Code|**—**> [Number of Symbols]

Description: For query only.

[:SOURce[1]|2]:PWM:TYPE < Mode>

Function description: This command sets the types of intra-pulse modulation, including LFM, Triangular FM, Cosine Fourth-law FM, Raised Cosine FM, Barker Code, PM Code and FSK.

Setting format: [:SOURce[1]|2]:PWM:TYPE LFM|TFM|NLFM|RCOSine|BARKer|PPCM|FSK

Query format: [:SOURce[1]|2]:PWM:TYPE?

Parameter description:

<Mode> Discrete type parameter, intra-pulse modulation type

LFM	: Linear frequency modulation
TFM	: Triangular frequency modulation
NLFM	: Cosine fourth-law FM (Non-linear frequency modulation
RCOSine	: Raised cosine frequency modulation
BARKer	: Barker code
PPCM	: Phase modulation code
FSK	: FSK

Example: :PWM:TYPE TFM

Set the intra-pulse modulation type to triangular FM

Reset state: LFM

Key path: [Baseband]**—**>[Intra-pulse Modulation] **—**>[Intra-pulse Modulation Type]

3.3.19 Status Subsystem

The status subsystem command detects the status of the signal generator by monitoring the operation status register, the question register and the event register.

Commands or events affecting the status register

Status register	***RST**	*CLS	**Start up**	STATus:PRESet
SCPI transition filter (NTR and PTR)	No effect	No effect	Preset	Preset

SCPI enable register	No effect	No effect	Preset	Preset
SCPI event register	No effect	Clear	Clear	No effect
SCPI error/event queue enable	No effect	No effect	Preset	Preset
SCPI error/event queue	No effect	Clear	Clear	No effect
IEEE488.2 Register ESE SRE	No effect	No effect	Clear	No effect
IEEE488.2 Register ESR STB	No effect	Clear	Clear	No effect

Preset status description: The preset value of PTR is 0x7fff (32767); both NTR and enable registers are cleared. Three levels of status registers are provided in the signal generator from bottom to top, as follows:

There are 6 first-level status registers, of which 4 of the Questionable type contain Power, Frequency, Modulation and Calibration; One of the Operation types is Baseband. All the registers of this type are 16-bit.

There are three second-level status registers, namely Questionable, Standard Event, and Standard Operation. The Questionable and Standard Operation registers in such status registers are 16-bits, and the Standard Event register is 8-bit.

The third level status register is the Status Byte register. This type of register is 8-bit.

The status register related instructions are as follows:

:STATus:OPERation:CONDition?

Function description: This command queries the value of condition register in the operation status register.

Query format: :STATus:OPERation:CONDition?

Parameter description: Return the value of the condition register in the operation status register.

Range: [0,32767].

Example: :STAT:OPER:COND? Query the condition register value in the operation status register.

:STATus:OPERation:ENABle <val>

Function description: This command queries or sets the operation status event enable register.

Setting format: :STATus:OPERation:ENABle <val>

Query format: :STATus:OPERation:ENABle?

Parameter description: Integer parameter

Value in the operation status event enable register.

Range: [0,32767].

Example: :STAT:OPER:ENAB 8

Enable the third bit of the Operation status register (sweep status) to 1.

:STATus:OPERation:NTRansition <val>

Function description: This command queries or sets the negative transition filter for operation state.

Setting format: :STATus:OPERation:NTRansition <val>

Query format: :STATus:OPERation:NTRansition?

Parameter description: Integer parameter

Value of the negative transition filter for operation state.

Range: [0,32767].

Example: :STAT:OPER:NTR 8

Set the third bit of the negative transition filter for operation state to 1, other bits to 0.

:STATus:OPERation:PTRansition <val>

Function description: This command queries or sets positive transition filter for operation state.

Setting format: :STATus:OPERation:PTRansition <val>

Query format: :STATus:OPERation:PTRansition?

Parameter description: Value of the positive transition filter for integer parameter operation state.

Range: [0,32767].

Example: :STAT:OPER:PTR 4

Set the second bit of the positive transition filter for integer parameter operation state to 1, other bits to 0.

:STATus:OPERation[:EVENt]?

Function description: This command queries the operation status event register. After the query, this register is automatically cleared.

Query format: :STATus:OPERation[:EVENt]?

Parameter description: Return the value in the operation status register.

Range: [0,32767].

Example: :STAT:OPER?

Query the value of operation status event register.

:STATus:PRESet

Function description: This command presets some status registers as follows, and other registers remain unchanged.

Setting format: :STATus:PRESet

Example: :STAT:PRES

Status register reset state description

Register	**Sub register**	**Preset status**
OPERation	ENABle	All 0
	PTR	All 1
	NTR	All 0
QUEStionable	ENABle	All 0
	PTR	All 1
	NTR	All 0
Other	ENABle	All 0
	PTR	All 1
	NTR	All 0

:STATus:QUEStionable:CONDition?

Function description: This command queries the value of the condition register in the question status register

Query format: :STATus:QUEStionable:CONDition?

Parameter description: Query the value of the condition register in the question status register.

Range: [0,32767].

Example: :STAT:QUES:COND?

Query the value of condition register in the question status register.

:STATus:QUEStionable:ENABle <val>

Function description: This command queries or sets the question status event enable register.

Setting format: :STATus:QUEStionable:ENABle <val>

Query format: :STATus:QUEStionable:ENABle?

Parameter description: Integer parameter, the value in the question status event enable register.

Range: [0,32767].

Example: :STAT:QUES:ENAB 32

Set the fifth bits of the question status register (frequency question) enabling to 1.

:STATus:QUEStionable:NTRansition <val>

Function description: This command queries or sets the question status negative transition filter.

Setting format: :STATus:QUEStionable:NTRansition <val>

Query format: :STATus:QUEStionable:NTRansition?

Parameter description: Integer parameter, the value in the question status negative hopping filter.

Range: [0,32767].

Example: :STAT:QUES:NTR 17

Set the 0th and fourth bit of the question status negative hopping filter to 1, other bits to 0.

:STATus:QUEStionable:PTRansition <val>

Function description: This command queries or sets the question status positive transition filter.

Setting format: :STATus:QUEStionable:PTRansition <val>

Query format: :STATus:QUEStionable:PTRansition?

Parameter description: Integer parameter, the value in the question status positive hopping filter.

Range: [0,32767].

Example: :STAT:QUES:PTR 6

Set the first and second bits of the question status positive hopping filter to 1, other bits to 0.

:STATus:QUEStionable[:EVENt]?

Function description: This command queries the question status event register. After the query, this register is automatically cleared.

Query format: :STATus:QUEStionable[:EVENt]?

Parameter description: Return the value in the question status register.

Range: [0,32767].

Example: :STAT:QUES?

Query the value of question status event register.

:STATus:QUEStionable:FREQuency:CONDition?

Function description: This command queries the value in the frequency question status event enable register..

Query format: :STATus:QUEStionable:FREQuency:CONDition?

Parameter description: Integer parameter, return the value in the frequency question status event enable register..

Range: [0,32767].

Example: :STAT:QUES:FREQ:COND?

Query the value of condition register in the frequency question status register.

:STATus:QUEStionable:FREQuency:ENABle <val>

Function description: This command queries or sets the frequency question status event enable register.

Setting format: :STATus:QUEStionable:FREQuency:ENABle <val>

Query format: :STATus:QUEStionable:FREQuency:ENABle?

Parameter description:

Integer parameter, the value in the frequency question status event enable register.

Range: [0,32767].

Example: :STAT:QUES:FREQ:ENAB 64

Set the fixth bit of the question status register (YO loop loss of lock) enabling to 1.

:STATus:QUEStionable:FREQuency:NTRansition <val>

Function description: This command queries or sets the frequency question status negative transition filter.

Setting format: :STATus:QUEStionable:FREQuency:NTRansition <val>

Query format: :STATus:QUEStionable:FREQuency:NTRansition?

Parameter description: Integer parameter, the value in the question status negative hopping filter.

Range: [0,32767].

Example: :STAT:QUES:FREQ:NTR 9

Set the 0th and third bits of the question status negative hopping filter to 1, other bits to 0.

:STATus:QUEStionable:FREQuency:PTRansition <val>

Function description: This command queries or sets the frequency question status positive transition filter.

Setting format: :STATus:QUEStionable:FREQuency:PTRansition <val>

Query format: :STATus:QUEStionable:FREQuency:PTRansition?

Parameter description: Integer parameter, the value in the question status positive hopping filter.

Range: [0,32767].

Example: :STAT:QUES:FREQ:PTR 2

Set the first bit of the frequency question status positive hopping filter to 1, other bits to 0.

:STATus:QUEStionable:FREQuency[:EVENt]?

Function description: This command queries the frequency question status event register. After the query, this register is automatically cleared.

Query format: :STATus:QUEStionable:FREQuency[:EVENt]?

Parameter description: Return the value in the frequency question status register.

Range: [0,32767].

Example: :STAT:QUES:FREQ?

Query the value of the frequency question status event register.

:STATus:QUEStionable:POWer:CONDition?

Function description: This command queries the value of condition register in the power question status register.

Query format: :STATus:QUEStionable:POWer:CONDition?

Parameter description: Return the value of the condition register in the power question status register.

Range: [0,32767].

Example: :STAT:QUES:POW:COND?

Query the value of condition register in the power question status register.

:STATus:QUEStionable:POWer:ENABle <val>

Function description: This command queries or sets the power state event enable register.

Setting format: :STATus:QUEStionable:POWer:ENABle <val>

Query format: :STATus:QUEStionable:POWer:ENABle?

Parameter description: Integer parameter, the value in the power question status event enable register.

Range: [0,32767].

Example: :STAT:QUES:POW:ENAB 2

Sets the first bit (Unleveled) of the power question status register enabling to 1.

:STATus:QUEStionable:POWer:NTRansition <val>

Function description: This command queries or sets the power question status negative transition filter.

Setting format: :STATus:QUEStionable:POWer:NTRansition <val>

Query format: :STATus:QUEStionable:POWer:NTRansition?

Parameter description: Integer parameter, the value in the power question status negative hopping filter.

Range: [0,32767].

Example: :STAT:QUES:POW:NTR 10

sets the first and third bits of the question status negative hopping filter to 1, other bits to 0.

:STATus:QUEStionable:POWer:PTRansition <val>

Function description: This command queries or sets the power question status positive transition filter.

Setting format: :STATus:QUEStionable:POWer:PTRansition <val>

Query format: :STATus:QUEStionable:POWer:PTRansition?

Parameter description: Integer parameter, the value in the power question status positive hopping filter.

Range: [0,32767].

Example: :STAT:QUES:POW:PTR 7

sets the 0th, first and second bits of the question status positive hopping filter to 1, other bits to 0.

:STATus:QUEStionable:POWer[:EVENt]?

Function description: This command queries the power question status event register. After the query, this register is automatically cleared.

Query format: :STATus:QUEStionable:POWer[:EVENt]?

Parameter description: Return the value in the power question status condition register.

Range: [0,32767].

Example: :STAT:QUES:POW?

Query the value of the power question status event register.

3.3.20 Function subsystem

The function generator subsystem is used to realize the configuration function of the function generator and generate different signals. The instruction of this subsystem will be valid only after the low-frequency output/function generator option (S14) is installed in the instrument, otherwise an error prompt of "undefined command" will be generated.

The commands are used to select the operating modes, including:

[:SOURce[1]|2]:FUNCtion:FREQuency <Frequency>

Function description: This command sets the output frequency of the function generator. It should be noted that when the low-frequency waveform is selected as sweep sine or dual sine, this command sets the start frequency of sweep sine and dual sine frequency 1.

Setting format: [:SOURce[1]|2]:FUNCtion:FREQuency <val>

Query format: [:SOURce[1]|2]:FUNCtion:FREQuency?

Parameter description:

<Frequency> Floating point type data. Values are taken as follows

Range:

Sine: [0.01Hz, 10MHz]

Square wave, triangle wave and ramp wave: [0.01Hz, 1MHz].

Sweep sine start frequency: [0.001Hz, 9.99MHz 999999]. When 10 MHz is input, the software will automatically set the sweep sine start frequency to 9.99999999MHz without error message.

Dual sine frequency 1: [0.001Hz, 10 MHz].

Example: :FUNCtion:FREQuency 2MHz Set the frequency of the function generator to 2MHz.

Reset state: 400

Key path: [Function Generator 1|2] — > [Data Source] — > [Frequency]

[Function Generator 1 | 2]-> [Data Source]-> [Waveform Selection Sweep Sine]-> [Sweep Sine Start Frequency]

[Function Generator 1|2] — > [Data Source] — > [Waveform Selection Dual Sine]-> [Dual Sine Frequency 1]

[:SOURce[1]|2]:FUNCtion:FREQuency:ALTernate

Function description: This command sets the stop frequency of sweep sine or dual sine frequency 2 when the waveform of function generator is selected as sweep sine or dual sine. If the waveform is not sweep sine or dual sine, the value of dual sine frequency 2 is modified by default.

Setting format: [:SOURce[1]|2]:FUNCtion:FREQuency:ALTernate <val>

Query format: [:SOURce[1]|2]:FUNCtion:FREQuency:ALTernate?

Parameter description:

<Frequency> Dual sine frequency 2.

Range: [-325GHz～ +325GHz].

The stop frequency of the sweep sine or

Range: [0Hz, 1MHz].

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

Reset state: Double-sine frequency 2: 2kHz

Sweep sine: 0.002Hz

Key path: [Function Generator 1 | 2]-> [Data Source]-> [Waveform Selection Sweep Sine]-> [Sweep Sine Stop

Stop Frequency]

[Function Generator 1|2] — > [Data Source] — > [Waveform Selection Dual Sine]-> [Dual Sine Frequency 2]

[:SOURce[1]|2]:FUNCtion:FREQuency:ALTernate:AMPLitude:PERCent

Function description: This command sets the percentage of the amplitude of the second tone in the total output signal when the output waveform of the generator is dual sine; For example, the second tone accounts for 20% of the total waveform power, then the first tone accounts for 80% of the total power output.

Setting format:

[:SOURce[1]|2]:FUNCtion:FREQuency:ALTernate:AMPLitude:PERCent

<val>

Query format:
[:SOURce[1]|2]:FUNCtion:FREQuency:ALTernate:AMPLitude:PERCent?

Parameter description:

<Percent> Amplitude percent of the dual sine frequency 2.
Range: [0, 100].

Example: FUNCtion:FREQuency:ALTernate:AMPLitude:PERCent 20
Set the second waveform of the dual sine to account for 20% of the total signal power output.

Reset state: 50%

Key path: [Function Generator 1|2] — > [Data Source] — > [Waveform Selection Dual Sine]-> [Dual Sine Frequency 2
Amplitude Percent]

[:SOURce[1]|2]:FUNCtion:PERiod?

Function description: This command queries the period value of the function generator, which is related to the frequency.

Query format: [:SOURce[1]|2]:FUNCtion:PERiod?

Example: :FUNCtion:PERiod?

Reset state: 2.5ms

Key path: [Function Generator 1 | 2]-> [Data Source]-> [Period]

[:SOURce[1]|2]:FUNCtion:SHAPe

Function description: This command sets the signal output waveform of the function generator, that is, setting the waveform of the function generator. When the waveform is set to ramp or noise, the waveform will be automatically set to a certain ramp or noise, depending on the ramp type or noise type set last time. The default is an up ramp or white noise.

Setting format: [:SOURce[1]|2]:FUNCtion:SHAPe
SINE|SQUare|TRIangle|RAMP|NOISe| SWEep|DUALsine

Query format: [:SOURce[1]|2]:FUNCtion:SHAPe?

Parameter description:

<Mode> discrete data. The values of LF signal output waveform are as follows:

SINE	sine wave,
SQUare	Square wave,
TRIangle	triangle wave,
RAMP	ramp wave,
NOISE	Noise,
SWEPtsine	Sweep sine
DUALsine	Dual sine

Example: :FUNCtion:SHAPe TRIangle The LF signal generator waveform is triangle.

Reset state: SINE

Key path: [Function Generator 1|2] — > [Data Source] — > [Waveform Selection]

[:SOURce[1]|[2]:FUNCtion:SHAPe:NOISe

Function description: When the output waveform of the function generator is selected as noise, this command sets the noise type.

Setting format: [:SOURce[1]|2]:FUNCtion:SHAPe:NOISe GAUSsian |UNIForm

Query format: [:SOURce[1]|2]:FUNCtion:SHAPe:NOISe?

Parameter description:

<Mode> discrete data. The values of noise type of function generator are as follows:

GAUSsian	Gaussian noise
UNIForm	white noise.

Example: FUNCtion:SHAPe:NOISe GAUSsian Set the noise type to Gaussian.

Reset state: GAUS

Key path: [Function Generator 1|2] — > [Data Source] — > [Waveform Selection]

[:SOURce[1]|[2]:FUNCtion:SHAPe:RAMP

Function description: This command sets the signal output type when the waveform of the function generator is ramp, including up and down.

Setting format: [:SOURce[1]|2]:FUNCtion:SHAPe:RAMP POSitive|NEGative

Query format: [:SOURce[1]|2]:FUNCtion:SHAPe:RAMP?

Parameter description:

<Mode> discrete data. The values of ramp signal type are as follows:

POSitive	up
NEGative	down.

Example: FUNCtion:SHAPe:RAMP NEGative The low-frequency ramp wave is down.

Reset state: POS

Key path: [Function Generator 1|2] — > [Data Source] — > [Waveform Selection]

[:SOURce[1]|[2]:FUNCtion:SWEep:TIME

Function description: This command sets the sweep time of sweep sine when output waveform of the function generator is sweep sine.

Setting format: [:SOURce[1]|2]:FUNCtion:SWEep:TIME <val>

Query format: [:SOURce[1]|2]:FUNCtion:SWEep:TIME?

Parameter description:

<Time> Sweep time of sweep sine.
Range:[10us, 2s].

Example: FUNCtion:SWEep:TIME 200ms The sweep time of sweeping sine is 200 ms.

Reset state: 10ms

Key path: [Function Generator 1 | 2]-> [Data Source]-> [Waveform Selection Sweep Sine]— [Sweep Sine Sweep Time]

3.3.21 EXTernal Subsystem

The external source subsystem is used to realize the configuration function of external output signal when the modulation source is selected as external in analog modulation and low frequency output.

The commands are used to select the operating modes, including:

[:SOURce]:EXTernal[1]|2[:SOURce]:COUPling <MODE>

Function description: This command is used to set external input coupling mode.

Setting format: [:SOURce]:EXTernal[1]|2:COUPling DC|AC

Query format: [:SOURce]:EXTernal[1]|2:COUPling?

Parameter description:

<Mode> discrete data. The values of external input coupling mode are as follows:

DC	DC coupling
AC	AC coupling

Example: :EXTernal:COUPling AC set the external 1 input coupling mode to AC coupling.

Reset state: DC

Key path: [Function Generator 1 | 2]-> [Data Source]—>[Ext]-> [Input Coupling Mode]

[:SOURce]:EXTernal[1]|2[:SOURce]:IMPedance <MODE>

Function description: This command is used to set the external impedance mode.

Setting format: :EXTernal[1]|2:IMPedance <MODE>

Query format: :EXTernal[1]|2:IMPedance?

Parameter description:

<Mode> discrete data. External input impedance, with the values listed below:

OHM50: 50 Ω,

OHM600: 600Ω.

OHM1M 1MΩ

Example: :EXTernal:IMPedance OHM600 Set the impedance of Ext 1 to 600Ω.

Reset state: OHM50

Key path: [Function Generator 1 | 2]-> [Data Source]—>[Ext]-> [Imped]

3.3.22 CORRection Subsystem

The CORRection subsystem command is used to realize functions related to user calibration compensation. This subsystem is used to compensate power and ensure power accuracy.

[:SOURce[1]|2]:CORRection:ACALculation:STATe <State>

Function description: This command is used to set the state of the inter-frequency interpolation switch. When the inter-frequency interpolation switch is on, if the current frequency point is not included in the calibration compensation file, the power compensation values of the left and right frequency points closest to the frequency in the compensation file will be used for interpolation calculation to obtain the power compensation value of the frequency point and use it. When the user calibration compensation switch is ON, this command is valid. Please refer to "[:SOURce[1]|2]:CORRection[:STATe]" for details.

Setting format: [:SOURce[1]|2]:CORRection:ACALculation:STATe ON|OFF|1|0

Query format: [:SOURce[1]|2]:CORRection:ACALculation:STATe?

Parameter description: Boolean data; the values are taken as follows:

<State>	ON | 1: Frequency reference ON,
	OFF | 0: frequency reference OFF.

Example: :CORRection:ACAL:STATe ON Set the interpolation switch between frequencies to ON.

Reset state: OFF

Key path: [Power]—>[User Calibration Compensation] —>[User Calibration Compensation Switch ON] —>[Interpolation Between Frequencies]

[:SOURce]:CORRection:CATalog?

Function description: This command is used to query the name of the stored calibration compensation file under the default calibration compensation file storage path. The default path is "SgData/user/Pltc".

Query format: [:SOURce]:CORRection:CATalog?

Description of returned value: returns all file names with the extension ".plt" in the default path, marked with ";" For segmentation.

Example: :CORRection:CATalog?

Key path: None

Reset state: None

Description: For query only.

[:SOURce[1]|2]:CORRection:EXECute

Function description: This instruction is used to start a calibration process. Before starting calibration, it is necessary to ensure that a calibration compensation file has been selected, and the frequency value in the calibration file is not empty, otherwise a prompt of "user power flatness compensation file is empty" will be generated; After the calibration is started, the signal source will be connected to the power meter in the specified way and calibrated, so make sure that the power meter is connected before calibration, otherwise the calibration process cannot be carried out. This command is for setting only. When the instrument executes this instruction, if the connection mode of the power meter is GPIB, the GPIB of the instrument will automatically switch to the control mode. At this time, it is impossible to remotely control the instrument through GPIB. When the power calibration is completed, the GPIB interface of the instrument will automatically switch back to the listening and speaking mode, which can be remotely controlled. If the connection mode of power meter is not GPIB, the programmed control function of GPIB interface will not be affected.

Setting format: [:SOURce[1]|2]:CORRection:EXECute

Example: :CORRection:EXECute. Start calibration

Reset state: None

Key path: [Power]—>[User Calibration Compensation] —>[Edit Calibration Compensation File] —>[Start Calibration]

[:SOURce]:CORRection:FILE:DELete <FileName>

Function description: This instruction is used to delete the specified calibration compensation file. The parameter is the compensation file name to be deleted, and the file name does not need to contain the path name. The default path is "SgData/user/Pltc/", and the file name needs to contain the file extension ".plt". If the specified file does not exist, an error “file does not exist” will be generated. This command is for setting only.

Setting format: [:SOURce]:CORRection:FILE:DELete “FileName”

Parameter description:

<FileName> Character string type parameter, compensation file name

Example: :CORRection:FILE:Delete “text.plt”. Delete the compensation file ”text.plt” in the path SgData/user/Pltc.

Reset state: None

Key path: None

Description: for setting only.

[:SOURce[1]|2]:CORRection:FLATness:CURRent:FREQuency <value >

Function description: This instruction is used to modify or query the frequency value of the currently specified line in the compensation file. Please refer to the instruction “[:SOURce[1]|2]:CORRection:FLATness:INDex”. If the frequency value of the modified current line is the same as that of other lines, the duplicate frequency value will be deleted when duplicate removal and sorting are performed, resulting in one line reduction of compensation file data. When sorting by frequency from small to large, the frequency value of the current line will be modified. Before calling this command, you need to confirm that a compensation file has been selected. Please refer to "● [:SOURce[1]|2]:CORRection:FLATness[:FILE]:SELect" for the command to select the compensation file. Otherwise, a prompt of "user power flatness compensation file is empty" will be generated.

Setting format: [:SOURce[1]|2]:CORRection:FLATness:CURRent:FREQuency <value >

Query format: [:SOURce[1]|2]:CORRection:FLATness:CURRent:FREQuency?

Parameter description:

<value> double type parameter, frequency value

Model	Range
1466C	[6kHz ~45GHz]
1466D	[6kHz~20GHz]
1466E	[6kHz ~33GHz]
1466G	[6kHz ~45GHz]
1466H [6kHz ~ 53GHz].	
1466L	[6kHz ~45GHz]
1466N	[6kHz ~90GHz]
1466P [6kHz ~ 53GHz].	

Example: :CORRection:FLATness:CURRent:FREQuency 1GHz. Modify the frequency of the current line to 1MHz.

Reset state: None

Key path: None

[:SOURce[1]|2]:CORRection:FLATness:CURRent:POWer <value >

Function description: This instruction is used to modify or query the power compensation value of the currently specified line in the compensation file. Please refer to the instruction "[:SOURce[1]|2]:CORRection:FLATness:INDex". Before calling this command, you need to confirm that a compensation file has been selected. Please refer to "[:SOURce[1]|2]:CORRection:FLATness[:FILE]:SELect" for the command to select the compensation file. Otherwise, a prompt of "user power flatness compensation file is empty" will be generated.

Setting format: [:SOURce[1]|2]:CORRection:FLATness:CURRent:POWer <value >

Query format: [:SOURce[1]|2]:CORRection:FLATness:CURRent:POWer?

Parameter description:

<value> double type parameter, power value

Range:[-35dB~ 35dB]

Example: :CORRection:FLATness:CURRent:POWer 3dB. Modify the compensation value of the current line to 3dB.

Reset state: None

Key path: None

[:SOURce[1]|2]:CORRection:FLATness:DELete <MODE >

Function description: This instruction is used to delete the data in the currently selected compensation file. When deleting data, it can be divided into deleting the current line and deleting all lines. When deleting the current line, only the data of the currently specified line will be deleted. When all rows of data are deleted, the data in the current compensation file is cleared. Please refer to the instruction "[:SOURce[1]|2]:CORRection:FLATness:INDex" for specifying the current line. Before calling this command, you need to confirm that a compensation file has been selected. Please refer to "[:SOURce[1]|2]:CORRection:FLATness[:FILE]:SELect" for the command to select the compensation file. Otherwise, a prompt of "user power flatness compensation file is empty" will be generated. This command is for setting only.

Setting format: [:SOURce[1]|2]:CORRection:FLATness:DELete CURRent|ALL

Parameter description:

<Mode> Discrete type data, data deletion mode

CURRent: Delete the current line

ALL : Delete all lines

Example: :CORRection:FLATness:DELete ALL. Clear the current compensation file data.

Reset state: None

Key path: [Power]—>[User Calibration Compensation] —>[Edit Calibration Compensation File] —>[Delete]

Description: for setting only.

[:SOURce[1]|2]:CORRection:FLATness[:FILE]:SELect <FileName>

Function description: This instruction is used to select a calibration compensation file. The parameter is a string type, indicating the selected calibration compensation file. It is not necessary to specify a path name in the parameter, but the default path "SgData/user/Pltc/" parameter needs to include the file extension ".plt". If the specified compensation file does not exist, an error message "File name not found" will be generated. You can use the instruction "[:SOURce]:CORRection:CATalog" to obtain the name of the compensation file saved in the instrument. This command is for setting only.

Setting format: [:SOURce[1]|2]:CORRection:FLATness[:FILE]:SELect "filename"

Parameter description:

<FileName> Character string type parameter, name of the compensation file selected

Example: :CORRection:FLATness:SELect “test.plt” selects the compensation file text.plt in the path ”SgData/user/Pltc/”

Reset state: None

Key path: [Power]—>[User Calibration Compensation] —>[Select Calibration Compensation File]

Description: for setting only.

[:SOURce[1]|2]:CORRection:FLATness[:FILE]:STORe <FileName>

Function description: This instruction is used to save the current compensation file data in another specified file. The parameter is a string type, which indicates the saved file name. There is no need to specify a path name in the parameter. The default path "SgData/user/Pltc/" parameter needs to include the file extension ".plt". If the saved file name already exists, the original file will be overwritten. This command is for setting only.

Setting format: [:SOURce[1]|2]:CORRection:FLATness[:FILE]:STORe “filename”

Parameter description:

<FileName> Character string type parameter, name of the compensation file to be saved

Example: :CORRection:FLATness:STORe “test.plt” saves the current compensation data to the file
”SgData/user/Pltc/”.

Reset state: None

Key path: [Power]—>[User Calibration Compensation] —>[Edit Calibration Compensation File] —>[Save As]

Description: for setting only.

[:SOURce[1]|2]:CORRection:FLATness:FREQuency <Val>{,{Val}}

Function description: This instruction is used to set the value of the frequency column in the compensation file. This instruction supports setting the frequency values of multiple lines of the frequency column. When modifying the frequency values of multiple lines, you need to input multiple frequency values in turn, separated by commas. If the number of input parameters a is greater than the number of rows b in the current compensation file, A-B sweep points are inserted in the list, the power compensation values of newly inserted frequency points are the default values, and the frequency values of all points are the input frequency values. The maximum number of parameters entered is 65,535, and the minimum number is 1. Before calling this command, you need to confirm that a compensation file has been selected. Please refer to "[:SOURce[1]|2]:CORRection:FLATness[:FILE]:SELect" for the command to select the compensation file. Otherwise, a prompt of "user power flatness compensation file is empty" will be generated.

Setting format: [:SOURce[1]|2]:CORRection:FLATness:FREQuency <val >{,{val}}

Query format: [:SOURce[1]|2]:CORRection:FLATness:FREQuency?

Parameter description:

<value> double type parameter, frequency value

Model	Range
1466C	[6kHz ~45GHz]
1466D	[6kHz~20GHz]
1466E	[6kHz ~33GHz]
1466G	[6kHz ~45GHz]
1466H [6kHz ~ 53GHz].	
1466L	[6kHz ~45GHz]
1466N	[6kHz ~90GHz]
1466P [6kHz ~ 53GHz].	

Example: :CORRection:FLATness:FREQuency 1GHz,2GH,3GHz Set the power compensation values to 1GHz, 2GHz, and 3GHz in turn.

Reset state: None

Key path: None

[:SOURce[1]|2]:CORRection:FLATness:INDex <Val>

Function description: This instruction is used to set the current line of the compensation file. The frequency at which the specified current line user modifies the current line is the power compensation value, which can also be used to delete the current line data.

Setting format: [:SOURce[1]|2]:CORRection:FLATness:INDex <val>

Query format: [:SOURce[1]|2]:CORRection:FLATness:INDex?

Parameter description:

<value> Integer data, index value

Range [0~current number of compensation points-1]

Example: :CORRection:FLATness:INDex 5 Set the data in line 6 with index 5 as the currently selected line

Reset state: None

Key path: None

[:SOURce[1]|2]:CORRection:FLATness:PAIR <FreqVal>,<PowerVal>

Function description: This command is used to add or set compensation data of a specified frequency. When the added FreqVal value already exists in the list, the power offset value corresponding to this frequency point is set as the PowerVal value. If the FreqVal value does not exist in the list, the frequency point is inserted in the list and the power offset value corresponding to this frequency point is set as PoweVal.

Setting format: [:SOURce[1]|2]:CORRection:FLATness:PAIR <FreqVal>,<PowerVal>

Parameter description:

< value > double data pair, where FreqVal is the frequency value and PowerVal is the power offset value.

FreqVal range : [6kHz~Max frequency value],

PowerVal range : [-35dB,35dB]

Example: :CORRection:FLATness:PAIR 1GHz,3dB Set the compensation value of 1GHz power to 3dB

Reset state: None

Key path: None

Description: for setting only.

[:SOURce[1]|2]:CORRection:FLATness:POINts?

Function description: This command is used to query the number of frequency points of the current power compensation file. Before calling this command, you need to confirm that a compensation file has been selected. Please refer to "[:SOURce[1]|2]:CORRection:FLATness[:FILE]:SELect" for the command to select the compensation file. Otherwise, a prompt of "user power flatness compensation file is empty" will be generated. The returned value is 0 at this time.

Query format: [:SOURce[1]|2]:CORRection:FLATness:POINts?

Example: :CORRection:FLATness:POINts? Query the number of frequency points of the current power compensation file.

Reset state: 0

Key path: None

Description: For query only.

[:SOURce[1]|2]:CORRection:FLATness:POWer <Val>{,{Val}}

Function description: This instruction is used to set the value of the power compensation column in the compensation file. This instruction supports setting the power compensation values of multiple lines of the power column. When modifying the values of multiple lines, you need to input multiple compensation values in turn, separated by commas. If the number of input parameters a is greater than the number of rows b in the current compensation file, A-B sweep points are inserted in the list, and the value of newly inserted frequency points is 1GHz. Because the power compensation data will be sorted and duplicated by frequency value, the number of rows of compensation data finally generated may be inconsistent with the number of input power compensation points.

The maximum number of parameters entered is 65,535, and the minimum number is 1. Before calling this command, you need to confirm that a compensation file has been selected. Please refer to "[:SOURce[1]|2]:CORRection:FLATness[:FILE]:SELect" for the command to select the compensation file. Otherwise, a prompt of "user power flatness compensation file is empty" will be generated.

Setting format: [:SOURce[1]|2]:CORRection:FLATness:POWer <val >{,{val}}

Query format: [:SOURce[1]|2]:CORRection:FLATness:POWer?

Parameter description:

<value> double type parameter, power compensation value
Range: [-35dB~ 35dB]

Example: :CORRection:FLATness:POWer -3dB,5dB,3dB Set the power compensation values of the first three frequency points to -3dB, 5dB and 3dB in turn.

Reset state: None

Key path: None

[:SOURce[1]|2]:CORRection:PMETer:COMMunicate:MODE <MODE>

Function description: This instruction is used to set the connection mode of the power meter. It is necessary to ensure that the setting mode is consistent with the actual connection mode of the power meter, otherwise the power meter cannot be connected correctly when the calibration is started.

Setting format: [:SOURce[1]|2]:CORRection:PMETer:COMMunicate:MODE LAN|GPIB|USB

Query format: [:SOURce[1]|2]:CORRection:PMETer:COMMunicate:MODE?

Parameter description:

<Mode> Discrete type parameter
LAN: Network
GPIB : GPIB
USB : USB

Example: :CORRection:PMETer:COMMunicate:MODE LAN Set the connection mode of power meter to Network.

Reset state: LAN

Key path: [Power]—>[User Calibration Compensation] —>[Edit Calibration Compensation File] —>[Power Calibration Connection Mode]

[:SOURce[1]|2]:CORRection:PMETer:GPIB:ADDRess <val>

Function description: This instruction is used to set the GPIB address of the power meter, which is valid when the power calibration connection mode is GPIB. The set value should be consistent with the actual GPIB address in the power meter, otherwise the power meter cannot be connected correctly when the calibration is started.

Setting format: [:SOURce[1]|2]:CORRection:PMETer:GPIB:ADDRess <val>

Query format: [:SOURce[1]|2]:CORRection:PMETer:GPIB:ADDRess?

Parameter description:

<val> integer parameter

Range [1~30], it is necessary to ensure that the GPIB address of the power meter is different from the GPIB address of the signal source itself.

Example: :CORRection:PMETer:GPIB:ADDRess 18 Set the GPIB address of power meter to 18

Reset state: None

Key path: [Power]**—**>[User Calibration Compensation] **—**>[GPIB Address]

[:SOURce[1]|2]:CORRection:PMETer:LAN:IP <Address>

Function description: This instruction is used to set the IP address of the power meter, which is valid when the power calibration connection mode is LAN. The set value should be consistent with the actual IP address in the power meter, and ensure that the IP address of the power meter and the IP address of the signal source are different and in the same network segment, otherwise the power meter cannot be connected correctly when the calibration is started.

Setting format: [:SOURce[1]|2]:CORRection:PMETer:LAN:IP “ipstring”

Query format: [:SOURce[1]|2]:CORRection:PMETer:LAN:IP?

Parameter description:

<Address> string type parameter, network IP address expressed in dotted decimal notation

Example: :CORRection:PMETer:LAN:IP “10.42.114.2” Set the IP address of the power meter to

10.42.114.2

Reset state: None

Key path: [Power]**—**>[User Calibration Compensation] **—**>[Edit Calibration Compensation File] **—**>[Power Meter: IP (Port 5025)]

[:SOURce[1]|2]:CORRection:PMETer:TYPE <TYPE>

Function description: This instruction is used to select the power meter type.

Setting format: [:SOURce[1]|2]:CORRection:PMETer:TYPE N191X|243X|2432|ML2437|

RSNRP

Query format: [:SOURce[1]|2]:CORRection:PMETer:TYPE?

Parameter description:

<TYPE> Discrete type parameter

N191X	: Power meter models N1911 and N1914, etc. of Keysight
243X	: Power meter series 2434, 2436, and 2438 of Ceyear
2432	: Power meter 2432 of Ceyear
ML2437	: Power meters of Anritsu
RSNRp	: Power meters of RS

Example: :CORRection:PMETer:TYPE 243X selects Ceyear 243 series power meter

Reset state: None

Key path: [Power]—>[User Calibration Compensation] —>[Edit Calibration Compensation File] —>[Power Meter Type]

[:SOURce[1]|2]:CORRection:PMETer:ZERO:STATe <State>

Function description: This command is used to set the state of the zeroing switch. When it is on, the power meter itself will be calibrated at the beginning of calibration, and then the power meter with set frequency will be compensated after calibration.

Setting format: [:SOURce[1]|2]:CORRection:PMETer:ZERO:STATe ON|OFF|1|0

Query format: [:SOURce[1]|2]:CORRection:PMETer:ZERO:STATe?

Parameter description:

<State> Boolean data, with the values listed below:

ON | 1 : Zeroing On,

OFF | 0 : Zeroing Off.

Example: :CORRection:PMETer:ZERO:STATe ON sets whether the zeroing switch is ON

Reset state: OFF

Key path: [Power]—>[User Calibration Compensation] —>[Edit Calibration Compensation File] —>[Zero ON/OFF]

[:SOURce[1]|2]:CORRection[:STATe] <State>

Function description: This instruction is used to set the user calibration compensation switch, and when it is on, the current power is compensated by using the compensation data in the calibration compensation file. Therefore, when the switch is on, you need to select a compensation file. When it is off, the compensation data in the calibration compensation file is not used.

Setting format: [:SOURce[1]|2]:CORRection:STATe ON|OFF|1|0

Query format: [:SOURce[1]|2]:CORRection:STATe?

Parameter description:

<State> Boolean data, with the values listed below:

ON | 1: User calibration compensation ON,

OFF | 0: User calibration compensation OFF.

Example: :CORRection:STATe ON sets whether user calibration compensation is ON

Reset state: OFF

Key path: [Power]—>[User Calibration Compensation] —>[User Calibration Compensation OFF]

3.3.23 AWGN Subsystem

The AWGN subsystem is used to configure noise parameters and generate noise signals. The instruction of this subsystem will be valid only after the additive Gaussian white noise generation function option (S06) is installed in the instrument, otherwise an error prompt of "undefined command" will be generated.

The commands are used to select the operating modes, including:

[:SOURce[1]|2]:AWGN:BRATe?

Function description: This command is used to query the bit rate of additive noise. When digital modulation is set to ON, this configuration item is visible in the interface.

Query format: [:SOURce[1]|2]:AWGN:BRATe?

Example: :AWGN:BRATe? Query the bit rate of additive noise in Channel A.

Reset state: 0

Key path: [Noise]—> [Add Noise] —>[Bit Rate]

Description: For query only.

[:SOURce[1]|2]:AWGN:BWIDth <value>

Function description: This command is used to set the additive noise bandwidth or pure noise bandwidth, which is related to the currently selected noise mode. When the noise mode is additive noise or continuous wave interference, this command sets the additive noise bandwidth; When the noise mode is pure noise, the instruction is set to pure noise bandwidth. For details about noise mode selection instructions, please refer to“[:SOURce[1]|2]:AWGN:MODE”.

Setting format: [:SOURce[1]|2]:AWGN:BWIDth <value>

Query format: [:SOURce[1]|2]:AWGN:BWIDth?

Parameter description:

<value> float parameter,

Standard Configuration: [-100MHz, 100MHz].

Option H31|Option H41: [-100MHz, 100MHz].

Option H31|Option H41: [-100MHz, 100MHz].

Option H33|Option H43: [-100MHz, 100MHz].

Example: :AWGN:BWIDth 100MHz Set the noise bandwidth of additive noise and pure noise in channel A to 100MHz.

Reset state: 0

Key path: [Noise]—> [Add Noise| Pure Noise] —>[Noise Bandwidth]

[:SOURce[1]|2]:PWM:BWIDth

Function description: This command is used to query the carrier signal bandwidth of additive noise.

Query format: [:SOURce[1]|2]:AWGN:CARRier:BWIDth?

Example: SOURce2:AWGN:CARRier:BWIDth? Query the carrier signal bandwidth of additive noise in channel B.

Key path: [Noise]—> [Add Noise] —>[Carrier Signal Bandwidth]

Description: For query only.

[:SOURce[1]|2]:AWGN:CNRatio <value>

Function description: This command is used to set the signal-to-noise ratio of additive noise or continuous wave interference, which is related to the currently selected noise mode. When the noise mode is additive noise or pure noise, this command sets the signal-to-noise ratio of additive noise; When the noise mode is continuous wave interference, the instruction is set to the signal-to-noise ratio of continuous wave interference. For details about noise mode selection instructions, please refer to“[:SOURce[1]|2]:AWGN:MODE”.

Setting format: [:SOURce[1]|2]:AWGN:CNRatio <value>

Query format: [:SOURce[1]|2]:AWGN:CNRatio?

Parameter description:

<value> float parameter,

Range:[-50dB, 45dB]

Example: :AWGN:CNRatio 15dB Set the signal-to-noise ratio of additive noise and continuous wave interference in channel A to 15dB.

Reset state: 0

Key path: [Noise]—> [Add Noise | Continuous Wave Interference] —>[Signal-to-Noise Ratio]

[:SOURce[1]|2]:AWGN:ENRatio <value>

Function description: This command is used to set the Eb/N0 value of additive noise.

Setting format: [:SOURce[1]|2]:AWGN:ENRatio <value>

Query format: [:SOURce[1]|2]:AWGN:ENRatio?

Parameter description:

<value> float parameter,
Range:[-100dB, 100dB]

Example: :AWGN:ENRatio 15dB Set the Eb/N0 value of additive noise in channel A to 15dB.

Reset state: 0

Key path: [Noise]—> [Add Noise] —>[Noise Power Calculation Method Eb/N0]—>[Eb/N0]

[:SOURce[1]|2]:AWGN:FREQuency:OFFSet <value>

Function description: This command is used to set the target CW frequency offset value in CW jamming mode.

Setting format: [:SOURce[1]|2]:AWGN:FREQuency:OFFSet <value>

Query format: [:SOURce[1]|2]:AWGN:FREQuency:OFFSet?

Parameter description:

<value> float parameter,
Range:
Standard Configuration: [-100MHz, 100MHz].
Option H31|Option H41: [-100MHz, 100MHz].
Option H31|Option H41: [-100MHz, 100MHz].
Option H33|Option H43: [-100MHz, 100MHz].

Example: :AWGN:FREQuency:OFFSet 50MHz Set the frequency offset of the target continuous wave under the interference of channel A continuous wave to 50MHz.

Reset state: 0

Key path: [Noise]—> [Continuous Wave Interference] —> [Target Continuous Wave Frequency Offset]

[:SOURce[1]|2]:AWGN:MODE <mode>

Function description: This command is used to set the noise mode, which can be divided into Add Noise, Pure Noise, and Continuous Wave Interference.

Setting format: [:SOURce[1]|2]:AWGN:MODE ADDNoise|PURenoise|CWDisturb

Query format: [:SOURce[1]|2]:AWGN:MODE?

Parameter description:

<mode> Discrete type parameter

ADDNoise	:Add Noise
PURenoise	: Pure Noise
CWDisturb	:CW Interferer

Example: :AWGN:MODE PUR Set the noise mode of Channel A to Pure Noise.

Reset state: ADDNoise

Key path: [Noise]—> [Basic Settings]—> [Mode Selection]

[:SOURce[1]|2]:AWGN:POWer:MODE <mode>

Function description: This command is used to set the noise power calculation method. The setting of this instruction is valid when the noise mode is selected as Add Noise and digital modulation is on. There are two ways to calculate noise power: CN and EN.

Setting format: [:SOURce[1]|2]:AWGN:POWer:MODE CN|EN

Query format: [:SOURce[1]|2]:AWGN:POWer:MODE?

Parameter description:

<mode> Discrete type parameter

CN: Signal-noise ratio

EN : Eb/N0

Example: :AWGN:POWer:MODE CN Set the calculation method of Channel A noise power as signal-to-noise ratio.

Reset state: CN

Key path: [Noise]—> [Mode Selection Add Noise] —>[Add Noise] —>[Noise Power Calculation Method]

[:SOURce[1]|2]:AWGN[:STATe] <State>

Function description: This command is used to set noise ON/Off

Setting format: [:SOURce[1]|2]:AWGN[:STATe] ON|OFF|1|0

Query format: [:SOURce[1]|2]:AWGN[:STATe]?

Parameter description:

<State> Boolean parameter

ON|1: noise ON

OFF|0 : noise OFF

Example: :AWGN:STATe ON Set Channel A noise switch to ON.

Reset state: 0

Key path: [Noise]—>[Basic Settings] —>[Noise ON/OFF]

3.3.24 FHOPping Subsystem

The frequency hopping subsystem is used to control frequency hopping signals, which is valid only after the instrument features the frequency hopping signal generation function (S09) is installed in the instrument, otherwise an error prompt of "Undefined command" will be generated.

The commands are used to select the operating modes, including:

[:SOURce[1]|2]:RADio:FHOPping:DATA[:SOURce] <MODE>

Function description: This command sets the selection of data sources for frequency hopping signals, including 11 data sources of PN9, PN11, PN15, PN16, PN20, PN21, PN23, ZERO, ONE, PATTern and PRAM.

Setting format: [:SOURce[1]|2]:RADio:FHOPping:DATA[:SOURce] PN9|PN11|PN15|PN16|PN20|PN21|PN23|ZERO|ONE|PATTern|PRAM

Query format: [:SOURce[1]|2]:RADio:FHOPping:DATA[:SOURce]?

Parameter description:

<Mode> discrete data. Please refer to the setting command format for data source type of frequency hopping signal.

Example: RADio:FHOPping:DATA PATTern Set the frequency hopping signal data source to Custom Sequence.

Reset state: PN9

Key path: [Baseband]-> [Frequency Hopping]-> [Basic Settings]-> [Data Source Selection]

[:SOURce[1]|2]:RADio:FHOPping:DATA[:SOURce]:FILes <MODE>

Function description: This command is used to query the available code stream files in the instrument. This command returns a list of all file names with the extension ".src" stored in the path /SgData/user/FreqHop/, and the file names are separated by ",".

Query format: [:SOURce[1]|2]:RADio:FHOPping:DATA[:SOURce]:FILes?

Example: RADio:FHOPping:DATA:FILes? Return all file names with the extension “.src” in the path /SgData/user/FreqHop/

Key path: None

[:SOURce[1]|2]:RADio:FHOPping:DATA[:SOURce]:PATTern <Val>,<BitCount>

Function description: This command is used to configure the sequence value of a custom sequence. Settings can be made in different decimal systems, but shall always be returned in hexadecimal when queried. The command contains two parameters. The first parameter Val is unsigned 64-bit shaped data, supporting binary, decimal and hexadecimal data and indicating the set PATTern value, which is displayed in binary form in the instrument software interface. The second parameter, BitCount, is shaped signed data, indicating the number of bits when the setting data is displayed in binary form. If the number of bits set for displaying Val as binary data does not match the number of bits specified in BitCount, automatic truncation or zeroing will be performed, with the former starting from the low position, and the latter performed at the high position.

Setting format: [:SOURce[1]|2]:RADio:FHOPping:DATA[:SOURce]:PATTern <Val>,<BitCount>

Query format: [:SOURce[1]|2]:RADio:FHOPping:DATA[:SOURce]:PATTern?

Parameter description:

< Val> Unsigned integer data, with the highest bit processed as 1 when set to a negative number.

<BitCount> Integer data, value range: [1,64]

Example: :RADio:FHOPping:DATA:PATT #HF,4 Set the user-defined sequence data to 1111 (binary), and the query will be returned as #HF,4.

Reset state: 0

Key path: [Baseband]-> [Frequency Hopping]-> [Basic Settings]-> [Data Source Selection Custom Sequence]—> [Custom Sequence]

[:SOURce[1]|2]:RADio:FHOPping:DATA[:SOURce]:PRAM <s>

Function description: When file stream is selected as the frequency hopping data source of the signal generator, this command is used to select the stream file that must be saved in /SgData/user/ FreqHop/ with the extension of ".src". The parameter is a string, containing only the specified file name and not the path name. When no file is selected, it returns to "NULL" when queried.

Setting format: [:SOURce[1]|2]:RADio:FHOPping:DATA[:SOURce]:PRAM <s>

Query format: [:SOURce[1]|2]:RADio:FHOPping:DATA[:SOURce]:PRAM?

Parameter description:

<S > the name of file stream selected contains the extension.

Example: :RADio:FHOPping:DATA:PRAM "Test.src" Select "Test.src" as the file code stream file.

Key path: [Baseband]-> [Frequency Hopping]-> [Basic Settings]-> [Data Source Selection Code Stream] **—**> [Select Code Stream File]

[:SOURce[1]|2]:RADio:FHOPping:DUTY <Val>

Function description: This instruction sets the duty cycle of frequency hopping signal.

Setting format: [:SOURce[1]|2]:RADio:FHOPping:DUTY <val>

Query format: [:SOURce[1]|2]:RADio:FHOPping:DUTY?

Parameter description:

<Val> float parameter, rounded to the nearest hundredth.
Range: [0,100.00]

Example: :RADio:FHOPping:DUTY 63.65 Set the duty cycle of frequency hopping signal to 63.65.

Reset state: 100.00

Key path: [Baseband]-> [Frequency Hopping]-> [Basic Settings]-> [Frequency Hopping Signal Duty Cycle]

[:SOURce[1]|2]:RADio:FHOPping:ENCoding[:TYPE] <MODE>

Function description: This instruction sets the frequency hopping signal coding mode, which determines the frequency hopping order in the frequency hopping list; It can be divided into three types: Positive, Negative, and M Sequence. Where, Positive indicates that frequency hopping changes from low sequence number to high sequence number according to the configuration in the frequency hopping list, and Negative indicates that frequency hopping jumps from high sequence number to low sequence number according to the configuration in the frequency hopping list. M Sequence means that the frequency hopping jumps according to the preset M sequence.

Setting format: [:SOURce[1]|2]:RADio:FHOPping:ENCoding[:TYPE] POSitive|NEGative|MSEQuence

Query format: [:SOURce[1]|2]:RADio:FHOPping:ENCoding[:TYPE]?

Parameter description:

<Val> Discrete parameter.

POSitive	: Positive
NEGative	: Negative
MSEQuence	: M sequence

Example: :RADio:FHOPping:ENCoding NEGative Set the frequency hopping coding mode to Negative.

Reset state: POSitive

Key path: [Baseband]-> [Frequency Hopping]-> [Basic Settings]-> [Frequency Hopping Coding Mode]

[:SOURce[1]|2]:RADio:FHOPping:MODulation[:STATe] <State>

Function description: This command sets the modulation switch of frequency hopping signal. When the modulation switch is on, the modulation type selection is effective, and the modulation signal will be added to the frequency hopping signal. When the modulation switch is off, the modulation type selection is invalid, and the frequency hopping signal has no modulation signal.

Setting format: [:SOURce[1]|2]:RADio:FHOPping:MODulation[:STATe] ON|OFF|1|0

Query format: [:SOURce[1]|2]:RADio:FHOPping:MODulation[:STATe]?

Parameter description:

<State> Boolean data, with the values listed below:

ON | 1 : modulation ON,

OFF | 0: modulation OFF.

Example: RADio:FHOPping:MODulation:STATe ON Set the modulation switch to ON.

Reset state: OFF

Key path: [Baseband]-> [Frequency Hopping]-> [Basic Settings]-> [Modulation ON/OFF]

[:SOURce[1]|2]:RADio:FHOPping:MODulation:TYPE <MODE>

Function description: This command sets the modulation type selection of frequency hopping signals.

Setting format: [:SOURce[1]|2]:RADio:FHOPping:MODulation:TYPE ASK|BPSK|QPSK |QPSK45|QEDGe|AQPSk|OQPSk|P4QPsk|P2DBpsk|P4DQpsk|P8D8psk|8PSK|8PEDG|16QAM|16QEDG|32QAM|32QEDG|64QAM|128QAM|256QAM|512QAM|1024QAM|2048QAM|4096QAM|MSK|2FSK|4FSK|8FSK|16FSK|32FSK|64FSK|16APSK|32APSK

Query format: [:SOURce[1]|2]:RADio:FHOPping:MODulation:TYPE?

Parameter description:

<Mode> discrete data. Please refer to the setting command format for data

source type of frequency hopping signal.

Example: RADio:FHOPping:MODulation:TYPE ASK Set the modulation type and select ASK

Reset state: QPSK

Key path: [Baseband]-> [Frequency Hopping]-> [Basic Settings]-> [Modulation Type Selection]

[:SOURce[1]|2]:RADio:FHOPping:RATE <Val>

Function description: This command sets the hopping rateof frequency hopping signals.

Setting format: [:SOURce[1]|2]:RADio:FHOPping:RATE <val>

Query format: [:SOURce[1]|2]:RADio:FHOPping:RATE?

Parameter description:

<int> Integer data, with the values listed below:

Range: [1,10000]

Example: RADio:FHOPping:RATE 5000 Set the frequency hopping rate to 5,000 hops/s.

Reset state: 2000

Key path: [Baseband]-> [Frequency Hopping]-> [Basic Settings]-> [Hopping Rate]

[:SOURce[1]|2]:RADio:FHOPping:SRATe <Val>

Function description: This command sets the symbol rate. The symbol rate is in units of sps, ksps, Msps and Gsps

Setting format: [:SOURce[1]|2]:RADio:FHOPping:SRATe <val>

Query format: [:SOURce[1]|2]:RADio:FHOPping:SRATe?

Parameter description:

<Val> Floating point data, the value range is related to the modulation bandwidth, with the value range as follow:

Standard configuration: [50sps,125Msps]

Optional H31|H41: [50sps,312.5Msps]

Optional H31|H42: [50sps,625Msps]

Optional H31|H43: [50sps,1.25Gsps]

Example: RADio:FHOPping:SRATe 40ksps Set the symbol rate to 40ksps 设置码元速率为 40ksps.

Reset state: 24.300000ksps

Key path: [Baseband]-> [Frequency Hopping]-> [Basic Settings]-> [Symbol Rate]

[:SOURce[1]|2]:RADio:FHOPping:STATe <State>

Function description: This command sets the frequency hopping signal switch, and when the switch is on, the frequency hopping signal is generated.

Setting format: [:SOURce[1]|2]:RADio:FHOPping:STATe ON|OFF|1|0

Query format: [:SOURce[1]|2]:RADio:FHOPping:STATe?

Parameter description:

<State> Boolean data, with the values listed below:

ON | 1 : Frequency hopping ON,

OFF | 0 : Frequency hopping OFF

Example: RADio:FHOPping:STATe ON Set the frequency hopping switch to ON.

Reset state: OFF

Key path: [Baseband]-> [Frequency Hopping]-> [Basic Settings]-> [Hopping ON/OFF]

[:SOURce[1]|2]:RADio:FHOPping:TABLe:APPend

Function description: This command appends a line of default frequency value at the end of the frequency hopping list, and the default frequency value is 1GHz. Because the frequency hopping list sorts and duplicates all the data when it is modified, the position of the appended default data may change due to sorting. At the same time, due to the de-duplication, if there is a point with a frequency value of 1GHz in the source frequency hopping list, only one 1GHz frequency point will be reserved due to the de-duplication. If the added default frequency value and the data in the original frequency hopping list exceed the baseband bandwidth, an error "The difference between the maximum and minimum frequency of the frequency hopping list is beyond the bandwidth range" will occur. If the frequency hopping switch is on at this time, it will be automatically set to off after an error occurs.

Setting format: [:SOURce[1]|2]:RADio:FHOPping:TABLe:APPend

Example: RADio:FHOPping:TABLe:APPend Add a line of frequency points with default frequency of 1GHz to the frequency hopping list.

Reset state: None

Key path: None

[:SOURce[1]|2]:RADio:FHOPping:TABLe:CLEar

Function description: This command clears the data of the frequency hopping list and sets the frequency hopping list to be empty. If the frequency hopping switch is on, the frequency hopping signal disappears after clearing.

Setting format: [:SOURce[1]|2]:RADio:FHOPping:TABLe:CLEar

Example: RADio:FHOPping:TABLe:CLEar Clear the data of the frequency hopping list.

Reset state: None

Key path: None

[:SOURce[1]|2]:RADio:FHOPping:TABLe:COUNts

Function description: This command is used to query the number of lines in the frequency hopping list. If the frequency hopping list is empty, the return value is +0.

Query format: [:SOURce[1]|2]:RADio:FHOPping:TABLe:COUNts?

Example: RADio:FHOPping:TABLe:COUNts? Query the number of data lines in the current frequency hopping list.

Reset state: None

Key path: None

[:SOURce[1]|2]:RADio:FHOPping:TABLe:DELete <Index>

Function description: This command is used to delete the data of a specified line in the frequency hopping list. The parameter represents the specified line sequence, and the value starts from 0. If the set parameter is greater than the serial number in the frequency hopping list, a "numerical data error" will be generated.

Setting format: [:SOURce[1]|2]:RADio:FHOPping:TABLe:DELete <Index>

Example: RADio:FHOPping:TABLe:DELete 1 Delete the frequency data with serial number 1 in the frequency hopping list.

Reset state: None

Key path: None

[:SOURce[1]|2]:RADio:FHOPping:TABLe:FREQuency <Val>{,{Val}}

Function description: This command is used to configure the frequency value of each line in the frequency hopping list. If the user needs to set a different frequency value, it must assign a corresponding frequency value to each frequency point in the list. It is just required to enter the frequency value of the list sweep point in turn, separated by commas. If the number of parameters a input by the user is less than the number of points b in the current list, only the frequency values of the first a points in the list are modified; If the number of input parameters a is greater than the number of list points b, A-B sweep points are inserted into the list, and the frequency values of all points are the input frequency values. The maximum number of parameters entered is 1024, and the minimum number is 1. When querying, the frequency values of all points are returned in turn, and the different frequency values are separated by commas. It should be noted that the difference between the maximum and minimum values of all set frequency points cannot be greater than the bandwidth value of the current baseband.

Setting format: [:SOURce[1]|2]:RADio:FHOPping:TABLe:FREQuency <val>{,{val}}

Query format: [:SOURce[1]|2]:RADio:FHOPping:TABLe:FREQuency?

Parameter description:

<Val> float data, multi-parameter frequency values separated by commas.

Example: RADio:FHOPping:TABle:FREQuency 1.01GHz,1.02GHz Set the first two frequency values in the frequency hopping list to 1.1 Ghz and 1.02 GHz.

Reset state: None

Key path: None

3.3.25 RADar Subsystem

The RADar subsystem is used to configure radar parameters and generate radar signals. The instruction of this subsystem will be valid only after the radar signal simulation function option is installed in the instrument (S50), otherwise an error message of "undefined command" will be generated.

The commands are used to select the operating modes, including:

[:SOURce]:RADar:CHANnel:SELect <MODE>

Function description: Set the output channel of radar signal. When the signal generator has two channels, this command is valid. When the signal generator has only one channel, this command cannot be set. The default is channel A..

Setting format: [:SOURce]:RADar:CHANnel:SELect ACHannel|BCHannel

Query format: [:SOURce]:RADar:CHANnel:SELect?

Parameter description:

<Mode> discrete data. Values are taken as follows:

ACHannel	: Channel A
BCHannel	: Channel B

Example: RADar:CHANnel:SELect BCH Set the playback channel of radar signal as B channel.

Reset state: ACHannel

Key path: [Baseband]—>[Radar Signal Simulation] —>[Pulse Sequence Setting]—>[Output Channel Selection]

[:SOURce]:RADar:PULSe[1]|2--32766:DELay[:TIME] <val>

Function description: Set the pulse delay time of the specified line in the sequence, and the set line is specified by the digital suffix after PULSe.

Setting format: [:SOURce]:RADar:PULSe[1]|2--32766:DELay[:TIME] <val>

Query format: [:SOURce]:RADar:PULSe[1]|2--32766:DELay[:TIME]?

Parameter description:

<Val>val> Float data. Values are taken as follows:

0[0,20ms]

Example: RADar:PULSe2:DELay 10us Set the delay time of the pulse in the second line of the sequence to 10us.

Reset state: 0

Key path: [Baseband]—>[Radar Signal Simulation] —>[Pulse X]—>[Pulse Shape]—>[Delay

Time]

[:SOURce]:RADar:PULSe[1]|2--32766:FALL[:TIME] <val>

Function description: Set the fall edge time of the pulse in the specified line in the sequence. The set line is specified by the digital suffix after pulse. When the pulse envelope is non-rectangular, the execution is effective. For details about setting the pulse envelope shape, please refer to "[:SOURce]:RADar:PULSe[1]|2--32766:SHAPe".

Setting format: [:SOURce]:RADar:PULSe[1]|2--32766:FALL[:TIME] <val>

Query format: [:SOURce]:RADar:PULSe[1]|2--32766:FALL[:TIME]?

Parameter description:

<Val>val> Float data. Values are taken as follows:

10us[10ns,100us]

Example: RADar:PULSe2:FALL 10us Set the fall edge time of the pulse in the second line of the sequence to
10us.

Reset state: 10us

Key path: [Baseband]—>[Radar Signal Simulation] —>[Pulse X]—>[Pulse Shape Non-rectangle]—>[Fall
Time]

[:SOURce]:RADar:PULSe[1]|2--32766:HOPPing:CLEar

Function description: Clear the period list of the repetition frequency mode. Please refer to "[:SOURce]:RADar:PULSe[1]|2--32766:REPetition[:TYPE]" for setting of pulse repetition frequency modes.

Setting format: [:SOURce]:RADar:PULSe[1]|2--32766:HOPPing:CLEar

Example: RADar:PULSe2:HOPPing:CLEar Clear the hopping period list of pulse 2.

Reset state: None

Key path: Not available

Description: for setting only.

[:SOURce]:RADar:PULSe[1]|2--32766:HOPPing:DELete <val>

Function description: Delete the data of a specified line in the period list of repetition frequency mode. Please refer to "[:SOURce]:RADar:PULSe[1]|2--32766:REPetition[:TYPE]" for setting of pulse repetition frequency modes.

Setting format: [:SOURce]:RADar:PULSe[1]|2--32766:HOPPing:DELete <val>

Parameter description:

<val> integer data. Values are taken as follows:

1[1 , number of cycles]

Example: RADar:PULSe2:HOPPing:DELete 1 Delete the first period value in the pulse hoppng period of the second line in the sequence.

Reset state: None

Key path: Not available

Description: for setting only.

[:SOURce]:RADar:PULSe[1]|2--32766:HOPPing:PERiod <val>{<val>……}

Function description: Set the period value when the repetition frequency mode is frequency hopping, and the parameters are multi-parameter values, indicating the set period value of each frequency hopping pulse, and the number of parameters cannot be greater than the number of repetition frequencies. Please refer to "[:SOURce]:RADar:PULSe[1]|2--32766:REPetition[:TYPE]" for setting of pulse repetition frequency modes

Setting format: [:SOURce]:RADar:PULSe[1]|2--32766:HOPPing:PERiod

Parameter description:

<Val>val> Float type. Values are taken as follows:

[Pulse Width +Rise Time + Fall Time +10ns,40ms]

Example: RADar:PULSe2:HOPPing:PERiod 1ms,2ms Set the pulse frequency hopping period of the second line in the sequence to 1ms and 2 ms

Reset state: None

Key path: Not available

Description: for setting only.

[:SOURce]:RADar:PULSe[1]|2--32766:JITTered:PERCent <val>

Function description: Set the repetition frequency mode to Jitter Percent. Please refer to "[:SOURce]:RADar:PULSe[1]|2--32766:REPetition[:TYPE]" for setting of pulse repetition frequency modes

Setting format: [:SOURce]:RADar:PULSe[1]|2--32766:JITTered:PERCent <val>

Query format: [:SOURce]:RADar:PULSe[1]|2--32766:JITTered:PERCent?

Parameter description:

<Val>val> Float data. Values are taken as follows:

[0.50]

Example: RADar:PULSe2:JITTered:PERCent 15 Set the jitter percent of the pulses in the second line of the sequence to 15.

Reset state: 2

Key path: [Baseband]—>[Radar Signal Simulation] —>[Pulse X]—>[Intra-pulse Modulation]—>[Repetition Frequency Mode
Jitter]—>[Jitter Quantity]

[:SOURce]:RADar:PULSe[1]|2--32766:MODulation:BARKer:TYPE <val>

Function description: Set the sequence length of Barker code when modulation mode is Barker code. Please refer to [:SOURce]:RADar:PULSe[1]|2- -32766:MODulation[:TYPE]" for setting pulse modulation types.

Setting format: [:SOURce]:RADar:PULSe[1]|2--32766:MODulation:BARKer:TYPE <val>

Query format: [:SOURce]:RADar:PULSe[1]|2--32766: MODulation:BARKer:TYPE?

Parameter description:

<val> discrete data. Values are taken as follows:
2 , 3, 4 , 5, 7, 11 , 13

Example: RADar:PULSe2:MODulation:BARKer:TYPE 11 Set the pulse in the second line of the sequence.
The sequence length of Barker code is 11.

Reset state: 2

Key path: [Baseband]—>[Radar Signal Simulation] —>[Pulse X]—>[Intra-pulse Modulation]—>[Modulation Mode
Barker Code]—>[Sequence Length]

[:SOURce]:RADar:PULSe[1]|2--32766:MODulation[:FREQuency]:BWIDth <val>

Function description: Set the FM bandwidth when the modulation mode is linear FM, triangular FM and nonlinear FM. Please refer to [:SOURce]:RADar:PULSe[1]|2- -32766:MODulation[:TYPE]" for setting pulse modulation types.

Setting format:
[:SOURce]:RADar:PULSe[1]|2--32766:MODulation[:FREQuency]:BWIDth <val>

Query format: [:SOURce]:RADar:PULSe[1]|2--32766: MODulation[:FREQuency]: BWIDth?

Parameter description:

<Val>val> Float data. Values are taken as follows:
0[0,200MHz]

Example: RADar:PULSe2:MODulation:BWIDth 150MHz Set the modulation bandwidth of the pulse in the second line of the sequence
To 150 MHz.

Reset state: 10MHz

Key path: [Baseband]—>[Radar Signal Simulation] —>[Pulse X]—>[Intra-pulse Modulation]—>[FM Bandwidth]

[:SOURce]:RADar:PULSe[1]|2--32766:MODulation:LFM:TYPE <val>

Function description: Set the FM direction when the modulation mode is LFM. Please refer to [:SOURce]:RADar:PULSe[1]|2--32766:MODulation[:TYPE]" for setting pulse modulation types.

Setting format: [:SOURce]:RADar:PULSe[1]|2--32766:MODulation:LFM:TYPE POSitive|NEGative

Query format: [:SOURce]:RADar:PULSe[1]|2--32766:MODulation:LFM:TYPE?

Parameter description:

<val> discrete data. Values are taken as follows:

POSitive:	for upward linear frequency modulation
NEGative:	for downward linear frequency modulation

Example: RADar:PULSe2:MODulation:LFM:TYPE NEGative Set the LFM direction of the pulse in the second line of the sequence to downward LFM.

Reset state: POSitive

Key path: [Baseband]—>[Radar Signal Simulation] —>[Pulse X]—>[Intra-pulse Modulation]—>[Modulation Mode
Linear FM] **—**>[Modulation Mode]

[:SOURce]:RADar:PULSe[1]|2--32766:MODulation:NLFM:SYMMetry <val>

Function description: Set the symmetry when the modulation mode is nonlinear FM. Please refer to [:SOURce]:RADar:PULSe[1]|2--32766:MODulation[:TYPE]" for setting pulse modulation types.

Setting format:
[:SOURce]:RADar:PULSe[1]|2--32766:MODulation:NLFM:SYMMetry ON|OFF|1|0

Query format:
[:SOURce]:RADar:PULSe[1]|2--32766:MODulation:NLFM:SYMMetry?

Parameter description:

<val> discrete data. Values are taken as follows:

ON | 1:	For symmetric
OFF | 0:	For asymmetric

Example: RADar:PULSe2:MODulation:NLFM:SYMM ON Set the non-linear FM symmetry of the pulse in the second line of the sequence
to symmetric

Reset state: ON

Key path: [Baseband]—>[Radar Signal Simulation] —>[Pulse X]—>[Intra-pulse Modulation]—>[Modulation Mode

Non-linear FM]—>[Symmetry]

[:SOURce]:RADar:PULSe[1]|2-32766:MODulation:NLFM:TYPE <val>

Function description: Set the frequency modulation mode when the modulation mode is nonlinear FM. Please refer to [:SOURce]:RADar:PULSe[1]|2--32766:MODulation[:TYPE]" for setting pulse modulation types.

Setting format: [:SOURce]:RADar:PULSe[1]|2-32766:MODulation:NLFM:TYPE TANGent|ARCTangent|COSine|ARCCosine

Query format: [:SOURce]:RADar:PULSe[1]|2-32766:MODulation:NLFM:TYPE?

Parameter description:

<val> discrete data. Values are taken as follows:

TANGent:	For tangent FM
ARCTangent:	For arc tangent FM
COSine:	For cosine FM
ARCCosine:	For arc cosine FM

Example: RADar:PULSe2:MODulation:NLFM:TYPE COSine Set the nonlinear FM type of the pulse in the second line of the sequence to cosine FM.

Reset state: TANGent

Key path: [Baseband]—>[Radar Signal Simulation] —>[Pulse X]—>[Intra-pulse Modulation]—>[Modulation Mode Nonlinear FM]-> [FM mode]

[:SOURce]:RADar:PULSe[1]|2--32766:MODulation:SYMBols <val>

Function description: Set the number of symbols of the FM mode corresponding to the modulation mode of BPSK or QPSK. Please refer to [:SOURce]:RADar:PULSe[1]|2- -32766:MODulation[:TYPE]" for setting pulse modulation types.

Setting format: [:SOURce]:RADar:PULSe[1]|2--32766:MODulation:SYMBols <val>

Query format: [:SOURce]:RADar:PULSe[1]|2--32766:MODulation:SYMBols?

Parameter description:

<val> Integer data. Values are taken as follows:

2[2,2000]

Example: RADar:PULSe2:MODulation:SYMBols 20 Set the number of symbols of the pulse in the second line of the sequence to 20.

Reset state: 2

Key path: [Baseband]—>[Radar Signal Simulation] —>[Pulse X]—>[Intra-pulse Modulation]—>[Modulation Mode BPSK|QPSK]—>[Number of Symbols]

[:SOURce]:RADar:PULSe[1]|2-32766:MODulation:TFM:TYPE <val>

Function description: Sets the Frequency modulation direction when the modulation mode is triangular frequency modulation. Please refer to [:SOURce]:RADar:PULSe[1]|2- -32766:MODulation[:TYPE]” for setting pulse modulation types.

Setting format: [:SOURce]:RADar:PULSe[1]|2-32766:MODulation:TFM:TYPE POSitive|NEGative

Query format: [:SOURce]:RADar:PULSe[1]|2-32766:MODulation:TFM:TYPE?

Parameter description:

<val> discrete data. Values are taken as follows:

POSitive:	For upward triangle
NEGative:	For downward triangle

Example: RADar:PULSe2:MODulation:TFM:TYPE NEGative Set the triangle frequency modulation style of the pulse in the second line of the sequence as downward triangle.

Reset state: POSitive

Key path: [Baseband]—>[Radar Signal Simulation] —>[Pulse X]—>[Intra-pulse Modulation]—>[Modulation Mode
Triangular FM]—>[FM Mode]

[:SOURce]:RADar:PULSe[1]|2--32766:MODulation[:TYPE] <val>

Function description: sets the in-pulse modulation type

Setting format: [:SOURce]:RADar:PULSe[1]|2--32766:MODulation[:TYPE] OFF|LFM|TFM|NLFM|BPSK|QPSK|BARKer

Query format: [:SOURce]:RADar:PULSe[1]|2--32766:MODulation[:TYPE]?

Parameter description:

<val> discrete data. Values are taken as follows:

OFF:	for no modulation
LFM:	For linear frequency modulation
TFM:	for triangular FM
NLFM:	for nonlinear frequency modulation
BPSK:	for BPSK modulation
QPSK:	for QPSK modulation
BARKer:	for Barker Code modulation

Example: RADar:PULSe2:MODulation LFM Set the modulation style of the pulse in the second line of the sequence to LFM.

Reset state: OFF

Key path: [Baseband]—>[Radar Signal Simulation] —>[Pulse X]—>[Intra-pulse Modulation]—>[Modulation Mode]

[:SOURce]:RADar:PULSe[1]|2--32766:OFFTime <val>

Function description: sets the pulse off time of the specified line in the sequence, and the set line is specified by the digital suffix after PULSe.

Setting format: [:SOURce]:RADar:PULSe[1]|2--32766:OFFTime<val>

Query format: [:SOURce]:RADar:PULSe[1]|2--32766:OFFTime?

Parameter description:

<Val>val> Float data. Values are taken as follows:

[10ns,20ms]

Example: RADar:PULSe2:OFFTime50us set sthe off time of the pulse in the second line of the sequence as

50us.

Reset state: 40ns

Key path: [Baseband]—>[Radar Signal Simulation] —>[Pulse X]—>[Pulse Shape]—>[Off Time]

[:SOURce]:RADar:PULSe[1]|2--32766:ONTime <val>

Function description: sets the pulse width time of the specified line in the sequence, and the set line is specified by the digital suffix after pulse.

Setting format: [:SOURce]:RADar:PULSe[1]|2--32766:ONTime<val>

Query format: [:SOURce]:RADar:PULSe[1]|2--32766:ONTime?

Parameter description:

<Val>val> Float data. Values are taken as follows:

[10ns,20ms]

Example: RADar:PULSe2:ONTime50us Set the pulse width time in the second line of the sequence as

50us.

Reset state: 10ns

Key path: [Baseband]—>[Radar Signal Simulation] —>[Pulse X]—>[Pulse Shape]—>[Pulse Width]

[:SOURce]:RADar:PULSe[1]|2--32766:REPentition[:TYPE] <val>

Function description: Set the repetition frequency method of pulse.

Setting format: [:SOURce]:RADar:PULSe[1]|2--32766:REPentition[:TYPE] FIXed|STAGger|JITTered|SLIDing|HOPPing

Query format: [:SOURce]:RADar:PULSe[1]|2--32766:REPentition [:TYPE]?

Parameter description:

<val> discrete data. Values are taken as follows:

FIXed:	for fixed
STAGger:	for staggered
JITTered:	for Jitter:

SLIDing: for sliding

HOPPing: for hopping

Example: RADar:PULSe2:REPentition STAG Set the pulse repetition frequency mode in the second line of the sequence to stagger.

Reset state: FIXed

Key path: [Baseband]—>[Radar Signal Simulation] —>[Pulse X]—>[Intra-pulse Modulation]—>[Repetition Frequency Mode]

[:SOURce]:RADar:PULSe[1]|2--32766:REPentition:COUNts <val>

Function description: Set the quantity of repetition frequencies of pulse.

Setting format: [:SOURce]:RADar:PULSe[1]|2--32766:REPentition:COUNts <val>

Query format: [:SOURce]:RADar:PULSe[1]|2--32766:REPentition:COUNts?

Parameter description:

<val> integer data. Values are taken as follows:

1[1,100]

Example: RADar:PULSe2:REPentition:COUNts 10 Set the quantity of pulse repetition frequencies in the second line of the sequence to 100

Reset state: 1

Key path: [Baseband]—>[Radar Signal Simulation] —>[Pulse X]—>[Intra-pulse Modulation]—>[Number of Pulses]

[:SOURce]:RADar:PULSe[1]|2--32766:RISE[:TIME] <val>

Function description: Set the fall edge time of the pulse in the specified line in the sequence. The set line is specified by the digital suffix after pulse. When the pulse envelope is non-rectangular, the execution is effective. For details about setting the pulse envelope shape, please refer to "[:SOURce]:RADar:PULSe[1]|2--32766:SHAPe".

Setting format: [:SOURce]:RADar:PULSe[1]|2--32766:RISE[:TIME] <val>

Query format: [:SOURce]:RADar:PULSe[1]|2--32766:RISE[:TIME]?

Parameter description:

<Val>val> Float data. Values are taken as follows:

10us[10ns,100us]

Example: RADar:PULSe2:RISE 20us Set the rising edge time of the pulse in the second line of the sequence to

20us.

Reset state: 10us

Key path: [Baseband]—>[Radar Signal Simulation] —>[Pulse X]—>[Pulse Shape Non-rectangle]—>[Rising

Time]

[:SOURce]:RADar:PULSe[1]|2--32766:SHAPe <val>

Function description: sets the pulse envelope pattern.

Setting format: [:SOURce]:RADar:PULSe[1]|2--32766:SHAPe RECTangle|TRAPezoid|RCOSine|RRCosine|EXPonent

Query format: [:SOURce]:RADar:PULSe[1]|2--32766:SHAPe?

Parameter description:

<val> discrete data. Values are taken as follows:

RECTangle:	for rectangle
TRAPezoid:	for keystone
RCOSine:	for raised cosine
RRCosine:	for root raised cosine
EXPonent:	for exponent

Example: RADar:PULSe2:SHAPe TRAPezoid Set the envelope of the pulse in the second line of the sequence to be keynote.

The shape change

Reset state: RECTangle

Key path: [Baseband]—>[Radar Signal Simulation] —>[Pulse X]—>[Pulse Shape]—>[Envelope Type]

[:SOURce]:RADar:PULSe[1]|2--32766:SLIDing:PERCent <val>

Function description: sets the maximum sliding range when the repetition frequency mode is sliding. Please refer to "[:SOURce]:RADar:PULSe[1]|2--32766:REPetition[:TYPE]" for setting of pulse repetition frequency modes

Setting format: [:SOURce]:RADar:PULSe[1]|2--32766:SLIDing:PERCent <val>

Query format: [:SOURce]:RADar:PULSe[1]|2--32766:SLIDing:PERCent?

Parameter description:

<Val>val> Float data. Values are taken as follows:

[0.50]

Example: RADar:PULSe2:SLIDing:PERCent 15 Set the maximum sliding of the pulses in the second line of the sequence.

ranges from 0 to 15.

Reset state: 2

Key path: [Baseband]—>[Radar Signal Simulation] —>[Pulse X]—>[Intra-pulse Modulation]—>[Repetition Frequency Mode Sliding]

—>[Max. Sliding Range]

[:SOURce]:RADar:PULSe[1]|2--32766:STAGger:PERiod <val>{<val>……}

Function description: Set the period value when the repetition frequency mode is staggered, and the parameters are multi-parameter values, indicating the set period value of each staggered pulse, and the number of parameters cannot be greater than the number of repetition frequencies. Please refer to "[:SOURce]:RADar:PULSe[1]|2--32766:REPetition[:TYPE]" for setting of pulse repetition frequency modes

Setting format: [:SOURce]:RADar:PULSe[1]|2--32766:STAGger:PERiod <val>{<val>……}

Parameter description:

<Val>val> Float type. Values are taken as follows: [Pulse Width +Rise Time + Fall Time +10ns,40ms]

Example: RADar:PULSe2:STAGger:PERiod 1ms,2ms Set the pulse stagger period of the second row in the sequence to 1ms and 2 ms.

Reset state: None

Key path: None

Description: for setting only.

[:SOURce]:RADar:SEQuence:APPend

Function description: adds a line of data to the pulse sequence, only at the end of the sequence, and the added data is the default data.

Setting format: [:SOURce]:RADar:SEQuence:APPend

Example: RADar:SEQuence:APPend add a line of data at the end of the pulse sequence list

Reset state: None

Key path: [Baseband]—>[Radar Signal Simulation] —>[Pulse Sequence Setting]—>[Add]

Description: for setting only.

[:SOURce]:RADar:SEQuence:CATalog?

Function description: This command is used to query the radar configuration files saved in the current instrument. All radar configuration file names in the path /home/ceyear/sgdata/pulgin/radar/user/state/allstate/ will be returned, with the file names separated by commas.

Query format: [:SOURce]:RADar:SEQuence:CATalog?

Example: RADar:SEQuence:CATalog? Query the file name of radar configuration saved in the instrument.

Reset state: None

Key path: Not available

Description: For query only.

[:SOURce]:RADar:SEQuence:CLEar

Function description: clears the data of the last line in the current pulse sequence list.

Setting format: [:SOURce]:RADar:SEQuence:CLEar

Example: RADar:SEQuence:CLEar clears the currently configured pulse sequence list

Reset state: None

Key path: [Baseband]—>[Radar Signal Simulation] —>[Pulse Sequence Setting]—>[Delete All Lines]

Description: for setting only.

[:SOURce]:RADar:SEQuence:CONFigure:LOAD <S>

Function description: This instruction is used to load the specified radar configuration file, and the parameter is a string type. Just specify the file name of the loaded configuration file (and the extension ".radarstate"), without the need to specify the path of the configuration file. The instruction will find the specified file in the default path /home/ceyear/sgdata/pulgin/radar/user/state/allstate/ and load it. If the specified file name does not exist, an error “file does not exist” will be generated.

Setting format: [:SOURce]:RADar:SEQuence:CONFigure:LOAD “FileName”

Example: RADar:SEQuence:CONFigure:LOAD “test.radarstate” loads The file test.radarstate under the path /home/ceyear/SgData/Pulgin/Radar/user/State/AllState/

Reset state: None

Key path: [Baseband]—>[Radar Signal Simulation] —>[Pulse Sequence Setting]—>[Call Configuration File]

Description: for setting only.

[:SOURce]:RADar:SEQuence:CONFigure:STORe <S>

Function description: This instruction is used to save the current radar configuration to a file. The parameter is a string type and specifies the file name (extension ".radarstate") where the current configuration is saved. It is not necessary to specify the saving path of the configuration file, and the instruction will be saved in the default path /home/ceyear/SgData/Pulgin/Radar/user/State/AllState/. If a file with the same name exists, the original configuration file will be replaced.

Setting format: [:SOURce]:RADar:SEQuence:CONFigure:STORe “FileName”

Example: RADar:SEQuence:CONFigure:STORe “test.radarstate” saves the current radar configuration to the file test.radarstate in the path

/home/ceyear/SgData/Pulgin/Radar/user/State/AllState/.

Reset state: None

Key path: [Baseband]—>[Radar Signal Simulation] —>[Pulse Sequence Setting]—>[Save Configuration File]

Description: for setting only.

[:SOURce]:RADar:SEQuence:DELete

Function description: deletes the data of the last line in the current pulse sequence list.

Setting format: [:SOURce]:RADar:SEQuence:DELete

Example: RADar:SEQuence:DELete deletes the data of the last line in the currently configured pulse sequence list

Reset state: None

Key path: [Baseband]—>[Radar Signal Simulation] —>[Pulse Sequence Setting]—>[Delete Last Line]

Description: for setting only.

[:SOURce]:RADar:SEQuence:PULSe:COUNts?

Function description: This instruction is used to query the number of pulses configured in the current radar pulse sequence.

Query format: [:SOURce]:RADar:SEQuence:PULSe:COUNts?

Example: RADar:SEQuence:PULSe:COUNts? Queries the number of pulses added to the current radar pulse sequence.

Reset state: 0

Key path: None

Description: For query only.

[:SOURce]:RADar:STATe <State>

Function description: This command sets the radar radiation signal switch. When the radar radiation signal switch is on, modifying any configuration item will take effect immediately, which will cause the instrument to regenerate radar according to the newly modified configuration item. When the generated radar signal data is large, it will take a long time to generate radar signal data. Therefore, it is recommended to turn off the radar signal radiation switch before modifying any configuration item, and then turn on the radar signal radiation switch after modifying all configuration items.

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

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

Parameter description:

<State>			Boolean data, with the values listed below:
	ON | 1	:	Radar radiation signal On,
	ON | 0	:	Radar radiation signal Off.

Example: RADar:STATe ON Set the radar radiation signal to ON.

Reset state: OFF

Key path: [Baseband]—>[Radar Signal Simulation] —>[Pulse Sequence Setting]—>[Modulation On/Off]

3.3.26 WLAN Subsystem

The WLAN subsystem is used to configure the WLAN parameters to generate WLAN signals; This subsystem instruction will be valid only after the wireless connection signal simulation function option (S21) is installed in the instrument, otherwise an error prompt of "Undefined command" will be generated.

The commands are used to select the operating modes, including:

[:SOURce]:WLAN:ANTenna:COORdinate[:MODE] <MODE>

Function description: Set or query the WLAN antenna mapping coordinate system.

Setting format: [:SOURce]:WLAN:ANTenna:COORdinate[:MODE] CARTesian|CYLindrical

Query format: [:SOURce]:WLAN:ANTenna:COORdinate[:MODE]?

Parameter description:

<Mode> discrete data. Values are taken as follows:

CARTesian	: Cartesian coordinate system
CYLindrical	: Cylindrical coordinate syste

Example: WLAN:ANTenna:COORdinate:MODE CYL Set the antenna mapping coordinate system of WLAN to cylindrical coordinate system

Reset state: CARTesian

Key path: [Baseband]—>[WLAN 802.11]—>[General Configuration]—>[Transmitting Antenna Setting] —>[Mapping Coordinate]

[:SOURce]:WLAN:ANTenna:MODE <MODE>

Function description: Set or query the WLAN antenna types (number).

Setting format: [:SOURce]:WLAN:ANTenna:MODE A1|A2|A3|A4|A5|A6|A7|A8

Query format: [:SOURce]:WLAN:ANTenna:MODE?

Parameter description:

<Mode> discrete data. Values are taken as follows:

A1	:1
A2	:2
A3	:3
A4	:4
A5	:5
A6	:6
A7	:7
A8	:8

Example: WLAN:ANTenna:MODE A5 设置 WLAN 的天线数为 5 Set the number of WLAN antennas to 5

Reset state: A1

Key path: [Baseband]—>[WLAN 802.11]—>[General Configuration]—>[Transmitting Antenna Setting] —>[Antenna Quantity]

[:SOURce]:WLAN:BWIDth:MODE <MODE>

Function description: Set or query the WLAN bandwidth type.

Setting format: [:SOURce]:WLAN:BWIDth:MODE BW20|BW40|BW80|BW160

Query format: [:SOURce]:WLAN:BWIDth:MODE?

Parameter description:

<Mode> discrete data. Values are taken as follows:

BW20	:20MHz
BW40	:80MHz
BW80	:80MHz
BW160	:80MHz

Example: WLAN:BWIDth:MODE BW80 设置 WLAN 的宽带为 80MHz Set the WLAN bandwidth to 80MHz

Reset state: BW20

Key path: [Baseband]—>[WLAN 802.11]—>[General Configuration]—>[Bandwidth]

[:SOURce]:WLAN:CONFigure:LOAD <FileName>

Function description: calls the saved WLAN configuration file. The parameter of this instruction is a string type, indicating the name of the specified configuration file to be called. The parameter only needs to contain the file name (the file name extension is WIFISTATE), and there is no need to specify a path in the parameter. The instrument calls the configuration file under the default path "/home/ceyear/SgData/Plugin/Wifi/user/State/AllState". If the specified configuration file does not exist, it will be generated.

Setting format: [:SOURce]:WLAN:CONFigure:LOAD “FileName”

Parameter description:

<FileName> string type represents the name of the configuration file

Example: WLAN:CONFigure:LAOD “test.WIFISTATE” calls the text.WIFISTATE file under the path/home/ceyear/SgData/Plugin/Wifi/user/State/AllState/

Reset state: None

Key path: [Baseband]—>[WLAN 802.11]—>[General Configuration]—>[Call Configuration File]

Description: for setting only.

[:SOURce]:WLAN:CONFigure:PRESet

Function description: restores the current WLAN configuration to the factory settings.

Setting format: [:SOURce]:WLAN:CONFigure:PRESet

Example: WLAN:CONFigure:PRESet Restore WLAN configuration to factory settings.

Reset state: None

Key path: [Baseband]—>[WLAN 802.11]—>[General Configuration]—>[Restore Factory Setting]

Description: for setting only.

[:SOURce]:WLAN:CONFigure:STORe <FileName>

Function description: saves the current LAN configuration to the file. The parameter of this instruction is a string type, indicating the name of the specified configuration file to be saved. The parameter only needs to contain the file name (the file name extension is WIFISTATE), and there is no need to specify a path in the parameter. The default saving path of the instrument is "/home/ceyear/SgData/Plugin/Wifi/user/State/AllState". If the specified configuration file does not for testing, it will be generated.

Setting format: [:SOURce]:WLAN:CONFigure:STORe "FileName"

Parameter description:

<FileName> string type represents the file name where the configuration file is saved.

Example: WLAN:CONFigure:STORe "test.WIFISTATE" save the current WLAN configuration to the file text.WIFISTATE in the directory /home/ceyear/SgData/Plugin/Wifi/user/State/AllState/.

Reset state: None

Key path: [Baseband]—>[WLAN 802.11]—>[General Configuration]—>[Save Current Setting]

Description: for setting only.

[:SOURce]:WLAN:FBLock:SEQuence:APPend

Function description: Append a line of default value at the end of the current physical frame block configuration list. The physical frame block configuration list can add up to 50 rows of data. When there are already 50 rows of data in the list, calling this instruction will generate an error prompt of "execution error".

Setting format: [:SOURce]:WLAN:FBLock:SEQuence:APPend

Example: WLAN:FBLock:SEQuence:APPend adds a line of default data to the end of the physical frame block configuration list.

Reset state: None

Key path: [Baseband]—>[WLAN 802.11]—>[Physical Frame Block Configuration]—>[Add]

Description: for setting only.

[:SOURce]:WLAN:FBLock:SEQuence:CLEar

Function description: Clear the current physical frame block configuration list.

Setting format: [:SOURce]:WLAN:FBLock:SEQuence:CLEar

Example: WLAN:FBLock:SEQuence:CLEar clears the current physical frame block configuration list

Reset state: None

Key path: [Baseband]—>[WLAN 802.11]—>[Physical Frame Block Configuration]—>[Delete All Lines]

Description: for setting only.

[:SOURce]:WLAN:FBLock:SEQuence:DELete

Function description: deletes the last line of data in the physical frame block configuration list. When the physical frame block configuration list is empty, executing this instruction will produce an error prompt of "execution error".

Setting format: [:SOURce]:WLAN:FBLock:SEQuence:DELete

Example: WLAN:FBLock:SEQuence:Delete deletes the data of the last line of the physical frame block configuration list.

Reset state: None

Key path: [Baseband]—>[WLAN 802.11]—>[Physical Frame Block Configuration]—>[Delete All Lines]

Description: for setting only.

[:SOURce]:WLAN:FBLock:SEQuence:ECOunt?

Function description: Query the total number of frame blocks whose physical frame block status is ON in the current physical frame block configuration list.

Query format: [:SOURce]:WLAN:FBLock:SEQuence:ECOunt?

Example: WLAN:FBLock:SEQuence:ECOunt? Query the total number of frame blocks whose status switch is ON in the physical frame block configuration table.

Reset state: None

Key path: None

Description: For query only.

[:SOURce]:WLAN:FBLock[1]|2--50:STATe <State>

Function description: Set or query the switch status of each physical frame block in the WLAN physical frame block configuration list. Where the digital suffix after FBLock indicates the index line number of the set or queried physical frame block in the physical frame block list.

Setting format: [:SOURce]:WLAN:FBLock[1]|2--50:STATe ON|OFF|1|0

Query format: [:SOURce]:WLAN:FBLock[1]|2--50:STATe?

Parameter description:

<value> discrete data. Values are taken as follows:

ON|1 : State set to ON.

OFF | 0: state OFF

Example: WLAN:FBLock:STATe 0 Set the state of the first row of physical frame blocks in the physical frame block list to off.

Reset state: None

Key path: [Baseband]—>[WLAN 802.11]—>[Physical Frame Block Configuration]—>[List State On/Off]

[:SOURce]:WLAN:FILTer:CLIPping:LEVel <value>

Function description: Set or query the WLAN waveform trimming level. When the waveform modification state is ON, the configured waveform trimming level is valid. Please refer to [:SOURce]:WLAN:FILTer:CLIPping[:STATe]" for details about the]waveform trimming state.

Setting format: [:SOURce]:WLAN:FILTer:CLIPping:LEVel <value>

Query format: [:SOURce]:WLAN:FILTer:CLIPping:LEVel?

Parameter description:

<value> Integer data. Values are taken as follows:

Range: [1,100]

Example: WLAN:FILTer:CLIPping:LEVel 50 Set the waveform trimming level to 50%

Reset state: 100

Key path: [Baseband]—>[WLAN 802.11]—>[General Configuration]—>[Waveform Trimming Settings]—>[Trim Level]

.

[:SOURce]:WLAN:FILTer:CLIPping:MODE <MODE>

Function description: Set or query the WLAN waveform trimming state. When the waveform modification state is ON, the configured waveform trimming mode is valid. Please refer to [:SOURce]:WLAN:FILTer:CLIPping[:STATe]" for details about the]waveform trimming state.

Setting format: [:SOURce]:WLAN:FILTer:CLIPping:MODE VECTor|SCALar

Query format: [:SOURce]:WLAN:FILTer:CLIPping:MODE?

Parameter description:

<Mode> Discrete data. Values are taken as follows:

VECTor : Vector|i+jq|

SCALar : Scalar;|i|,|q|

Example: WLAN:FILTer:CLIPping:MODE SCALar Set the waveform trimming state to Scalar;|i|,|q|

Reset state: VECTor

Key path: [Baseband]—>[WLAN 802.11]—>[General Configuration]—>[Waveform Trimming Settings]**—**>[Trim Mode]

.

[:SOURce]:WLAN:FILTer:CLIPping[:STATe] <State>

Function description: Set or query the WLAN waveform trimming state.

Setting format: [:SOURce]:WLAN:FILTer:CLIPping[:STATe] ON|OFF|1|0

Query format: [:SOURce]:WLAN:FILTer:CLIPping[:STATe]?

Parameter description:

<State> Boolean data. Values are taken as follows:

ON|1	:ON
OFF|0	:OFF

Example: WLAN:FILTer:CLIPping:STATe 1 Set the waveform trimming state to ON.

Reset state: 0

Key path: [Baseband]—>[WLAN 802.11]—>[General Configuration]—>[Waveform Trimming Settings]**—**>[Trim Status]

.

[:SOURce]:WLAN:MARKer[1]|2|3:FBINdex <value>

Function description: Set or query the frame block index values of signal markers 1, 2 and 3. When the marking mode is frame block start, frame start, frame active start, or frame inactive start, the set frame block index values are effective. For detailed marking methds, refer to "[:SOURce]:WLAN:MARKer[1]|2|3:MODE".

Setting format: [:SOURce]:WLAN:MARKer[1]|2|3:FBINdex <value>

Query format: [:SOURce]:WLAN:MARKer[1]|2|3:FBINdex?

Parameter description:

<value> Integer data. Values are taken as follows:

Range: [1, Number of Physical Frames in ON Status], the number of physical frames in ON status.

A column of status switches in the block configuration list.

Example: WLAN:MARKer3:FBINdex 1 Set the frame block index of marker signal 3 to 1.

Reset state: 1

Key path: [Baseband]-> [WLAN 802.11]-> [Marker Configuration]-> [Marking Method Frame Block Start | Frame Start | Frame Active Start | Frame

Inactive Start]-> [Frame Block Index]

[:SOURce]:WLAN:MARKer[1]|2|3:FESHift <value>

Function description: Set or query the falling edge shift of signal markers 1, 2 and 3. When the marking mode is frame active start or frame inactive start style, the configured falling edge shift is valid. For detailed marking methds, refer to "[:SOURce]:WLAN:MARKer[1]|2|3:MODE".

Setting format: [:SOURce]:WLAN:MARKer[1]|2|3:FESHift <value>

Query format: [:SOURce]:WLAN:MARKer[1]|2|3:FESHift?

Parameter description:

<value> Integer data. Values are taken as follows:

Range: [-100000,23119]

Example: WLAN:MARKer2:FESHift 100 Set the falling edge shift of the marker signal 2 to 100.

Reset state: 5

Key path: [Baseband]—>[WLAN 802.11]—>[Marker Configuration]-—>[Marking Method Frame Active Start|Frame Inactive Start]—>[Falling Edige Shift]

[:SOURce]:WLAN:MARKer[1]|2|3:FINDex <value>

Function description: Set or query the frame index values of signal markers 1, 2 and 3. When the marking mode is frame start, frame active start, or frame inactive start, the set frame block index values are effective. For detailed marking methds, refer to "[:SOURce]:WLAN:MARKer[1]|2|3:MODE".

Setting format: [:SOURce]:WLAN:MARKer[1]|2|3:FINDex <value>

Query format: [:SOURce]:WLAN:MARKer[1]|2|3:FINDex?

Parameter description:

<value> Integer data. Values are taken as follows:

Range: [1, PPDU Number of Physical Frame Block Led by Frame Block Undex]

Example: WLAN:MARKer3:FINDex 1 设置标记信号 3 的帧索引为 1

Reset state: 1

Key path: [Baseband]**—**>[WLAN 802.11]**—**>[Marker Configuration]-**—**>[Marking Method Frame Start| Frame Active Start | Frame Inactive Start]—>[Frame Block Index]

[:SOURce]:WLAN:MARKer[1]|2|3:MODE <MODE>

Function description: Set or query the marking methods of marker signals 1, 2, and 3.

Setting format: [:SOURce]:WLAN:MARKer[1]|2|3:MODE

RESTart|FBLock|FRAMe|FAPart|FIPart|PULSe|PATTern|RATio

Query format: [:SOURce]:WLAN:MARKer[1]|2|3:MODE?

Parameter description:

<Mode>	Discrete data. Values are taken as follows:	
	RESTart	: Restart
	FBLock	: Frame block start
	FRAMe	: Frame start
	FAPart	: Frame active start
	FIPart	: Frame inactive start
	PULSe	:Pulse
	PATTern	: Pattern
	RATio	: ON/OFF rati.

Example: WLAN:MARKer:MODE PULSe Set the marking method of signal marker 1 to pulse

Reset state: RESTart

Key path: [Baseband]—>[WLAN 802.11]—>[Marker Configuration]-—>[Marking Method]

[:SOURce]:WLAN:MARKer[1]|2|3:PATTern <Val>,<BitCount>

Function description: Set the marking method to the predefined pattern. Settings can be made in different decimal systems, but shall always be returned in hexadecimal when queried. The command contains two parameters. The first parameter Val is unsigned 64-bit shaped data, supporting binary, decimal and hexadecimal data and indicating the set PATTern value, which is displayed in binary form in the instrument software interface. The second parameter, BitCount, is shaped signed data, indicating the number of bits when the setting data is displayed in binary form. If the number of bits set for displaying Val as binary data does not match the number of bits specified in BitCount, automatic truncation or zeroing will be performed, with the former starting from the low position, and the latter performed at the high position. For marking methods, refer to "[:SOURce]:WLAN:MARKer[1]|2|3:MODE".

Setting format: [:SOURce]:WLAN:MARKer[1]|2|3:PATTern <Val>,<BitCount>

Query format: [:SOURce]:WLAN:MARKer[1]|2|3:PATTern?

Parameter description:

< Val> Unsigned integer data, with the highest bit processed as 1 when set to a negative number.

<BitCount> Integer data, value range: [1,64]

Example: Hexadecimal data example: WLAN:MARKer2:PATT #HF,4
Binary data example: WLAN:MARKer2:PATT #B1111,4
Decimal data example: WLAN:MARKer2:PATT 15,4

Set the predefined value of Marker 2 to 1111 (binary), and the query is returned as #HF,4.

Reset state: 10(binary)

Key path: [Baseband]**—>** [WLAN 802.11] [Marker Configuration]-—>[Predefined Pattern]

[:SOURce]:WLAN:MARKer[1]|2|3:PULSe:DIVider <value>

Function description: Set or query the divisor factor of signal markers 1, 2 and 3. When the marking mode is pulse, the configured divisor factor is valid. For detailed marking methdos, refer to "[:SOURce]:WLAN:MARKer[1]|2|3:MODE"

Setting format: [:SOURce]:WLAN:MARKer[1]|2|3:PULSe:DIVider <value>

Query format: [:SOURce]:WLAN:MARKer[1]|2|3:PULSe:DIVider?

Parameter description:

<value> Integer data. Values are taken as follows:

Range: [2,1024]

Example: WLAN:MARKer3:PULSe:DIVider 20 Set the divisor factor of signal marker 3 to 20.

Reset state: 32

Key path: [Baseband]**—>[WLAN 802.11]—>[Marker Configuration]-—>[Marking Method Pulse]—>[Diviisor Factor]**

[:SOURce]:WLAN:MARKer[1]|2|3:PULSe:FREQuency?

Function description: Query the pulse frequencies of signal marks 1, 2 and 3.

Query format: [:SOURce]:WLAN:MARKer[1]|2|3:PULSe:FREQuency?

Example: WLAN:MARKer3:PULSe:FREQuency? Query the pulse frequency of signal marker 3

Reset state: 1.25MHz

Key path: [Baseband]-> [WLAN 802.11]-> [Marker Configuration]-> [Marking Method Pulse]-> [Pulse Frequency]

Description: For query only.

[:SOURce]:WLAN:MARKer[1]|2|3:RATio:OFFTime <value>

Function description: Set or query the number of sampling points in the closing phase of signal markers 1, 2 and 3. When the marking method is switch ratio, the configured value is valid. For detailed marking methds, refer to "[:SOURce]:WLAN:MARKer[1]|2|3:MODE".

Setting format: [:SOURce]:WLAN:MARKer[1]|2|3:RATio:OFFTime <value>

Query format: [:SOURce]:WLAN:MARKer[1]|2|3:RATio:OFFTime?

Parameter description:

<value> Integer data. Values are taken as follows:
Range: [1,16777215]

Example: WLAN:MARKer3:RATio:OFFTime 40 Set the number of off-phase sampling points of signal marker 3 to 40.

Reset state: 1

Key path: [Baseband]**—>[WLAN 802.11]—>[Marker Configuration]-—>[Marking Method On/Off Ratio] —>[Off-phase Sampling Points]**

[:SOURce]:WLAN:MARKer[1]|2|3:RATio:ONTime <value>

Function description: Set or query the number of sampling points in the Off-phase of signal markers 1, 2 and 3. When the marking method is switch ratio, the configured value is valid. For detailed marking methds, refer to "[:SOURce]:WLAN:MARKer[1]|2|3:MODE".

Setting format: [:SOURce]:WLAN:MARKer[1]|2|3:RATio:ONTime <value>

Query format: [:SOURce]:WLAN:MARKer[1]|2|3:RATio:ONTime?

Parameter description:

<value> Integer data. Values are taken as follows:
Range: [1,16777215]

Example: WLAN:MARKer3:RATio:ONTime 40 Set the number of On-phase sampling points of signal marker 3 to 40.

Reset state: 1

Key path: [Baseband]—>[WLAN 802.11]—>[Marker Configuration]-—>[Marking Method On/Off Ratio] —>[On-phase Sampling Points]

[:SOURce]:WLAN:MARKer[1]|2|3:RESHift <value>

Function description: Set or query the rising edge shift of signal markers 1, 2 and 3. When the marking mode is frame active start or frame inactive start style, the configured rising edge shift is valid. For detailed marking methds, refer to "[:SOURce]:WLAN:MARKer[1]|2|3:MODE".

Setting format: [:SOURce]:WLAN:MARKer[1]|2|3:RESHift <value>

Query format: [:SOURce]:WLAN:MARKer[1]|2|3:RESHift?

Parameter description:

<value> Integer data. Values are taken as follows:
Range: [-23994,100000]

Example: WLAN:MARKer2:RESHift 100 Set the rising edge shift of the marker signal 2 to 100.

Reset state: 120

Key path: [Baseband]**—>[WLAN 802.11]—>[Marker Configuration]-—>[Marking Method Frame Active Start|Frame Inactive Start]—>[Rising Edige Shift]**

[:SOURce]:WLAN:SRATe?

Function description: Query WLAN sampling rate.

Query format: [:SOURce]:WLAN:SRATe?

Example: WLAN:SRATe? Query the sampling rate of WLAN

Reset state: 20MHz

Key path: [Baseband]—>[WLAN 802.11]—>[General Configuration]—>[Sampling Rate]

Description: For query only.

[:SOURce]:WLAN:STATe <State>

Function description: Set or query the WLAN ON/OFF.

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

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

Parameter description:

<State> Boolean data. Values are taken as follows:

ON|1	:ON
OFF|0	:OFF

Example: WLAN:STATe 1 Set WLAN to On

Reset state: 0

Key path: [Baseband]—>[WLAN 802.11]—>[General Configuration]—>[WLAN On/Off]

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 name for the standard I/O function library and its associated specifications. VISA library function is a set of functions that can be easily called. Its core function can control various types of devices without considering the interface type of devices and the use of different I/O interface software. These library functions are used to write the driver program of the instrument and complete the command and data transmission between the computer and the instrument, so as to realize program control of the instrument. By initializing the addressing string ("VISA resource string"), a connection to an instrument with a program port (LAN, USB, GPIB and RS-232, etc.) can be established.

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. The signal generator supports such programming interfaces as GPIB, LAN and RS-232. These interfaces, combined with the VISA library and programming language, allow remote control of the signal generator. At present, Agilent I/O Library provided by Agilent is often used as the underlying I/O Library.

Figure 4.1 takes GPIB interface as an example to show the relationship among programmed control interface, VISA library, programming language and signal generator.

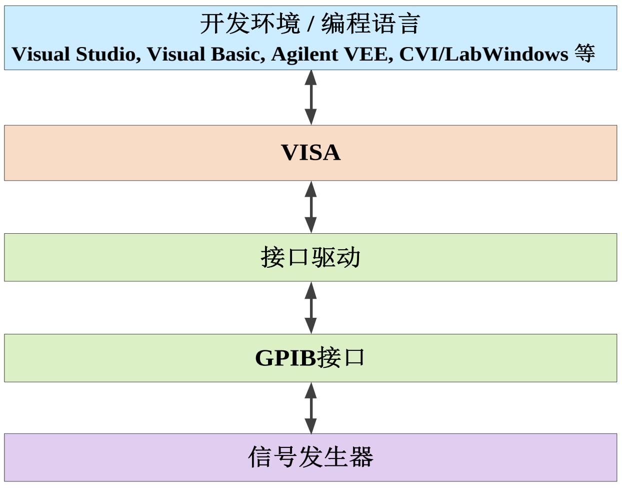


Fig.4.1 Programmable software/hardware layer

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, Windows 7 or Windows 8, Windows 10 operating systems.
- Visual Studio 2010/2012 integrated development environment;
- PCI-GPIB interface card of NI or GPIB interface card of Keysight;
- VISA library of NI or Keysight;
- GPIB card;
- Network card.

4.1.2.2 Files included

To run an example program written in C/C++, you must include the required files in the VC project.

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 Keysight respectively.

4.1.3 Initialize and Set Default State

The program starts by initializing the VISA explorer, and then it is required to open and establish a communication connection between the VISA library and the instrument. The specific steps are as follows:

4.1.3.1 Generate Global Variables

Start by generating global variables that other program modules will call, such as instrument handle variables. The following example programs should contain the following global variables:

```
ViSession      iDevHandle;
ViSession      iDefaultRM;
const char rgcDevResource[VI_FIND_BUFLEN] = “GPIB0::20::INSTR”;
ons tint analyzerTimeout = 5000;
```

Whereas, the constant rgcDevResource indicates the network analyzer descriptor, “GPIB0” indicates the master, and “20” indicates the network analyzer connected with the master. If it is assumed that the network analyzer is connected to the LAN socket, the IP address is "192.168.1.1", and port number is “5025”, then the value of the variable should be:

```
const char rgcDevResource [VI_FIND_BUFLEN] =
“TCPIP::192.168.1.1::5025::SOCKET”;
```

4.1.3.2 Initialize the Controller

```
/***************************************************************************
The following example shows how to open and establish a communication
connection between the VISA library and the instrument (as specified by the instrument
descriptor).
//Initialize the master: open the default resource manager and return the instrument
handle iDevHandle.
***************************************************************************/
void InitController()
{
    ViStatus iStatus;
    iStatus = viOpenDefaultRM(&iDefaultRM);
    iStatus = viOpen(defaultRM, rgcDevResource, VI_NULL, VI_NULL, &
iDevHandle);
}
```

4.1.3.3 Initialize the Network Analyzer

```
/*************************************************************************
The following example shows how to initialize the default state of the instrument and
empty the status register.
/*************************************************************************
void InitDevice()
{
        ViStatus iStatus;
        ViUInt32   uiRetCnt;
        //Reset state register
        iStatus = viWrite(iDevHandle, “*CLS\n”,strlen(“*CLS\n”), & uiRetCnt);
        //Reset the instrument
        iStatus = viWrite(iDevHandle, “*RST\n”, strlen(“*RST\n”), & uiRetCnt);
        //Set the continuous wave frequency of the signal generator to 1GHz.
        iStatus = viWrite(iDevHandle, “freq 1GHz\n”, strlen(“freq 1GHz \n”), & uiRetCnt);
}
```

4.1.4 Send Setting Command

```
/*************************************************************************
The following examples show how to set the point frequency and amplitude of 1466
series signal generator.
/*************************************************************************
void SimpleSettings()
{
        ViStatus    istatus;
        ViUInt32    uiRetCnt;
        char* pcFreqCmd = “FREQ 128MHz\n”;
        char* pcPowCmd = “POW -10dBm\n”;
        //Set the point frequency to 128MHz
        iStatus = viWrite(iDevHandle, pcFreqCmd, strlen(pcFreqCmd), &uiRetCnt);
        //Set the amplitude to -10dBm
        iStatus = viWrite(iDevHandle, pcPowCmd, 1 strlen(pcPowCmd), &uiRetCnt);
}
```

4.1.5 Read the State of Measuring Instrument

```
/*************************************************************************/
The following examples show how to read the set state of the instrument.
/*************************************************************************/
void ReadSettings()
```

```
{
        ViStatus        iStatus;
        ViUInt32   uiRetCnt;
        char rd_Buf_CW[VI_READ_BUFLEN];    // #define VI_READ_BUFLEN 256
        char rd_Buf_LVL[VI_READ_BUFLEN];

        //Query the point frequency
        iStatus = viWrite(iDevHandle, "FREQ:CW?\n", strlen("FREQ:CW?\n"), &
        uiRetCnt);
        Sleep(10);
        iStatus = viRead(iDevHandle, rd_Buf_CW, sizeof(rd_Buf_CW), & uiRetCnt);
        //Query amplitude
        iStatus = viWrite(iDevHandle, "POW?\n", strlen(POW?\n), & uiRetCnt);
        Sleep(10);
        iStatus = viRead(iDevHandle, rd_Buf_LVL,strlen(rd_Buf_LVL), & uiRetCnt);
        //Print debugging information
        printf("frequency    %s", rd_Buf_CW);
        Printf("power %s", rd_Buf_ LVL);
}
```

4.1.6 Command Synchronization

/**/

The following examples illustrate the methods for command synchronization with sweep process.

/**/

```
void SweepSync()
{
        ViStatus iStatus;
        ViInt32    uiRetCnt;
        ViEventType  eType;
        ViEvent  eEvent;
        int iStat;
        char rgcOpcOk [2];

        /*************************************************************************/
        /* The command INITiate[:IMMediate] is used to start single sweep (when
        continuous sweep is OFF, INIT:CONT OFF)*/
        /* Only at the end of single sweep can the next command in the command buffer
        be executed                                  */
        /*************************************************************************/
        iStatus = viWrite(iDevHandle, "INIT:CONT OFF", 13, & uiRetCnt);
        //Method 1 for waiting for the sweep to end: use *WAI
```

```
    iStatus = viWrite(iDevHandle, “ABOR;INIT:IMM;*WAI”, 18, & uiRetCnt);

    //Method 2 for waiting for the sweep to end: use *OPC?
    iStatus = viWrite(iDevHandle, “ABOR;INIT:IMM; *OPC?”, 20, & uiRetCnt);
    //Wait *OPC to return “1”
    iStatus = viRead(iDevHandle, rgcOpcOk, 2, &retCnt);
    //Method 3 for waiting for the sweep to end: use *OPC
    //To use the GPIB service request, set "Disable Auto Serial Poll” to "yes"

    //Enable svice rquest ESR
    iStatus = viWrite(iDevHandle, “*SRE 32”, 7, & uiRetCnt);
    //Set the event enabling position, and end the operation.
    iStatus = viWrite(iDevHandle, “*ESE 1”, 6, & uiRetCnt);
    //Enable the SRQ event
    iStatus = viEnableEvent(iDevHandle, VI_EVENT_SERVICE_REQ, VI_QUEUE,
    VI_NULL);
    //Start sweeping together with OPC
    iStatus = viWrite(iDevHandle, “ABOR;INIT:IMM;*OPC”, 18, & uiRetCnt);
    //Wait for service request
    iStatus = viWaitOnEvent(iDevHandle, VI_EVENT_SERVICE_REQ, 10000, &
    eType, &eEvent)
    iStatus = viReadSTB(iDevHandle, & iStat);
    iStatus = viClose(eEvent); //close event handle
    //Disable SRQ event
    iStatus = viDisableEvent(iDevHandle, VI_EVENT_SERVICE_REQ, VI_QUEUE);
    //Main program continues……
}
```

4.2 Advanced Operation Examples

4.2.1 Network Programmed Control Example

4.2.1.1 Before Using the Examples

In order to be able to use the following example correctly, you must match your host address with the IP address of your signal generator. The two IP addresses must be in the same network segment. For example, if the IP of the host is 192.168.1.168, you need to configure the IP of the signal generator in the form of 192.168.1.X, where X can be 0~167/169~254.

If you use VISA library for network programmed control, you need to install VISA library, such as KeysightIO17.2, NI VISA5, etc.

Note that lower versions of the VISA library do not support the network programmed control.

4.2.1.2 Examples

1) Using VISA library and C++ language to implement network programmed control

```
/****************************************************************
In this example, network remote control is implemented via the VISA library.
Start VC6.0 or VC2013, etc., and add the necessary files. Enter the following code into
your .cpp file
****************************************************************/
    #include "stdafx.h"
    #include <visa.h>
    #include <stdio.h>
    #include <stdlib.h>

    #define  SOURCE_IP_ADDR_A       "192.168.1.199"     //IP address of the signal
    generator
    #define  SOURCE_SOCKET_PORT   5025              //port number of the signal
    generator

    void ShowMsg(PCHAR lpszText)
    {
    #ifdef _UNICODE
        AfxMessageBox((Cstring)lpszText);
    #else
        AfxMessageBox(lpszText);
    #endif
    }

    void SocketTest1(void)
    {
        ViSession   defaultRM;
        ViSession   vi;
        ViStatus   iStatus = 0;
        ViChar          rgcSocket[VI_FIND_BUFLEN];          //for          example:
```

```
"TCPIP0::192.168.1.2::21::SOCKET"
    ViChar rgcBuf[256];
    ViReal64 dFreq;

    iState = viOpenDefaultRM(&defaultRM);
    if (iStatus < VI_SUCCESS)
    {
        ShowMsg("Could not open a session to the VISA Resource Manager!\n");
    }
    else
    {
        sprintf(rgcSocket,   "TCPIP0::%s::%d::SOCKET",   SOURCE_IP_ADDR_A,
SOURCE_SOCKET_PORT);
        iState = viOpen(defaultRM, rgcSocket, VI_NULL, 6000, &vi);
        if (VI_SUCCESS > iStatus)
        {
            ShowMsg("An error occurred opening the session:SOCKET\n");
        }
        else
        {
            viSetAttribute(vi, VI_ATTR_TERMCHAR, '\n');//Set the terminator to LF
            viSetAttribute(vi,  VI_ATTR_TERMCHAR_EN,  VI_TRUE);//Enable  the
terminator
            viSetAttribute(vi,  VI_ATTR_TMO_VALUE,  5000);//Timeout  is  set  to
seconds

            // Query the frequency of channel A
            viPrintf(vi, ":SOURce1:FREQ?\n");
            viScanf(vi, „%t", rgcBuf);
            sscanf(rgcBuf, „%lf", &dFreq);
            sprintf(rgcBuf, "Freq of Channel A is:%lg\n", dFreq);
            ShowMsg(rgcBuf);

            viClose(vi); //Close the device
        }
        viClose(defaultRM); //Close default tasks
    }
}
```

2) Use socket and C++ to implement programmed control

Use socket to implement remote control.

4.2 Advanced Operation Examples

Start VC6.0 or VC2013, and add the necessary files. Enter the following code into your .cpp file

```
****************************************************************/
#include "stdafx.h"
#include <afxsock.h>
#include <stdio.h>
#include <stdlib.h>

#ifdef _UNICODE
#define  SOURCE_IP_ADDR          L"192.168.1.199"      //IP address of the signal
generators
#else
#define  SOURCE_IP_ADDR          "192.168.1.199"       //IP address of the signal
generator
#endif
#define  SOURCE_SOCKET_PORT    5025                    //port number of the signal
generator

void ShowMsg(PCHAR lpszText)
{
#ifdef _UNICODE
      AfxMessageBox((Cstring)lpszText);
#else
      AfxMessageBox(lpszText);
#endif
}

void SocketTest2(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(SOURCE_IP_ADDR, SOURCE_SOCKET_PORT);
/Connect the network port
            if (!iFlag)
            {
                ShowMsg("Connection failed");
            }

            // Turn on the RF output switch of channel A
            sprintf(rgcBuf, "%s\n", "OUTP1:STATe 1\n");
            iBufLen = (int)strlen(rgcBuf);
            iFlag = client.Send(rgcBuf, iBufLen);
            if (!iFlag)
            {
                ShowMsg("Send failed");
            }
            else
            {
                // Query the frequency of channel A
               sprintf(rgcBuf, „%s\n“, „SOURce1:FREQ?\n“);
               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 GPIB Programmed Control Example

4.2.2.1 Before Using the Examples

If you are using a GPIB interface card from Keysight , then you must install the Keysight VISA library correctly. Likewise, if you are using a PCI-GPIB interface card from NI, you must also install the NI-488.2 library correctly.

In this program, it is assumed that only one GPIB card is connected in the system. If there are multiple GPIB cards, GPIB0 indicates the first GPIB card, GPIB1 indicates the second, and so on.

Set the GPIB addresses of the instrument to 13.

4.2.2.2 Use VISA Library and C Language to Implement Settings and Query Function

```
/*******************************************************************
This example is used to query and set the frequency of channel A, and finally restore the
frequency before setting.
Start VC6.0 or VC2010, and add the necessary files. Enter the following code into
your .cpp file
*******************************************************************/
#include "stdafx.h"
#include <visa.h>
#include <stdio.h>
#include <stdlib.h>

#define GPIB_CARD_ID        0        //GPIB card No.
#define MY_GPIB_ADDR     13       //Instrument's GPIB address

void GpibTest1(void)
{
        ViSession defaultRM;
        ViSession vi;
        ViState iState = 0;
```

```
    ViChar rgcBuf[256];
    ViByte cRead[256];
    ViReal64 rgdFreq[2];
    ViUInt32 uiRetCnt;

    sprintf(rgcBuf, "GPIB%d::%d::INSTR", GPIB_CARD_ID, MY_GPIB_ADDR);
    iState = viOpenDefaultRM(&defaultRM); //enable GPIB task
    if (iState)
    {
        ShowMsg ("The task cannot be opened. Please recheck the device and
connect\n").
    }
    else
    {
        iState = viOpen(defaultRM, rgcBuf, VI_NULL, 5000, &vi);
        if (iState)
        {
            ShowMsg ("The devicee cannot be opened. Please recheck the device and
connect\n");
        }
        else
        {
            // Reset signal generator
            viPrintf(vi, “*RST\n”);
            Sleep(10);
            //2) Set the frequency of channel A to 500MHz
            viPrintf(vi, “SOURce1:FREQ    %lfMHz\n”, 500);

            //3) Query the frequency of Channel A, and store it in rgdFreq[1]
            viPrintf(vi, “SOURce1:FREQ?\n”);
            viScanf(vi, "%t", rgcBuf);          //Put the query results into an array
            ShowMsg(rgcBuf);

            sscanf(rgcBuf, “%lf”, &rgdFreq[1]);
            sprintf(rgcBuf, "Frequency of Channel A: %lg\n", rgdFreq[1]);
            ShowMsg(rgcBuf);

            //4) Set the frequency of Channel A to -2dBm
            viPrintf(vi, “SOURce1:POW      %lg\n”,-2);

            viClose(vi); //Close the device
        }
```

```
            viClose(defaultRM); //Close default tasks
      }
}
```

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.

5.1 Error Message

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 error message

1) Error message viewing

View via interface operation:

In case of any error prompt at the bottom of the signal source during use, 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 network analyzer, to see all errors, do the following:

Step 1. Double-click the error display area, and an error message collection window will pop up.

Step 2. The prompt will be displayed in the window.

Step 3: Use the mouse to browse the error message and close the dialog window.

Step 4. Select "Clear historical error" to clear historical error message.

Step 5. Click the Save Data to File button to save the error data.

2) Error message 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 message description

Key error field	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	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 Remote control error message

1) Error message formats and descriptions

In remote control mode, errors are recorded in the error/event queue of the status reporting system, and can be queried with the command "SYSTem:ERRor?". The format is as follows:

"<Error code>, "<Error in error queue>; <Detailed error description>"

Example:

“-190,” data exceeding the maximum range; alreadt set to the maximum value.”

A negative error code defined by the SCPI standard. This type of error is not specified here.

Table 5.2 List of descriptions on instrument feature error messages

Failure code	Error Description
-101	Invalid character Invalid character: There are invalid characters in the command string (command or parameter). For example: FREQ:!ERD 10GHz
-108	Parameter not allowed Parameters are not allowed: the command has too many parameters, or the command without parameters follows the parameters.

	E.g.: *TRG 10
-109	Missing parameter Missing parameters: The command has too few parameters. For example, POW
-112	Program mnemonic too long Command string is too long: A single segment of the command has more than 12 characters. For example: OUTPutROSCillatorSTATe ON
-113	Undefined header Undefined header: signal source receives an unrecognized command. Possible causes: Wrong spelling, or wrong abbreviation of the command, etc. E.g.: FREQ:ALC 1
-114	Header suffix out of range Command suffix out of range: when the set command suffix is within the allowable range, but there is no corresponding configuration item in the instrument. For example, when there are only 10 data in the multi-carrier configuration list, set RADio:ARB:MCAR:CARR53:FREQ 10.
-115	Unexpected number of parameters Wrong number of parameters: for multi-parameter commands, when the number of parameters passed is too small or too large. E.g.: PULM:INT:STAG:DATA 10us
-121	Invalid character in number Invalid characters in the numeric value: there are invalid characters in the numeric parameter. E.g.: FREQ 8
-128	Numeric data not allowed Numeric parameters are not allowed: commands that cannot receive numeric parameters receive a numeric value.
-131	Invalid suffix Invalid suffix: when the set command suffix is within the allowable range and there is this configuration item in the instrument, but it’s not allowed to be set due to certain status.
-150	String data error Character data error: check whether quotation marks need to be added.

	E.g.: MEM:DEL 123.txt (File 123.txt exists) Correct: MEM:CLE "123.txt"
-151	Invalid string data Invalid string data: the set string type data is illegal. E.g.: SYSTem:COMM:LAN:IP "10.10.0."
-161	Invalid block data Invalid block data: Check according to Section 7.7.6 of IEEE 488.2.
-190	Out of range,set max value Data exceeding the max. range: it is detected that the parameter exceeds the latest value, and the signal source has sety the data to the max. value. For example: FREQ 10000GHz
-191	Out of range,set min value Data exceeding the min. range: it is detected that the parameter exceeds the latest value, and the signal source has sety the data to the min. value. For example: FREQ 1Hz
-195	No Channal B Channel B option not installed: when the instrument adopts single channel, the instructions sent begin with SOURce2 or OUTput2. For example: SOURce2:POWer 10
-200	Execution error Execution error: The signal source has received the instruction, but there was an error during execution.
-224	Illegal parameter value Illegal parameter value: a discrete parameter was received, but it is invalid for this command.
-256	File name not found The file name is not found: when the specified file does not exist

2) Error message types

The error event corresponds only to one type of error message, and the error message types are introduced in details below:

- Query error (–499 to –400): indicates the output queue control of the instrument detects the message exchange protocol error specified in Chapter 6 of IEEE 488.2. At this point, the query error bit (bit2) of the event status register is set to 1 (please refer to IEEE 488.2, 6.5 for details). The data cannot be successfully read from the output queue at this time (no kind of error is produced at present).

- 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 to 1(no kind of error is produced at present).
- 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 to 1.
- **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 to 1.

5.2 Method to Obtain After-sales Services

5.2.1 Contact us

If there is a problem with 1466 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:

Free customer service number: 800--8687--041

Technical support: **0532-86889847 86897262**

Fax: **0532-86889056 86897258**

Website: **www.ceyear.com**

Email: **techbb@ceyear.com**

Postal code: **266555**

Address: **No. 98, Xiangjiang Road, Qingdao Economic & Technological Development Zone, Shandong Province, China**

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 that the signal generator has to be returned for repairing, pack it 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.
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 network analyzer 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 consign it as precision instruments.
7) Keep a copy of all shipping documents.

Notice

Precautions on packing the signal generator

Using other materials to pack the signal generator may damage the instrument. Never use polystyrene beads as packing materials because on the one hand, they cannot provide sufficient protection on the instrument, and on the other hand, they can be sucked in to the instrument fan by the static electricity generated, resulting in instrument damage.

Tips

8)

Instrument package and transportation

Please follow carefully the precautions described in “3.1.1.1 Unpacking” of the User's Manual when transporting or handling the instrument (for example, damage occurred during delivery).

Annex

Appendix A Zoom Table of SCPI Classified by Subsystem

Table 1 Zoom table of 1466 SCPI commands

No.	Command	Function
1	*CLS	Universal command
2	*ESE	Universal command
3	*ESR	Universal command
4	*IDN?	Universal command
5	*OPC	Universal command
6	*RCL	Universal command
7	*RST	Universal command
8	*SAV	Universal command
9	*SRE	Universal command
1	*STB	Universal command
1	*TRG	Universal command
1	*TST	Universal command
1	:OUTPut:ALL[:STATe]	Set the RF ON/OFF for all channels
1	:OUTPut[1]|2:BLANking[:STATe](?)	Set the RF blanking ON/OFF

1	:OUTPut[:STATe](?)	Set RF output to ON/OFF state
1	:OUTPut:MODulation:ALL[:STATe]	Set all-channel modulation ON/OFF
1	:OUTPut[1]|2:MODulation[:STATe](?)	Set each-channel modulation ON/OFF
1	:OUTPut[1]|2[:STATe](?)	Set each-channel RF ON/OFF
1	[:SOURce[1]|2]:FREQuency:CENTer(?)	Set the step sweep center frequency
2	[:SOURce[1]|2]:FREQuency[:CW|FIXed](?)	Set the output frequency of the signal Generator
2	[:SOURce[1]|2]:FREQuency[:CW|FIXed]:AUTO(?)	Set the frequency following ON/OFF
2	[:SOURce[1]|2]:FREQuency:MODE(?)	Set the frequency generation mode
2	[:SOURce[1]|2]:FREQuency:MULTiplier(?)	Set the frequency multiplier
2	[:SOURce[1]|2]:FREQuency:OFFSet(?)	Set the frequency offset
2	[:SOURce[1]|2]:FREQuency:REFerence(?)	Set the relative frequency
2	[:SOURce[1]|2]:FREQuency:REFerence:SET	Sett frequency reference value to 0

2	[:SOURce[1]|2]:FREQuency:REFerence:STATe(?)	Set the relative frequency to ON/OFF state
2	[:SOURce[1]|2]:FREQuency:SPAN(?)	Set the frequency span of the step sweep
2	[:SOURce[1]|2]:FREQuency:STARt(?)	Set the step sweep start frequency
3	[:SOURce[1]|2]:FREQuency:STEP[:INCRement](?)	Set the frequency step
3	[:SOURce[1]|2]:FREQuency:STEP:STATe(?)	Set the frequency step ON/OFF
3	[:SOURce[1]|2]:FREQuency:STOP(?)	Set the step sweep stop frequency
3	[:SOURce[1]|2]:POWer:ALC:BANDwidth|BWIDth(?)	Set the bandwidth of ALC loop
3	[:SOURce[1]|2]:POWer:ALC:LEVel(?)	Set ALC level value
3	[:SOURce[1]|2]:POWer:ALC:SEARch(?)	Set the power search mode
3	[:SOURce[1]|2]:POWer:ALC:SOURce(?)	Set the power stabilization mode
3	[:SOURce[1]|2]:POWer:ALC:SOURce:EXTernal:COUPling(?)	Set the coupling coefficient of external detection
3	[:SOURce[1]|2]:POWer:ALC[:STATe](?)	Set the ALC loop state

3	[:SOURce[1]|[2]:POWer:ATTenuation(?)]	Set the power attenuation
4	[:SOURce[1]|[2]:POWer:ATTenuation:AUTO(?)]	Set the power attenuation to ON/OFF state
4	[:SOURce[1]|[2]:POWer:CENTer(?)]	Set the center power of power sweep
4	[:SOURce[1]|[2]:POWer[:LEVel][:IMMediate][:AMPLitude](?)]	Set the power level
4	[:SOURce[1]|[2]:POWer[:LEVel][:IMMediate]:OFFSet(?)]	Set the power offset
4	[:SOURce[1]|[2]:POWer:PEP?	Query the power PEP value
4	[:SOURce[1]|[2]:POWer:PROTection[:STATe](?)	Set the power search output status
4	[:SOURce[1]|[2]:POWer:REFerence(?)	Set the relative power
4	[:SOURce[1]|[2]:POWer:REFerence:STATe(?)	Set the relative power to ON/OFF state
4	[:SOURce[1]|[2]:POWer:SPAN(?)	Set the power sweep span
4	[:SOURce[1]|[2]:POWer:STATt(?)	Set the start power during power sweep.
5	[:SOURce[1]|[2]:POWer:STEP(?)	Set the power step value

5	[:SOURce[1]|2]:POWer:STEP:STATe(?)	Set the power step ON/OFF
5	[:SOURce[1]|2]:POWer:STOP(?)	Set the stop power during power sweep.
5	[:SOURce[1]|2]:POWer:SWEep[:STATe](?)	Set the power sweep to ON/OFF state
5	[:SOURce[1]|2]:POWer:USER:ENABle(?)	Set the power limit ON/OFF
5	[:SOURce[1]|2]:POWer:USER:MAX(?)	Set the maximum output power
5	[:SOURce[1]|2]:LIST:CATalog?	Query list file under default column path
5	[:SOURce[1]|2]:LIST:DWELl	Set the dwell time for each sweep point in the list.
5	[:SOURce[1]|2]:LIST:DWELl:ALL(?)	Set the dwell time of all list sweep points
5	[:SOURce[1]|2]:LIST:DWELl:TYPE(?)	Set the list sweep dwell time type
6	[:SOURce[1]|2]:LIST:FREQuency(?)	Set the sweep frequency value in the list
6	[:SOURce[1]|2]:LIST:FREQuency:POINts？	Query the frequency points of list sweep
6	[:SOURce[1]|2]:LIST:INDex:STARt(?)	Set the list sweep start index

6	[:SOURce[1]|2]:LIST:INDex:STOP(?)	Set the list sweep stop index
6	[:SOURce[1]|2]:LIST:PLAY[:INDex]?	Query the currently playing index swept by list
6	[:SOURce[1]|2]:LIST:POWer(?)	Set the sweep power of the list
6	[:SOURce[1]|2]:LIST:POWer:POINts?	Query the number of list sweep power points
6	[:SOURce[1]|2]:LIST:RETRace(?)	Set the sweep to ON/OFF state
6	[:SOURce[1]|2]:LIST:TRIGger	Trigger list sweep
6	[:SOURce]:LIST:TRIGger:SOURce(?)	Set the list sweep
7	[:SOURce[1]|2]:LIST[:WAVeform](?)	Select the List Sweep file
7	[:SOURce[1]|2]:LIST[:WAVeform]:DELete	Delete the specified sweep file
7	[:SOURce[1]|2]:LIST[:WAVeform]:DELete:ALL	Delete all swept files
7	[:SOURce[1]|2]:LIST[:WAVeform]:EXPort	Export list file
7	[:SOURce[1]|2]:LIST[:WAVeform]:IMPort	Import list file

7	[:SOURce[1]|2]:LIST[:WAVeform]:POINts?	Query the list points
7	[:SOURce[1]|2]:LIST[:WAVeForm]:RESet	Reset the file selected by list
7	[:SOURce[1]|2]:LIST[:WAVeform]:STORe	Save the current list sweep data to a file
7	[:SOURce[1]|2]:LFOutput:AMPLitude	Set LF signal output amplitude
7	[:SOURce[1]|2]:LFOutput[:FUNCtion]:FREQuency(?)	Set LF signal output frequency
8	[:SOURce[1]|2]:LFOutput[:FUNCtion]:FREQuency:ALTernate	Set the LF output dual sine frequency 2 or the sweep sine stop frequency
8	[:SOURce[1]|2]:LFOutput[:FUNCtion]:FREQuency:ALTernate:AMPLitude:PERCent(?)	Set the LF dual sine frequency 2 amplitude percent
8	[:SOURce[1]|2]:LFOutput[:FUNCtion]:SHAPe(?)	Set the waveform of the function generator
8	[:SOURce[1]|2]:LFOutput[:FUNCtion]:SHAPe:NOISe(?)	Set the function generator noise type
8	[:SOURce[1]|2]:LFOutput[:FUNCtion]:SHAPe:RAMP(?)	Set the function generator ramp wave type
8	[:SOURce[1]|2]:LFOutput[:FUNCtion]:SWEep:TIME(?)	Set the sweep time of the function generator sweep frequency sine
8	[:SOURce[1]|2]:LFOutput:OFFSet(?)	Set the DC offset of low frequency

		output
8	[:SOURce[1]|2]:LFOutput:STATe(?)	Set LF ON/OFF state
8	[:SOURce[1]|2]:SWEep[:FREQuency]:DWELl(?)	Set the step sweep dwell time
8	[:SOURce[1]|2]:SWEep[:FREQuency]:MODE(?)	Set the step sweep mode
9	[:SOURce[1]|2]:SWEep[:FREQuency]:SHAPe(?)	Set the step sweep shape
9	[:SOURce[1]|2]:SWEep[:FREQuency]:SINGle:RESet	Restart single sweep
9	[:SOURce[1]|2]:SWEep[:FREQuency]:SPACing(?)	Set the step mode of step sweep
9	[:SOURce[1]|2]:SWEep[:FREQuency]:STARt:TRIGger	Trigger initial sweep
9	[:SOURce[1]|2]:SWEep[:FREQuency]:STARt:TRIGger:SOURce(?)	Set the trigger source of start sweep
9	[:SOURce[1]|2]:SWEep[:FREQuency]:STEP[:LINear](?)	Set the step value of the linear sweep
9	[:SOURce[1]|2]:SWEep[:FREQuency]:STEP:LOGarithmic(?)	Set the step value of the logarithm step sweep
9	[:SOURce[1]|2]:SWEep[:FREQuency]:STEP:TYPE(?)	Set the step mode of step sweep
9	[:SOURce[1]|2]:SWEep[:FREQuency]:TRIGger	Trigger the step sweep to the next

		point
9	[:SOURce[1]|2]:SWEep[:FREQuency]:TRIGger:SOURce(?)	Set the step sweep trigger source
1	[:SOURce[1]|2]:SWEep:POWer:DWELl(?)	Set the power sweep dwell time
1	[:SOURce[1]|2]:SWEep:POWer:MODE(?)	Set the power sweep mode
1	[:SOURce[1]|2]:SWEep:POWer:PLAY[:POWer](?)	Query the current value of the power sweep
1	[:SOURce[1]|2]:SWEep:POWer:SINGle:RESet	Restart single power sweep
1	[:SOURce[1]|2]:SWEep:POWer:SPACing(?)	Set the power sweep step mode
1	[:SOURce[1]|2]:SWEep:POWer:STARt:TRIGger	Trigger initial power sweep
1	[:SOURce[1]|2]:SWEep:POWer:STARt:TRIGger:SOURce(?)	Set initial power sweep trigger source
1	[:SOURce[1]|2]:SWEep:POWer:STEP[:LINear](?)	Set the linear step value of power sweep
1	[:SOURce[1]|2]:SWEep:POWer:STEP:LOGarithmic(?)	Set the step value

		of the logarithm of power sweep
1	[:SOURce[1]|2]:SWEep:POWer:TRIGger	Trigger power sweep step
1	[:SOURce[1]|2]:SWEep:POWer:TRIGger:SOURce(?)	Set power sweep trigger source
1	[:SOURce[1]|2]:SWEep:RETRace(?)	Set the flyback ON/OFF for frequency sweep
1	[:SOURce[1]|2]:PULM:INTernal:DELay(?)	Set the pulse delay time
1	[:SOURce[1]|2]:PULM:INTernal:DELay:STEP(?)	Set the pulse delay step
1	[:SOURce[1]|2]:PULM:INTernal:FREQuency(?)	Set the pulse repetition frequency
1	[:SOURce[1]|2]:PULM:INTernal:FREQuency:STEP(?)	Set the pulse repetition frequency step
1	[:SOURce[1]|2]:PULM:INTernal:JITTered:MODE(?)	Set the jitter mode of pulse
1	[:SOURce[1]|2]:PULM:INTernal:JITTered:PERCent (?)	Set the jitter percent of pulse

1	[:SOURce[1]|[2]:PULM:INTernal:PERiod(?)	Set the pulse period
1	[:SOURce[1]|[2]:PULM:INTernal:PERiod:STEP[:INCRement](?)	Set the pulse modulation step
1	[:SOURce[1]|[2]:PULM:INTernal:PTRain:DATA	Set pulse train data
1	[:SOURce[1]|[2]:PULM:INTernal:PTRain:DELete	Delete a row of data in the pulse train
1	[:SOURce[1]|[2]:PULM:INTernal:PTRain:POINts?	Query the number of pulse trains
1	[:SOURce[1]|[2]:PULM:INTernal:PTRain:PRESet	Delete all data in the pulse train
1	[:SOURce[1]|[2]:PULM:INTernal:PTRain:SELect	Select Load Pulse Train File
1	[:SOURce[1]|[2]:PULM:INTernal:PTRain:STORe	Save the current pulse train data to a file
1	[:SOURce[1]|[2]:PULM:INTernal:PWIDth(?)	Set pulse width

1	[:SOURce[1]|2]:PULM:INTernal:PWIDth:STEP(?)	Set the pulse width step value
1	[:SOURce[1]|2]:PULM:INTernal:SLIDing:MODE(?)	Set the pulse slip mode
1	[:SOURce[1]|2]:PULM:INTernal:SLIDing:POINts(?)	Set pulse slip points
1	[:SOURce[1]|2]:PULM:INTernal:SLIDing:STEP(?)	Set the pulse slip step
1	[:SOURce[1]|2]:PULM:INTernal:STAGger:DATA	Set pulse stagger data
1	[:SOURce[1]|2]:PULM:INTernal:STAGger:DELete	Delete any row in the pulse stagger list
1	[:SOURce[1]|2]:PULM:INTernal:STAGger:INSert	Inserts a row of stagger data into the specified row
1	[:SOURce[1]|2]:PULM:INTernal:STAGger:POINts?	Query the stagger points
1	[:SOURce[1]|2]:PULM:INTernal:STAGger:PRESet	Delete all stagger data
1	[:SOURce[1]|2]:PULM:INTernal:STAGger:SELect	Select stagger file for loading and

		playing
1	[:SOURce[1]|2]:PULM:INTernal:STAGger:STORe	Store stagger data to a file
1	[:SOURce[1]|2]:PULM:SOURce(?)	Set the pulse source type
1	[:SOURce[1]|2]:PULM:SOURce:EXTernal(?)	Set external pulse source type
1	[:SOURce[1]|2]:PULM:SOURce:EXTernal[1]|2:IMPedance(?)	Set external pulse source impedance
1	[:SOURce[1]|2]:PULM:STATe(?)	Set the pulse modulation to ON/OFF state
1	[:SOURce[1]|2]:AM[1]|2[:DEPTh]:EXPonential(?)	Set exponential AM depth
1	[:SOURce[1]|2]:AM[1]|2[:DEPTh][:LINear](?)	Set Linear AM Depth
1	[:SOURce[1]|2]:AM[:DEPTh]:STEP[:INCRement](?)	Set the linear amplitude modulation depth step
1	[:SOURce[1]|2]:AM[1]|2:EXTernal[1]|2:COUPling (?)	Set the external coupling coefficient

1	[:SOURce[1]|2]:AM[1]|2:INTernal:FREQuency(?)	Set the amplitude modulation rate
1	[:SOURce[1]|2]:AM[1]|2:INTernal:FREQuency:ALTernate(?)	Set the amplitude modulation dual sine frequency 2 or the sweep sine stop frequency
1	[:SOURce[1]|2]:AM[1]|2:INTernal:FREQuency:ALTernate:AMPLitude:PERCent(?)	Set the dual sine frequency 2 amplitude percent
1	[:SOURce[1]|2]:AM[1]|2:INTernal:FREQuency:STEP(?)	Set the step value of the amplitude modulation rate
1	[:SOURce[1]|2]:AM[1]|2:INTernal:SHAPe(?)	Set the amplitude modulation waveform
1	[:SOURce[1]|2]:AM[1]|2:INTernal:SHAPe:NOISe(?)	Set the AM nose type
1	[:SOURce[1]|2]:AM[1]|2:INTernal:SHAPe:RAMP(?)	Set the AM ramp wave type
1	[:SOURce[1]|2]:AM[1]|2:INTernal:SWEep:TIME(?)	Set the sweep time for the frequency sweep sine wave
1	[:SOURce[1]|2]:AM:MODE(?)	Set AM depth mode

Appendix A Zoom Table of SCPI Classified by Subsystem

1	[:SOURce[1]|2]:AM[1]|2:SOURce(?)	Set the AM source
1	[:SOURce[1]|2]:AM[1]|2:STATe(?)	Set AM ON/OFF state
1	[:SOURce[1]|2]:AM:TYPE(?)	Set the AM type
1	[:SOURce[1]|2]:FM[1]|2:[DEViation](?)	Set FM frequency offset
1	[:SOURce[1]|2]:FM[:DEViation]:STEP[:INCRement](?)	Set FM frequency offset step
1	[:SOURce[1]|2]:FM[1]|2:EXTernal[1]|2:COUPling(?)	Set the coupling coefficient of the external frequency modulation source
1	[:SOURce[1]|2]:FM[1]|2:INTernal:FREQuency(?)	Set the frequency modulation rate
1	[:SOURce[1]|2]:FM[1]|2:INTernal:FREQuency:ALTernate(?)	Set frequency modulation dual sine frequency 2 or the sweep sine stop frequency
1	[:SOURce[1]|2]:FM[1]|2:INTernal:FREQuency:ALTernate:AMPLitude:PERCent(?)	Set the dual sine frequency 2 amplitude percent

1	[:SOURce[1]|2]:FM[1]|2:INTernal:FREQuency:STEP(?)	Set the frequency modulation step
1	[:SOURce[1]|2]:FM[1]|2:INTernal:SHAPe(?)	Set the internal waveform of frequency modulation
1	[:SOURce[1]|2]:FM[1]|2:INTernal:SHAPe:NOISe(?)	Set FM noise type
1	[:SOURce[1]|2]:FM[1]|2:INTernal:SHAPe:RAMP(?)	Set the frequency modulation ramp wave type
1	[:SOURce[1]|2]:FM[1]|2:INTernal:SWEep:TIME(?)	Set the sweep time for the frequency modulation sweep sine
1	[:SOURce[1]|2]:FM[1]|2:SOURce(?)	Set the FM source type
1	[:SOURce[1]|2]:FM[1]|2:STATe(?)	Set the FM to ON/OFF state
1	[:SOURce[1]|2]:PM:BANDwidth|BWIDth(?)	Set phase modulation bandwidth
1	[:SOURce[1]|2]:PM[1]|2 [:DEViation](?)	Set the phase modulation photo
1	[:SOURce[1]|2]:PM[:DEViation]:STEP[:INCRement](?)	Set the phase modulation photo

		step
1	[:SOURce[1]|2]:PM[1]|2:EXTernal[1]|2:COUPling(?)	Set the coupling coefficient of the external phase modulation source
1	[:SOURce[1]|2]:PM[1]|2:INTernal:FREQuency(?)	Set the phase modulation rate
1	[:SOURce[1]|2]:PM[1]|2:INTernal:FREQuency:ALTernate(?)	Set the phase modulation dual sine frequency 2 or the sweep sine stop frequency
1	[:SOURce[1]|2]:PM[1]|2:INTernal:FREQuency:ALTernate :AMPLitude:PERCent(?)	Set the dual sine frequency 2 amplitude percent for phase modulation
1	[:SOURce[1]|2]:PM[1]|2:INTernal:SHAPe(?)	Set PM waveform
1	[:SOURce[1]|2]:PM[1]|2:INTernal:SHAPe:NOISe(?)	Set PM noise type
1	[:SOURce[1]|2]:PM[1]|2:INTernal:SHAPe:RAMP(?)	Set the phase modulation ramp wave type
1	[:SOURce[1]|2]:PM[1]|2:INTernal:SWEep:TIME(?)	Set the sweep time for the sweep sine of phase modulation
1	[:SOURce[1]|2]:PM[1]|2:SOURce(?)	Set the PM source type

1	[:SOURce[1]|2]:PM[1]|2:STATe(?)	Set the phase modulation ON/OFF
1	[:SOURce[1]|2]:DM:ATTenuation(?)	Set the gain adjustment for I/Q modulation
1	[:SOURce[1]|2]:DM:CORRection	Calibrate the current frequency point IQ modulation
1	[:SOURce[1]|2]:DM:IQADjustment:GAIN(?)	Set I/Q input adjustment gain balance
1	[:SOURce[1]|2]:DM:IQADjustment:IOFFset(?)	Set I offset of I/Q input adjustment
1	[:SOURce[1]|2]:DM:IQADjustment:OUTPut:GAIN(?)	Set I/Q output adjustment gain balance
1	[:SOURce[1]|2]:DM:IQADjustment:OUTPut:IOFFset(?)	Set I offset of I/Q output adjustment
1	[:SOURce[1]|2]:DM:IQADjustment:OUTPut:QOFFset(?)	Set Q offset of I/Q output modulation
1	[:SOURce[1]|2]:DM:IQADjustment:OUTPut:SKEW(?)	Set orthogonal offset of I/Q output adjustment

1	[:SOURce[1]|2]:DM:IQADjustment:OUTPut[:STATe](?)	Set IQ output adjustment to ON/OFF state
1	[:SOURce[1]|2]:DM:IQADjustment:QOFFset(?)	Set Q offset of I/Q input adjustment
1	[:SOURce[1]|2]:DM:IQADjustment:QSKew(?)	Set orthogonal offset of I/Q input modulation
1	[:SOURce[1]|2]:DM:IQADjustment[:STATe](?)	Set IQ input adjustment to ON/OFF state
1	[:SOURce[1]|2]:DM:IQEXchange:STATe(?)	Set the I/Q exchange ON/OFF
1	[:SOURce[1]|2]:DM:OUTPut:AMPLitude(?)	Set the I/Q output amplitude
1	[:SOURce[1]|2]:DM:OUTPut:ICOFfset(?)	Set the I/Q output I-channel common-mode voltage
1	[:SOURce[1]|2]:DM:OUTPut:QCOFfset(?)	Set the I/Q output Q-channel common-mode voltage
2	[:SOURce[1]|2]:DM:OUTPut[:STATe](?)	Set IQ output ON/OFF state
2	[:SOURce[1]|2]:DM:SOURce(?)	Set the I/Q data

		source
2	[:SOURce[1]|2]:DM:STATe(?)	Set the I/Q modulation ON/OFF
2	:MEMory:CATalog:ALL?	Query the file information under the directory of /home/ceyear/SgData/user/
2	:MEMory:CATalog:FSK?	Query the file information under the directory of /home/ceyear/SgData/user/Fsk/
2	:MEMory:CATalog:IQ?	Query the file information under the directory of /home/ceyear/SgData/user/IQ/
2	:MEMory:CATalog:LIST?	Query the file information under the directory of /home/ceyear/SgData/user/List/
2	:MEMory:CATalog:MTONe?	Query the file information under the directory of /home/ceyear/SgData/user/Mtone/
2	:MEMory:CATalog:PTRain?	Query the file information under the directory of /home/ceyear/SgData/user/Train/
2	:MEMory:CATalog:SEQ?	Query the file information under

		the directory of /home/ceyear/SgData/user/Wav/
2	:MEMory:COPY[:NAME]	Copy the file
2	:MEMory:DATA	Transfer block data and store it in a file
2	:MEMory:DATA:APPend	Append a data block to a specified file
2	:MEMory:DELete:FSK	Delete the file specified under the directory of /home/ceyear/SgData/user/Fsk/
2	:MEMory:DELete:IQ	Delete the file specified under the directory of /home/ceyear/SgData/user/IQ/
2	:MEMory:DELete:LIST	Delete the file specified under the directory of /home/ceyear/SgData/user/List/
2	:MEMory:DELete:MTONe	Delete the file specified under the directory of /home/ceyear/SgData/user/Mtone/
2	:MEMory:DELete[:NAME]	Delete the specified file

2	:MEMory:DELete:SEQ	Delete the file specified under the directory of /home/ceyear/SgData/user/Wav/
2	:MEMory:MOVE	Rename the specified file
2	[:SOURce]:ROSCillator[:ADJust]:REFerence(?)	Set the internal reference accuracy
2	[:SOURce]:ROSCillator:DEFaults	For settings, refer to Restoring Factory Defaults
2	[:SOURce]:ROSCillator:FREQuency:EXTernal(?)	Set the external reference frequency
2	[:SOURce]:ROSCillator:SOURce(?)	Set reference source
2	[:SOURce]:ROSCillator:SOURce:AUTO(?)	Set reference source Man/Auto
2	:DIAGnostic[:CPU]:INFormation:CCOunt:PON?	Query startup counts
2	:DIAGnostic[:CPU]:INFormation:OPTions?	Query option information

2	:DIAGnostic[:CPU]:INFormation:OTIMe?	Query start time.
2	:DIAGnostic:SNUM?	Query the serial number of the instrument
2	:SYSTem:COMMunicate:GPIB[:SELF]:ADDRess(?)	Set the GPIB address
2	:SYSTem:COMMunicate:GTLocal	Set the instrument to the local status
2	:SYSTem:COMMunicate:LAN:ADDRess|IP(?)	Set the IP address
2	:SYSTem:COMMunicate:LAN:DGATeway|GATeway(?)	Set the default gateway
2	:SYSTem:COMMunicate:LAN:HNAMe|HOSTname(?)	Set the host name
2	:SYSTem:COMMunicate:LAN:MAC?	Query the MAC address of the instrument
2	:SYSTem:COMMunicate:LAN:PORT(?)	Set the instrument port number
2	:SYSTem:COMMunicate:LAN:RESTart	Restart the instrument network

		configuration
2	:SYSTem:COMMunicate:LAN:SMASkSUBNet(?)	Set the subnet mask
2	:SYSTem:DEVice:LANGuage(?)	Set instrument display language
2	:SYSTem:ERRor:CLEar	Clear error list
2	:SYSTem:ERRor:COUNt?	Query quantity of errors
2	:SYSTem:ERRor[:NEXT]?	Query the latest error information
2	:SYSTem:HELP:HEADers?	Query a programmed control command string
2	:SYSTem:PRESet:TYPE	Set the instrument reset type
2	:SYSTem:PRESet[:USER]:SVAE	Save user status
2	:SYSTem:SHUTdown	Turn off the device

2	:SYSTem:VERSion?	Query instrument SCPI version information
2	[:SOURce]:RADio:ARB:IMAP[1]|3:TYPE	Set the arbitrary wave interface type
2	[:SOURce]:RADio:ARB:IMAP[1]|2|3|4:USED	Set the channel to output arbitrary wave marker
2	[:SOURce[1]|2]:RADio:ARB:MARKer[1]|2|3|4:DIVider	Set the average value when arbitrary wave is marked as a pulse
2	[:SOURce[1]|2]:RADio:ARB:MARKer[1]|2|3|4:OFFTime	Set the number of sampling points of OFF time for arbitrary wave marked as fixed ON/OFF ratio
2	[:SOURce[1]|2]:RADio:ARB:MARKer[1]|2|3|4:ONTime	Set the number of sampling points of ON time for arbitrary wave marked as fixed ON/OFF ratio
2	[:SOURce[1]|2]:RADio:ARB:MARKer[1]|2|3|4:PATTern	Set the predefined style value for arbitrary wave marked as predefined style
2	[:SOURce[1]|2]:RADio:ARB:MARKer[1]|2|3|4[:TYPE]	Set the arbitrary wave marker type
2	[:SOURce[1]|2]:RADio:ARB:MCARrier:CARRier[1]|2--512:C	

	ONFlict?	
2	[:SOURce[1]|2]:RADio:ARB:MCARrier:CARRier:COUNt(?)	Query whether the specified carrier conflicts
2	[:SOURce[1]|2]:RADio:ARB:MCARrier:CARRier[1]|2--512:DELay(?)	Set the delay for the specified carrier
2	[:SOURce[1]|2]:RADio:ARB:MCARrier:CARRier[1]|2--512:FREQuency(?)	Set the frequency offset of the specified carrier
2	[:SOURce[1]|2]:RADio:ARB:MCARrier:CARRier:MODE(?)	Set the multi-carrier interval mode
2	[:SOURce[1]|2]:RADio:ARB:MCARrier:CARRier[1]|2--512:PHASe(?)	Set the phase of the specified carrier
2	[:SOURce[1]|2]:RADio:ARB:MCARrier:CARRier[1]|2--512:POWer(?)	Set the gain of the specified carrier
2	[:SOURce[1]|2]:RADio:ARB:MCARrier:CARRier:SPACing(?)	Set multi-carrier carrier spacing
2	[:SOURce[1]|2]:RADio:ARB:MCARrier:CARRier[1]|2--512:STATe(?)	Set the state of the specified carrier
2	[:SOURce[1]|2]:RADio:ARB:MCARrier:CARRier[1]|2--512:WAVeform(?)	Set the file of the specified carrier

2	[:SOURce[1]|2]:RADio:ARB:MCARrier:CATalog?	Query the list of saved multi-carrier profiles
2	[:SOURce[1]|2]:RADio:ARB:MCARrier:CONFigure:LOAD	Load a multi-carrier profile
2	[:SOURce[1]|2]:RADio:ARB:MCARrier:CONFigure:STORe	Save the current multi-carrier configuration as a file
2	[:SOURce[1]|2]:RADio:ARB:MCARrier:EDIT:CARRier:DELay:STARt(?)	Set the delay start value for carrier autofill
2	[:SOURce[1]|2]:RADio:ARB:MCARrier:EDIT:CARRier:DELay:STEP(?)	Set the delay step value for carrier autofill
2	[:SOURce[1]|2]:RADio:ARB:MCARrier:EDIT:CARRier:EXECute	Perform carrier autofill
2	[:SOURce[1]|2]:RADio:ARB:MCARrier:EDIT:CARRier:PHASe:STARt(?)	Set the phase start value for carrier autofill
2	[:SOURce[1]|2]:RADio:ARB:MCARrier:EDIT:CARRier:PHASe:STEP(?)	Set the phase step value for carrier autofill
2	[:SOURce[1]|2]:RADio:ARB:MCARrier:EDIT:CARRier:POWer:STARt(?)	Set the gain start value for carrier autofill

2	[:SOURce[1]|2]:RADio:ARB:MCARrier:EDIT:CARRier:POWer:STEP(?)	Set the gain step value for carrier autofill
2	[:SOURce[1]|2]:RADio:ARB:MCARrier:EDIT:CARRier:STARt(?)	Set the start index value for carrier autofill
2	[:SOURce[1]|2]:RADio:ARB:MCARrier:EDIT:CARRier:STATe(?)	Set the carrier status for carrier autofill
2	[:SOURce[1]|2]:RADio:ARB:MCARrier:EDIT:CARRier:STOP(?)	Set the end index for carrier autofill
2	[:SOURce[1]|2]:RADio:ARB:MCARrier:EDIT:CARRier::WAVeform(?)	Set waveform file selection for carrier autofill
2	[:SOURce[1]|2]:RADio:ARB:MCARrier:PERiod(?)	Set a custom period value for the multi-carrier
2	[:SOURce[1]|2]:RADio:ARB:MCARrier:PERiod:MODE(?)	Set the signal period style for the multi-carrier
2	[:SOURce[1]|2]:RADio:ARB:MCARrier:POWer:REFerence(?)	Set the multi-carrier power reference
2	[:SOURce[1]|2]:RADio:ARB:MCARrier:RESet	Set multi-carrier to the default value
2	[:SOURce[1]|2]:RADio:ARB:MCARrier:WAVeform:CLOad	Create, load and output an arbitrary

		wave file based on the current multi-carrier configuration
2	[:SOURce[1]|2]:RADio:ARB:MCARrier:WAVeform:CREate	Create an arbitrary wave file based on the current multi-carrier configuration
2	[:SOURce[1]|2]:RADio:ARB:SCLock:RATE	Set the arbitrary clock frequency
2	[:SOURce[1]|2]:RADio:ARB:SEQuence:CLOad	Create and Load Sequence File
2	[:SOURce[1]|2]:RADio:ARB:SEQuence:CLOCk	Set the clock type when creating sequence files
2	[:SOURce[1]|2]:RADio:ARB:SEQuence:CLOCk:RATE	Set the clock type when creating sequence files to the custom clock frequency
2	[:SOURce[1]|2]:RADio:ARB:SEQuence:CONFigure:APPend	Append waveform segment files to the sequence configuration list
2	[:SOURce[1]|2]:RADio:ARB:SEQuence:CONFigure:DELete	Delete the waveform segment file with the specified index from the sequence configuration list
2	[:SOURce[1]|2]:RADio:ARB:SEQuence:CONFigure:RESet	Delete all waveform segment

		information from the sequence configuration list
2	[:SOURce[1]|2]:RADio:ARB:SEQuence:CONFigure:SELect	Select a sequence profile
2	[:SOURce[1]|2]:RADio:ARB:SEQuence:CONFigure:STORe	Save the sequence configuration data to a file
2	[:SOURce[1]|2]:RADio:ARB:SEQuence:CREate	Create Sequence File
2	[:SOURce[1]|2]:RADio:ARB:SEQuence:INDex	Specify the index number of the waveform segment to be played
2	[:SOURce[1]|2]:RADio:ARB:SEQuence:MARKer:ESEGment	Set the start marker type for each waveform segment in the sequence
2	[:SOURce[1]|2]:RADio:ARB:SEQuence:MARKer:MODE	Set the marker type for each waveform segment in the sequence
2	[:SOURce[1]|2]:RADio:ARB:SEQuence:MARKer:STARt	Set the sequence start marker type
2	[:SOURce[1]|2]:RADio:ARB:SEQuence:NEXT[:SEGMent]	Play the next waveform segment in the sequence.
2	[:SOURce[1]|2]:RADio:ARB:SEQuence[:TYPE]	Set arbitrary wave play mode

3	[:SOURce[1]|2]:RADio:ARB[:STATe](?)	Set arbitrary to ON/OFF state
3	[:SOURce[1]|2]:RADio:ARB:TRIGger:EXECute	Trigger to play an arbitrary wave
3	[:SOURce[1]|2]:RADio:ARB:TRIGger[:SOURce](?)	Set the arbitrary trigger source
3	[:SOURce[1]|2]:RADio:ARB:TRIGger[:SOURce]:EXTernal:DELay(?)	Set the delay time for external triggering source of an arbitrary wave
3	[:SOURce[1]|2]:RADio:ARB:TRIGger[:SOURce]:EXTernal:SAMPle(?)	Set the number of delayed sampling points for external triggering source of an arbitrary wave
3	[:SOURce[1]|2]:RADio:ARB:TRIGger[:SOURce]:EXTernal:DELay:STATe(?)	Set the arbitrary wave external trigger source delay ON/OFF state
3	[:SOURce[1]|2]:RADio:ARB:TRIGger[:SOURce]:EXTernal:SLOPe(?)	Set the arbitrary wave external trigger source slope
3	[:SOURce[1]|2]:RADio:ARB:TRIGger:TYPE(?)	Set the trigger mode during arbitrary wave play
3	[:SOURce[1]|2]:RADio:ARB:TRIGger:TYPE:CONTinuous[:TYPE](?)	Set the arbitrary continuous trigger type

3	[:SOURce[1]|2]:RADio:ARB:TRIGger:TYPE:GATE[:ACTive](?)	Set the arbitrary wave gate trigger type
3	[:SOURce[1]|2]:RADio:ARB:TRIGger:TYPE:SINGle(?)	Set the arbitrary single trigger mode
3	[:SOURce[1]|2]:RADio:ARB:TWAVeform:SINE:CLOad	Create and load a sine wave test waveform
3	[:SOURce[1]|2]:RADio:ARB:TWAVeform:SINE:CREate	Create a sine wave test waveform
3	[:SOURce[1]|2]:RADio:ARB:TWAVeform:SINE:FREQuency(?)	Set the frequency value of the sine wave test waveform
3	[:SOURce[1]|2]:RADio:ARB:TWAVeform:SINE:PHASe(?)	Set the Q-channel phase offset of the sine wave test waveform
3	[:SOURce[1]|2]:RADio:ARB:TWAVeform:SINE:SAMPles(?)	Set the sampling points per cycle of the sine wave test waveform
3	[:SOURce[1]|2]:RADio:ARB:TWAVeform:SQUare:AMPLitude(?)	Set the amplitude of the square wave test waveform
3	[:SOURce[1]|2]:RADio:ARB:TWAVeform:SQUare:CLOad	Create and load the square wave test waveform
3	[:SOURce[1]|2]:RADio:ARB:TWAVeform:SQUare:CREate	Create a square

		wave test waveform
3	[:SOURce[1]|2]:RADio:ARB:TWAVeform:SQUare:FREQuency(?)	Set the frequency value of the square wave test waveform
3	[:SOURce[1]|2]:RADio:ARB:TWAVeform:SQUare:OFFSet(?)	Set the DC bias of the square wave test waveform
3	[:SOURce[1]|2]:RADio:ARB:TWAVeform:SQUare:SAMPles(?)	Set the number of sampling points for the square wave test waveform
3	[:SOURce[1]|2]:RADio:ARB:TWAVeform:TYPE(?)	Set the type of test waveform
3	[:SOURce[1]|2]:RADio:ARB:WAVeform(?)	Select and load the arbitrary wave file
3	[:SOURce[1]|2]:RADio:ARB:TWAVeform:RESet	Reset arbitrary wave playback settings
3	[:SOURce[1]|2]:RADio:CUSTom:ALPHa(?)	Set filter factor
3	[:SOURce[1]|2]:RADio:CUSTom:DATA(?)	Set the data source for digital modulation
3	[:SOURce[1]|2]:RADio:CUSTom:DATA:PATTern(?)	Set custom sequence values for the data source

3	[:SOURce[1]|2]:RADio:CUSTom:DATA:PRAM(?)	Select file code stream file
3	[:SOURce[1]|2]:RADio:CUSTom:DATA:SCOunt(?)	Set the number of sampling points for generating waveform files
3	[:SOURce[1]|2]:RADio:CUSTom:FILTer(?)	Set the digital modulation filter type
3	[:SOURce[1]|2]:RADio:CUSTom:FILTer:PRESet	Restore default filtering mode
3	[:SOURce]:RADio:CUSTom:IMAP[1]|3:TYPE(?)	Set the interface type of the digital modulation interface mapping
3	[:SOURce]:RADio:CUSTom:IMAP[1]|2|3|4:USED(?)	Set the channel to which the digital modulation interface belongs
3	[:SOURce[1]|2]:RADio:CUSTom:IQData	Transmit and load arbitrary wave data pair
3	[:SOURce[1]|2]:RADio:CUSTom:MARKer[1]|2|3|4:CUSTom(?)	Set a custom marker file
3	[:SOURce[1]|2]:RADio:CUSTom:MARKer[1]|2|3|4:DIVider(?)	Set the average value of the pulse marker
3	[:SOURce[1]|2]:RADio:CUSTom:MARKer[1]|2|3|4:OFFTime(	Set the number of code elements for

	?)	the OFF time of the fixed ON/OFF ratio marker
3	[:SOURce[1]|2]:RADio:CUSTom:MARKer[1]|2|3|4:ONTime(?)	Set the number of code elements for the ON time of the fixed ON/OFF ratio marker
3	[:SOURce[1]|2]:RADio:CUSTom:MARKer[1]|2|3|4:PATTern(?)	Set the predefined marker value
3	[:SOURce[1]|2]:RADio:CUSTom:MARKer[1]|2|3|4:[:TYPE](?)	Set the digital modulation marker type
3	[:SOURce[1]|2]:RADio:CUSTom:MODulation:APSK[16]|32:GAMMa(?)	Set the APSK-modulated gamma value
3	[:SOURce[1]|2]:RADio:CUSTom:MODulation:APSK[16]|32:GAMMa:CUSTom[1]|2(?)	Set the custom parameter values for APSK modulation
3	[:SOURce[1]|2]:RADio:CUSTom:MODulation:ASK[:DEPTh](?)	Set ASK modulation depth
3	[:SOURce[1]|2]:RADio:CUSTom:MODulation:FSK[:DEViation](?)	Set FSK frequency offset
3	[:SOURce[1]|2]:RADio:CUSTom:MODulation:PRESet	Restore the default modulation type
3	[:SOURce[1]|2]:RADio:CUSTom:MODulation[:TYPE](?)	Set the digital modulation type

3	[:SOURce[1]|2]:RADio:CUSTom:MODulation:UFSK:DEViation[1]|2|3|4|5|6|7|8|9|10|11|12|13|14|15|16(?)	Set each frequency offset of the custom FSK
3	[:SOURce[1]|2]:RADio:CUSTom:MODulation:UFSK:TYPE(?)	Set the custom FSK type
3	[:SOURce[1]|2]:RADio:CUSTom:MODulation:UIQ	If the modulation type is custom IQ file, select IQ file
3	[:SOURce[1]|2]:RADio:CUSTom:SRATe(?)	Set the code element rate for digital modulation
3	[:SOURce[1]|2]:RADio:CUSTom[:STATe](?)	Set the digital modulation ON/OFF state
3	[:SOURce[1]|2]:RADio:CUSTom:TRIGger:EXECute	Set the digital modulation triggering
3	[:SOURce[1]|2]:RADio:CUSTom:TRIGger[:SOURce](?)	Set the baseband signal trigger source
3	[:SOURce[1]|2]:RADio:CUSTom:TRIGger[:SOURce]:EXTernal:DELay:SAMPle(?)	Set the number of delayed sampling points of the external digital modulation source
3	[:SOURce[1]|2]:RADio:CUSTom:TRIGger[:SOURce]:EXTernal:DELay:STATe(?)	Set the delay ON/OFF for the external digital

		modulation source
3	[:SOURce[1]|2]:RADio:CUSTom:TRIGger[:SOURce]:EXTernal:DELay[:TIME](?)	Set the delay time for the external digital modulation source
3	[:SOURce[1]|2]:RADio:CUSTom:TRIGger[:SOURce]:EXTernal:SLOPe(?)	Set the external digital modulation source slope
3	[:SOURce[1]|2]:RADio:CUSTom:TRIGger:TYPE(?)	Set the digital modulation triggering mode
3	[:SOURce[1]|2]:RADio:CUSTom:TRIGger:TYPE:CONTinuous[:TYPE](?)	Set the continuous triggering type of digital modulation
3	[:SOURce[1]|2]:RADio:CUSTom:TRIGger:TYPE:GATE:ACTive(?)	Set the gating triggering type of the digital modulation
3	[:SOURce[1]|2]:RADio:CUSTom:WAVeform:CREate	Create a waveform segment file
3	[:SOURce[1]|2]:RADio:MTONe:ARB:CFACtor (?)	Set the target peak-to-average ratio in the multitone configuration
3	[:SOURce[1]|2]:RADio:MTONe:ARB:CFACtor:MODE(?)	Set the optimization model for the target peak-to-average ratio in the multitone configuration
3	[:SOURce[1]|2]:RADio:MTONe:ARB:SETup	Select the multitone file to play

3	[:SOURce[1]|2]:RADio:MTONe:ARB:SETup:RESet	Restore the default value of multitone
3	[:SOURce[1]|2]:RADio:MTONe:ARB:SETup:STORe	Save the multitone data to a file
3	[:SOURce[1]|2]:RADio:MTONe:ARB:SETup:TABLe	Configure multitone table data
3	[:SOURce[1]|2]:RADio:MTONe:ARB:SETup:TABLe:FSPacing(?)	Set the multitone frequency spacing
3	[:SOURce[1]|2]:RADio:MTONe:ARB:SETup:TABLe:NTONes(?)	Set the number of multitone
3	[:SOURce[1]|2]:RADio:MTONe:ARB:SETup:TABLe:PHASe(?)	Set the initial phase value of multitone
3	[:SOURce[1]|2]:RADio:MTONe:ARB:SETup:TABLe:PHASe:INITialize(?)	Set the initial phase of multitone
3	[:SOURce[1]|2]:RADio:MTONe:ARB:SETup:TABLe:PHASe:INITialize:SEED (?)	Set the relationship between tone phases
3	[:SOURce[1]|2]:RADio:MTONe:ARB:SETup:TABLe:ROW	Set a row of data in multitone
3	[:SOURce]:RADio:MTONe:ARB[:STATe](?)	Set multitone modulation ON/OFF

		state
3	[:SOURce[1]|2]:PWM:BARKer:TYPE(?)	Set the Barker code type of intra-pulse modulation
3	[:SOURce[1]|2]:PWM:BWIDth(?)	Set the bandwidth for intra-pulse modulation
3	[:SOURce[1]|2]:PWM:DIRection(?)	Set the direction for intra-pulse modulation
3	[:SOURce[1]|2]:PWM:FSK[:DEViation](?)	Set the frequency offset of FSK modulation
3	[:SOURce[1]|2]:PWM:LENGth(?)	Set the sequence length of the phase modulation code
3	[:SOURce[1]|2]:PWM:SYMBols?	Query the number of symbols of PM code.
3	[:SOURce[1]|2]:PWM:PPCM:TYPE(?)	Set the phase modulation code type
3	[:SOURce[1]|2]:PWM:STATe(?)	Set intrapulse modulation ON/OFF
3	[:SOURce[1]|2]:PWM:TYPE(?)	Set the type of intra-pulse modulation

3	:STATus:OPERation:CONDition?	Query the value of condition register in the operation state register
3	:STATus:OPERation:ENABle(?)	Set the operation state event enable register
3	:STATus:OPERation:NTRansition(?)	Set the operation state negative transition filter
3	:STATus:OPERation:PTRansition(?)	Set the operation state positive transition filter
3	:STATus:OPERation[:EVENt]?	Query the operation status event register
3	:STATus:PRESet	Preset the value of the state register
3	:STATus:QUEStionable:CONDition?	Query the value of condition register in the question status register
3	:STATus:QUEStionable:ENABle(?)	Set the question status event enable register
3	:STATus:QUEStionable:NTRansition(?)	Set the question status negative transition filter
3	:STATus:QUEStionable:PTRansition(?)	Set the question stat positive

		transition filter
3	:STATus:QUEStionable[:EVENt]?	Query the question status event register.
3	:STATus:QUEStionable:FREQuency:CONDition?	Query the value of condition register in the frequency question status register
3	:STATus:QUEStionable:FREQuency:ENABle(?)	Set the value of the enabling register of the frequency question status event
3	:STATus:QUEStionable:FREQuency:NTRansition(?)	Set the negative transition filter of the frequency question status
3	:STATus:QUEStionable:FREQuency:PTRansition(?)	Set the positive transition filter of the frequency question status
3	:STATus:QUEStionable:FREQuency[:EVENt]?	Query the value of the frequency question status event register
4	:STATus:QUEStionable:POWer:CONDition?	Query the value of condition register in the power question status register
4	:STATus:QUEStionable:POWer:ENABle(?)	Set the value of the power question status event enable register
4	:STATus:QUEStionable:POWer:NTRansition(?)	Set the value of the negative transition

		filter of the power question status register
4	:STATus:QUEStionable:POWer:PTRansition(?)	Set the value of the positive transition filter of the power question status register
4	:STATus:QUEStionable:POWer[:EVENt]?	Query the value of the power question status event register
4	[:SOURce[1]|2]:FUNCtion:FREQuency(?)	Set the output frequency of the function generator
4	[:SOURce[1]|2]:FUNCtion:FREQuency:ALTernate(?)	Set function generator dual sine frequency 2 and the sweep sine stop frequency
4	[:SOURce[1]|2]:FUNCtion:FREQuency:AMPLitude:PERCent (?)	Set the function generator dual sine frequency 2 amplitude percent
4	[:SOURce[1]|2]:FUNCtion:PERiod?	Query the period value of the function generator
4	[:SOURce[1]|2]:FUNCtion:SHAPe(?)	Set the output waveform of the function generator
4	[:SOURce[1]|2]:FUNCtion:SHAPe:NOISe(?)	Set the function generator noise type
4	[:SOURce[1]|2]:FUNCtion:SHAPe:RAMP(?)	Set the function

Appendix A Zoom Table of SCPI Classified by Subsystem

		generator ramp wave type
4	[:SOURce[1]|2]:FUNCtion:SWEep:TIME(?)	Set the sweep time of the function generator sweep frequency sine
4	[:SOURce]:EXTernal[1]|2[:SOURce]:COUPling(?)	Set the external coupling coefficient
4	[:SOURce]:EXTernal[1]|2[:SOURce]:IMPedance(?)	Set the external impedance value
4	[:SOURce[1]|2]:CORRection:ACALculation:STATe(?)	Set the frequency device interpolation ON/OFF
4	[:SOURce]:CORRection:CATalog?	Check the calibration compensation file saved in the instrument
4	[:SOURce[1]|2]:CORRection:EXECute	Start calibration
4	[:SOURce]:CORRection:FILE:DELete	Set the specified calibration compensation file
4	[:SOURce[1]|2]:CORRection:FLATness:CURRent:FREQuency(?)	Set frequency value in the current line.
4	[:SOURce[1]|2]:CORRection:FLATness:CURRent:POWer(?)	Set the power compensation value

		of the current row
4	[:SOURce[1]|2]:CORRection:FLATness:DELete(?)	Set the deletion type and delete
4	[:SOURce[1]|2]:CORRection:FLATness[:FILE]:SELect	Select calibration compensation file
4	[:SOURce[1]|2]:CORRection:FLATness[:FILE]:STORe	Save calibration compensation data to a file
4	[:SOURce[1]|2]:CORRection:FLATness:FREQuency(?)	Set the frequency value in the compensation data list
4	[:SOURce[1]|2]:CORRection:FLATness:INDex(?)	Set the currently selected row
4	[:SOURce[1]|2]:CORRection:FLATness:PAIR	Modify the frequency and the corresponding power compensation value
4	[:SOURce[1]|2]:CORRection:FLATness:POINts?	Query the number of lines in the current compensation file
4	[:SOURce[1]|2]:CORRection:FLATness:POWer(?)	Set the power compensation value in the compensation data list
4	[:SOURce[1]|2]:CORRection:PMETer:COMMunicate:MODE(?)	Set the power meter connection mode

4	[:SOURce[1]|2]:CORRection:PMETer:GPIB:ADDRess(?)	Set the power meter GPIB address
4	[:SOURce[1]|2]:CORRection:PMETer:LAN:IP(?)	Set the power meter IP address
4	[:SOURce[1]|2]:CORRection:PMETer:TYPE(?)	Set the power meter type
4	[:SOURce[1]|2]:CORRection:PMETer:ZERO:STATe(?)	Set whether to reset the ON/OFF state
4	[:SOURce[1]|2]:CORRection[:STATe](?)	Set the user calibration compensation ON/OFF state
4	[:SOURce[1]|2]:AWGN:BRATe?	Query the bit rate of additive noise
4	[:SOURce[1]|2]:AWGN:BWIDth(?)	Set the additive noise or pure noise bandwidth
4	[:SOURce[1]|2]:AWGN:CARRier:BWIDth(?)	Set the carrier signal bandwidth of additive noise
4	[:SOURce[1]|2]:AWGN:CNRatio(?)	Set the signal-to-noise ratio of additive noise or continuous wave interference

4	[:SOURce[1]|2]:AWGN:ENRaio(?)	Set the Eb/N0 of additive noise
4	[:SOURce[1]|2]:AWGN:FREQuency:OFFSet(?)	Set the target CW frequency offset of CW interference
4	[:SOURce[1]|2]:AWGN:MODE(?)	Set noise mode
4	[:SOURce[1]|2]:AWGN:POWer:MODE(?)	Set the noise power calculation mode
4	[:SOURce[1]|2]:AWGN[:STATe](?)	Set the noise ON/OFF state
4	[:SOURce]:RADar:CHANnel:SELect(?)	Set output channel
4	[:SOURce]:RADar:PULSe[1]|2--32766:DELay[:TIME](?)	Set the pulse delay time
4	[:SOURce]:RADar:PULSe[1]|2--32766:FALL[:TIME](?)	Set the pulse falling edge time
4	[:SOURce]:RADar:PULSe[1]|2--32766:HOPPing:CLEar	Clear the periodic table of transition
4	[:SOURce]:RADar:PULSe[1]|2--32766:HOPPing:DELete	Deletes the specified row in the

		periodic table of transition
4	[:SOURce]:RADar:PULSe[1]|2--32766:HOPPing:PERiod	Set the periodic value of transition
4	[:SOURce]:RADar:PULSe[1]|2--32766:JITTered:PERCent(?)	Set the jitter percent
4	[:SOURce]:RADar:PULSe[1]|2--32766:MODulation:BARKer:TYPE(?)	Set the sequence length of the Barker code
4	[:SOURce]:RADar:PULSe[1]|2--32766:MODulation[:FREQuency]:BWIDth(?)	Set frequency modulation bandwidth
4	[:SOURce]:RADar:PULSe[1]|2--32766:MODulation:LFM:TYPE(?)	Set the linear frequency modulation direction
4	[:SOURce]:RADar:PULSe[1]|2--32766:MODulation:NLFM:SYMMetry(?)	Set nonlinear frequency modulation symmetry
4	[:SOURce]:RADar:PULSe[1]|2-32766:MODulation:NLFM:TYPE(?)	Set the mode of nonlinear frequency modulation
4	[:SOURce]:RADar:PULSe[1]|2--32766:MODulation:SYMBols(?)	Set the number of code elements
4	[:SOURce]:RADar:PULSe[1]|2-32766:MODulation:TFM:TYPE(?)	Set the triangle frequency modulation direction

4	[:SOURce]:RADar:PULSe[1]|2--32766:MODulation[:TYPE](?)	Set the intra-pulse modulation style
4	[:SOURce]:RADar:PULSe[1]|2--32766:OFFTime(?)	Set the pulse OFF time
4	[:SOURce]:RADar:PULSe[1]|2--32766:ONTime(?)	Set pulse width
4	[:SOURce]:RADar:PULSe[1]|2--32766:REPentition[:TYPE](?)	Set the pulse repetition frequency mode
4	[:SOURce]:RADar:PULSe[1]|2--32766:REPentition:COUNts(?)	Set the quantity of pulse repetition frequencies
4	[:SOURce]:RADar:PULSe[1]|2--32766:RISE[:TIME](?)	Set the pulse rise edge time
4	[:SOURce]:RADar:PULSe[1]|2--32766:SHAPe(?)	Set the pulse envelope pattern.
4	[:SOURce]:RADar:PULSe[1]|2--32766:SLIDing:PERCent(?)	Set the maximum slip range
4	[:SOURce]:RADar:PULSe[1]|2--32766:STAGger:PERiod(?)	Set the stagger period value
4	[:SOURce]:RADar:SEQuence:APPend	Append a row to the

		pulse train
4	[:SOURce]:RADar:SEQuence:CATalog ?	Query the saved radar profile
4	[:SOURce]:RADar:SEQuence:CLEar	Clear the pulse sequence table
4	[:SOURce]:RADar:SEQuence:CONFigure:LOAD	Load the specified radar profile
4	[:SOURce]:RADar:SEQuence:CONFigure:STORe	Save the current radar configuration to a file
4	[:SOURce]:RADar:SEQuence:DELete	Delete the last row of data in the current pulse train table
4	[:SOURce]:RADar:SEQuence:PULSe:COUNts?	Query the number of pulses in a pulse sequence
4	[:SOURce]:RADar:STATe(?)	Set the radar signal ON/OFF
4	[:SOURce]:WLAN:ANTenna:COORdinate[:MODE](?)	Set the antenna mapping coordinate system
4	[:SOURce]:WLAN:ANTenna:MODE(?)	Set antenna quantity

4	[:SOURce]:WLAN:BWIDth:MODE(?)	Set bandwidth
4	[:SOURce]:WLAN:CONFigure:LOAD	Call the WALN profile
4	[:SOURce]:WLAN:CONFigure:PRESet	Restore the WLAN configuration to factory settings
4	[:SOURce]:WLAN:CONFigure:STORe	Save the current settings
4	[:SOURce]:WLAN:FBLock:SEQuence:APPend	Append a row of default data to the end of the physical frame block configuration list
4	[:SOURce]:WLAN:FBLock:SEQuence:CLEar	Clear the physical frame block configuration list
4	[:SOURce]:WLAN:FBLock:SEQuence:DELete	Set the last row of the physical frame block configuration list
4	[:SOURce]:WLAN:FBLock:SEQuence:ECOunt?	Query the number of physical frame blocks whose state is ON in the physical frame block configuration list
4	[:SOURce]:WLAN:FBLock[1]|2--50:STATe(?)	Set the ON/OFF state of each

		physical frame block in the physical frame block configuration list.
4	[:SOURce]:WLAN:FILTer:CLIPping:LEVel(?)	Set waveform clipping level
4	[:SOURce]:WLAN:FILTer:CLIPping:MODE(?)	Set waveform clipping mode
4	[:SOURce]:WLAN:FILTer:CLIPping[:STATe](?)	Set waveform clipping state
4	[:SOURce]:WLAN:MARKer[1]|2|3:FBINdex(?)	Set the signal marker frame block index
4	[:SOURce]:WLAN:MARKer[1]|2|3:FESHift(?)	Set the signal marker falling edge shift
4	[:SOURce]:WLAN:MARKer[1]|2|3:FINDex(?)	Set the signal marker frame index
4	[:SOURce]:WLAN:MARKer[1]|2|3:MODE(?)	Set signal marking method
4	[:SOURce]:WLAN:MARKer[1]|2|3:PATTern(?)	Set predefined style
4	[:SOURce]:WLAN:MARKer[1]|2|3:PULSe:DIVider(?)	Set the divisor factor

4	[:SOURce]:WLAN:MARKer[1]|2|3:PULSe:FREQuency?	Query pulse frequency
4	[:SOURce]:WLAN:MARKer[1]|2|3:RATio:OFFTime(?)	Set number of sampling points in the OFF phase
4	[:SOURce]:WLAN:MARKer[1]|2|3:RATio:ONTime(?)	Set number of sampling points in the ON phase
4	[:SOURce]:WLAN:MARKer[1]|2|3:RESHift(?)	Set the signal marker rising edge shift
4	[:SOURce]:WLAN:SRATe?	Query the sampling rate of WLAN
5	[:SOURce]:WLAN:STATe(?)	Set WLAN ON/OFF

Annex B Zoom table of error message

Table 2 Local error message

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 unlocked	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.