1435 Series

Signal Generators

Program Control Manual

China Electronics Technology Instruments Co., Ltd.

This manual is applied to the following models of signal generators, based on firmware version 1.0 and above.

- 1435A signal generator (9kHz - 3GHz)
- 1435B signal generator (9kHz - 6GHz)
- 1435C signal generator (9kHz - 12GHz)
- 1435D signal generator (9kHz - 20GHz)
- 1435F signal generator (9kHz - 40GHz)
- 1435A-V signal generator (9kHz - 3GHz)
- 1435B-V signal generator (9kHz - 6GHz)

Options other than standard accessories are as follows:

- 115dB programmable step attenuator: extend the dynamic range of output power;
- Analog modulation: add the function of analogue modulation, including AM, FM, Φ M, low-frequency (LF) output;
- Pulse modulation: add the function of pulse modulation, with the minimum pulse width of 100ns;
- Narrow pulse modulation: add the function of pulse modulation, with the minimum pulse width of 20ns;
- Multi-function generator: add more rich analog modulation signal format;
- Low phase noise: optimize SSB phase noise, 10GHz@10kHz: -113dBc/Hz;
- High power output: increase the maximum output power;
- High stability time base option: internal time base aging rate;
- Calibration certificate: instrument calibration;
- N-type connector for RF output: N-type connector for RF output is applicable for 1435D;
- RF output moved to the rear panel: RF output on the rear panel;
- Portable handle: 3U handle;
- Rack mounting kit: mounting kit for upper cabinet;
- Aluminum alloy transport case: high strength light aluminum alloy transport case with handle and universal roller makes transportation convenient;
- English kits: English panel, English manual, English interface and English operating system;
- English options: English menu, English panel, etc., for export;

Version: A.1 2018.12, China Electronics Technology Instruments Co., Ltd.

Add.: No. 98, Xiangjiang Road, Qingdao Economic & Technological Development Zone, Shandong, China

Free customer service number: 800-868-7041

Technical support: 0532-86889847 86897262

Fax: 0532-86889056 86897258

Website: www.ceyear.com

Email: eiqd@ceyear.com

Postal code: 266555

Foreword

Thank you for choosing and using 1435 series signal generator developed and produced by China Electronics Technology Instruments Co., Ltd.! Integrating high, sophisticated and cutting-edge technologies, our products offer high cost performance among similar products.

With maximally meeting your requirements as our duty, we will provide you with high-quality measuring instruments as well as first-class after-sales service. With the consistent tenet of "high quality and considerate service", we promise to provide the users with satisfactory products and services.

Manual No.

2.827.1101CK

Version

A.1 2018.12

China Electronics Technology Instruments Co., Ltd.

Manual Authorization

The contents of this manual are subject to change without notice. The final right to interpret the contents and terms used in this manual belongs to China Electronics Technology Instruments Co., Ltd.

The copyright of this manual belongs to China Electronics Technology Instruments Co., Ltd. No organization or individual is allowed to modify or alter the contents of this manual or copy or disseminate this manual for profit without our authorization. China Electronics Technology Instruments Co., Ltd. reserves the right to take legal actions against any infringer.

Product Warranty

The warranty period of this product is 18 months from the date of shipment. During the warranty period, the manufacturer will repair or replace damaged parts according to actual conditions. 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 Certification

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. China Electronics Technology Instruments Co., Ltd. already has the required qualifications and has passed the certification of ISO 9001 and ISO 14001 management systems.

Safety Precautions

Note

The "Note" symbol indicates some important information which will not cause danger. It reminds the user to pay attention to 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.

Tip

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.

Contents

1. Manual Navigation

This chapter introduces the function, chapter structure and main content of the programming manual for 1435 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 1435 series signal generator and application of SCPI. Meanwhile, in order to make it convenient for users to quickly master the remote control programming methods, some programming examples are listed, and the basic concept of I/O function library is introduced. To facilitate your skillful use of such instrument, please read carefully and follow this manual in advance for correct operation.

SCPI (Standard Commands for Programmable Instruments) define the standards and methods for remote control of the instrument, and are the remote control programming language for programmable electronic test and measuring instruments. SCPI are based on the IEEE-488.2 standard and form. Please refer to http://www.scpiconsortium.org for details. The manual describes the program control commands of 1435 series signal generators in details.

The chapters of the programming manual include:

- **Remote Control**

The methods for remote control of the instrument are summarized to make users get familiar with remote control quickly. It is divided into three parts: remote control basics, introducing program related concepts, software configuration, program port, SCPI, etc.; instrument port configuration method, introducing the connection method and software configuration method for program ports of 1435 series signal generator; I/O function library, introducing the basic concept of instrument driver and basic installation instructions of IVI-COM/IVI-C driver.

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

- **Annexes**

Necessary reference information related to program control of 1435 series signal generator is provided, including zoom table of SCPI and zoom table of errors.

1.2 Related Documents

Documents of 1435 series signal generator include:

- Quick Start Guide
- User's Manual
- Program 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 basic operation methods. Main chapters include:

- Get Prepared
- Typical Applications
- Get Help

User's Manual

This manual describes the functions and operation methods of the instrument in detail, including configuration, measurement, program 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 instrument. Main chapters include:

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

Programming manual

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 program control commands and methods of the instrument. Main chapters include:

- Remote Control
- Program Control Commands
- Programming Examples
- Error Description
- Annexes

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 1435 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 Program Control Interface

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

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

Remote control interface	VISA addressing string	Description
LAN (Local Area Network)	**VXI-11 protocol:** TCPIP::host_address[::LAN_device_name][::INSTR] **Raw socket protocol:** TCPIP::host_address::port::SOCKET	The controller realizes remote control by connecting the instrument with the network port on the rear panel of the instrument. For details of the protocol, please refer to: 2.1.1.1 LAN Interface
GPIB (IEC/IEEE Bus Interface)	GPIB::primary address[::INSTR] 4	The controller realizes remote control by connecting the instrument with the port on the rear panel of the instrument. The bus interface standard IEC 625.1/IEEE 418 is met. For details, please refer to: 2.1.1.2 GPIB 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. Knowledges about the LAN interface are introduced below:

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 192.168.12.XXX, where XXX is the value between 1 and 255. **The default network port number used by the signal generator for communication is 5025.**

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

TCPIP::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::INSTR

- To establish a raw socket connection, use:

TCPIP::192.1.2.3::5025::SOCKET

Tip

Multi-instrument identification method in program 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 and 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 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, and the data transmission rate can reach 8MBps, which is relatively 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 15m or twice the number of instruments in the system. Generally, the maximum length of transmission cables between devices should not exceed 2m.
- ➢ If multiple instruments are connected in parallel, a "live" cable is required.
- ➢ The terminal of the IEC bus cable should be connected to the instrument or master computer.

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**

Depending on the direction of transmission, instrument message can be divided into commands and instrument responses. 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 instrument for remote control of instrument functions and query of status information. Commands are divided into the following two categories:

- Based on the impact on the instrument:

  -- 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 program 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) Program control commands are oriented to test function rather than describing instrument operation;
2) Program control commands reduce the repetition of similar test function realization process, and

ensure the compatibility of programming.

3) Program messages are defined in layers that are hardware independent of the communication physical layer.
4) Program control commands are independent of programming methods and languages. The test program of SCPI is easy to transplant.
5) Program control commands are scalable and can adapt to different scale of measurement control.
6) The extensibility of SCPI makes it a "living" standard.

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

IEEE Standard 488.1-1987, IEEE Standard Digital Interface for

Programmable Instrumentation. New York, NY, 1998.

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

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

Standard Commands for Programmable Instruments (SCPI) VERSION 1999.0.

For program command set, classification and description of 1435 series signal generator, please refer to:

1) "3. Program control commands" in this manual;
2) "Annex A Zoom Table of SCPI command" in 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 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 status 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 program 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.

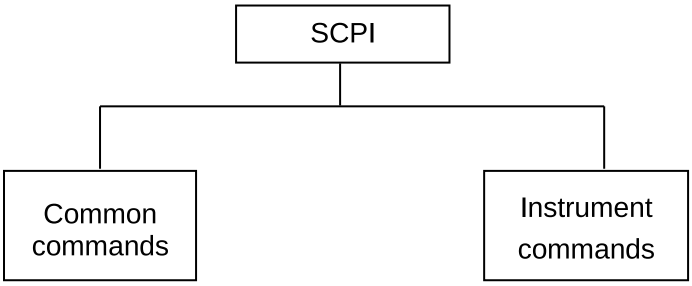


Figure 2.1 Types of SCPI

3) Instrument Command Syntax

A typical command consists of a keyword prefixed with a colon. The keyword is followed by parameters. Here is an example of a syntax statement.

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

In the example above, [:LEVel] in the command immediately follows :POWer 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 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 character represents the minimum set of characters required to execute the command.	[:SOURce]:FREQuency[:CW]?, FREQ is the short format part of the command.
Lowercase character of the command is optional; such flexible format is called "flexible listening". Please refer to "Command Parameters and Responses" 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. Please refer to the command path of "Command Tree" for more information.	:TRIGger[:SEQuence]:SOURce? TRIGger is the top-level keyword for the 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>}
A semicolon separates two adjacent commands without affecting the 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 or within keywords. However, you must use blank characters to separate commands and parameters without affecting 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:

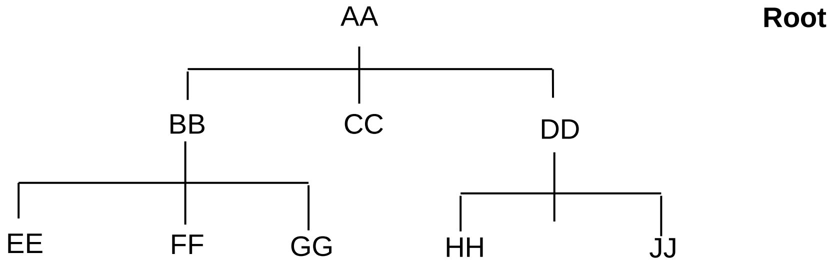


Figure 2.2 Simplified command tree

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 starting or *RST (resetting) the instrument, reset the current command path 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:

: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

Numerical parameters can be used in both instrument commands and common commands. A numeric parameter receives all the usual decimal counting methods, including signs, decimals, and scientific notation. If a device only accepts a specified numeric type, such as an integer, it will automatically round up the received numeric parameters.

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 millivolts
10DEG	10 degrees
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.

Examples 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 status register, most of the response data returned are of integer type.

13Examples 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 mode is millimeter wave source module

Digital Boolean 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 10 dBm (or a value of the equivalent value of the currently selected unit, such as DBUV or DBUVEMF) with a hexadecimal value of 000A.

:POW #H000A

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

7) Command Line Structure

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 belonging to the MEM subsystem and the second to the FREQ subsystem. If adjacent commands belong to the same subsystem, the command paths is partially repeated and the commands can be abbreviated. For example:

FREQ:CW 5GHZ; FREQ:OFFSet 3GHz

The command line contains two commands, both of which belong to the FREQ subsystem and have the same level 1. Therefore, the second command can start below FREQ, and the colon at the beginning of the command can be omitted. It can be abbreviated to the following command line:

FREQ:CW 5GHz; OFFS 3GHz

8) 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.

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)

A variable value is returned for the following command:

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

The returned result may be the current starting 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.**

Tip

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:

➢ **The controller program applies multithreading**

Multithreading is used for synchronization of waiting for command completion and user interface and program control, i.e., waiting in a separate thread for *OPC? completion without blocking GUI or program thread 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 to ESE BIT0; Set to SRE BIT5; 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 to OPC mask bit (bit0) of ESE: *ESE 1;
2) Set to 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 to 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 to OPC mask bit (bit0) of ESE: *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 status.

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

- *** OPC? and short timeout**

1) Send only overlapping command instead of *OPC, *OPC or *WAI
2) Send "<short timeout>; *OPC?" in timer for cyclic query of the operation status.;
3) The returned value (LSB) 1 indicates completion of the overlapping command. 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 Status Reporting System

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

2.1.5.1 Structure of Status Register

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

As shown in the following figure, the status information is of hierarchical structure.

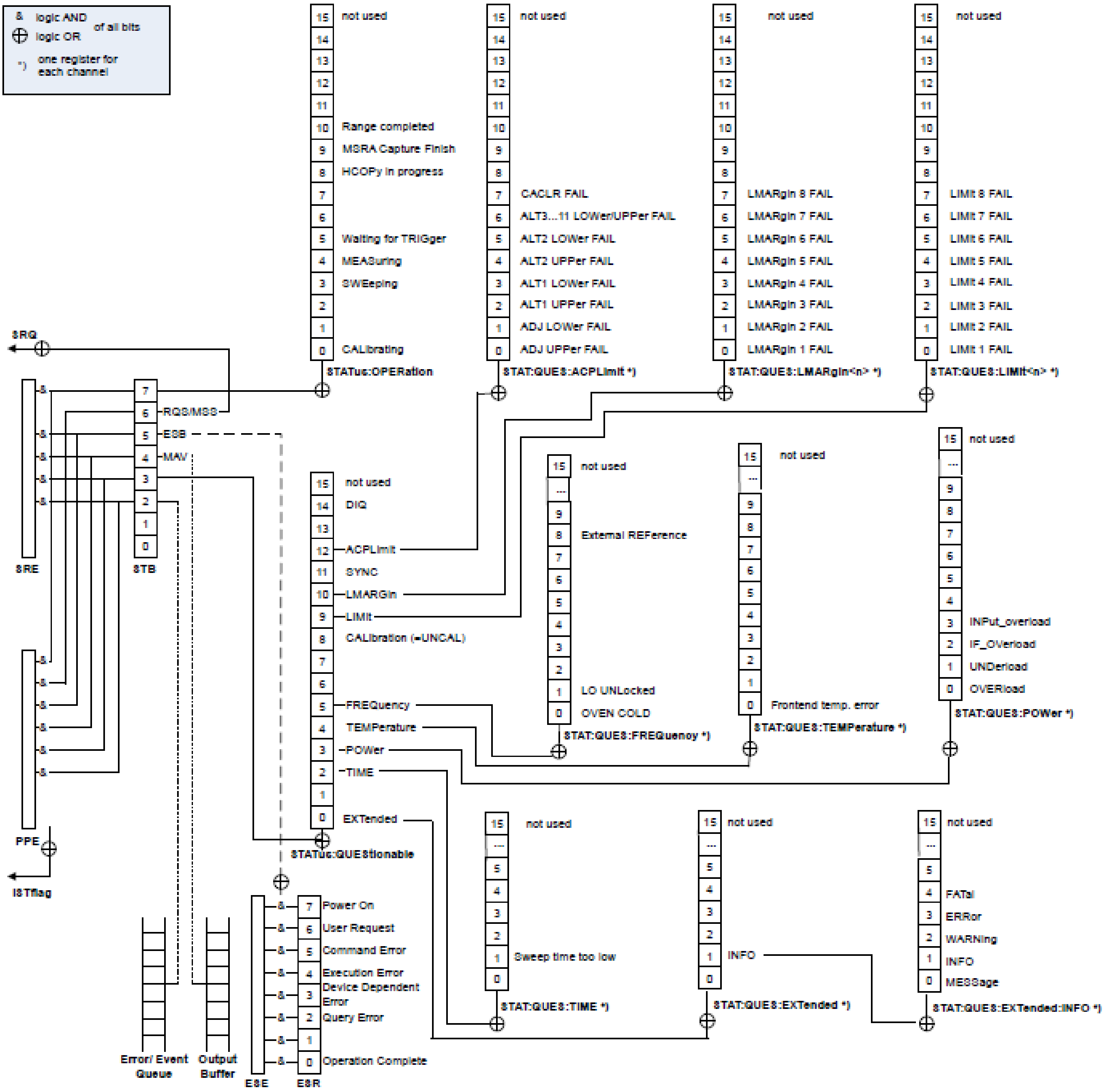


Figure 2.3 Hierarchical structure of status register

The register classification is described 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 programming manual for details).

3) IST, PPE

Similar to SRQ, as an individual bit of IST symbol ("Individual STatus"), it is composed of all the statuses of the instrument. Related parallel poll enable register (PPE) determines which bits of the STB act on the IST symbol.

4) Output buffer

The message returned by the instrument to the controller is stored. It is not part of the status reporting system, but determines the value of the MAV bit of STB.

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

Tip

SRE, ESE

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 Structure of SCPI Status Register

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

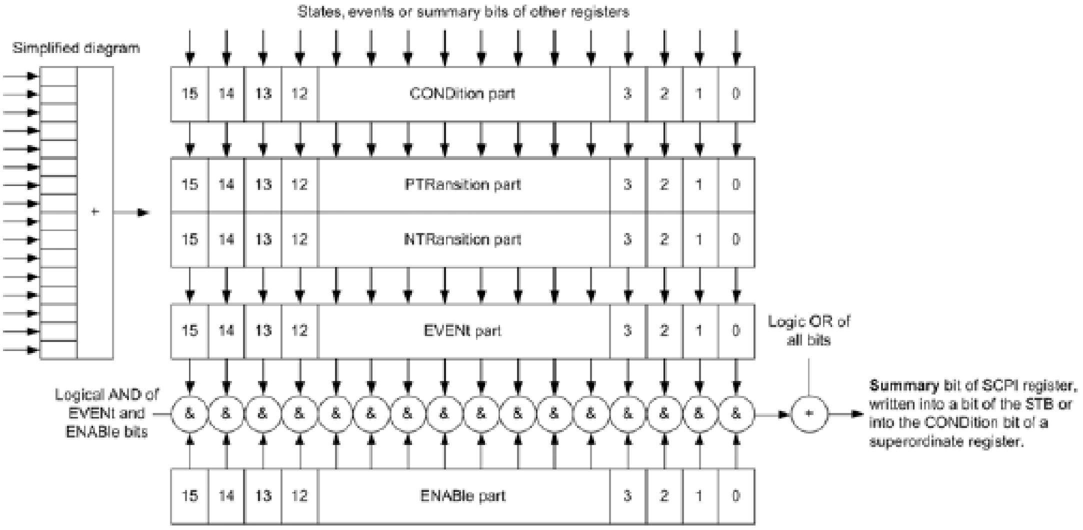

States, events or summary bits of other registers
Simplified diagram
+
15 14 13 12 CONDition part 3 2 1 0
15 14 13 12 PTRansition part 3 2 1 0
15 14 13 12 NTRansition part 3 2 1 0
15 14 13 12 EVENt part 3 2 1 0
Logic OR of all bits
Logical AND of EVENt and ENABle bits
&
+
Summary bit of SCPI register,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

Figure 2.4 Structure of status register

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

- **Condition register**

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

- **Positive/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 instrument and read by the user. The value will be cleared after reading. The value of this part is often equal to the value of the entire register.

- **Enable register**

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

- enable bit = 0: the related event bit does not act on the 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 instrument automatically generates the summary bit for each register so that events can cause different levels of service requests.

2.1.5.3 Description of Status Register

The status registers are detailed as follows:

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

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

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

Table 2.6 Description of status byte

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

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

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

Table 2.7 Description of event status byte

Bit	Meaning
0	Operation completed Set to this bit when the previous commands have been executed and the command *OPC has been received.
1	Not used.
2	Query error Set to this bit when the controller reads the instrument data without sending the query command, or sends a new command before reading the query data. It indicates that there is a query error, for which the query cannot be executed.
3	Instrument error Set to this bit when there is an instrument error. Error code range: -300 - -399, or positive error code. Specific errors can be found in relevant information in the error queue.
4	Execution error Set to this bit when a syntactically correct command is received but cannot be executed, and an error with code ranging from -200 to -300 is generated in the error queue.
5	Command error Set to this bit when the syntax of the command received is incorrect. Error code range: -100 - -200.

	Specific errors can be found in relevant information in the error queue.
6	User request Set to this bit when the instrument is switched to manual control mode.
7	Power ON Set to this bit when the instrument power is turned on.

3) Status: inquiry register

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

Table 2.8 Description of status: inquiry register

Bit	Meaning
0-2	Not used.
3	Set to this bit when the local power setting is wrong.
4	Set to this bit when the time base is not hot.
5	Set to this bit in case of frequency error of the local oscillator or reference frequency error of any active path.
6	Not used.
7	Set to this bit in case of setting error of local modulation.
8	Set to this bit when the instrument is not calibrated (the prompt "Not calibrated" is displayed on the interface).
9	Set to this bit in case of self-test error.
10-14	Not used
15	The bit is always 0.

Tip

Query register

Status: the inquiry register has collected the information of all lower sub-registers (for example, bit2 has collected all time related information). Since each path corresponds to a separate sub-register, in case of a status bit error of the inquiry register, it is required to go back to the sub-register of the path to check for the specific error source. By default, the sub-register status being queried belongs to the currently selected path.

4) Status: inquiry: frequency 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]?.

The register is described in Table 2.9 below.

Table 2.9 Description of status: inquiry: frequency register

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

5) Status: inquiry: power register

This register contains information about power overload when operating the instrument. Each active path corresponds to a separate power register. The register value may be read by using the command STATus:QUEStionable: POWer:CONDition? or STATus:QUEStionable:POWer [:EVENt]?. POWer [:EVENt]?.

The register is described in Table 2.10 below.

Table 2.10 Description of status: inquiry: power register

Bit	Meaning
0	Not used
1	If the amplitude is unstable, the bit is 1, meaning that the power ALC loop is out of lock and the power is inaccurate.
2-14	Not used.
15	**The bit is always 0.**

2.1.5.4 Application of Status Reporting System

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

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) Parallel query of all instruments;
4) Program command to query the status of specific instruments;
5) 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. Please refer to Section 4.2.1.1. "Service Request" for specific routines.

2) Serial query

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

3) Parallel query

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

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

4) Query instrument status

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

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

The returned value of the 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:

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

Figure 2.5 Relationship between the bit and the weight

5) Error Queue

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

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

2.1.5.5 Reset Status Reporting System

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

Table 2.11 Reset status reporting system

Event / Function	Power ON/OFF (Powered status cleared)		DCL, SDC (Instrument cleared, instrument selected to be cleared)	*RST or SYSTem: PRESet	STATus: PRESet	*CLS
	0	1				
Clear STB, ESR	—	Yes	—	—	—	Yes
Clear SRE, ESE	—	Yes	—	—	—	—
Clear PPE	—	Yes	—	—	—	—
Clear the event part of the register	—	Yes	—	—	—	Yes
Clear the enable part of the operation and inquiry registers. Fill the enable part of other registers with 1.	—	Yes	—	—	Yes	—

Fill the positive transition part with 1. Clear the negative transition part.	—	Yes	—	—	Yes	—
Clear the error queue	Yes	Yes	—	—	—	Yes
Clear the output buffer	Yes	Yes	Yes	—	—	—
Clear the command processing and input buffers	Yes	Yes	Yes	—	—	—

2.1.6 Programming Precautions

1) **Please initialize the instrument status 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 instrument. The controller program in the test system should guide the instrument 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 instrument instead of the status 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 instrument.

2.2 Instrument Program Port and Configuration

2.2.1 LAN

SICL-LAN is used to control 1435 series signal generator in Local Area Network (LAN).

Note

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 1435 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 1435 series vector generator and external controller (computer) to the LAN with network cable, as shown in Figure 2.6:

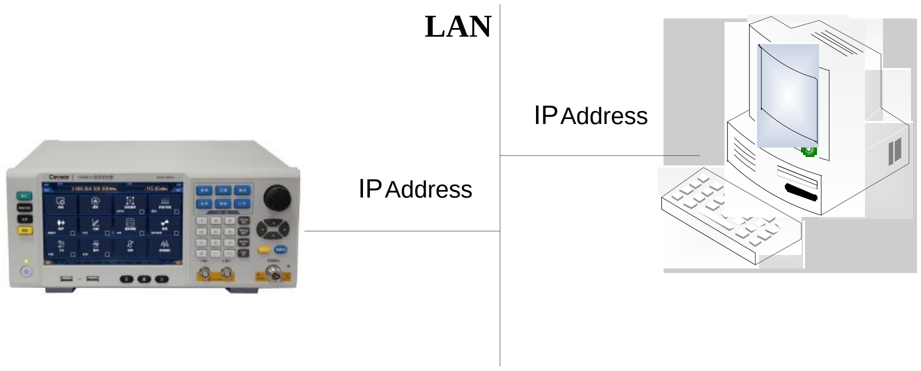
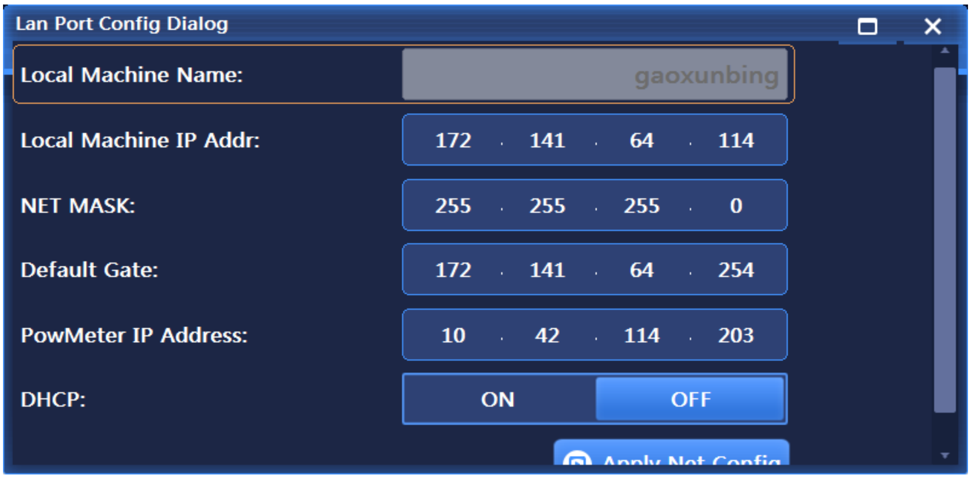


Figure 2.6 LAN interface connection

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



Lan Port Config Dialog

Local Machine Name:	gaoxunbing
Local Machine IP Addr:	172 . 141 . 64 . 114
NET MASK:	255 . 255 . 255 . 0
Default Gate:	172 . 141 . 64 . 254
PowMeter IP Address:	10 . 42 . 114 . 203
DHCP:	ON [✓ OFF]

Figure 2.7 LAN interface setting

Note

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 1435 series signal generator with external controller (computer), as shown in Figure 2.8:

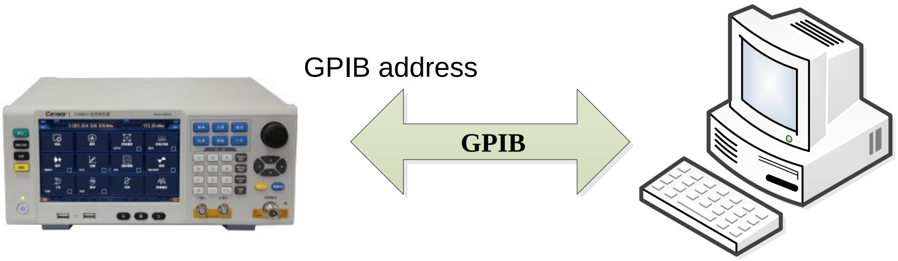
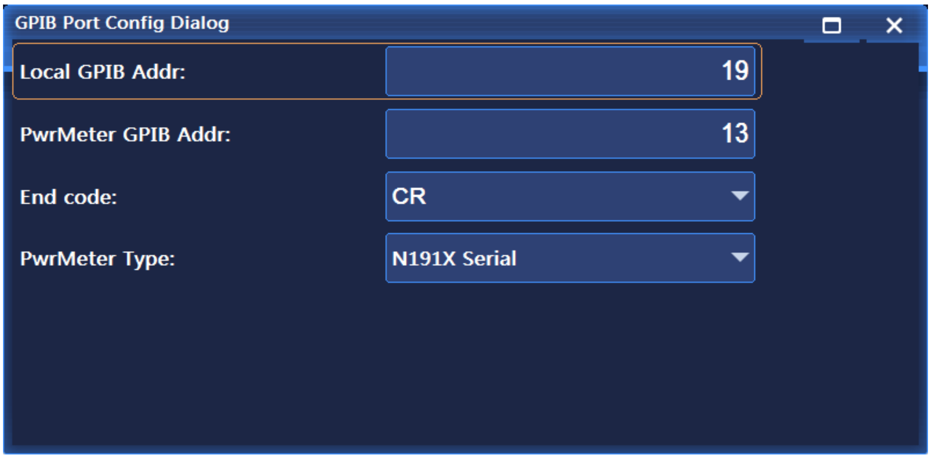


Figure 2.8 GPIB interface connection

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 2.9, and then use the number key on the front panel to modify in the GPIB address input box of the machine.


GPIB Port Config Dialog

Local GPIB Addr: 19

PwrMeter GPIB Addr: 13

End code: CR

PwrMeter Type: N191X Serial

Figure 2.9 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 instrument 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 instrument, reading from the instrument, writing to the instrument and triggering the instrument, etc. Residing in the computer, it is the bridge and link between the computer and the instrument. 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 2.10.

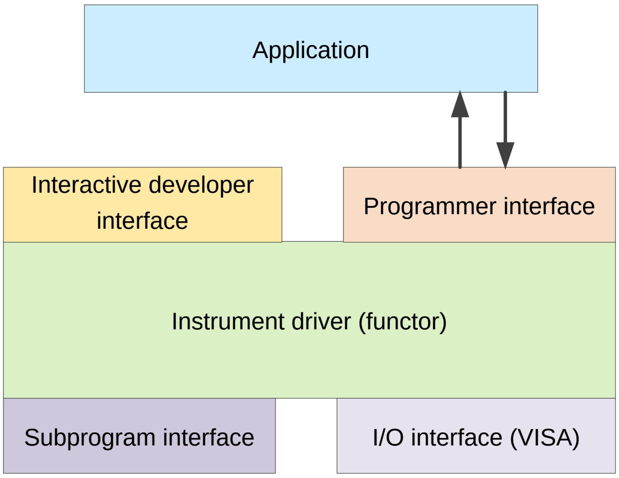


Figure 2.10 Structure model of instrument driver

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.

Tip

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.

Tip

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 1435.
- 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 status registers, status reports, synchronization, data storage and other common functions. The use and function of common commands apply to different instruments. All common commands may be identified by the first "*" in the command word, and are defined in detail in IEEE488.2.

IEEE488.2 common command is interpreted and explained below.

Tip

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.

*IDN?

Function description: return the instrument identification.

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

For example: CETC41,1435,2017008,1.0.11

Note: For query only.

*RCL <Value>

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

Parameter description: range [0, 99].

Note: For setting only.

*RST

Function description: this command is used to complete the signal generator reset function

Note: 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.

Parameter description: range [0, 99].

Note: For setting only.

*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.

Note: For setting only.

*ESE <Value>

Function description: enable the event status. Set the event status enable register. Query the register to return a decimal value.

Parameter description: range [0, 255].

*ESR?

Function description: read the decimal value of the event status register, and then set the register value to zero.

Returned value: range [0, 255], indicating calibration error.

Note: For query only.

***STB?**

Function description: this command is used to query the status byte register in the status reporting system.

Returned value: range [0, 127].

Note: For query only.

***TRG**

Function description: this command is used when the signal generator selects the bus as the trigger source.

Note: 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.

Returned value: 0: the self-test is passed;

1: the self-test fails.

Note: For query only.

3.3 Instrument Subsystem Command

This section details the subsystem commands of 1435 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[:STATe] <State>

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

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

Query format: OUTPut[:STATe] ?

Parameter Description:

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

ON | 1: RF output

OFF | 0: RF OFF.

Example: OUTPut 1 set RF output of the signal generator。

Reset state: 0

Key path: [RF ON/OFF]

:OUTPut:MODulation[:STATe] <State>

Function description: This command is used to set the modulation state.

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

Query format: OUTPut:MODulation[:STATe]?

Parameter Description:

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

ON | 1: Modulation ON

OFF | 0: modulation OFF.

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

Reset state: 0

Key path: [Modulation ON/OFF]

3.3.2 FREQuency Subsystem

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

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

[:SOURce]:FREQuency[:CW] <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 “:FREQuency:MODE” for setting of other frequency generation modes.

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

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

Parameter Description:

<Frequency> output frequency in continuous wave mode.

Model	Range
1435A	[9kHz ~ 3GHz]
1435A-V	[9kHz ~ 3GHz]
1435B	[9kHz ~ 6GHz]
1435B-V	[9kHz ~ 6GHz]
1435C	[9kHz ~ 12GHz]
1435D	[9kHz~20GHz]
1435F	[9kHz~40GHz]

Example: [:SOURce]:FREQuency 10GHz set the point frequency of the signal generator to 10GHz.

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

Key path: [Frequency] —> [Continuous wave]

[:SOURce]:FREQuency:MODE <Mode>

Function description: Set the frequency generation mode of the signal generator.

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

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

Parameter Description:

<Mode> discrete data, frequency generation mode to be configured. 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.

Please refer to the commands “:FREQuency[:CW]” and “:FREQuency[:FIXed]” for setting of point frequency.

STEP this parameter is used to set the current frequency generation to step sweep mode.

LIST set the frequency generation to list mode. If the current list is empty, the signal generator will remind that the list is empty. It will start the sweep only when at least one sweep point is stored in the list.

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

Reset state: CW

[:SOURce]:FREQuency:MULTiplier <FreqMult>

Function description: This command is used to set multiplier factor for the source frequency. When the frequency multiplier is set to a value greater than 1, the multiplier indicator "Multiplier" will be displayed above the frequency display 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]:FREQuency:MULTiplier <val>

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

Parameter Description:

<FreqMult> multiplier factor.

Range: 1[1, 36].

Example: FREQuency: MULTiplier 8 the multiplier factor of the signal generator is 8.

Reset state: 1

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

[:SOURce]:FREQuency:OFFSet <FreqOffs>

Function description: When the frequency offset is not set to zero, the offset indicator "Offset" will be displayed above the frequency display area, and the displayed value becomes the frequency after adding the offset. At this time, the displayed frequency value = RF output frequency value * multiplier factor + frequency offset, but the real frequency output is still the frequency before multiplying the multiplier factor and adding the frequency offset. When the frequency offset is set to zero, the indicator will disappear.

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

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

Parameter Description:

<FreqOffs> frequency offset.

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

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

Reset state: 0Hz

Key path: [Frequency] —> [Base Config] —> [Freq Offset]

[:SOURce]:FREQuency:REFerence <FreqRef>

Function description: This command is used to set frequency reference, and the set value may be used normally when frequency reference is set to ON state. Please refer to the command “: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, 0Hz 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]:FREQuency:REFerence <val>

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

Parameter Description:

<FreqRef> frequency reference.

Model	Range
1435A	[0Hz - 3GHz]
1435A-V	[9kHz~3GHz]
1435B	[9kHz~6GHz]
1435B-V	[9kHz~6GHz]
1435C	[9kHz~12GHz]
1435D	[9kHz~20GHz]
1435F	[9kHz ~ 40GHz

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

Reset state: 0Hz

Key path: [Frequency] —> [Base Config] —> [Freq Ref]

[:SOURce]: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 above the frequency display area. The frequency value displayed in the frequency display area is based on the frequency reference. Please refer to ":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.

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

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

Parameter Description:

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

ON | 1: frequency reference ON

OFF | 0: frequency reference OFF.

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

Reset state: 0

Key path: [Frequency] —>[Base Config] —>[Freq Ref Switch]

[:SOURce]:FREQuency:STARt <StartFreq>

Function description: This command is used to set the step sweep start frequency of the instrument. Please refer to the command ":FREQuency:STOP".

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

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

Parameter Description:

<StartFreq> sweep start frequency.

Model	Range
1435A	[9kHz~3GHz]
1435A-V	[9kHz~3GHz]
1435B	[9kHz~6GHz]
1435B-V	[9kHz~6GHz]
1435C	[9kHz~12GHz]
1435D	[9kHz~20GHz]
1435F	[9kHz~40GHz

Example: FREQuency:STARt 1MHz the step sweep start frequency of the signal generator is 1MHz.

Reset state: 9kHz

Key path: [Sweep] —>[Step Sweep] —>[Freq Start]

[:SOURce]:FREQuency:STOP <StopFreq>

Function description: This command is used to set the step sweep stop frequency of the instrument. Please refer to the command.“FREQuency:STARt”.

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

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

Parameter Description:

<StopFreq> sweep stop frequency.

Model	Range
1435A	[9kHz~3GHz]
1435A-V	[9kHz~3GHz]
1435B	[9kHz~6GHz]
1435B-V	[9kHz~6GHz]
1435C	[9kHz~12GHz]
1435D	[9kHz~20GHz]
1435F	[9kHz~40GHz]

Example: FREQuency:STOP 100MHz the step sweep stop frequency of the signal generator is 100MHz.

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

Key path: [Sweep] —>[Step 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]:POWer:ALC:LEVel <AlcLevel>

Function description: This command is used to set the ALC level value when the attenuator is set to manual. Please refer to the command ":POWer:ATTenuation:AUTO" for selection of manual/automatic mode for the attenuator.

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

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

Parameter Description:

<AlcLevel> ALC level.

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

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

Reset state: 0dBm

Key path: [Amplitude] —> [Attenuation] —>[ALC]

[:SOURce]: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. Please refer to the command ":POWer:ALC[:STATe]" for ALC loop state.

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

Query format: [:SOURce]: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.

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

Reset state: Manual

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

[:SOURce]: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

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

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

Parameter description:

<State > discrete data. The values of power stabilization mode are as follows:

INTernal: the power stabilization mode is internal

EXTernal: the power stabilization mode is external diode detection.

Example: POWer:ALC:SOURce INT the stabilization mode of the signal generator is internal.

Reset state: Internal

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

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

Function description: This command is used to set the coupling coefficient of external detection. When the power stabilization mode is external diode detection, this command is used to set the coupling factor to be used for external stabilization. Please refer to the command ":POWer:ALC:SOURce" for power stabilization mode.

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

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

Parameter description:

<CouplingValue> coupling coefficient of external detection.

Range: 16dB[-90dB, +90dB].

Example: POWer:ALC:SOURce:EXTernal:COUPling 16dBm the coupling factor for external stabilization is 16dBm.

Reset state: 16.00dBm

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

[:SOURce]: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.

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

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

Parameter description:

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

ON | 1: ALC loop ON

OFF | 0: ALC loop OFF

Example: POWer:ALC 1. This example shows that the ALC loop is set to OFF state.

Reset state: 1

Key path: [Amplitude] —> [ALC Loop] —> [ALC Loop State: ALC-ON/ALC-OFF]

[:SOURce]:POWer:ATTenuation <Atten>

Function description: This command is used to set the power attenuation of the mechanical attenuator of the signal generator. Only when the attenuator is kept in the manual state can the value set by this command be activated. Please refer to the command ":POWer:ATTenuation:AUTO" for setting of auto/manual attenuation.

The minimum attenuation step value set by this command is 5dB, that is, the user can only set the attenuation value as 0dB, 5dB, 10dB and 15dB, with 5dB as the step 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]:POWer:ATTenuation <value>

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

Parameter description:

<Atten> power attenuation.

Range: 115dB[0dB, 115dB].

Example: POWer:ATTenuation 15dB the attenuation value is 15dB.

Reset state: 115dB

Key path: [Amplitude] —>[Atten Config] —>[Attenuation]

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

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

Parameter description:

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

ON | 1: Auto attenuation

OFF | 0: manual attenuation.

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

Reset state: 1

Key path: [Amplitude] —> [Atten Config] —> [Attenuation coupling: auto/manual]

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

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

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

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

Parameter description:

<Ampl> power level.

Range: -135 DBM [-135 DBM, +30dBm].

Example: POWer 0dBm the power output level is 0dBm.

Reset state: -115dBm

Key path: [Amplitude]

[:SOURce]: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 above the power display area, 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]:POWer[:LEVel][:IMMediate]:OFFSet <value>

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

Parameter description:

<PowOffset> power offset.

Range: 0dB [-100db, +100dB].

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

Reset state: 0dB

Key path: [Amplitude] —> [Base Config] —> [Ampl Offset]

[:SOURce]:POWer:REFerence <PowRef>

Function description: When the power reference is set to ON state, the power reference value may be set. Please refer to the command ":POWer:REFerence:STATe" for power reference state. When the power reference is set to ON state, the indicator "*" will be displayed in the power display area, 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]:POWer:REFerence <value>

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

Parameter description:

<PowRef> power reference.

Range: 0dBm [-135dBm, +30dBm].

Example: POWer:REFerence -10dBm the power reference is -10dBm.

Reset state: 0dBm

Key path: [Amplitude] —> [Base Config] —> [Ampl Ref]

[:SOURce]: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, the power reference value is not zero, and the power level of the signal generator is changed, the power value displayed in the power display area is based on the power reference. Please refer to the command ":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.

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

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

Parameter description:

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

ON | 1: Power reference ON

OFF | 0: power reference OFF.

Example: POWer:REFerence:STATe 1 power reference ON.

Reset state: 0

Key path: [Amplitude] —>[Base Config] —>[Ampl Ref ON/OFF]

[:SOURce]:POWer:STEP <PowStep>

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

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

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

Parameter description:

<PowStep> power reference value.

Range: 0.10dB [0.01dB, 20dB].

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

Reset state: 0.10dB

Key path: [Amplitude] —> [Base Config] —> [Ampl Step]

[:SOURce]: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: 100Hz, 1kHz, 10kHz and 100kHz.

Note:

1. Please refer to the commands ":POWer:ALC:BANDwidth:AUTO" and ":POWer:ALC:BWIDth:AUTO" when the bandwidth of ALC is set to automatic mode; such setting is invalid when the bandwidth of ALC is not selected properly.

2. 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]:POWer:ALC:BANDwidth|BWIDth 100Hz|1kHZ|10kHz|100kHz

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

Parameter description:

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

100Hz	|	0: loop bandwidth: 100Hz
1kHz	|	1: loop bandwidth: 1kHz
10kHz	|	2: Loop bandwidth at 10kHz,
100kHz	|	3: loop bandwidth: 100kHz.

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

ALC loop bandwidth set to 100Hz.

Reset state: 10kHz

Key path: [Amplitude] —> [ALC Band]

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

Function description: This command is used to set the bandwidth selection mode of ALC (automatic leveling control) loop. When it is automatic, the signal generator will automatically select an appropriate ALC loop bandwidth. When it is manual, the ALC loop bandwidth is the value set by the user. Please refer to the command "[:SOURce]:POWer:ALC:BANDwidth|BWIDth " for details.

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

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

Parameter description:

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

ON | 1: ALC loop bandwidth is set to automatic mode.

OFF | 0: ALC loop bandwidth is set to manual mode.

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

Reset state: 1

Key path: [Amplitude] —> [ALC band: manual/auto]

[:SOURce]: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: POWer:SWEep ON|OFF|1|0

Query format: POWer:SWEep?

Parameter description:

<State> Boolean data; with values taken as follows:

ON|1: power sweep ON

OFF|0: power sweep OFF

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

Reset state: 0

Key path: **[Amplitude]** —> [Ampl Sweep] —> [Ampl Sweep ON/OFF]

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]:LIST:DIRection <Direc>

Function description: This command is used to set the sweep direction of the list. The user may choose two directions: Up and down, where the former means sweep from the first point in the list to the last point in the list, and the latter means sweep from the last point in the list to the first point in the list.

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

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

Parameter description:

<Direc> discrete data. The values of sweep direction are as follows:

UP sweep up from the first point in the list

DOWN sweep down from the last point in the list.

Example: LIST:DIRection UP set the list sweep direction to up.

Reset state: UP

Key path: [Sweep] —> [List Sweep] —> [Sweep direction: Forward/Backward]

[:SOURce]: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 a 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 of points entered by the user is less than the current list number, the points for which the dwell time is not entered are the current default. Note that the list needs to be filled in before setting, to ensure that the list is not empty. If it is empty, the query program will not respond. Please refer to "[:SOURce]:LIST:FILL:EXECute" for list filling.

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

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

Parameter description:

<Val> dwell time of list sweep point.

Range: 10ms [100us, 100s].

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…] —> [Time]

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

Function description: This command is used to set the continuous wave frequency 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 points entered by the user is less than the current list number, the points for which the list frequency is not entered are the current default.

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

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

Parameter description:

<Val> list sweep point frequency.

Model	Range
1435A	[9kHz~3GHz]
1435A-V	[9kHz~3GHz]
1435B	[9kHz~6GHz]
1435B-V	[9kHz~6GHz]
1435C	[9kHz~12GHz]
1435D	[9kHz~20GHz]
1435F	[9kHz~40GHz]

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...] --> [Frequency]

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

Function description: This command is used to set the number of list points to generate.

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

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

Parameter description:

<Num> number of list sweep points

Range: 3[2, 801].

Example: LIST:FILL:POINts 100 set 100 frequency points for the list.

Reset state: 3

Key path: [Sweep] —> [List Sweep] —> [Insert Counts]

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

Function description: This command is used to set the list sweep start frequency , which is used in conjunction with the list stop frequency and list points to generate list sweep points. Please refer to the commands ":LIST:FILL:STOP", ":LIST:FILL:POINts", :LIST:FILLL:POWer" and "LIST:FILL:DWELl" for setting of list stop frequency and sweep points.

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

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

Parameter description:

<FreqStart> list sweep start frequency.

Model	Range
1435A	[9kHz~3GHz]
1435A-V	[9kHz~3GHz]
1435B	[9kHz~6GHz]
1435B-V	[9kHz~6GHz]
1435C	[9kHz~12GHz]
1435D	[9kHz~20GHz]
1435F	[9kHz~40GHz]

Example: LIST:FILL:STARt 300MHz set the list sweep start frequency to 300MHz.

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

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

Function description: This command is used to set the list sweep stop frequency, which is used in conjunction with the list start frequency and list points to generate list sweep points. Please refer to the commands ":LIST:FILL:STARt", ":LIST:FILL:POINts", ":LIST:FILLL:POWer" and "LIST:FILL:DWELl" for setting of list start frequency and sweep points.

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

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

Parameter description:

<FreqStop> list sweep stop frequency.

Model	Range
1435A	[9kHz~3GHz]
1435A-V	[9kHz~3GHz]
1435B	[9kHz~6GHz]

1435B-V	[9kHz~6GHz]
1435C	[9kHz~12GHz]
1435D	[9kHz~20GHz]
1435F	[9kHz~40GHz]

Example: LIST:FILL:STOP 1GHz set the list sweep stop frequency to 1GHz.

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

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

Function description: This command is used to set the power of each sweep point in the current list. If the user needs to set a different offset for each sweep point in the list, it must enter a corresponding offset value for each point in the list. It is just required to enter the power offset value of the list sweep point in turn, separated by commas. If the number of points entered by the user is less than the current list number, the points for which the list offset power is not entered are the current default. Note that the list needs to be filled in before setting, to ensure that the list is not empty. If it is empty, the query program will not respond. Please refer to “[:SOURce]:LIST:FILL:EXECute” for list filling.

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

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

Parameter description:

<Val> list sweep point power offset.

Range: 0dBm [-100dB, +100dB].

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

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

Key path: [Sweep] —> [List Sweep] —> [Edit list...] —> [Offset]

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

Function description: This command is used to set the trigger source of the start list sweep. There are four trigger sources including automatic, bus, external and trigger key. Please refer to “TRIGger[:SEQuence]:SOURce” for relevant command.

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

Query format: [:SOURce]: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 after completing a sweep.
BUS	bus; a trigger is performed by a group from GPIB or after receiving the "*TRG" command.
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.

Example:: LIST:TRIGger:SOURce BUS set the list sweep trigger source to bus.

Reset state: IMM

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

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

Function description: This command is used to set the list sweep power offset, which is used in conjunction with the list start frequency and list points to generate list sweep points. Please refer to the commands ":LIST:FILL:STARt", ":LIST:FILL:POINts", ":LIST:FILLL:STOP" and "LIST:FILL:DWELl" for setting of list start frequency and sweep points.

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

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

Parameter description:

<val> power offset of all list sweep points.

Range: 0dBm [-100dB, +100dB]

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

Key path: [Sweep] —> [List Sweep] —> [All List Ampl Offset]

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

Function description: This command is used to set the dwell time of all list sweep points, which is used in conjunction with the list start frequency and list points to generate list sweep points. Please refer to the commands ":LIST:FILL:STARt", ":LIST:FILL:POINts", ":LIST:FILLL:STOP" and "LIST:FILL:POWer" for setting of list start frequency and sweep points.

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

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

Parameter description:

<val> dwell time of all list sweep points.

Range: 10ms [100us, 100s]. ,

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

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

[:SOURce]:LIST:FIll:EXECute

Function description: This command is used to generate the list sweep points according to the start frequency, stop frequency, list points, all point power offset and all point dwell time set. Please refer to the commands ":LIST:FILL:STARt", ":LIST:FILL:POINts", ":LIST:FILLL:STOP", "LIST:FILL:POWer" and "LIST:FILL:STOP".

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

Parameter description:

Example: LIST:FILL:EXECute complete automatic list fill.

Key path: [Sweep] —> [List Sweep] —> [Auto Fill]

Description: For setting only.

[:SOURce]:LIST:DELete <Mode>

Function description: This command is used to delete the list sweep points.

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

Parameter description: Discrete data. The values of list points to be deleted are as follows:

CURRent current point

ALL all points

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

Key path: [Sweep] —> [List Sweep] —> [Edit List] —> [Del All]

Description: For setting only.

3.3.5 LFOutput Subsystem

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

[:SOURce]:LFOutput:AMPLitude <Ampl>

Function description: This command is used to set the signal amplitude output from the LF output BNC connector of the signal generator.

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

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

Parameter description:

<Ampl> LF output signal amplitude.

Range: 2.000Vpp[0.002Vpp, 5.000Vpp].

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

Reset state: 2.000VPP

Key path: [Frequency] —> [LF Out] —> [LF Ampl]

[:SOURce]:LFOutput:FREQuency <Frequency>

Function description: This command is used to set the LF output frequency. Please refer to "LFOutput:SHAPe" for selection of LF waveform.

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

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

Parameter description:

<Frequency> LF output signal frequency.

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

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

Reset state: 400Hz

Key path: [Frequency] —> [LF Out] —> [LF]

[:SOURce]:LFOutput:SHAPe:RAMP <Mode>

Function description: This command is used to set the signal output type when the LF signal waveform is zigzag, including up and down. Please refer to the command ":LFOutput:SHAPe" for selection of LF waveform.

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

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

Parameter description:

<Mode> Discrete data. The values of zigzag signal type are as follows:

POSitive	up
NEGative	down.

Example: LFOutput:RAMP NEGative the LF zigzag is down.

Reset state: POS

Key path: [Frequency] —> [LF Out] —> [LF Waveform] —> [Zigzag>>]

[:SOURce]:LFOutput:SHAPe <Mode>

Function description: This command is used to set the LF signal output waveform. Users may select sine, square, triangle and Zigzag.

Setting format: [:SOURce]:LFOutput:SHAPe SINE|SQUare|TRIangle|RAMP|FUNCtion[1]|FUNCtion2|DUAL|SWEep|NOISe[1]|NOISe2| DC

Query format: [:SOURce]:LFOutput: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	Zigzag wave,
FUNCtion[1]	Function generator 1
FUNCtion2	Function generator 2
DUAL	Dual function generator
SWEep	Sweep function generator
NOISe[1]	Noise generator 1
NOISe2	Noise generator 2
DC	Direct current

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

Reset state: SINE

Key path: [Frequency] —> [LF Out] —> [LF Waveform]

[:SOURce]:LFOutput:STATe <State>

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

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

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

Parameter description:

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

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: [Frequency] —> [LF Out] —> [LF Out ON/OFF]

[:SOURce]:LFOutput:OFFSet <Offset>

Function description: This command is used to set the LF amplitude offset of the signal generator.

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

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

Parameter description:

<Offset> the values of LF DC offset are as follows:

Range: 0v[-5v, 5v]

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

Reset state: 0

Key path: [Frequency] —> [LF Out] —> [LF Offset]

[:SOURce]:LFOutput:DUAL:FUNCtion:AMPLitude:PERCent <val>

Function description: When the dual function generator is selected as LF output waveform, this command is used to set the amplitude ratio of the dual function generator relative to audio 1.

Setting format: [:SOURce]:LFOutput:DUAL:FUNCtion:AMPLitude:PERCent <val>

Query format: [:SOURce]:LFOutput:DUAL:FUNCtion:AMPLitude:PERCent?

Parameter description:

<val> the values of amplitude ratio relative to audio 1 are as follows:

Range: 50 [0,100]

Example: LFOutput:DUAL:FUNCtion:AMPLitude:PERCent 50 set the amplitude ratio relative to audio 1 to 50

Reset state: 50

Key path: [Frequency] —> [LF Out] —> [LF Waveform] —> [Dual Fun-Generator]

[:SOURce]:LFOutput:DUAL:FUNCtion[1]|2:FREQuency <Frequency>

Function description: When the dual function generator is selected as LF output waveform, this command is used to set the value of frequency 1 (default) or frequency 2 of the dual function generator.

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

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

Parameter description:

<Frequency> the **values** of frequency 1 or frequency 2 are as follows:

Range: 1kHz [0.001Hz, 1MHz]

Example: LFOutput:DUAL:FUNCtion:FREQuency 20kHz set frequency 1 to 20kHz.

Reset state: 1kHz

Key path: [Frequency] —> [LF Out] —> [LF Waveform] —> [Dual Fun-Generator]

[:SOURce]:LFOutput:DUAL:FUNCtion[1]|2:PERCent <val>

Function description: When the dual function generator is selected as LF output waveform and pulse is selected as the waveform of dual function generator, this command is used to set the pulse duty factor of the dual function generator relative to audio 1 (default) or audio 2.

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

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

Parameter description:

<val> the values of pulse duty factor of audio 1 (default) or audio 2 are as follows:

Range: 50 [0,100]

Example: LFOutput:DUAL:FUNCtion:PERCent 25 set the pulse duty factor of frequency audio 1 to 25%.

Reset state: 50%

Key path: [Frequency] —> [LF Out] —> [LF Waveform] —> [Dual Fun-Generator]

[:SOURce]:LFOutput:DUAL:FUNCtion:POFFset <val>

Function description: When the dual function generator is selected as LF output waveform, this command is used to set the phase offset of the dual function generator relative to audio 1.

Setting format: [:SOURce]:LFOutput:DUAL:FUNCtion:POFFset <val>

Query format: [:SOURce]:LFOutput:DUAL:FUNCtion: POFFset?

Parameter description:

<val> the values of phase offset relative to audio 1 are as follows:

Range: 0deg [0deg, 360deg]

Example: LFOutput: DUAL:FUNCtion:POFFset 90 set the phase offset relative to audio 1 to 90deg.

Reset state: 0deg

Key path: [Frequency] —> [LF Out] —> [LF Waveform] —> [Dual Fun-Generator]

[:SOURce]:LFOutput:DUAL:FUNCtion:SHAPe <Mode>

Function description: When the dual function generator is selected as LF output waveform, this command is used to set the output waveform of the dual function generator.

Setting format: [:SOURce]:LFOutput:DUAL:FUNCtion:SHAPe SINE|SQUare|TRIangle|RAMP|PULSe

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

Parameter description:

<Mode> Discrete data. The values of output waveform are as follows:

Sine	sine wave,
SQUare	square wave,
TRIangle	triangle wave,
Ramp	ramp wave,
PULSe	pulse

Example: LFOutput: DUAL:FUNCtion:SHAPe TRIangle set the output waveform to triangle.

Reset state: SINE

Key path: [Frequency] —> [LF Out] —> [LF Waveform] —> [Dual Fun-Generator]

[:SOURce]:LFOutput:DUAL:FUNCtion:SHAPe:RAMP <Mode>

Function description: When the dual function generator is selected as LF output waveform, this command is used to set the signal output type when the output waveform of the dual function generator is ramp, including up and down.

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

Query format: [:SOURce]:LFOutput:DUAL:FUNCtion:SHAPe:RAMP?

Parameter description:

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

POSitive	up

NEGative down.

Example: LFOutput:DUAL:FUNCtion:SHAPe:RAMP NEGative set the ramp to down.

Reset state: POS

Key path: [Frequency] —> [LF Out] —> [LF Waveform] —> [Dual Fun-Generator]

[:SOURce]:LFOutput:FUNCtion[1]|2:FREQuency <Frequency>

Function description: When the function generator 1|2 is selected as LF output waveform, this command is used to set the output frequency of function generator 1|2.

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

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

Parameter description:

<Frequency> LF **output** signal frequency.

Range: 1kHz[0.001Hz, 1MHz].

Example: LFOutput:FUNCtion2:FREQuency 20kHz set the output frequency of function generator 2 to 20kHz.

Reset state: 1kHz

Key path: [Frequency] —> [LF Out] —> [LF Waveform] —> [Function Generator 1|2]

[:SOURce]:LFOutput:FUNCtion[1]|2:PERCent <val>

Function description: When the function generator 1|2 is selected as LF output waveform and pulse is selected as the waveform of function generator 1|2, this command is used to set the pulse duty factor of function generator 1|2.

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

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

Parameter description:

<val> the values of pulse duty factor of function generator 1|2 are as follows:

Range: 50 [0,100]

Example: LFOutput: FUNCtion:PERCent 25 set the pulse duty factor of function generator 1 to 25%.

Reset state: 50%

Key path: [Frequency] —> [LF Out] —> [LF Waveform] —> [Function Generator 1|2]

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

Function description: When the function generator 1|2 is selected as LF output waveform, this command is used to set the output waveform of function generator 1|2.

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

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

Parameter description:

<Mode> Discrete data. The values of output waveform are as follows:

Sine sine wave,

SQUare	square wave,
TRIangle	triangle wave,
Ramp	ramp wave,
PULSe	pulse

Example: LFOutput:FUNCtion:SHAPe TRIangle set the output waveform of function generator 1 to triangle.

Reset state: SINE

Key path: [Frequency] —> [LF Out] —> [LF Waveform] —> [Function Generator 1|2]

[:SOURce]:LFOutput:FUNCtion[1]|2:SHAPe:RAMP <Mode>

Function description: When the function generator 1|2 is selected as LF output waveform, this command is used to set the signal output type when the output waveform of function generator 1|2 is ramp, including up and down.

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

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

Parameter description:

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

POSitive	up
NEGative	down.

Example: LFOutput:FUNCtion:SHAPe:RAMP NEGative set the ramp to down when the output waveform of function generator 1 is ramp.

Reset state: POS

Key path: [Frequency] —> [LF Out] —> [LF Waveform] —> [Function Generator 1|2]

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

Function description: When the noise generator 1|2 is selected as LF output waveform, this command is used to set the noise type of noise generator 1|2.

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

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

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:NOISe2:TYPe GAUSsian set the noise type of noise generator 2 to Gaussian.

Reset state: GAUS

Key path: [Frequency] —> [LF Out] —> [LF Waveform] —> [Noise Generator 1|2]

[:SOURce]:LFOutput:SWEep:FUNCtion:FREQuency:STARt <Frequency>

Function description: When the sweep function generator is selected as LF output waveform, this command is used to set the start frequency of the sweep function generator.

Setting format: [:SOURce]:LFOutput:SWEep:FUNCtion:FREQuency:STARt <val>

Query format: [:SOURce]:LFOutput:SWEep:FUNCtion:FREQuency:STARt?

Parameter description:

<Frequency> start frequency of sweep function generator.

Range: 1kHz[0.001Hz, 1MHz].

Example: LFOutput:SWEep:FUNCtion:FREQuency:STARt 20kHz set the start frequency of the sweep function generator to 20kHz.

Reset state: 1kHz

Key path: [Frequency] —> [LF Out] —> [LF Waveform] —> [Sweep Fun-Generator]

[:SOURce]:LFOutput:SWEep:FUNCtion:FREQuency:STOP <Frequency>

Function description: When the sweep function generator is selected as LF output waveform, this command is used to set the stop frequency of the sweep function generator.

Setting format: [:SOURce]:LFOutput:SWEep:FUNCtion:FREQuency:STOP <val>

Query format: [:SOURce]:LFOutput:SWEep:FUNCtion:FREQuency:STOP?

Parameter description:

<Frequency> stop frequency of sweep function generator.

Range: 1kHz[0.001Hz, 1MHz].

Example: LFOutput:SWEep:FUNCtion:FREQuency:STOP 20kHz set the stop frequency of the sweep function generator to 20kHz.

Reset state: 1kHz

Key path: [Frequency] —> [LF Out] —> [LF Waveform] —> [Sweep Fun-Generator]

[:SOURce]:LFOutput:SWEep:FUNCtion:SHAPe <Mode>

Function description: When the sweep function generator is selected as LF output waveform, this command is used to set the output waveform of the sweep function generator.

Setting format: [:SOURce]:LFOutput:SWEep:FUNCtion:SHAPe SINE|SQUare|TRIangle|RAMP

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

Parameter description:

<Mode> Discrete data. The values of output waveform are as follows:

Sine	sine wave,
SQUare	square wave,
TRIangle	triangle wave,
RAMP	ramp wave.

Example: LFOutput: DUAL:SWEep:SHAPe TRIangle set the output waveform to triangle.

Reset state: SINE

Key path: [Frequency] —> [LF Out] —> [LF Waveform] —> [Sweep Fun-Generator]

[:SOURce]:LFOutput:SWEep:FUNCtion:SHAPe:RAMP <Mode>

Function description: When the sweep function generator is selected as LF output waveform, this command is used to set the signal output type when the output waveform of the sweep function generator is ramp, including up and down.

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

Query format: [:SOURce]:LFOutput:SWEep:FUNCtion:SHAPe:RAMP?

Parameter description:

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

POSitive	up
NEGative	down.

Example: LFOutput:SWEep:FUNCtion:SHAPe:RAMP NEGative set the ramp to down.

Reset state: POS

Key path: [Frequency] —> [LF Out] —> [LF Waveform] —> [Sweep Fun-Generator]

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

Function description: When the sweep function generator is selected as LF output waveform, this command is used to set the sweep time of the sweep function generator.

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

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

Parameter description:

< Time > sweep time of sweep function generator.

Range: 0.1ms[0.01us, 40s].

Example: LFOutput:SWEep:FUNCtion:TIME 1s set the LF output signal frequency to 1s.

Reset state: 0.1ms

Key path: [Frequency] —> [LF Out] —> [LF Waveform] —> [Sweep Fun-Generator]

[:SOURce]:LFOutput:SWEep:FUNCtion:TRIGger:MODE <Mode>

Function description: When the sweep function generator is selected as LF output waveform, this command is used to set the trigger mode of the sweep function generator.

Setting format: [:SOURce]:LFOutput:SWEep:FUNCtion:TRIGger:MODE <val>

Query format: [:SOURce]:LFOutput:SWEep:FUNCtion:TRIGger:MODE?

Parameter description:

<Mode> Discrete data, with values taken as follows:

CONTinuous:	Continuous
SINGle:	Single

Example: LFOutput:SWEep:FUNCtion:TRIGger:MODE CONTinuous set the trigger mode of the sweep function generator to continuous.

Reset state: CONTinuous

Key path: [Frequency] —> [LF Out] —> [LF Waveform] —> [Sweep Fun-Generator]

[:SOURce]:LFOutput:SWEep:FUNCtion:TRIGger:PERiod <Time>

Function description: When the sweep function generator is selected as LF output waveform and the trigger type is timed trigger, this command is used to set the sweep timer period of the sweep function generator.

Setting format: [:SOURce]:LFOutput:SWEep:FUNCtion:TRIGger: PERiod <val>

Query format: [:SOURce]:LFOutput:SWEep:FUNCtion:TRIGger: PERiod?

Parameter description:

<Time> sweep timer period.

Range: 0.1ms[10ns, 40s].

Example: LFOutput:SWEep:FUNCtion:TRIGger: PERiod 1s set the sweep timer period of the sweep function generator to 1s.

Reset state: 0.1ms

Key path: [Frequency] —> [LF Out] —> [LF Waveform] —> [Sweep Fun-Generator]

[:SOURce]:LFOutput:SWEep:FUNCtion:TRIGger:TYPE <Mode>

Function description: When the sweep function generator is selected as LF output waveform, this command is used to set the trigger type of the sweep function generator.

Setting format: [:SOURce]:LFOutput:SWEep:FUNCtion:TRIGger:TYPE <val>

Query format: [:SOURce]:LFOutput:SWEep:FUNCtion:TRIGger:TYPE?

Parameter description:

<Mode> Discrete data, with values taken as follows:

IMMediate:	Auto
KEY:	Trigger key
BUS: :	Bus
INTernal:	Internal
EXTernal:	External
TIMer:	Timed trigger

Example: LFOutput:SWEep:FUNCtion:TRIGger:TYPE BUS set the trigger type of the sweep function generator to bus.

Reset state: IMMediate

Key path: [Frequency] —> [LF Out] —> [LF Waveform] —> [Sweep Fun-Generator]

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]:SWEep:DIRection <Direction>

Function description: This command is used to set the step sweep direction, including: up and down. Up represents the step sweep from the start frequency to the stop frequency, and down represents the sweep from the stop frequency to the start frequency.

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

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

Parameter description:

<Direction> discrete data. The values of step sweep direction are as follows:

UP | Up

DOWN | Down

Example: SWEep:DIRection DOWN the step sweep direction is down.

Reset state: UP

Key path: [Sweep] —> [Current Sweep Type>>] —> [Step>>] —>[Sweep direction: Forward/Backward]

[:SOURce]:SWEep: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 “SWEep:TRIGger:SOURce” for setting of trigger source.

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

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

Parameter description:

<Val> step sweep dwell time.

Range: 10.000ms[100us, 100s].

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

Reset state: 10.000ms

Key path: [Sweep] —> [Step Sweep] —> [Step Dwell]

[:SOURce]:SWEep:POINts <Num>

Function description: This command is used to set the number of step sweep points.

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

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

Parameter description:

<Num> number of step sweep points.

Range: 11[2, 801].

Example: SWEep:POINts 101 set the number of step sweep points to 101.

Reset state: 11

Key path: [Sweep] —> [Step Sweep] —> [Step Counts]

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

Function description: This command is used to set the trigger source to start the step sweep. There are four trigger sources: automatic, bus, external and trigger key.

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

Query format: [:SOURce]:SWEep: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 system will automatically trigger the next sweep after completing a sweep.

BUS bus, a trigger is performed by a group from GPIB or after receiving the "*TRG" command.

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.

Example: SWEep: TRIGger:SOURce BUS set the step sweep trigger mode to bus.

Reset state: IMM

Key path: [Sweep] —> [Step Sweep] —> [Step Trig]

[:SOURce]: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.

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

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

Parameter description:

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

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: LIST:RETRace 0 when the 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] —> [Mode] —> [Sweep ON/OFF]

[:SOURce]:SWEep:STEP:TYPE <Mode >

Function description: Set the step sweep mode.

Setting format: [:SOURce]:SWEep:TYPE LINEar|LOGarithm

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

Parameter description:

<Mode> Discrete data, with values taken as follows:

LINEar: Linear.

LOGarithm: Logarithm.

Example: LIST:SWEep:TYPE LINEar set the step sweep mode to linear.

Reset state: LINEar

Key path: [Sweep] —> [Step Sweep] —> [Step mode]

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

Function description: Set the start sweep trigger type.

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

Query format: [:SOURce]:SWEep:STARt:TRIGge?

Parameter description:

<Mode> Discrete data, with values taken as follows:

AUTO: Auto; the trigger signal is always true. When the sweep is set to ON state, the system will sweep automatically.

BUS: Bus; a trigger is performed by a group from GPIB or after receiving the "*TRG" command.

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

Example: SWEep:STARt:TRIGger BUS set the start sweep trigger to bus.

Reset state: AUTO

Key path: [Sweep] —> [Mode] —> [Trig Style of Start]

[:SOURce]:SWEep:MODE <Mode>

Function description: Set the sweep mode.

Setting format: [:SOURce]:SWEep:MODE CONTinuous|SINGle

Query format: [:SOURce]: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 to continuous.

Reset state: CONTinuous

Key path: [Sweep] —> [Mode] —> [Mode]

3.3.7 PULM Subsystem

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

The following commands are used to set the pulse modulation mode, including:

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

Function description: This command is used to perform a logical flip on the external input pulse signal, that is, when the external mode is selected as the pulse source, the pulse signal input from the pulse input port on the front panel of the signal generator is TTL high level signal, or flipped to TTL low level signal. Please refer to ":PULM:SOURce" for selection of pulse source.

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

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

Parameter description:

<Mode> Discrete data. The values of pulse input inverted state are as follows:

NORMal Pulse input inverted OFF; the pulse signal input is TTL

		high level.
	INVerted	Pulse input inverted ON; the pulse signal input is TTL low level.

Example: PULM:ENTernal:POLarity INV the external input pulse signal is flipped to TTL low level.

Reset state: NORM

Key path: [Analog Modulation] —> [Pulse] —> [Base Config] —> [Input Inverted ON/OFF]

[:SOURce]: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 ":PULM:INTernal:PERiod" and ":PULM:SOURce" for relevant commands.

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

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

Parameter description:

<DelayTime> pulse delay time for pulse modulation.

Range: non-trigger mode: 0s[0ns, 42.000000000s]

Trigger mode: 0s[100ns, 42.000000000s].

Example: PULM:INTernal:DELay 1ms set the pulse delay to 1ms

Reset state: 0s

Key path: [Analog Modulation] —> [Pulse] —> [Base Config] —> [Delay]

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

Function description: This command is used to set the pulse modulation frequency. When square is selected as the pulse source, the pulse signal output will be a signal with a duty ratio of 50%. This command can change the frequency of the square signal. Please refer to ":PULM:SOURce" for the pulse source.

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

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

Parameter description:

<Frequency> pulse modulation frequency.

Range: 1kHz [0.023Hz, 25MHz]

Example: PULM:INTernal:FREQuency 1MHz set the pulse frequency to 1MHz.

Reset: 1kHz

Key path: [Analog Modulation] —> [Pulse Modulation] —> [Pulse Setting] —> [Multi-pulse]

[:SOURce]: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. Please refer to ":PULM:INTernal:PWIDth" for the pulse width.

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

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

Parameter description:

<Percent> pulse period.

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

Example: PULM:INTernal:PERiod 10ms the pulse signal period is 10ms.

Reset state: 1.000000ms

Key path: [Analog Modulation] —> [Pulse] —> [Base Config] —> [Period]

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

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

Parameter description:

<PWidth> pulse width.

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

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

Reset state: 50.000us

Key path: [Analog Modulation] —> [Pulse] —> [Base Config] —> [Width]

[:SOURce]:PULM:SOURce <Mode>

Function description: The command is used to set the pulse source mode of pulse modulation, including: external, scalar, auto, square, doublet, pulse train, gated, triggered, jittered, stagger 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]:PULM:SOURce EXTernal|SCALar|INTernal|SQUare|DOUBlet

|PTRain|GATEd|TRIGgered|JITTered|STAGger|SLIDing

Query format: [:SOURce]: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 auto.
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.
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.
JITTered	The pulse source is jittered.
STAGger	The pulse source is stagger.
SLIDing	The pulse source is sliding.

Example: PULM:SOURce SQUare set the pulse source to square mode.

Reset state: INT

Key path: [Analog Modulation] —> [Pulse] —> [Base Config] —> [Source]

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

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

Parameter description:

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

ON | 1: Pulse modulation ON,

OFF | 0: Pulse modulation OFF.

Example: PULM:STATe 1 the pulse modulation state is ON.

Reset: 0

Key path: [Analog modulation] —> [Pulse Modulation] —> [Pulse Setting] —> [Pulse modulation ON/OFF]

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

Query format: [:SOURce]: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: [:SOURce]:PULM:INTernal:JITTered:MODE GAUS set the jittered mode of pulse to Gaussian.

Reset state: GAUS

Key path: [Analog modulation] —> [Pulse] —> [Jittered] —> [Dither Style]

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

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

Parameter description:

<Percent> jittered percent of pulse modulation.

Range: 0[0, 10].

Example: [:SOURce]:PULM:INTernal:JITTered:PERCent 5 the jittered percent of pulse is 5%.

Reset state: 10

Key path: [Analog modulation] —> [Pulse] —> [Jittered] —> [Dither Percent]

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

Function description: When the pulse source for pulse modulation of the signal generator is in multi-pulse mode, this command is used to set its pulse train. The parameter components 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.

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

{pls_width, pls_period…}

Parameter description:

<PlsWidth> Pulse width of pulse train.

Range: 50us[0.02us, 42s].

< PlsPerd> Pulse period of pulse train.

Range: 1ms[0.03us, 42s].

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

Set pulse trains with pulse width of 100us and period of 1ms as well as pulse width of 200us and period of 2ms.

Key path: [Analog modulation] —> [Pulse] —> [Pulse Train] —> [Edit pulse train...]

Description: For setting only.

[:SOURce]: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 number 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 "PULM:INTernal:PTRain:POINts" for pulse train points.

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

Parameter description:

<Index> pulse train list index.

Range: 0[0, 1023].

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

Key path: NONE

Description: For setting only.

[:SOURce]: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.

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

Returned value:

<Num> integer data, pulse train list points returned.

Range: 0[0, 1023].

Example: [:SOURce]:PULM:INTernal:PTRain:POINts? query the current pulse train list points.

Reset state: 0

Key path: NONE

Description: For query only.

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

Function description: This command is used to delete all points in the pulse train list. If the current list is empty, there is no action. Users may query the current list points before deletion. Please refer to "PULM:INTernal:PTRain:POINts" for pulse train points.

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

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

Key path: [Analog modulation] —> [Pulse] —> [Pulse Train] —> [Edit pulse train...] —> [Del All]

Description: For setting only.

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

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

Parameter description:

<StepTime> sliding step.

Range: 100ns[0s, 42ms].

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

Reset: 100ns

Key path: [Analog modulation] —> [Pulse] —> [Sliding] —> [Sliding step]

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

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

Parameter description:

<StepTime> sliding points.

Range: [2, 1024].

Example: [:SOURce]:PULM:INTernal:SLIDing:POINts 10 set the number of pulse sliding points to 10.

Reset: 1024

Key path: [Analog modulation] —> [Pulse] —> [Sliding] —> [Sliding Counts]

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

Function description: This command is used to insert pulse stagger points. Users shall write parameters in the order of the index number and pulse period. At most five pulse stagger points are supported by the signal generator, for which users may query the current stagger points before using the command. If the maximum number of stagger points is exceeded, the stagger point currently set will not be inserted into the current stagger list. In addition, the pulse width is uniform in the stagger list. Please refer to "PULM:INTernal:STAGger:POINts" and ":PULM:INTernal:PWIDth" for relevant commands.

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

Parameter description:

<Index> stagger list point index.

Range: 0[0, 4].

< PlsPerd> pulse period.

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

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

Insert a stagger point with period of 1ms at the position with index number of 1 in the current stagger list.

Key path: [Analog modulation] —> [Pulse] —> [Staggered] —> [Edit Stagg List...] --> [Insert]

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]: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.

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

Returned value:

<Num> integer data, stagger list points returned.

Range: 0[0, 5].

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

Query the current stagger list points.

Reset state: 0

Key path: NONE

Description: For query only.

[:SOURce]: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. Please refer to “PULM:INTernal:STAGger:POINts” for stagger list points.

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

Parameter description:

<Index> pulse stagger list index.

Range: 0[0, 4].

Example: [:SOURce]:PULM:INTernal:STAGger:DELete 1 delete the stagger point with index number 1 in the current pulse stagger list.

Reset state: 0

Key path: [Analog modulation] —> [Pulse Modulation] —> [Staggered] —> [Edit Stag List...] --> [Del Cur]

Description: For setting only.

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

Function description: This command is used to clear all points in the pulse stagger list. If there are no stagger points in the current list, the operation is invalid. Therefore, the stagger points in the current list may be queried before deletion. Please refer to "PULM:INTernal:STAGger:POINts" for query of stagger list points.

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

Example: [:SOURce]:PULM:INTernal:STAGger:PRESet delete all points in the stagger list.

Key path: [Analog modulation] —> [Pulse Modulation] —> [Staggered] —> [Edit Stag List...] —> [Del All]

Description: For setting only.

3.3.8 AMPLitude MODulation Subsystem

The following commands are used to set the amplitude modulation (AM) mode, including:

[:SOURce]:AM[1]|2[:DEPTh]:EXPonential <AmDepthExp>

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. Please refer to ":AM:TYPE" for AM type.

Setting format: [:SOURce]:AM[1]|2:DEPTh:EXPonential <.val>

Query format: [:SOURce]:AM[1]|2:DEPTh:EXPonential?

Parameter description:

<AmDepthExp> AM depth (exponential).

Range: 0.00dB[0.00dB, 40.00dB].

Example: AM2:DEPTh:EXPonential 10dBset the AM depth of AM Path 2 to 10dB.

Reset state: 30dB

Key path: [Modulation] —> [Amplitude Modulation] —> [Base Config] —> [Path 1|Path 2] —> [Depth]

[:SOURce]: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 ":AM:TYPE".

Setting format: [:SOURce]:AM[1]|2:DEPTh [:LINear] <.val>

Query format: [:SOURce]:AM[1]|2:DEPTh [:LINear]?

Parameter description:

<AmDepthExp> AM depth (linear).

Range: 30 [0, 100].

Example: AM:DEPTh 10 set the linear AM depth of Path 1 to 10%

Reset state: 30

Key path: [Modulation] —> [Amplitude Modulation] —> [Base Config] —> [Path 1|Path 2] —> [Depth]

[:SOURce]:AM[1]|2:INTernal:FREQuency <Frequency>

Function description: This command is used to set the internal AM rate of the AM path of the signal generator.

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

Query format: [:SOURce]:AM[1]|2:INTernal:FREQuency?

Parameter description:

<Frequency> AM rate.

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

Example: AM:INTernal:FREQuency 100kHz set the internal AM rate of Path 1 to 100kHz.

Reset state: AM rate: 1kHz.

Key path: [Modulation] —> [Amplitude Modulation] —> [Base Config] —> [Path 1|Path 2] —> [AM rate]

[:SOURce]: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 “:AM[1]|2:INTernal:SHAPe” for the output waveform of AM signal.

Setting format: [:SOURce]:AM:INTernal:RAMP POSitive|NEGative

Query format: [:SOURce]:AM:INTernal: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:RAMP NEG the AM ramp signal is up.

Reset state: POS

Key path: [Modulation] —> [Amplitude Modulation] —> [Base Config] —> [Path 1|Path 2] —> [AM Waveform] —> [Zigzag]

[:SOURce]:AM[1]|2:INTernal:SHAPe <Mode>

Function description: This command is used to set the output waveform of AM signal, including sine, square, triangle and zigzag.

Setting format: [:SOURce]:AM:INTernal:SHAPe SINE|SQUare|TRIangle|RAMP

NOISe|SWEPtsine|DUALsine

Query format: [:SOURce]:AM: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	zigzag wave.

Example: AM:INTernal:SHAP RAMP set the AM signal waveform of Path 1 to ramp.

Reset state: SINE

Key path: [Modulation] —> [Amplitude Modulation] —> [Base Config] —> [Path 1|Path 2] —> [AM Waveform]

[:SOURce]: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, AM and modulation are all set to ON state can the AM signal be output. Please refer to “AM:MODulation:STATe” for AM state and “OUTPut:MODulation:STATe” for modulation state.

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

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

Parameter description:

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

ON | 1: Path output ON

OFF | 0: Path output OFF.

Example: AM:STATe 1 Path 1 output ON.

Reset state: 0

Key path: [Modulation] —> [Amplitude Modulation] —> [Base Config] —> [Path 1|Path 2 ON/OFF]

[:SOURce]: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 1435 series signal generator.

Setting format: [:SOURce]:AM:MODE DEEP|NORMal

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

Parameter description:

<Mode> Discrete data. The values of AM mode are as follows:

DEEP Deep AM ON

NORMal Deep AM OFF.

Example: AM:MODE NORM deep AM OFF.

Reset state: NORM

Key path: [Modulation] —> [Amplitude Modulation] —> [AM Depth] —> [AM Depth ON/OFF]

[:SOURce]:AM:SOURce <Mode>

Function description: This command is used to select the AM source, including: internal 50Ω, external 50Ω, external 600Ω and external 1MΩ. 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]:AM:SOURce INTernal

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

Parameter description:

<Mode> Discrete data. The values of AM source are as follows:

INTernal internal AM.

EXT50 external 50Ω

EXT600 external 600Ω

EXT1M external 1MΩ

Example: AM:SOURce INT the AM source is internal.

Reset state: INT

Key path: [Modulation] —> [Amplitude Modulation] —> [AM source] —> [AM source]

[:SOURce]:AM:MODulation:STATe <State>

Function description: This command is used to set the AM signal output state of the signal generator.

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

Query format: [:SOURce]:AM:MODulation:STATe?

Parameter description:

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

ON | 1: AM output ON

OFF | 0: AM output OFF.

Example: AM:MODulation:STATe 1 AM ON

Reset state: 0

Key path: [Modulation] —> [Amplitude Modulation] —> [Base Config] —> [AM ON/OFF]

[:SOURce]: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]:AM:TYPE EXPonential|LINear

Query format: [:SOURce]: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: LIN

Key path: [Modulation] —> [Amplitude Modulation] —> [Base Config] —> [AM type: Exp/Linear]

[:SOURce]:AM:EXTernal:COUPling <Mode>

Function description: This command is used to set the AM external input coupling mode.

Setting format: [:SOURce]:AM:EXTernal:COUPling DC|AC

Query format: [:SOURce]:AM:EXTernal: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: [Modulation] —> [Amplitude Modulation] —> [AM source] —> [EXT Couple Type]

[:SOURce]:AM:EXTernal:PATH <Mode>

Function description: This command is used to set the AM external input path.

Setting format: [:SOURce]:AM:EXTernal:PATH EXTernal[1]|2

Query format: [:SOURce]:AM:EXTernal:PATH?

Parameter description:

<Mode> Discrete data. The values of AM external input path are as follows:

EXTernal1	external path 1
EXTernal2	external path 2

Example: AM:EXTernal:PATH EXTernal2 set the external input path to external 2.

Reset state: EXTernal1

Key path: [Modulation] —> [Amplitude Modulation] —> [AM source] —> [Ext input path]

[:SOURce]:AM[1]|2:INTernal:DUAL:FUNCtion:AMPLitude:PERCent <val>

Function description: This command is used to set the amplitude ratio of dual function generator relative to audio 1 when the waveform of AM Path 1 or Path 2 is dual function generator.

Setting format: [:SOURce]:AM[1]|2:INTernal:DUAL:FUNCtion:AMPLitude:PERCent <val>

Query format: [:SOURce]:AM[1]|2:INTernal:DUAL:FUNCtion:AMPLitude:PERCent?

Parameter description:

<val> the values of amplitude ratio of dual function generator relative to audio 1 are as follows:

Range: 50 [0,100]

Example: AM:INTernal:DUAL:FUNCtion:AMPLitude:PERCent 30 set the amplitude ratio of dual function generation in Path 1 relative to audio 1 to 30%.

Reset state: 50

Key path: [Modulation] --> [Amplitude Modulation] --> [Base Config] --> [Path 1|Path 2] —> [AM Waveform] --> [Dual Fun-Generator]

[:SOURce]:AM[1]|2:INTernal:DUAL:FUNCtion[1]|2:FREQuency <Frequency >

Function description: This command is used to set the frequency of dual function generator relative to audio 1 when the waveform of AM Path 1 or Path 2 is dual function generator.

Setting format: [:SOURce]:AM[1]|2:INTernal:DUAL:FUNCtion[1]|2:FREQuency <Frequency >

Query format: [:SOURce]:AM[1]|2:INTernal:DUAL:FUNCtion[1]|2:FREQuency?

Parameter description:

<Frequency> frequency of dual function generator relative to audio 1.

Range: 1kHz[0.001Hz, 1MHz].

Example: AM:INTernal:DUAL:FUNCtion:FREQuency 20kHz set the frequency of dual function generation in Path 1 relative to audio 1 to 20kHz.

Reset state: 1kHz

Key path: [Modulation] —> [Amplitude Modulation] --> [Base Config] --> [Path 1|Path 2] —> [AM Waveform] —> [Dual Fun-Generator]

[:SOURce]:AM[1]|2:INTernal:DUAL:FUNCtion[1]|2:PERCent <val>

Function description: This command is used to set the pulse duty factor of dual function generator relative to audio 1|2 when the waveform of AM Path 1 or Path 2 is dual function generator.

Setting format: [:SOURce]:AM[1]|2:INTernal:DUAL:FUNCtion[1]|2:PERCent <val>

Query format: [:SOURce]:AM[1]|2:INTernal:DUAL:FUNCtion[1]|2:PERCent?

Parameter description:

<val> the values of pulse duty factor of dual function generator relative to audio 1 are as follows:

Range: 50 [0,100]

Example: AM:INTernal:DUAL:FUNCtion: PERCent 20 set the pulse duty factor of dual function generator in Path 1 relative to audio 1 to 20%.

Reset state: 50%

Key path: [Modulation] —> [Amplitude Modulation] --> [Base Config] --> [Path 1|Path 2] —> [AM Waveform] --> [Dual Fun-Generator]

[:SOURce]:AM[1]|2:INTernal:DUAL:FUNCtion:POFFset <val>

Function description: This command is used to set the phase offset of dual function generator relative to audio 1 when the waveform of AM Path 1 or Path 2 is dual function generator.

Setting format: [:SOURce]:AM[1]|2:INTernal:DUAL: FUNCtion:POFFset <val>

Query format: [:SOURce]:AM[1]|2:INTernal:DUAL: FUNCtion:POFFset?

Parameter description:

<val> the values of phase offset of dual function generator relative to audio 1 are as follows:

Range: 0deg [0deg, 360deg]

Example: AM:INTernal:DUAL: FUNCtion:POFFset 60 set the phase offset of dual function generator in Path 1relative to audio 1 to 60deg.

Reset state: 0

Key path: [Modulation] —> [Amplitude Modulation] —> [Base Config] —> [Path 1|Path 2] —> [AM Waveform] --> [Dual Fun-Generator]

[:SOURce]:AM[1]|2:INTernal:DUAL:FUNCtion:SHAPe <Mode>

Function description: This command is used to set the output waveform of dual function generator when the waveform of AM Path 1 or Path 2 is dual function generator.

Setting format: [:SOURce]:AM[1]|2:INTernal:DUAL:FUNCtion:SHAPe <Mode>

Query format: [:SOURce]:AM[1]|2:INTernal:DUAL:FUNCtion:SHAPe?

Parameter description:

<Mode> Discrete data. The values of output waveform are as follows:

Sine	sine wave,
SQUare	square wave,
TRIangle	triangle wave,
Ramp	zigzag wave,
PULSe	pulse

Example: AM:INTernal:DUAL:FUNCtion:SHAPe TRIangle set the output waveform of dual function generator in Path 1 to triangle.

Reset state: SINE

Key path: [Modulation] —> [Amplitude Modulation] —> [Base Config] —> [Path 1|Path 2] —> [AM Waveform] —> [Dual Fun-Generator]

[:SOURce]:AM[1]|2:INTernal:DUAL:FUNCtion:SHAPe:RAMP <Mode>

Function description: This command is used to set the signal output type when the waveform of AM Path 1 or Path 2 is dual function generator and the output waveform of the generator is ramp, including up and down.

Setting format: [:SOURce]:AM[1]|2:INTernal:DUAL:FUNCtion:SHAPe:RAMP <Mode>

Query format: [:SOURce]:AM[1]|2:INTernal:DUAL:FUNCtion:SHAPe:RAMP?

Parameter description:

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

POSitive	up
NEGative	down.

Example: AM:INTernal:DUAL:FUNCtion:SHAPe:RAMP POSitive set the ramp to up when the output waveform of dual function generator in Path 1 is ramp.

Reset state: POS

Key path: [Modulation] —> [Amplitude Modulation] —> [Base Config] —> [Path 1|Path 2] —> [AM Waveform] —> [Dual Fun-Generator]

[:SOURce]:AM[1]|2:INTernal:FUNCtion[1]|2:FREQuency <Frequency>

Function description: This command is used to set the output frequency of function generator 1|2 when the waveform of AM Path 1 or Path 2 is function generator 1|2.

Setting format: [:SOURce]:AM[1]|2:INTernal:FUNCtion[1]|2:FREQuency <Frequency>

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

Parameter description:

<Frequency> output frequency of function generator 1|2.

Range: 1kHz[0.001Hz, 1MHz].

Example: AM:INTernal:FUNCtion2:FREQuency 10kHz set the output frequency of function generator 2 in Path 1 to 10kHz.

Reset state: 1kHz

Key path: [Modulation] —> [Amplitude Modulation] —> [Base Config] —> [Path 1|Path 2] —> [AM Waveform] —> [Generator 1|2]

[:SOURce]:AM[1]|2:INTernal:FUNCtion[1]|2:PERCent <val>

Function description: This command is used to set the pulse duty factor of function generator 1|2 when the waveform of AM Path 1 or Path 2 is function generator 1|2.

Setting format: [:SOURce]:AM[1]|2:INTernal:FUNCtion[1]|2:PERCent <val>

Query format: [:SOURce]:AM[1]|2:INTernal:FUNCtion[1]|2:PERCent?

Parameter description:

<val> the values of pulse duty factor of function generator 1|2 are as follows:

Range: 50 [0,100]

Example: AM:INTernal:FUNCtion2:PERCent 50 set the pulse duty factor of function generator 2 in Path 1 to 50%.

Reset state: 50%

Key path: [Modulation] —> [Amplitude Modulation] —> [Base Config] —> [Path 1|Path 2] —> [AM Waveform] —> [Generator 1|2]

[:SOURce]:AM[1]|2:INTernal:FUNCtion[1]|2:SHAPe <Mode>

Function description: This command is used to set the output waveform of function generator 1|2 when the waveform of AM Path 1 or Path 2 is function generator 1|2.

Setting format: [:SOURce]:AM[1]|2:INTernal:FUNCtion[1]|2:SHAPe <Mode>

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

Parameter description:

<Mode> Discrete data. The values of output waveform are as follows:

Sine	sine wave,
SQUare	square wave,
TRIangle	triangle wave,
Ramp	zigzag wave,
PULSe	pulse.

Example: AM:INTernal:FUNCtion2:SHAPe TRIangle set the output waveform of function generator 2 in Path 1 to triangle.

Reset state: 50%

Key path: [Modulation] —> [Amplitude Modulation] —> [Base Config] —> [Path 1|Path 2] —> [AM Waveform] —> [Generator 1|2]

[:SOURce]:AM[1]|2:INTernal:FUNCtion[1]|2:SHAPe:RAMP <Mode>

Function description: This command is used to set the signal output type when the waveform of AM Path 1 or Path 2 is function generator 1|2 and the output waveform of the generator 1|2 is ramp, including up and down.

Setting format: [:SOURce]:AM[1]|2:INTernal:FUNCtion[1]|2:SHAPe:RAMP <Mode>

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

Parameter description:

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

POSitive	up
NEGative	down.

Example: AM:INTernal:FUNCtion2:SHAPe:RAMP NEGative set the zigzag to down when the output waveform of function generator 2 in Path 1 is ramp.

Reset state: POS

Key path: [Modulation] —> [Amplitude Modulation] —> [Base Config] —> [Path 1|Path 2] —> [AM Waveform] —> [Generator 1|2]

[:SOURce]:AM[1]|2:INTernal:NOISe:FUNCtion[1]|2:TYPE <Mode>

Function description: This command is used to set the noise type of noise generator 1|2 when the waveform of AM Path 1 or Path 2 is noise generator 1|2.

Setting format: [:SOURce]:AM[1]|2:INTernal:NOISe:FUNCtion[1]|2:TYPE <Mode>

Query format: [:SOURce]:AM[1]|2:INTernal:NOISe:FUNCtion[1]|2:TYPE?

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: AM:INTernal:FUNCtion2:SHAPe GAUSsian set the noise type of function generator 2 in Path 1 to Gaussian.

Reset state: GAUS

Key path: [Modulation] —> [Amplitude Modulation] —> [Base Config] —> [Path 1|Path 2] —> [AM Waveform] —> [Noise generator 1|2]

[:SOURce]:AM[1]|2:INTernal:SWEep:FUNCtion:FREQuency:STARt <Frequency>

Function description: This command is used to set the start frequency of sweep generator when the waveform of AM Path 1 or Path 2 is sweep generator.

Setting format: [:SOURce]:AM[1]|2:INTernal:SWEep:FUNCtion:FREQuency:STARt <Frequency>

Query format: [:SOURce]:AM[1]|2:INTernal:SWEep:FUNCtion:FREQuency:STARt?

Parameter description:

<Frequency> start frequency of sweep function generator.

Range: 1kHz[0.001Hz, 1MHz].

Example: AM:INTernal:SWEep:FUNCtion:FREQuency:STARt 30kHz set the start frequency of sweep function generator in Path 1 to 30kHz.

Reset state: 1kHz

Key path: [Modulation] —> [Amplitude Modulation] —> [Base Config] —> [Path 1|Path 2] —> [AM Waveform] —> [Sweep Fun-Generator]

[:SOURce]:AM[1]|2:INTernal:SWEep:FUNCtion:FREQuency:STOP <Frequency>

Function description: This command is used to set the stop frequency of sweep generator when the waveform of AM Path 1 or Path 2 is sweep generator.

Setting format: [:SOURce]:AM[1]|2:INTernal:SWEep:FUNCtion:FREQuency:STOP <Frequency>

Query format: [:SOURce]:AM[1]|2:INTernal:SWEep:FUNCtion:FREQuency:STOP?

Parameter description:

<Frequency> stop frequency of sweep function generator.

Range: 1kHz[0.001Hz, 1MHz].

Example: AM:INTernal:SWEep:FUNCtion:FREQuency: STOP 50kHz set the stop frequency of sweep function generator in Path 1 to 50kHz.

Reset state: 1kHz

Key path: [Modulation] —> [Amplitude Modulation] —> [Base Config] —> [Path 1|Path 2] —> [AM Waveform] —> [Sweep Fun-Generator]

[:SOURce]:AM[1]|2:INTernal:SWEep:FUNCtion:SHAPe <Mode>

Function description: This command is used to set the sweep type of sweep generator when the waveform of AM Path 1 or Path 2 is sweep generator.

Setting format: [:SOURce]:AM[1]|2:INTernal:SWEep:FUNCtion:SHAPe <Mode>

Query format: [:SOURce]:AM[1]|2:INTernal:SWEep:FUNCtion:SHAPe?

Parameter description:

<Mode> Discrete data. The values of output waveform are as follows:

Sine	sine wave,
SQUare	square wave,
TRIangle	triangle wave,
RAMP	zigzag wave.

Example: AM:INTernal:SWEep:FUNCtion:SHAPe TRIangle set the sweep type of sweep function generator in Path 1 to triangle.

Reset state: 1kHz

Key path: [Modulation] —> [Amplitude Modulation] —> [Base Config] —> [Path 1|Path 2] —> [AM Waveform] —> [Sweep Fun-Generator]

[:SOURce]:AM[1]|2:INTernal:SWEep:FUNCtion:SHAPe:RAMP <Mode>

Function description: This command is used to set the signal output type when the waveform of AM Path 1 or Path 2 is sweep generator and the sweep type is zigzag, including up and down.

Setting format: [:SOURce]:AM[1]|2:INTernal:SWEep:FUNCtion:SHAPe:RAMP <Mode>

Query format: [:SOURce]:AM[1]|2:INTernal:SWEep:FUNCtion:SHAPe:RAMP?

Parameter description:

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

POSitive	up
NEGative	down.

Example: AM:INTernal:SWEep:FUNCtion:SHAPe:RAMP NEGative set the signal output type to down when the waveform of Path 1 is sweep generator and the sweep type is zigzag.

Reset state: POS

Key path: [Modulation] —> [Amplitude Modulation] —> [Base Config] —> [Path 1|Path 2] —> [AM Waveform] —> [Sweep Fun-Generator]

[:SOURce]:AM[1]|2:INTernal:SWEep:FUNCtion:TIME <Time>

Function description: This command is used to set the sweep time of sweep generator when the waveform of AM Path 1 or Path 2 is sweep generator.

Setting format: [:SOURce]:AM[1]|2:INTernal:SWEep:FUNCtion:TIME <Time>

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

Parameter description:

< Time > sweep time of sweep function generator.

Range: 0.1ms[0.01us, 40s].

Example: AM:INTernal:SWEep:FUNCtion:TIME 5s set the sweep time of sweep generator to 5s when the waveform of Path 1 is sweep generator.

Reset state: 0.1ms

Key path: [Modulation] —> [Amplitude Modulation] —> [Base Config] —> [Path 1|Path 2] —> [AM Waveform] —> [Sweep Fun-Generator]

[:SOURce]:AM[1]|2:INTernal:SWEep:FUNCtion:TRIGger:MODE <Mode>

Function description: This command is used to set the trigger mode of sweep generator when the waveform of AM Path 1 or Path 2 is sweep generator.

Setting format: [:SOURce]:AM[1]|2:INTernal:SWEep:FUNCtion:TRIGger:MODE <val>

Query format: [:SOURce]:AM[1]|2:INTernal:SWEep:FUNCtion:TRIGger:MODE?

Parameter description:

<Mode> Discrete data, with values taken as follows:

CONTinuous:	Continuous
SINGle:	Single

Example: AM:INTernal:SWEep:FUNCtion:TRIGger:MODE CONTinuous set the trigger mode of the sweep function generator to continuous.

Reset state: CONTinuous

Key path: [Modulation] —> [Amplitude Modulation] —> [Base Config] —> [Path 1|Path 2] —> [AM Waveform] —> [Sweep Fun-Generator]

[:SOURce]:AM[1]|2:INTernal:SWEep:FUNCtion:TRIGger:PERiod <Time>

Function description: This command is used to set the sweep timer period of sweep generator when the waveform of AM Path 1 or Path 2 is sweep generator and the trigger type is timed trigger.

Setting format: [:SOURce]:AM[1]|2:INTernal:SWEep:FUNCtion:TRIGger:PERiod <val>

Query format: [:SOURce]:AM[1]|2:INTernal:SWEep:FUNCtion:TRIGger:PERiod?

Parameter description:

<Time> sweep timer period.

Range: 0.1ms[10ns, 40s].

Example: AM:INTernal:SWEep:FUNCtion:TRIGger:PERiod 1s set the sweep timer period of the sweep function generator to 1s.

Reset state: 0.1ms

Key path: [Modulation] —> [Amplitude Modulation] —> [Base Config] —> [Path 1|Path 2] —> [AM Waveform] —> [Sweep Fun-Generator]

[:SOURce]:AM[1]|2:INTernal:SWEep:FUNCtion:TRIGger:TYPE <Mode>

Function description: This command is used to set the trigger type of sweep generator when the waveform of AM Path 1 or Path 2 is sweep generator.

Setting format: [:SOURce]:AM[1]|2:INTernal:SWEep:FUNCtion:TRIGger:TYPE <val>

Query format: [:SOURce]:AM[1]|2:INTernal:SWEep:FUNCtion:TRIGger:TYPE?

Parameter description:

<Mode> Discrete data, with values taken as follows:

IMMediate: Auto

KEY: Trigger key

BUS: Bus

INTernal: Internal

EXTernal: External

TIMer: Timed trigger

Example: AM:INTernal:SWEep:FUNCtion:TRIGger:TYPE BUS set the trigger type of the sweep function generator to bus.

Reset state: IMMediate

Key path: [Modulation] —> [Amplitude Modulation] —> [Base Config] —> [Path 1|Path 2] —> [AM Waveform] —> [Sweep Fun-Generator]

3.3.9 FREQuency MODulation Subsystem

The following commands are used to select the frequency modulation (FM) mode, including:

[:SOURce]: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]:FM[1]|2:DEViation <val>

Query format: [:SOURce]:FM[1]|2:DEViation?

Parameter description:

<Deviation> the **relationship** between the current frequency and the frequency deviation is as follows:

1435	Current frequency	Frequency deviation
	9kHz – 250MHz	0 – 4MHz
	250MHz – 375MHz	0 – 1MHz
	375MHz – 750GHz	0 – 2MHz
	750MHz – 1.5GHz	0 – 4MHz
	1.5GHz – 3GHz	0 – 8MHz
	3GHz – 6GHz	0 – 16MHz
	6GHz – 12GHz	0 – 32MHz
	12GHz – 24GHz	0 – 64MHz
	24GHz – 40GHz	0 – 128MHz

Example: FM:DEViation 500kHz set the frequency deviation of FM signal for FM Path 1 to 500kHz.

Key path: [Modulation] —> [Frequency Modulation] **—**> [Base Config] **—**> [FM Dev]

[:SOURce]: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. Please refer to ":FM:SOURce" for relevant command.

Setting format: [:SOURce]:FM[1]|2:INTernal:FREQuency <val>.

Query format: [:SOURce]:FM[1]|2:INTernal:FREQuency?

Parameter description:

<Frequency> the relationship between the FM waveform and the FM rate range is as follows:

Sine: [0.005Hz, 10.000000000MHz]

Square: [0.005Hz, 10.000000000MHz]

Triangle: [0.005Hz, 10.000000000MHz]

Ramp: [0.005Hz, 10.000000000MHz]

Example: FM:INTernal:FREQuency 300kHz set the FM rate of FM Path 1 to 300kHz.

Reset state: 0.001MHz

Key path: [Modulation] —> [Frequency Modulation] —> [Base Config] —> [Path 1|Path 2] —> [FM rate]

[:SOURce]: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 ":FM[1]|2:INTernal:SHAPe" for selection of FM waveform.

Setting format: [:SOURce]:FM[1]|2:INTernal:SHAPe:RAMP POSitive|NEGative

Query format: [:SOURce]: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:RAMP NEG set the type of ramp for FM Path 1 to down.

Reset state: POS

Key path: [Modulation] —> [Frequency Modulation] —> [Base Config] —> [Path 1|Path 2] —> [FM Waveform] **—>** [Zigzag]

[:SOURce]: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 zigzag.

Setting format: [:SOURce]:FM[1]|2:INTernal:SHAPe SINE|SQUare|TRIangle|RAMP

Query format: [:SOURce]: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	zigzag wave.

Example: [:SOURce]:FM2:INTernal:SHAP RAMP set the FM waveform of Path 2 to ramp.

Reset state: SINE

Key path: [Modulation] —> [Frequency Modulation] —> [Base Config] —> [Path 1|Path 2] —> [FM Waveform]

[:SOURce]: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 path, FM and modulation are all set to ON state can the FM signal be output. Please refer to “FM:MODulation:STATe” for FM state and “OUTPut:MODulation:STATe” for modulation state.

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

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

Parameter description:

<State> **Boolean** data; the values are as follows:

ON | 1: Path output ON

OFF | 0: Path output OFF.

Example: FM:STATe 1 Path 1 output ON.

Reset state: 0

Key path: [Modulation] —> [Frequency Modulation] —> [Base Config] —> [Path 1|Path 2 ON/OFF]

[:SOURce]:FM:SOURce <Mode>

Function description: This command is used to select the FM source, including: internal 50Ω, external 50Ω, external 600Ω and external 1MΩ. 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]:FM:SOURce INTernal| EXT50Ω| EXT600Ω|EXT1MΩ

Query format: [:SOURce]:FM:SOURce?

Parameter description:

<Mode> Discrete data. The values of FM source are as follows:

INTernal	internal FM
EXT50	external 50Ω
EXT600	external 600Ω
EXT1M	external 1MΩ

Example: FM:SOURce INT the FM source is internal.

Reset state: INT

Key path: [Modulation] —> [Frequency Modulation] **—>** [FM source] **—>** [FM source]

[:SOURce]:FM:MODulation:STATe <State>

Function description: This command is used to set the FM signal output state of the signal generator.

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

Query format: [:SOURce]:FM:MODulation:STATe?

Parameter description:

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

ON | 1: FM output ON

OFF | 0: FM output OFF.

Example: FM:STATe 0 FM OFF.

Reset state: 0

Key path: [Modulation] —> [Frequency Modulation] —> [Base Config] —> [FM ON/OFF]

[:SOURce]:FM:EXTernal:COUPling <Mode>

Function description: This command is used to set the FM external input coupling mode.

Setting format: [:SOURce]:FM:EXTernal:COUPling DC|AC

Query format: [:SOURce]:FM:EXTernal: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: [Modulation] —> [Frequency Modulation] —> [FM Source] —> [EXT Couple Type]

[:SOURce]:FM:EXTernal:PATH <Mode>

Function description: This command is used to set the FM external input path.

Setting format: [:SOURce]:FM:EXTernal:PATH EXTernal[1]|2

Query format: [:SOURce]:FM:EXTernal:PATH?

Parameter description:

<Mode> Discrete data. The values of FM external input path are as follows:

EXTernal1	external path 1
EXTernal2	external path 2

Example: FM:EXTernal:PATH EXTernal2 set the external input path to external 2.

Reset state: EXTernal1

Key path: [Modulation] —> [Frequency Modulation] —> [FM Source] —> [External input path]

[:SOURce]:FM[1]|2:INTernal:DUAL:FUNCtion:AMPLitude:PERCent <val>

Function description: This command is used to set the amplitude ratio of dual function generator relative to audio 1 when the waveform of FM Path 1 or 2 is dual function generator.

Setting format: [:SOURce]:FM[1]|2:INTernal:DUAL:FUNCtion:AMPLitude:PERCent <val>

Query format: [:SOURce]:FM[1]|2:INTernal:DUAL:FUNCtion:AMPLitude:PERCent?

Parameter description:

<val> the values of amplitude ratio of dual function generator relative to audio 1 are as follows:

Range: 50 [0,100]

Example: FM:INTernal:DUAL:FUNCtion:AMPLitude:PERCent 30 set the amplitude ratio of dual function generation in Path 1 relative to audio 1 to 30%.

Reset state: 50

Key path: [Modulation] —> [Frequency Modulation] —> [Base Config] —> [Path 1|Path 2] —> [FM Waveform] —> [Dual Fun-Generator]

[:SOURce]:FM[1]|2:INTernal:DUAL:FUNCtion[1]|2:FREQuency <Frequency >

Function description: This command is used to set the frequency of dual function generator relative to audio 1 when the waveform of FM Path 1 or Path 2 is dual function generator.

Setting format: [:SOURce]:FM[1]|2:INTernal:DUAL:FUNCtion[1]|2:FREQuency <Frequency >

Query format: [:SOURce]:FM[1]|2:INTernal:DUAL:FUNCtion[1]|2:FREQuency?

Parameter description:

<Frequency> frequency of dual function generator relative to audio 1.

Range: 1kHz[0.001Hz, 1MHz].

Example: FM:INTernal:DUAL:FUNCtion:FREQuency 20kHz set the frequency of dual function generation in Path 1 relative to audio 1 to 20kHz.

Reset state: 1kHz

Key path: [Modulation] —> [Frequency Modulation] —> [Base Config] —> [Path 1|Path 2] —> [FM Waveform] —> [Dual Fun-Generator]

[:SOURce]:FM[1]|2:INTernal:DUAL:FUNCtion[1]|2:PERCent <val>

Function description: This command is used to set the pulse duty factor of dual function generator relative to audio 1|2 when the waveform of FM Path 1 or Path 2 is dual function generator.

Setting format: [:SOURce]:FM[1]|2:INTernal:DUAL:FUNCtion[1]|2:PERCent <val>

Query format: [:SOURce]:FM[1]|2:INTernal:DUAL:FUNCtion[1]|2:PERCent

Parameter description:

<val> the values of pulse duty factor of dual function generator relative to audio 1 are as follows:

Range: 50 [0,100]

Example: FM:INTernal:DUAL:FUNCtion: PERCent 20 set the pulse duty factor of dual function generator in Path 1 relative to audio 1 to 20%.

Reset state: 50%

Key path: [Modulation] —> [Frequency Modulation] —> [Base Config] —> [Path 1|Path 2] —> [FM Waveform] —> [Dual Fun-Generator]

[:SOURce]:FM[1]|2:INTernal:DUAL:FUNCtion:POFFset <val>

Function description: This command is used to set the phase offset of dual function generator relative to audio 1 when the waveform of FM Path 1 or Path 2 is dual function generator.

Setting format: [:SOURce]:FM[1]|2:INTernal:DUAL: FUNCtion:POFFset <val>

Query format: [:SOURce]:FM[1]|2:INTernal:DUAL: FUNCtion:POFFset?

Parameter description:

<val> the values of phase offset of dual function generator relative to audio 1 are as follows:

Range: 0deg [0deg, 360deg]

Example: FM:INTernal:DUAL: FUNCtion:POFFset 60 set the phase offset of dual function generator in Path 1relative to audio 1 to 60deg.

Reset state: 0

Key path: [Modulation] —> [Frequency Modulation] —> [Base Config] —> [Path 1|Path 2] —> [FM Waveform] —> [Dual Fun-Generator]

[:SOURce]:FM[1]|2:INTernal:DUAL:FUNCtion:SHAPe <Mode>

Function description: This command is used to set the output waveform of dual function generator when the waveform of FM Path 1 or Path 2 is dual function generator.

Setting format: [:SOURce]:FM[1]|2:INTernal:DUAL:FUNCtion:SHAPe <Mode>

Query format: [:SOURce]:FM[1]|2:INTernal:DUAL:FUNCtion:SHAPe?

Parameter description:

<Mode> Discrete data. The values of output waveform are as follows:

Sine	sine wave,
SQUare	square wave,
TRIangle	triangle wave,
RAMP	zigzag wave,
PULSe	pulse

Example: FM:INTernal:DUAL:FUNCtion:SHAPe TRIangle set the output waveform of dual function generator in Path 1 to triangle.

Reset state: SINE

Key path: [Modulation] —> [Frequency Modulation] —> [Base Config] —> [Path 1|Path 2] —> [FM Waveform] —> [Dual Fun-Generator]

[:SOURce]:FM[1]|2:INTernal:DUAL:FUNCtion:SHAPe:RAMP <Mode>

Function description: This command is used to set the signal output type when the waveform of FM Path 1 or Path 2 is dual function generator and the output waveform of the generator is zigzag, including up and down.

Setting format: [:SOURce]:FM[1]|2:INTernal:DUAL:FUNCtion:SHAPe:RAMP <Mode>

Query format: [:SOURce]:FM[1]|2:INTernal:DUAL:FUNCtion:SHAPe:RAMP?

Parameter description:

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

POSitive	up
NEGative	down.

Example: FM:INTernal:DUAL:FUNCtion:SHAPe:RAMP POSitive set the zigzag to up when the output waveform of dual function generator in Path 1 is ramp.

Reset state: POS

Key path: [Modulation] —> [Frequency Modulation] —> [Base Config] —> [Path

1|Path 2] —> [FM Waveform] —> [Dual Fun-Generator]

[:SOURce]:FM[1]|2:INTernal:FUNCtion[1]|2:FREQuency <Frequency>

Function description: This command is used to set the output frequency of function generator 1|2 when the waveform of FM Path 1 or Path 2 is function generator 1|2.

Setting format: [:SOURce]:FM[1]|2:INTernal:FUNCtion[1]|2:FREQuency <Frequency>

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

Parameter description:

<Frequency> output frequency of function generator 1|2.

Range: 1kHz[0.001Hz, 1MHz].

Example: FM:INTernal:FUNCtion2:FREQuency 10kHz set the output frequency of function generator 2 in Path 1 to 10kHz.

Reset state: 1kHz

Key path: [Modulation] —> [Frequency Modulation] —> [Base Config] —> [Path 1|Path 2] —> [FM Waveform] —> [Generator 1|2]

[:SOURce]:FM[1]|2:INTernal:FUNCtion[1]|2:PERCent <val>

Function description: This command is used to set the pulse duty factor of function generator 1|2 when the waveform of FM Path 1 or Path 2 is function generator 1|2.

Setting format: [:SOURce]:FM[1]|2:INTernal:FUNCtion[1]|2:PERCent <val>

Query format: [:SOURce]:FM[1]|2:INTernal:FUNCtion[1]|2:PERCent?

Parameter description:

<val> the values of pulse duty factor of function generator 1|2 are as follows:

Range: 50 [0,100]

Example: FM:INTernal:FUNCtion2:PERCent 50 set the pulse duty factor of function generator 2 in Path 1 to 50%.

Reset state: 50%

Key path: [Modulation] —> [Frequency Modulation] —> [Base Config] —> [Path 1|Path 2] —> [FM Waveform] —> [Generator 1|2]

[:SOURce]:FM[1]|2:INTernal:FUNCtion[1]|2:SHAPe <Mode>

Function description: This command is used to set the output waveform of function generator 1|2 when the waveform of FM Path 1 or Path 2 is function generator 1|2.

Setting format: [:SOURce]:FM[1]|2:INTernal:FUNCtion[1]|2:SHAPe <Mode>

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

Parameter description:

<Mode> Discrete data. The values of output waveform are as follows:

Sine	sine wave,
SQUare	square wave,
TRIangle	triangle wave,
Ramp	zigzag wave,
PULSe	pulse.

Example: FM:INTernal:FUNCtion2:SHAPe TRIangle set the output waveform of function generator 2 in Path 1 to triangle.

Reset state: 50%

Key path: [Modulation] —> [Frequency Modulation] —> [Base Config] —> [Path 1|Path 2] —> [FM Waveform] —> [Generator 1|2]

[:SOURce]:FM[1]|2:INTernal:FUNCtion[1]|2:SHAPe:RAMP <Mode>

Function description: This command is used to set the signal output type when the waveform of FM Path 1 or Path 2 is function generator 1|2 and the output waveform of the generator 1|2 is ramp, including up and down.

Setting format: [:SOURce]:FM[1]|2:INTernal:FUNCtion[1]|2:SHAPe:RAMP <Mode>

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

Parameter description:

<Mode> Discrete data. The values of zigzag signal type are as follows:

POSitive	up
NEGative	down.

Example: FM:INTernal:FUNCtion2:SHAPe:RAMP NEGative set the zigzag to down when the output waveform of function generator 2 in Path 1 is ramp.

Reset state: POS

Key path: [Modulation] —> [Frequency Modulation] —> [Base Config] —> [Path 1|Path 2] —> [FM Waveform] —> [Generator 1|2]

[:SOURce]:FM[1]|2:INTernal:NOISe:FUNCtion[1]|2:TYPE <Mode>

Function description: This command is used to set the noise type of noise generator 1|2 when the waveform of FM Path 1 or Path 2 is noise generator 1|2.

Setting format: [:SOURce]:FM[1]|2:INTernal:NOISe:FUNCtion[1]|2:TYPE <Mode>

Query format: [:SOURce]:FM[1]|2:INTernal:NOISe:FUNCtion[1]|2:TYPE?

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: FM:INTernal:FUNCtion2:SHAPe GAUSsian set the noise type of function generator 2 in Path 1 to Gaussian.

Reset state: GAUS

Key path: [Modulation] —> [Frequency Modulation] —> [Base Config] —> [Path 1|Path 2] —> [FM Waveform] —> [Noise generator 1|2]

[:SOURce]:FM[1]|2:INTernal:SWEep:FUNCtion:FREQuency:STARt <Frequency>

Function description: This command is used to set the start frequency of sweep generator when the waveform of FM Path 1 or Path 2 is sweep generator.

Setting format: [:SOURce]:FM[1]|2:INTernal:SWEep:FUNCtion:FREQuency:STARt <Frequency>

Query format: [:SOURce]:FM[1]|2:INTernal:SWEep:FUNCtion:FREQuency:STARt?

Parameter description:

<Frequency> start frequency of sweep function generator.

Range: 1kHz[0.001Hz, 1MHz].

Example: FM:INTernal:SWEep:FUNCtion:FREQuency:STARt 30kHz set the start frequency of sweep function generator in Path 1 to 30kHz.

Reset state: 1kHz

Key path: [Modulation] —> [Frequency Modulation] —> [Base Config] —> [Path 1|Path 2] —> [FM Waveform] —> [Sweep Fun-Generator]

[:SOURce]:FM[1]|2:INTernal:SWEep:FUNCtion:FREQuency:STOP <Frequency>

Function description: This command is used to set the stop frequency of sweep generator when the waveform of FM Path 1 or Path 2 is sweep generator.

Setting format: [:SOURce]:FM[1]|2:INTernal:SWEep:FUNCtion:FREQuency: STOP <Frequency>

Query format: [:SOURce]:FM[1]|2:INTernal:SWEep:FUNCtion:FREQuency: STOP?

Parameter description:

<Frequency> stop frequency of sweep function generator.

Range: 1kHz[0.001Hz, 1MHz].

Example: FM:INTernal:SWEep:FUNCtion:FREQuency: STOP 50kHz set the stop frequency of sweep function generator in Path 1 to 50kHz.

Reset state: 1kHz

Key path: [Modulation] —> [Frequency Modulation] —> [Base Config] —> [Path 1|Path 2] —> [FM Waveform] —> [Sweep Fun-Generator]

[:SOURce]:FM[1]|2:INTernal:SWEep:FUNCtion:SHAPe <Mode>

Function description: This command is used to set the sweep type of sweep generator when the waveform of FM Path 1 or Path 2 is sweep generator.

Setting format: [:SOURce]:FM[1]|2:INTernal:SWEep:FUNCtion:SHAPe <Mode>

Query format: [:SOURce]:FM[1]|2:INTernal:SWEep:FUNCtion:SHAPe?

Parameter description:

<Mode> Discrete data. The values of output waveform are as follows:

Sine	sine wave,
SQUare	square wave,
TRIangle	triangle wave,
RAMP	ramp wave.

Example: FM:INTernal:SWEep:FUNCtion:SHAPe TRIangle set the sweep type of sweep function generator in Path 1 to triangle.

Reset state: 1kHz

Key path: [Modulation] —> [Frequency Modulation] —> [Base Config] —> [Path 1|Path 2] —> [FM Waveform] —> [Sweep Fun-Generator]

[:SOURce]:FM[1]|2:INTernal:SWEep:FUNCtion:SHAPe:RAMP <Mode>

Function description: This command is used to set the signal output type when the waveform of FM Path 1 or Path 2 is sweep generator and the sweep type is ramp, including up and down.

Setting format: [:SOURce]:FM[1]|2:INTernal:SWEep:FUNCtion:SHAPe:RAMP <Mode>

Query format: [:SOURce]:FM[1]|2:INTernal:SWEep:FUNCtion:SHAPe:RAMP?

Parameter description:

<Mode>	Discrete data. The values of ramp signal type are as follows:	
	POSitive	up
	NEGative	down.

Example: FM:INTernal:SWEep:FUNCtion:SHAPe:RAMP NEGative set the signal output type to down when the waveform of Path 1 is sweep generator and the sweep type is ramp.

Reset state: POS

Key path: [Modulation] —> [Frequency Modulation] —> [Base Config] —> [Path 1|Path 2] —> [FM Waveform] —> [Sweep Fun-Generator]

[:SOURce]:FM[1]|2:INTernal:SWEep:FUNCtion:TIME <Time>

Function description: This command is used to set the sweep time of sweep generator when the waveform of FM Path 1 or Path 2 is sweep generator.

Setting format: [:SOURce]:FM[1]|2:INTernal:SWEep:FUNCtion:TIME <Time>

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

Parameter description:

< Time > sweep time of sweep function generator.

Range: 0.1ms[0.01us, 40s].

Example: FM:INTernal:SWEep:FUNCtion:TIME 5s set the sweep time of sweep generator to 5s when the waveform of Path 1 is sweep generator.

Reset state: 0.1ms

Key path: [Modulation] —> [Frequency Modulation] —> [Base Config] —> [Path 1|Path 2] —> [FM Waveform] —> [Sweep Fun-Generator]

[:SOURce]:FM[1]|2:INTernal:SWEep:FUNCtion:TRIGger:MODE <Mode>

Function description: This command is used to set the trigger mode of sweep generator when the waveform of FM Path 1 or Path 2 is sweep generator.

Setting format: [:SOURce]:FM[1]|2:INTernal:SWEep:FUNCtion:TRIGger:MODE <val>

Query format: [:SOURce]:FM[1]|2:INTernal:SWEep:FUNCtion:TRIGger:MODE?

Parameter description:

<Mode>	discrete data; with values taken as follows:	
	CONTinuous:	Continuous
	SINGle:	Single

Example: FM:INTernal:SWEep:FUNCtion:TRIGger:MODE CONTinuous set the trigger mode of the sweep function generator to continuous.

Reset state: CONTinuous

Key path: [Modulation] —> [Frequency Modulation] —> [Base Config] —> [Path 1|Path 2] —> [FM Waveform] —> [Sweep Fun-Generator]

[:SOURce]:FM[1]|2:INTernal:SWEep:FUNCtion:TRIGger:PERiod <Time>

Function description: This command is used to set the sweep timer period of sweep generator when the waveform of FM Path 1 or Path 2 is sweep generator and the trigger type is timed trigger.

Setting format: [:SOURce]:FM[1]|2:INTernal:SWEep:FUNCtion:TRIGger: PERiod <val>

Query format: [:SOURce]:FM[1]|2:INTernal:SWEep:FUNCtion:TRIGger: PERiod?

Parameter description:

<Time> sweep timer period.

Range: 0.1ms[10ns, 40s].

Example: FM:INTernal:SWEep:FUNCtion:TRIGger:PERiod 1s set the sweep timer period of the sweep function generator to 1s.

Reset state: 0.1ms

Key path: [Modulation] —> [Frequency Modulation] —> [Base Config] —> [Path 1|Path 2] —> [FM Waveform] —> [Sweep Fun-Generator]

[:SOURce]:FM[1]|2:INTernal:SWEep:FUNCtion:TRIGger:TYPE <Mode>

Function description: This command is used to set the trigger type of sweep generator when the waveform of FM Path 1 or Path 2 is sweep generator.

Setting format: [:SOURce]:FM[1]|2:INTernal:SWEep:FUNCtion:TRIGger:TYPE <val>

Query format: [:SOURce]:FM[1]|2:INTernal:SWEep:FUNCtion:TRIGger:TYPE?

Parameter description:

<Mode> Discrete data, with values taken as follows:

IMMediate:	Auto
KEY:	Trigger key
BUS:	Bus
INTernal:	Internal
EXTernal:	External
TIMer:	Timed trigger

Example: FM:INTernal:SWEep:FUNCtion:TRIGger:TYPE BUSset the trigger type of the sweep function generator to bus.

Reset state: IMMediate

Key path: [Modulation] —> [Frequency Modulation] —> [Base Config] —> [Path 1|Path 2] —> [FM Waveform] —> [Sweep Fun-Generator]

3.3.10 PHASe MODulation Subsystem

The following commands are used to select the phase modulation (PM) mode, including:

[:SOURce]: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:

1435	Current frequency	Phase deviation
	9kHz – 250MHz	0 – 4.000rad
	250MHz – 375MHz	0 – 1.000rad
	375MHz – 750GHz	0 – 2.000rad
	750MHz – 1.5GHz	0 – 4.000rad
	1.5GHz – 3GHz	0 – 8.000rad
	3GHz – 6GHz	0 – 16.000rad
	6GHz – 12GHz	0 – 32.000rad
	12GHz – 24GHz	0 – 64.000rad
	24GHz – 40GHz	0 – 128.000rad

Example: PM2:DEViation 3rad set the phase deviation of PM Path 2 to 3rad.

Reset state: 0.001rad

Key path: [Modulation] —> [Phase Modulation] —> [Base Config] —> [Path 1|Path 2] —> [PM Bias]

[:SOURce]: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. Please refer to “:PM:SOURce” for relevant command.

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

Query format: [:SOURce]:PM[1]|2:INTernal:FREQuency?

Parameter description:

<**Frequency**> the relationship between the PM waveform and the PM rate range is as follows:

Sine: [0.005Hz, 10.000000000MHz]

Square: [0.005Hz, 10.000000000MHz]

Triangle: [0.005Hz, 10.000000000MHz]

Ramp: [0.005Hz, 10.000000000MHz]

Example: PM:INTernal:FREQuency 300KHz set the PM rate of Path 1 to 300kHz.

Reset state: 0.001MHz

Key path: [Modulation] —> [Phase Modulation] —> [Base Config] —> [Path 1|Path 2] —> [PM rate]

[:SOURce]: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 zizag, including up and down. Please refer to “:PM[1]|2:INTernal:SHAPe” for PM waveform.

Setting format: [:SOURce]:PM[1]|2:INTernal:RAMP POSitive|NEGative

Query format: [:SOURce]:PM[1]|2:INTernal:RAMP?

Parameter description:

<Mode> Discrete data. The values of signal output type when the PM waveform is zigzag are as follows:

POSitive	up
NEGative	down

Example: PM:INTernal:SHAPe:RAMP NEG set the ramp for PM Path 1 to down.

Reset state: POS

Key path: [Modulation] —> [Phase Modulation] —> [Base Config] —> [Path 1|Path 2] —> [PM Waveform] —> [Zigzag]

[:SOURce]:PM[1]|2:INTernal:SHAPe <Mode>

Function description: This command is used to set the output waveform of PM Path 1 or Path 2, including sine, square, triangle and zigzag.

Setting format: [:SOURce]:PM[1]|2:INTernal:SHAPe SINE|SQUare|TRIangle|RAMP

Query format: [:SOURce]: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	zigzag wave.

Example: PM2:INTernal:SHAP RAMP set the PM signal waveform of Path 2 to zigzag.

Reset state: SINE

Key path: [Modulation] —> [Phase Modulation] —> [Base Config] —> [Path 1|Path 2] —> [PM Waveform]

[:SOURce]: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 path, PM and modulation are all set to ON state can the PM signal be output. Please refer to “PM:MODulation:STATe”for PM state and “OUTPut:MODulation:STATe” for modulation state.

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

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

Parameter description:

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

ON | 1: Path output ON

OFF | 0: Path output OFF.

Example: PM:STATe 1 Path 1 output ON.

Reset state: 0

Key path: [Modulation] —> [Phase Modulation] —> [Base Config] —> [Path 1|Path 2 ON/OFF]

[:SOURce]:PM:SOURce <Mode>

Function description: This command is used to select the PM source, including: internal 50Ω, external 50Ω, external 600Ω and external 1MΩ. 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]:PM:SOURce INTernal|EXT50Ω| EXT600Ω| EXT1MΩ

Query format: [:SOURce]:PM:SOURce?

Parameter description:

<Mode> Discrete data. The values of PM source are as follows:

INTernal	internal PM
EXT50Ω	external 50Ω
EXT600Ω	external 600Ω
EXT1MΩ	external 1MΩ

Example: PM:SOURce INT the PM source is internal

Reset: INT

Key path: [Modulation] —> [Phase Modulation] —> [PM Source] —> [PM source]

[:SOURce]:PM:MODulation:STATe <State>

Function description: This command is used to set the PM signal output state of the signal generator.

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

Query format: [:SOURce]:PM: MODulation:STATe?

Parameter description:

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

ON | 1: PM output ON

OFF | 0: PM output OFF.

Example: PM:STATe 0 PM OFF.

Reset state: 0

Key path: [Modulation] —> [Phase Modulation] —> [Base Config] —> [PM ON/OFF]

[:SOURce]:PM:EXTernal:COUPling <Mode>

Function description: This command is used to set the PM external input coupling mode.

Setting format: [:SOURce]:PM:EXTernal:COUPling DC|AC

Query format: [:SOURce]:PM:EXTernal: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: [Modulation] —> [Phase Modulation] —> [PM Source] —> [EXT Couple Type]

[:SOURce]:PM:EXTernal:PATH <Mode>

Function description: This command is used to set the PM external input path.

Setting format: [:SOURce]:PM:EXTernal:PATH EXTernal[1]|2

Query format: [:SOURce]:PM:EXTernal:PATH?

Parameter description:

<Mode> Discrete data. The values of PM external input path are as follows:

EXTernal1	external path 1
EXTernal2	external path 2

Example: PM:EXTernal:PATH EXTernal2 set the external input path to external 2.

Reset state: EXTernal1

Key path: [Modulation] —> [Phase Modulation] —> [PM Source] —> [Ext input path]

[:SOURce]:PM[1]|2:INTernal:DUAL:FUNCtion:AMPLitude:PERCent <val>

Function description: This command is used to set the amplitude ratio of dual function generator relative to audio 1 when the waveform of PM Path 1 or Path 2 is dual function generator.

Setting format: [:SOURce]:PM[1]|2:INTernal:DUAL:FUNCtion:AMPLitude:PERCent <val>

Query format: [:SOURce]:PM[1]|2:INTernal:DUAL:FUNCtion:AMPLitude:PERCent?

Parameter description:

<val> the values of amplitude ratio of dual function generator relative to audio 1 are as follows:

Range: 50 [0,100]

Example: PM:INTernal:DUAL:FUNCtion:AMPLitude:PERCent 30 set the amplitude ratio of dual function generation in Path 1 relative to audio 1 to 30%.

Reset state: 50

Key path: [Modulation] —> [Phase Modulation] —> [Base Config] —> [Path 1|Path 2] —> [PM Waveform] —> [Dual Fun-Generator]

[:SOURce]:PM[1]|2:INTernal:DUAL:FUNCtion[1]|2:FREQuency <Frequency >

Function description: This command is used to set the frequency of dual function generator relative to audio 1 when the waveform of PM Path 1 or Path 2 is dual function generator.

Setting format: [:SOURce]:PM[1]|2:INTernal:DUAL:FUNCtion[1]|2:FREQuency <Frequency >

Query format: [:SOURce]:PM[1]|2:INTernal:DUAL:FUNCtion[1]|2:FREQuency?

Parameter description:

<Frequency> frequency of dual function generator relative to audio 1.

Range: 1kHz[0.001Hz, 1MHz].

Example: PM:INTernal:DUAL:FUNCtion:FREQuency 20kHz set the frequency of dual function generation in Path 1 relative to audio 1 to 20kHz.

Reset state: 1kHz

Key path: [Modulation] —> [Phase Modulation] —> [Base Config] —> [Path 1|Path 2] —> [PM Waveform] —> [Dual Fun-Generator]

[:SOURce]:PM[1]|2:INTernal:DUAL:FUNCtion[1]|2:PERCent <val>

Function description: This command is used to set the pulse duty factor of dual function generator relative to audio 1|2 when the waveform of PM Path 1 or Path 2 is dual function generator.

Setting format: [:SOURce]:PM[1]|2:INTernal:DUAL:FUNCtion[1]|2:PERCent <val>

Query format: [:SOURce]:PM[1]|2:INTernal:DUAL:FUNCtion[1]|2:PERCent?

Parameter description:

<val> the values of pulse duty factor of dual function generator relative to audio 1 are as follows:

Range: 50 [0,100]

Example: PM:INTernal:DUAL:FUNCtion: PERCent 20 set the pulse duty factor of dual function generator in Path 1 relative to audio 1 to 20%.

Reset state: 50%

Key path: [Modulation] —> [Phase Modulation] —> [Base Config] —> [Path 1|Path 2] —> [PM Waveform] —> [Dual Fun-Generator]

[:SOURce]:PM[1]|2:INTernal:DUAL:FUNCtion:POFFset <val>

Function description: This command is used to set the phase offset of dual function generator relative to audio 1 when the waveform of PM Path 1 or Path 2 is dual function generator.

Setting format: [:SOURce]:PM[1]|2:INTernal:DUAL: FUNCtion:POFFset <val>

Query format: [:SOURce]:PM[1]|2:INTernal:DUAL: FUNCtion:POFFset?

Parameter description:

<val> the values of phase offset of dual function generator relative to audio 1 are as follows:

Range: 0deg [0deg, 360deg]

Example: PM:INTernal:DUAL: FUNCtion:POFFset 60 set the phase offset of dual function generator in Path 1relative to audio 1 to 60deg.

Reset state: 0

Key path: [Modulation] —> [Phase Modulation] —> [Base Config] —> [Path 1|Path 2] —> [PM Waveform] —> [Dual Fun-Generator]

[:SOURce]:PM[1]|2:INTernal:DUAL:FUNCtion:SHAPe <Mode>

Function description: This command is used to set the output waveform of dual function generator when the waveform of PM Path 1 or Path 2 is dual function generator.

Setting format: [:SOURce]:PM[1]|2:INTernal:DUAL:FUNCtion:SHAPe <Mode>

Query format: [:SOURce]:PM[1]|2:INTernal:DUAL:FUNCtion:SHAPe?

Parameter description:

<Mode> Discrete data. The values of output waveform are as follows:

Sine	sine wave,
SQUare	square wave,
TRIangle	triangle wave,
Ramp	zigzag wave,
PULSe	pulse

Example: PM:INTernal:DUAL:FUNCtion:SHAPe TRIangle set the output waveform of dual function generator in Path 1 to triangle.

Reset state: SINE

Key path: [Modulation] —> [Phase Modulation] —> [Base Config] —> [Path 1|Path 2] —> [PM Waveform] —> [Dual Fun-Generator]

[:SOURce]:PM[1]|2:INTernal:DUAL:FUNCtion:SHAPe:RAMP <Mode>

Function description: This command is used to set the signal output type when the waveform of PM Path 1 or Path 2 is dual function generator and the output waveform of the generator is zigzag, including up and down.

Setting format: [:SOURce]:PM[1]|2:INTernal:DUAL:FUNCtion:SHAPe:RAMP <Mode>

Query format: [:SOURce]:PM[1]|2:INTernal:DUAL:FUNCtion:SHAPe:RAMP?

Parameter description:

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

POSitive up

NEGative down.

Example: PM:INTernal:DUAL:FUNCtion:SHAPe:RAMP POSitive set the ramp to up when the output waveform of dual function generator in Path 1 is zigzag.

Reset state: POS

Key path: [Modulation] —> [Phase Modulation] —> [Base Config] —> [Path 1|Path 2] —> [PM Waveform] —> [Dual Fun-Generator]

[:SOURce]:PM[1]|2:INTernal:FUNCtion[1]|2:FREQuency <Frequency>

Function description: This command is used to set the output frequency of function generator 1|2 when the waveform of PM Path 1 or Path 2 is function generator 1|2.

Setting format: [:SOURce]:PM[1]|2:INTernal:FUNCtion[1]|2:FREQuency <Frequency>

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

Parameter description:

<Frequency> output frequency of function generator 1|2.

Range: 1kHz[0.001Hz, 1MHz].

Example: PM:INTernal:FUNCtion2:FREQuency 10kHz set the output frequency of function generator 2 in Path 1 to 10kHz.

Reset state: 1kHz

Key path: [Modulation] —> [Phase Modulation] —> [Base Config] —> [Path 1|Path 2] —> [PM Waveform] —> [Generator 1|2]

[:SOURce]:PM[1]|2:INTernal:FUNCtion[1]|2:PERCent <val>

Function description: This command is used to set the pulse duty factor of function generator 1|2 when the waveform of PM Path 1 or Path 2 is function generator 1|2.

Setting format: [:SOURce]:PM[1]|2:INTernal:FUNCtion[1]|2:PERCent <val>

Query format: [:SOURce]:PM[1]|2:INTernal:FUNCtion[1]|2:PERCent?

Parameter description:

<val> **the** values of pulse duty factor of function generator 1|2 are as follows:

Range: 50 [0,100]

Example: PM:INTernal:FUNCtion2:PERCent 50 set the pulse duty factor of function generator 2 in Path 1 to 50%.

Reset state: 50%

Key path: [Modulation] —> [Phase Modulation] —> [Base Config] —> [Path 1|Path 2] —> [PM Waveform] —> [Generator 1|2]

[:SOURce]:PM[1]|2:INTernal:FUNCtion[1]|2:SHAPe <Mode>

Function description: This command is used to set the output waveform of function generator

1|2 when the waveform of PM Path 1 or Path 2 is function generator 1|2.

Setting format: [:SOURce]:PM[1]|2:INTernal:FUNCtion[1]|2:SHAPe <Mode>

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

Parameter description:

<Mode> Discrete data. The values of output waveform are as follows:

Sine	sine wave,
SQUare	square wave,
TRIangle	triangle wave,
Ramp	zigzag wave,
PULSe	pulse.

Example: PM:INTernal:FUNCtion2:SHAPe TRIangle set the output waveform of function generator 2 in Path 1 to triangle.

Reset state: 50%

Key path: [Modulation] —> [Phase Modulation] —> [Base Config] —> [Path 1|Path 2] —> [PM Waveform] —> [Generator 1|2]

[:SOURce]:PM[1]|2:INTernal:FUNCtion[1]|2:SHAPe:RAMP <Mode>

Function description: This command is used to set the signal output type when the waveform of PM Path 1 or Path 2 is function generator 1|2 and the output waveform of the generator 1|2 is zigzag, including up and down.

Setting format: [:SOURce]:PM[1]|2:INTernal:FUNCtion[1]|2:SHAPe:RAMP <Mode>

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

Parameter description:

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

POSitive	up
NEGative	down.

Example: PM:INTernal:FUNCtion2:SHAPe:RAMP NEGative set the ramp to down when the output waveform of function generator 2 in Path 1 is ramp.

Reset state: POS

Key path: [Modulation] —> [Phase Modulation] —> [Base Config] —> [Path 1|Path 2] —> [PM Waveform] —> [Generator 1|2]

[:SOURce]:PM[1]|2:INTernal:NOISe:FUNCtion[1]|2:TYPE <Mode>

Function description: This command is used to set the noise type of noise generator 1|2 when the waveform of PM Path 1 or Path 2 is noise generator 1|2.

Setting format: [:SOURce]:PM[1]|2:INTernal:NOISe:FUNCtion[1]|2:TYPE <Mode>

Query format: [:SOURce]:PM[1]|2:INTernal:NOISe:FUNCtion[1]|2:TYPE?

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: PM:INTernal:FUNCtion2:SHAPe GAUSsian set the noise type of function generator 2 in Path 1 to Gaussian.

Reset state: GAUS

Key path: [Modulation] —> [Phase Modulation] —> [Base Config] —> [Path 1|Path 2] —> [PM Waveform] —> [Noise generator 1|2]

[:SOURce]:PM[1]|2:INTernal:SWEep:FUNCtion:FREQuency:STARt <Frequency>

Function description: This command is used to set the start frequency of sweep generator when the waveform of PM Path 1 or Path 2 is sweep generator.

Setting format: [:SOURce]:PM[1]|2:INTernal:SWEep:FUNCtion:FREQuency:STARt <Frequency>

Query format: [:SOURce]:PM[1]|2:INTernal:SWEep:FUNCtion:FREQuency:STARt?

Parameter description:

<Frequency> start frequency of sweep function generator.

Range: 1kHz[0.001Hz, 1MHz].

Example: PM:INTernal:SWEep:FUNCtion:FREQuency:STARt 30kHz set the start frequency of sweep function generator in Path 1 to 30kHz.

Reset state: 1kHz

Key path: [Modulation] —> [Phase Modulation] —> [Base Config] —> [Path 1|Path 2] —> [PM Waveform] —> [Sweep Fun-Generator]

[:SOURce]:PM[1]|2:INTernal:SWEep:FUNCtion:FREQuency:STOP <Frequency>

Function description: This command is used to set the stop frequency of sweep generator when the waveform of PM Path 1 or Path 2 is sweep generator.

Setting format: [:SOURce]:PM[1]|2:INTernal:SWEep:FUNCtion:FREQuency: STOP <Frequency>

Query format: [:SOURce]:PM[1]|2:INTernal:SWEep:FUNCtion:FREQuency: STOP?

Parameter description:

<Frequency> stop frequency of sweep function generator.

Range: 1kHz[0.001Hz, 1MHz].

Example: PM:INTernal:SWEep:FUNCtion:FREQuency: STOP 50kHz set the stop frequency of sweep function generator in Path 1 to 50kHz.

Reset state: 1kHz

Key path: [Modulation] —> [Phase Modulation] —> [Base Config] —> [Path 1|Path 2] —> [PM Waveform] —> [Sweep Fun-Generator]

[:SOURce]:PM[1]|2:INTernal:SWEep:FUNCtion:SHAPe <Mode>

Function description: This command is used to set the sweep type of sweep generator when the waveform of PM Path 1 or Path 2 is sweep generator.

Setting format: [:SOURce]:PM[1]|2:INTernal:SWEep:FUNCtion:SHAPe <Mode>

Query format: [:SOURce]:PM[1]|2:INTernal:SWEep:FUNCtion:SHAPe?

Parameter description:

<Mode> Discrete data. The values of output waveform are as follows:

Sine	sine wave,
SQUare	square wave,

	TRIangle	triangle wave,
	RAMP	zigzag wave.

Example: PM:INTernal:SWEep:FUNCtion:SHAPe TRIangle set the sweep type of sweep function generator in Path 1 to triangle.

Reset state: 1kHz

Key path: [Modulation] —> [Phase Modulation] —> [Base Config] —> [Path 1|Path 2] —> [PM Waveform] —> [Sweep Fun-Generator]

[:SOURce]:PM[1]|2:INTernal:SWEep:FUNCtion:SHAPe:RAMP <Mode>

Function description: This command is used to set the signal output type when the waveform of PM Path 1 or Path 2 is sweep generator and the sweep type is zigzag, including up and down.

Setting format: [:SOURce]:PM[1]|2:INTernal:SWEep:FUNCtion:SHAPe:RAMP <Mode>

Query format: [:SOURce]:PM[1]|2:INTernal:SWEep:FUNCtion:SHAPe:RAMP?

Parameter description:

<Mode> Discrete data. The values of zigzag signal type are as follows:

POSitive	up
NEGative	down.

Example: PM:INTernal:SWEep:FUNCtion:SHAPe:RAMP NEGative set the signal output type to down when the waveform of Path 1 is sweep generator and the sweep type is ramp.

Reset state: POS

Key path: [Modulation] —> [Phase Modulation] —> [Base Config] —> [Path 1|Path 2] —> [PM Waveform] —> [Sweep Fun-Generator]

[:SOURce]:PM[1]|2:INTernal:SWEep:FUNCtion:TIME <Time>

Function description: This command is used to set the sweep time of sweep generator when the waveform of PM Path 1 or Path 2 is sweep generator.

Setting format: [:SOURce]:PM[1]|2:INTernal:SWEep:FUNCtion:TIME <Time>

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

Parameter description:

< Time > sweep time of sweep function generator.

Range: 0.1ms[0.01us, 40s].

Example: PM:INTernal:SWEep:FUNCtion:TIME 5s set the sweep time of sweep generator to 5s when the waveform of Path 1 is sweep generator.

Reset state: 0.1ms

Key path: [Modulation] —> [Phase Modulation] —> [Base Config] —> [Path 1|Path 2] —> [PM Waveform] —> [Sweep Fun-Generator]

[:SOURce]:PM[1]|2:INTernal:SWEep:FUNCtion:TRIGger:MODE <Mode>

Function description: This command is used to set the trigger mode of sweep generator when the waveform of PM Path 1 or Path 2 is sweep generator.

Setting format: [:SOURce]:PM[1]|2:INTernal:SWEep:FUNCtion:TRIGger:MODE <val>

Query format: [:SOURce]:PM[1]|2:INTernal:SWEep:FUNCtion:TRIGger:MODE?

Parameter description:

<Mode>	Discrete data, with values taken as follows:
	CONTinuous: Continuous
	SINGle: Single
Example:	PM:INTernal:SWEep:FUNCtion:TRIGger:MODE CONTinuous set the trigger mode of the sweep function generator to continuous.
Reset state:	CONTinuous
Key path:	[Modulation] —> [Phase Modulation] —> [Base Config] —> [Path 1|Path 2] —> [PM Waveform] —> [Sweep Fun-Generator]

[:SOURce]:PM[1]|2:INTernal:SWEep:FUNCtion:TRIGger:PERiod <Time>

Function description:	This command is used to set the sweep timer period of sweep generator when the waveform of PM Path 1 or Path 2 is sweep generator and the trigger type is timed trigger.
Setting format:	[:SOURce]:PM[1]|2:INTernal:SWEep:FUNCtion:TRIGger: PERiod <val>
Query format:	[:SOURce]:PM[1]|2:INTernal:SWEep:FUNCtion:TRIGger: PERiod?
Parameter description:	
<Time>	sweep timer period.
	Range: 0.1ms[10ns, 40s].
Example:	PM:INTernal:SWEep:FUNCtion:TRIGger:PERiod 1s set the sweep timer period of the sweep function generator to 1s.
Reset state:	0.1ms
Key path:	[Modulation] —> [Phase Modulation] —> [Base Config] —> [Path 1|Path 2] —> [PM Waveform] —> [Sweep Fun-Generator]

[:SOURce]:PM[1]|2:INTernal:SWEep:FUNCtion:TRIGger:TYPE <Mode>

Function description:	This command is used to set the trigger type of sweep generator when the waveform of PM Path 1 or Path 2 is sweep generator.
Setting format:	[:SOURce]:PM[1]|2:INTernal:SWEep:FUNCtion:TRIGger:TYPE <val>
Query format:	[:SOURce]:PM[1]|2:INTernal:SWEep:FUNCtion:TRIGger:TYPE?
Parameter description:	
<Mode>	Discrete data, with values taken as follows:
	IMMediate: Auto
	KEY: Trigger key
	BUS: Bus
	INTernal: Internal
	EXTernal: External
	TIMer: Timed trigger
Example:	PM:INTernal:SWEep:FUNCtion:TRIGger:TYPE BUSset the trigger type of the sweep function generator to bus.
Reset state:	IMMediate
Key path:	[Modulation] —> [Phase Modulation] —> [Base Config] —> [Path 1|Path 2] —> [PM Waveform] —> [Sweep Fun-Generator]

3.3 Instrument Subsystem Command

3.3.11 Digital MODulation Subsystem

The following commands are used to select the digital modulation (DM) mode (some commands do not support for the time being and will be improved later), including:

[:SOURce]:DM:IQADjustment:GAIN <Gain>

Function description: When I/Q adjustment is ON, set the gain of signal I of the signal generator relative to signal Q. Please refer to ":DM:IQADjustment[:STATe]" for the state of I/Q adjustment.

Setting format: [:SOURce]:DM:IQADjustment:GAIN <val>

Query format: [:SOURce]:DM:IQADjustment:GAIN?

Parameter description:

<Gain> signal I/Q gain balance.

Range: 0dB[-4.00dB, +4.00dB].

Example: DM:IQADjustment:GAIN 0dB

Set the gain balance of signal I and signal Q to 0dB。

Reset state: 0dB

Key path: **[I/Q]** —> [I/Q Input Adj] —> [Gain balance]

[:SOURce]:DM:IQADjustment:IOFFset <Offset>

Function description: When I/Q 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 and the minimum resolution being 0.025%. 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]:DM:IQADjustment:IOFFset <val>

Query format: [:SOURce]:DM:IQADjustment:IOFFset?

Parameter description:

<Offset> signal I offset in I/Q

Range: 0 [-50, +50].

Example: DM:IQADjustment:IOFfset 30

Set I offset to 30%.

Reset state: 0

Key path: **[I/Q]** —> [I/Q Input Adj] —> [I Offset]

[:SOURce]: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 and the minimum resolution being 0.025%. 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]:DM:IQADjustment:QOFFset <val>

Query format: [:SOURce]:DM:IQADjustment:QOFFset?

Parameter description:

<Offset> signal Q offset in I/Q

Value range: 0 [-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]:DM:IQADjustment:QSKew <Offset>

Function description: When I/Q 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. If the current carrier frequency exceeds 3.2GHz, the orthogonal deviation error may exceed the value specified for product sample index of 1435 series signal generator.

Setting format: [:SOURce]:DM:IQADjustment:QSKew <val>

Query format: [:SOURce]:DM:IQADjustment:QSKew?

Parameter description:

<Offset> orthogonal offset of I/Q adjustment.

Range: 0deg [-10.00deg, +10.00deg].

Example: DM:IQADjustment:QSKew 30deg

Set the orthogonal offset of I/Q adjustment to 30deg.

Reset state: 0deg

Key path: **[I/Q]** **—>** [I/Q Input Adj] **—>** [Orthogonal Offset]

[:SOURce]:DM:IQADjustment[:STATe] <State>

Function description: This command is used to set the I/Q adjustment enable to ON/OFF state. After this function is enabled, I/Q adjustment parameters, such as gain balance, I offset, Q offset and orthogonal offset, will be added to the adjustment circuit. When this function is disabled, the above parameters will not be used, but the modulator attenuation will not be affected by the I/Q adjustment state. Please refer to “:DM:MODulation:ATTenuation” and “:DM:IQADjustment:EXTernal:IQATten” for relevant commands.

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

Query format: [:SOURce]:DM:IQADjustment[:STATe] ?

Parameter description:

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

ON | 1: I/Q adjustment ON

OFF | 0: I/Q adjustment OFF.

Example: DM:IQADjustment 1 enable I/Q adjustment function.

Reset state: 0

Key path: **[I/Q]** **—>** [I/Q Input Adj] **—>** [I/Q adjust ON/OFF]

[:SOURce]:DM:MODulation:ATTenuation <Atten>

Function description: This command is used to set the attenuation of signal I/Q modulated through the RF path of the signal generator. The output attenuation may be set when the attenuator state is manual. Even if the I/Q adjustment function is disabled at this time, the attenuation is still valid. Please refer to “:DM:MODulation:ATTenuation:AUTO” and “:DM:IQADjustment[:STATe]” for relevant commands.

Setting format: [:SOURce]:DM:MODulation:ATTenuation <val>

Query format: [:SOURce]:DM:MODulation:ATTenuation?

Parameter description:

<Atten> I/Q modulator attenuation.

Range: 12.00dB[0.00dB, 40.00dB].

Example: DM:MODulation:ATTenuation 10dB I/Q modulator attention is 10dB.

Reset state: 0dB

Key path: **[I/Q] —>** [Attenuation] —> [Attenuation]

[:SOURce]:DM:MODulation:ATTenuation:AUTO <State>

Function description: This command is used to set the attenuator in Path I/Q of the signal generator to manual state. When the manual state is enabled, maintain the current attenuation. Please refer to ":DM:MODulation:ATTenuation" for the modulator attenuation. After the manual state is disabled, users cannot change the attenuation, and the signal generator will automatically select the attenuation most suitable for its current state.

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

Query format: [:SOURce]:DM:MODulation:ATTenuation:AUTO?

Parameter description:

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

ON | 1: manual modulator attenuation control

OFF | 0: automatic modulator attenuation control

Example: DM:MODulation:ATTenuation:AUTO 1

The modulator attenuation control is in manual state.

Reset state: 0

Key path: **[I/Q] —>** [Attenuation] —> [Modulation attenuation: Manual/Auto]

[:SOURce]:DM:STATe <State>

Function description: This command is used to enable internal I/Q modulator to ON/OFF state.

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

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

Parameter description:

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

ON | 1: I/Q modulation output ON

OFF | 0: I/Q modulation output OFF.

Example: DM:STATe 1 turn on I/Q modulator.

Reset state: 0

Key path: [I/Q] **—>** [Base Config] —> [I/Q modulate ON/OFF]

[:SOURce]: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]:DM:SOURce EXTernal| INTernal

Query format: [:SOURce]:DM:SOURce?

Parameter description:

<Mode> Discrete data. When the I/Q filter is in manual mode, the values of filter

selection are as follows:

EXTernal | 0: external 50ohm impedance matched I/Q signal input

INTernal | 1: internal I/Q signal input I/Q modulator

Example: [:SOURce]:DM:SOURce EXT select external as I/Q modulation source.

Reset state: EXT

Key path: [I/Q] —> [Base Config] —> [Data Source]

[:SOURce]:DM:EXTernal:BWIDth[:STATe] <State>

Function description: This command is used to set external broadband I/Q input to ON/OFF state.

Setting format: [:SOURce]:DM:EXTernal:BWIDth[:STATe] ON|OFF|1|0

Query format: [:SOURce]:DM:EXTernal:BWIDth[:STATe]?

Parameter description:

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

ON | 1: External broadband I/Q input ON

OFF | 0: external broadband I/Q input OFF.

Example: DM:EXT:BWID:STATe 1 external broadband I/Q input ON

Reset state: 0

Key path: **[**I/Q**]** **—**> [Base Config] —> [External Wideband I/Q Input ON/OFF]

[:SOURce]:DM:IQADjustmentOUTPut[:STATe] <State>

Function description: This command is used to set I/Q input adjustment to ON/OFF state.

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

Query format: [:SOURce]:DM:IQADjustment:OUTPut [:STATe]?

Parameter description:

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

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**]** **—**> [I/Q Output Adj] —> [I/Q Output Adj ON/OFF]

[:SOURce]:DM:IQADjustment:OUTPut:ATTen <Atten >

Function description: This command is used to set the attenuation of I/Q output adjustment. When I/Q output adjustment is ON, the command works. Please refer to "DM:IQADjustment:OUTPut" for I/Q output adjustment state.

Setting format: [:SOURce]:DM:IQADjustment:OUTPut:ATTen <val>.

Query format: [:SOURce] :DM:IQADjustment:OUTPut:ATTen?

Parameter description:

<Atten> attenuation of I/Q output adjustment

Range: 0dB [0dB, 94.5dB].

Example: DM:IQADjustment:OUTPut:ATTen 10dB set the I/Q output attention to 10dB.

Reset state: 0

Key path: **[**I/Q**] —**> [I/Q Output Adj] —> [Attenuation]

[:SOURce]: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 "DM:IQADjustment:OUTPut" for I/Q output adjustment state.

Setting format: [:SOURce]:DM:IQADjustment:OUTPut:GAIN <val>.

Query format: [:SOURce] :DM:IQADjustment:OUTPut:GAIN?

Parameter description:

<Gain> I/Q output adjustment gain balance.

Range: 0dB [-4dB, 4dB].

Example: DM:IQADjustment:OUTPut:GAIN 2dB set I/Q output gain balance to 2dB.

Reset state: 0

Key path: [I/Q] —> [I/Q Output Adj] —> [Gain Balance]

[:SOURce]:DM:IQADjustment:OUTPut:IOFFset <offset >

Function description: This command is used to set I offset of I/Q output adjustment. When I/Q output adjustment is ON, the command works. Please refer to "DM:IQADjustment:OUTPut" for I/Q output adjustment state.

Setting format: [:SOURce]:DM:IQADjustment:OUTPut:IOFFset <val>.

Query format: [:SOURce] :DM:IQADjustment:OUTPut: IOFFset?

Parameter description:

<offset> I offset of I/Q output adjustment

Range: 0V[-1V, 1V].

Example: DM:IQADjustment:OUTPut:IOFFset 1V set I offset of I/Q output to 1V.

Reset state: 0

Key path: [I/Q] —> [I/Q Output Adj] —> [I Offset]

[:SOURce]:DM:IQADjustment:OUTPut:UIOFfset <offset >

Function description: This command is used to set I/offset of I/Q output adjustment. When I/Q output adjustment is ON, the command works. Please refer to "DM:IQADjustment:OUTPut" for I/Q output adjustment state.

Setting format: [:SOURce]:DM:IQADjustment:OUTPut:UIOFfset <val>.

Query format: [:SOURce] :DM:IQADjustment:OUTPut:UIOFfset?

Parameter description:

<offset> I/offset of I/Q output adjustment

Range: 0V [-1V, 1V].

Example: DM:IQADjustment:OUTPut:UIOFfset 1V set I/offset of I/Q output to 1V.

Reset state: 0

Key path: [I/Q] —> [I/Q Output Adj] —> [I/Offset]

[:SOURce]:DM:IQADjustment:OUTPut:QOFFset <offset >

Function description: This command is used to set Q offset of I/Q output adjustment. When I/Q output adjustment is ON, the command works. Please refer to “DM:IQADjustment:OUTPut” for I/Q output adjustment state.

Setting format: [:SOURce]:DM:IQADjustment:OUTPut:QOFFset <val>.

Query format: [:SOURce] :DM:IQADjustment:OUTPut:QOFFset?

Parameter description:

<offset> Q offset of I/Q output adjustment

Range: 0V [-1V, 1V].

Example: DM:IQADjustment:OUTPut:QOFFset 1V set Q offset of I/Q output to 1V.

Reset state: 0

Key path: [I/Q] —> [I/Q Output Adj] —> [Q Offset]

[:SOURce]:DM:IQADjustment:OUTPut:UQOFfset <offset >

Function description: This command is used to set Q/offset of I/Q output adjustment. When I/Q output adjustment is ON, the command works. Please refer to “DM:IQADjustment:OUTPut” for I/Q output adjustment state.

Setting format: [:SOURce]:DM:IQADjustment:OUTPut:UQOFfset <val>.

Query format: [:SOURce] :DM:IQADjustment:OUTPut:UQOFfset?

Parameter description:

<offset> Q/offset of I/Q output adjustment

Range: 0V [-1V, 1V].

Example: DM:IQADjustment:OUTPut:UQOFfset 1V setQ/offset of I/Q output to 1V.

Reset state: 0

Key path: [I/Q] —> [I/Q Output Adj] —> [Q/Offset]

[:SOURce]:DM:IQADjustment:OUTPut:SKEW <skew >

Function description: This command is used to set orthogonal offset of I/Q output adjustment. When I/Q output adjustment is ON, the command works. Please refer to “DM:IQADjustment:OUTPut” for I/Q output adjustment state.

Setting format: [:SOURce]:DM:IQADjustment:OUTPut:SKEW <val>.

Query format: [:SOURce] :DM:IQADjustment:OUTPut:SKEW?

Parameter description:

<skew> orthogonal offset of I/Q output adjustment

Range: 0V [-10deg, 10deg].

Example: DM:IQADjustment:OUTPut:SKEW 1deg To set I/Q output orthogonality offset to 1deg.

Reset state: 0

Key path: [I/Q] —> [I/Q Output Adj] —> [Orthogonal Offset]

[:SOURce]:RADio:CUSTom:ALPHa <FilterAlpha>

Function description: This command is used to set the alpha value of Nyquist filter, Root Nyquist filter and Gaussian filter. If the user changes the value, it will affect the bandwidth occupied by the radio signal spectrum. Please refer to the command “:RADio:CUSTom:FILTer” for change of filter type.

Setting format: [:SOURce]:RADio:CUSTom:ALPHa <val>.

Query format: [:SOURce]:RADio:CUSTom:ALPHa?

Parameter description:

<FilterAlpha> filter factor.

Range: 0.350 [0, 1.000].

Example: RADio:CUSTom:ALPHa 0.350 set the filter factor to 0.35.

Reset state: 0.350

Key path: [Base] —> [Filter] —> [Filter Factor a]

[:SOURce]:RADio:CUSTom:DATA <Mode>

Function description: This command is used to set the data source of radio modulation signal for the signal generator. Users may select such 15 data sources as PN9, PN11, PN15, PN16, PN20, PN21, PN23, FIX4, P4, P8, P16, P32, P64, PRAM and EXT (not currently supported).

Setting format: [:SOURce]:RADio:CUSTom:DATA PN9|PN11|PN15

|PN16|PN20|PN21|PN23|FIX4|P4|P8|P16|P32|P64PRAM

Query format: [:SOURce]:RADio:CUSTom:DATA?

Parameter description:

<Mode> Discrete data. Please refer to the setting command format for data source type of radio modulation signal.

Example: RADio:CUSTom:DATA FIX4

Fixed 4-bit code is selected as radio data source.

Reset state: PN9

Key path: **[Base] —**> [Base Config] —> [Data Source Selection]

Remarks: EXT is not currently supported.

[:SOURce]:RADio:CUSTom:DATA:PARM <S>

Function description: When file stream is selected as the data source in the real-time radio of the signal generator, this command is used to select the stream file that must be saved in D:\1435data\user\DataSrc with the extension of ".src". This command is for setting only.

Setting format: [:SOURce]:RADio:CUSTom:DATA

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: **[Base] —**> [Base Config] —> [Data Source: file] —> [Select File]

[:SOURce]:RADio:CUSTom:FILTer <Mode>

Function description: This command is used to select the type of radio preset filter for the signal generator, including RNYQuist, NYQuist, GAUSsian and RECTangle, where RECTangle is applicable for digital frequency modulation signals, such as FSK and MSK. Please refer to the command ":RADio:CUSTom:MODulation[:TYPE]".

Setting format: [:SOURce]:RADio:CUSTom:FILTer RNYQuist|NYQuist |GAUSsian| RECTangle

Query format: [:SOURce]:RADio:CUSTom:FILTer?

Parameter description:

<Mode> Discrete data. The values of radio preset filter type are as follows:

RNYQuist	Rnyquist filter
NYQuist	Nyquist filter
GAUSsian	Gaussian filter
RECTangle	Rectangle filter

Example: RADio:CUSTom:FILTer RNYQuist

The radio preset filter type is Rnyquist filter.

Reset state: RNYQuist

Key path: **[Base]** —> [Filter] —> [Filter Select]

[:SOURce]: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. Please refer to ":RADio:ARB:STATe" for play of remote arbitrary data block.

Setting format: [:SOURce]: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: [:SOURce]: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.

[:SOURce]:RADio:CUSTom:DATA:FIX4 <val>

Function description: This command is used to set the value of the code data when fixed 4-bit code is selected as the data source.

Setting format: [:SOURce]:RADio:CUSTom:DATA:FIX4 <val>.

Query format: [:SOURce]:RADio:CUSTom:DATA:FIX4?

Parameter description:

< val > value of code data.

Range: 0 [0, 15].

Example: RADio:CUSTom:DATA:FIX4 10 set the code data to 10.

Reset state: 0

Key path: **[Base]** —> [Base Config] —> [Data Source: fix 4] —> [Code Data]

[:SOURce]:RADio:CUSTom:MODulation:FSK[:DEViation] <Dev>

Function description: This command is used to set the frequency deviation of FSK when the radio modulation type is FSK mode. The value works after the user select FSK mode as the modulation type. Please refer to ":RADio:CUSTom:MODulation[:TYPE]" for the radio modulation type.

Setting format: [:SOURce]:RADio:CUSTom:MODulation:FSK[:DEViation] <val>

Query format: [:SOURce]:RADio:CUSTom:MODulation:FSK[:DEViation]?

Parameter description:

<Dev> frequency deviation of FSK when the modulation type is FSK mode.

Range: 0.4kHz [0.4kHz, 20MHz].

Example: RADio:CUSTom:MODulation:FSK 1MHz

The frequency deviation of FSK is 1MHz.

Reset state: 0.4kHz

Key path: **[Base]** —> [Modul Type] —> [FM Dev]

[:SOURce]:RADio:CUSTom:MODulation:MSK:PHASe <Phase>

Function description: This command is used to set the phase deviation of MSK when the radio modulation type is MSK mode. The value works after MSK mode is selected as the modulation type. Please refer to ":RADio:CUSTom:MODulation[:TYPE]" for the radio modulation type.

Setting format: [:SOURce]:RADio:CUSTom:MODulation:MSK:PHASe <val>

Query format: [:SOURce]:RADio:CUSTom:MODulation:MSK:PHASe?

Parameter description:

<Phase> phase deviation of MSK when the modulation type is MSK mode.

Range: 90rad [0rad, 90rad].

Example: RADio:CUSTom:MODulation:MSK:PHASe 30rad

The phase deviation is 30rad.

Reset state: 90.000rad

Key path: **[Base]** —> [Modul Type] —> [PM Bias]

[:SOURce]:RADio:CUSTom:MODulation[:TYPE] <Mode>

Function description: This command is used to set the radio modulation type.

Setting format: [:SOURce]:RADio:CUSTom:MODulation[:TYPE] BPSK|QPSK

|IS95QPSK|GRAYQPSK|OQPSK|IS95OQPSK|P4DQPSK|8PSK

|16PSK|D8PSK|MSK|2FSK|4FSK|8FSK|16FSK|C4FM|4QAM

|16QAM|32QAM|64QAM|128QAM|256QAM|512QAM|1024Q

AM|ASK

Query format: [:SOURce]:RADio:CUSTom:MODulation[:TYPE]?

Parameter description:

<Mode> Discrete data. Please refer to the setting command format for radio modulation type.

Example: RADio:CUSTom:MODulation 8PSK

The radio modulation type is 8PSK.

Reset state: QPSK

Key path: **[Base] —>** [Modul Type] —> [Modulation Type]

[:SOURce]:RADio:CUSTom:MODulation:ASK:DEPTh:PERCent <val>

Function description: This command is used to set the modulation depth of ASK when the radio modulation type is ASK mode. The value works after ASK mode is selected as the modulation type. Please refer to ":RADio:CUSTom:MODulation[:TYPE]" for the radio modulation type.

Setting format: [:SOURce]:RADio:CUSTom:MODulation:ASK:DEPTh:PERCent <val>

Query format: [:SOURce]:RADio:CUSTom:MODulation:ASK:DEPTh:PERCent?

Parameter description:

<val> modulation depth of ASK when the modulation type is ASK mode.

Range: 0[0, 100].

Example: RADio:CUSTom:MODulation:ASK:DEPTh:PERCent 30

The modulation depth of ASK is 30%.

Reset state: 100%

Key path: **[Base] —>** [Modul Type] —> [Modul Type: ASK] —> [Askn Depth]

[:SOURce]:RADio:CUSTom:MODulation:UFSK <S>

Function description: This command is used to set the user FSK file to be loaded when the radio modulation type is user FSK. Please refer to ":RADio:CUSTom:MODulation[:TYPE]" for radio modulation type. The command parameter is a file name containing the extension .fsk, without file path. The default path is D:\1435data\user\Fsk.

Setting format: [:SOURce]:RADio:CUSTom: MODulation:UFSK <S>

Parameter description:

<S> select the file to be loaded when the modulation type is user FSK.

Example: RADio:CUSTom:MODulation:UFSK "test.fsk"

Select test.fsk file.

Key path: **[Base] —>** [Modul Type] —> [Modul Type: User FSK] —> [Select File...]

[:SOURce]:RADio:CUSTom:MODulation:UIQ <S>

Function description: This command is used to set the user I/Q file to be loaded when the radio modulation type is user I/Q. Please refer to ":RADio:CUSTom:MODulation[:TYPE]" for radio modulation type. The command parameter is a file name containing the extension .iqm, without file path. The default path is D:\1435data\user\IqMap.

Setting format: [:SOURce]:RADio:CUSTom:MODulation:UIQ <S>

Parameter description:

<S> select user I/Q file when the modulation type is user I/Q

Example: RADio:CUSTom:MODulation:UIO “test.iqm”

Select test.iqm file.

Key path: **[Base] —>** [Modul Type] —> [Modul Type: User I/Q] —> [Select File]

[:SOURce]:RADio:CUSTom:SRATe <Val>

Function description: This command is used to set the symbol rate for radio signal of the signal generator, with sps, ksps, Msps and Gsps as the unit.

Setting format: [:SOURce]:RADio:CUSTom:SRATe <val>

Query format: [:SOURce]:RADio:CUSTom:SRATe?

Parameter description:

<Val> symbol rate for radio signal

The relationship among radio modulation type, symbol digit and symbol rate range is as follows:

Modulation type	Symbol digit	Symbol rate range
BPSK	1	0.00005Msps – 50Msps
MSK	1	0.00005Msps – 50Msps
2FSK	1	0.00005Msps – 50Msps
OQPSK	2	0.00005Msps – 50Msps
QPSK	2	0.00005Msps – 50Msps
8FSK	3	0.00005Msps – 50Msps
QAM16	4	0.00005Msps – 50Msps

Example: RADio:CUSTom:SRATe 3Msps the symbol rate is 3Msps.

Reset state: 24.300000ksps

Key path: **[Base] —>** [Base Config] —> [Symbo Rate]

[:SOURce]:RADio:CUSTom:STATe <State>

Function description: This command is used to enable the real-time radio function of the signal generator. When the radio function is enabled, instructions on radio and IQ modulation will be displayed in the main information display area on the user interface of the signal generator.

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

Query format: [:SOURce]:RADio:CUSTom:STATe?

Parameter description:

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

ON | 1: Radio ON

OFF | 0: Radio OFF.

Example: RADio:CUSTom:STATe 1 real-time radio ON.

Reset state: 0

Key path: **[Base] —>** [BaseBand ON/OFF]

[:SOURce]:RADio:CUSTom:POLarity[:ALL] <Mode>

Function description: This command is used to set the direction of rotation of the radio signal phase, including: normal and invert. If normal mode is selected, the signal modulation will be normal; if invert mode is selected, signal Q will be inverted to complete the inversion of carrier signal.

Setting format: [:SOURce]:RADio:CUSTom:POLarity[:ALL] NORMal|INVert

Query format: [:SOURce]:RADio:CUSTom:POLarity[:ALL]?

Parameter description:

<Mode> Discrete data. The values of rotation mode of radio signal phase are as follows:

NORMal | 0: normal

INVert | 1: invert

Example: [:SOURce]:RADio:CUSTom:POLarity[:ALL] INV

The radio signal phase is invert.

Reset state: NORM

Key path: **[Base]** **—>** [Base Config] **—>**]

[:SOURce]:RADio:CUSTom:DENCode <State>

Function description: This command is used to enable the differential encoding. When differential encoding is enabled, if the bit is different from the one before it, the modulation bit will be set to 1; if the bit is the same, the modulation bit will be set to 0. For example, when the bit is 1010 and differential encoding is enabled, the modulation bit will be 1111.

Setting format: [:SOURce]:RADio:CUSTom:DENCode ON|OFF|1|0

Query format: [:SOURce]:RADio:CUSTom:DENCode?

Parameter description:

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

ON | 1: Differential encoding ON

OFF | 0: Differential encoding OFF.

Example: [:SOURce]:RADio:CUSTom:DENCode 1 differential encoding ON

Reset state: 0

Key path: **[Base]** **—>** [Base Config] **—>** [Differential Code ON/OFF]

[:SOURce]:RADio:CUSTom:VCO:CLOCk <Mode>

Function description: This command is used to set the radio sampling clock type.

Setting format: [:SOURce]:RADio:CUSTom:VCO:CLOCk INTernal|EXTernal

Query format: [:SOURce]:RADio:CUSTom:VCO:CLOCk?

Parameter description:

<Mode> Discrete data, with values taken as follows:

INTernal: Internal sampling clock

EXTernal: External sampling clock

Example: [:SOURce]:RADio:CUSTom:VCO:CLOCk INT the radio sampling clock is internal.

Reset state: 0

Key path: **[Base]** —> [Clock] —> [Base Sample Clock]

[:SOURce]:RADio:CUSTom:TRIGger:EXTernal:SOURce <Mode>

Function description: This command is used to select the trigger input for the instrument rear panel to input the trigger signal when external is selected as trigger source of the signal generator. Users may select EXT1 or EXT2. Please refer to the command “:RADio:CUSTom:TRIGger:SOURce” for the radio signal trigger source.

Setting format: [:SOURce]:RADio:CUSTom:TRIGger:EXTernal:SOURce EXT1|EXT2

Query format: [:SOURce]:RADio:CUSTom:TRIGger:EXTernal:SOURce?

Parameter description:

<Mode> discrete data. The values of external radio trigger source are as follows:

EXT1 | 0: external trigger source 1

EXT2 | 1: external trigger source 2

Example: [:SOURce]:RADio:CUSTom:TRIGger:EXTernal:SOURce EXT1

Set to external trigger source 1.

Reset state: EXT1

Key path: **[Base]** —> [Trig Source] —> [Ext>>]

[:SOURce]:RADio:CUSTom:TRIGger:EXTernal:SOURce:DELay <Val>

Function description: This command is used to set the bits of external trigger signal delay for the radio 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 “:RADio:CUSTom:TRIGger:EXTernal:SOURce:DELay:STATe” for external delay state.

Setting format: [:SOURce]:RADio:CUSTom:TRIGger:EXTernal:SOURce:DELay <val>

Query format: [:SOURce]:RADio:CUSTom:TRIGger:EXTernal:SOURce:DELay?

Parameter description:

<Val> delay time when external is selected as the radio trigger source.

Range: 0[0, 1048575].

Example: [:SOURce]:RADio:CUSTom:TRIGger:EXTernal:SOURce:DELay 3000

The external trigger delay is 3000 bits.

Reset state: 0

Key path: **[Base]** —> [Trigger] —> [Trig source] —> [Ext] —> [Trigger Source - Ext]—> [Delay Time]

[:SOURce]:RADio:CUSTom:TRIGger:EXTernal:SOURce: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 bits set take effect. Please refer to “:RADio:CUSTom:TRIGger:EXTernal:SOURce:DELay” for external delay time.

Setting format: [:SOURce]:RADio:CUSTom:TRIGger:EXTernal:SOURce:DELay:STATe

ON|OFF|1|0

Query format: [:SOURce]:RADio:CUSTom:TRIGger:EXTernal:SOURce: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: [:SOURce]:RADio:CUSTom:TRIGger:EXTernal:SOURce:DELay:STATe 1 external trigger delay ON

Reset state: 0

Key path: **[Base]** **—>** [Trigger] **—>** [Trig Source] **—>** [Ext] —> [Trigger Source - Ext] —> [Delay ON/OFF]

[:SOURce]:RADio:CUSTom:TRIGger:EXTernal:SOURce: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]:RADio:CUSTom:TRIGger:EXTernal:SOURce:SLOPe POSitive|NGEative

Query format: [:SOURce]:RADio:CUSTom:TRIGger:EXTernal:SOURce:SLOPe?

Parameter description:

<Mode> discrete data. The values of external trigger polarity are as follows:

POSitive | 0: positive

NEGative | 1: negative

Example: [:SOURce]:RADio:CUSTom:TRIGger:EXTernal:SOURce:SLOPe NEGative the external trigger polarity is low effective.

Reset state: POS

Key path: [Base] —> [Trigger] —> [Trig Source] —> [Ext]

—> [Trigger Source - Ext] **—>** [Ext Trig Polar: POS/NEG]

[:SOURce]:RADio:CUSTom:TRIGger:SOURce <Mode>

Function description: This command is used to set the radio signal trigger source of the signal generator, including KEY, BUS and EXT. Please refer to Section "4.2.9 Configuration of Radio Trigger Function" in the user's manual for 1435 series microwave synthetic signal generator for details.

Setting format: [:SOURce]:RADio:CUSTom:TRIGger:SOURce KEY|BUS|EXT

Query format: [:SOURce]:RADio:CUSTom:TRIGger:SOURce?

Parameter description:

<Mode> Discrete data. The values of radio signal trigger source are as follows:

KEY | 0: the trigger source is the trigger key on the instrument front panel

BUS | 1: a trigger is performed by a group from GPIB or after receiving the "*TRG" command.

EXT | 2: the trigger source is the trigger input at the interface of the instrument rear panel.

Example: [:SOURce]:RADio:CUSTom:TRIGger:SOURce BUS

The radio signal trigger source is bus.

Reset state: KEY

Key path: **[Base]** —> [Trigger] —> [Trig Source]

[:SOURce]:RADio:CUSTom:TRIGger:TYPE <Mode>

Function description: This command is used to set the radio signal trigger mode for controlling data transmission, including continuous, single and gate. Please refer to Section "4.2.9 Configuration of Radio Trigger Function" in the user's manual for 1435 series microwave synthetic signal generator for details.

Setting format: [:SOURce]:RADio:CUSTom:TRIGger:TYPE CONTinuous|SINGle|GATE

Query format: [:SOURce]:RADio:CUSTom:TRIGger:TYPE?

Parameter description:

<Mode> Discrete data. The values of radio signal trigger mode are as follows:

CONTinuous | 0: the radio trigger mode is set to continuous

SINGle | 1: the radio trigger mode is set to single

GATE | 2: the radio trigger mode is set to gate

Example: [:SOURce]:RADio:CUSTom:TRIGger:TYPE SING

The radio trigger mode is single.

Reset state: CONT

Key path: **[Base]** —> [Trigger] —> [Trig Style]

[:SOURce]: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 ":RADio:CUSTom:TRIGger:TYPE" for radio signal trigger mode.

Setting format: [:SOURce]:RADio:CUSTom:TRIGger:TYPE:CONTinuous:TYPE FREE|TRIGger|RESet

Query format: [:SOURce]: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 | 0: when continuous automatic mode is selected, the radio signal output will last until the radio is turned off or the trigger mode is switched;

TRIGger | 1: when trigger mode is selected, the radio signal will be output only after a trigger signal is received;

RESet | 2: when real-time trigger mode is selected, the current radio signal output will stop, and the output will be loaded after recalculating the radio data.

Example: [:SOURce]:RADio:CUSTom:TRIGger:TYPE:CONTinuous:TYPE FREE

Set the continuous trigger to continuous automatic mode.

Reset state: FREE

Key path: **[Base]** **—**> [Trigger] **—**> [Trig Style] **—**> [Continuous>>]

[:SOURce]:RADio:CUSTom:TRIGger:TYPE:GATE:ACTive <Mode>

Function description: This command is used to set the gate trigger mode in radio trigger mode. Users may select low effective or high effective. When the external trigger signal is high or low, the radio signal output will be triggered. Please refer to ":RADio:CUSTom:TRIGger:TYPE" for radio signal trigger mode.

Setting format: [:SOURce]:RADio:CUSTom:TRIGger:TYPE:GATE:ACTive LOW|HIGH

Query format: [:SOURce]: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:

LOW | 0: low effective

HIGH | 1: high effective

Example: [:SOURce]:RADio:CUSTom:TRIGger:TYPE:GATE:ACTive HIGH

The trigger signal is output when the external trigger signal is high level.

Reset state: LOW

Key path: [Base] —> [Trigger] —> [Trig Style] —> [Gate>>]

[:SOURce]: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.

Setting format: [:SOURce]:RADio:MTONe:ARB:SETup <file_name>

Parameter description:

<FileName> Character string type, namely, the multiple-tone file name.

Example: [:SOURce]:RADio:MTONe:ARB:SETup "mtone1.mtn"

Load mtone1.mtn file into the memory of the signal generator.

Key path: **[Tone]** **—**> [Multi Tone] **—**> [Base Config] **—**> [File Load]

Description: For setting only.

[:SOURce]:RADio:MTONe:ARB:SETup:STORe <FileName>

Function description: This command is used to save the waveform data from current multitone list in the multitone file of the signal generator. It is only required to set the name of the multitone file instead of specifying the absolute path.

Setting format: [:SOURce]:RADio:MTONe:ARB:STORe <file_name>

Parameter description:

<FileName> Character string type, namely, the multiple-tone file name.

Example: [:SOURce]:RADio:MTONe:ARB:STORe "mtone1.mtn"

Save the multitone list in the multitone file mtone1.mtn of the signal generator.

Key path: **[Tone]** **—>** [Multi Tone] **—>** [Base Config] **—>** [Save]

Description: For setting only.

[:SOURce]: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 200M bandwidth of the baseband and the number of tones in the multiple-tone list, which is the same between tones and frequency spacing.

Setting format: [:SOURce]: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: 1MHz[100Hz, 200MHz].

<NumTones> number of tones.

Range: 2[2, 64].

<Pow> power attenuation.

Range: 0dB[-100dB, 0dB].

<Phase> initial phase.

Range: 0deg [0deg, 359deg].

<State> state. Boolean data; the values are taken as follows:

1: ON

0: OFF.

Example: [:SOURce]: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 attenuation of the first tone is 10dB, the phase is 90deg, and the state is OFF. The power attenuation of the second tone is 20dB, the phase is 0deg, and the state is ON. The power attenuation 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]:RADio:MTONe:ARB:SETup:TABLe:FSPacing <FreqSpacing>

Function description: This command is used to set the frequency spacing between tones. Such setting only takes effect after the multitone modulation is turned on. Please refer to “:RADio:MTONe:ARB:STATe” for multitone state. Please refer to the command “:RADio:MTONe:ARB:SETup:TABLe” for specifying other multitone modulation parameters or configuring multitone list.

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: 1MHz[100Hz, 200MHz].

Example: [:SOURce]:RADio:MTONe:ARB:SETup:TABLe:FSPacing 200kHz

Set the frequency spacing of multitone list to 200kHz.

Reset state: 1MHz

Key path: **[Tone]** —> [Multi Tone] —> [Base Config] —> [Freq Interval]

[:SOURce]:RADio:MTONe:ARB:SETup:TABLe:NTONes <NumTones>

Function description: This command is used to set the number of tones in multitone modulation list. Such setting only takes effect after the multitone modulation is turned on. Please refer to ":RADio:MTONe:ARB:STATe" for multitone state. Please refer to the command ":RADio:MTONe:ARB:SETup:TABLe" for specifying other multitone modulation parameters or configuring multitone list.

Setting format: [:SOURce]:RADio:MTONe:ARB:SETup:TABLe:NTONes <val>

Query format: [:SOURce]:RADio:MTONe:ARB:SETup:TABLe:NTONes?

Parameter description:

<NumTones> number of tones in multitone modulation list.

Range: 2[2, 64].

Example: [:SOURce]:RADio:MTONe:ARB:SETup:TABLe:NOTNes 3

Set the number of tones in multitone list to 3.

Reset state: 2

Key path: **[Tone]** —> [Multi Tone] —> [Base Config] —> [Tone Count]

[:SOURce]:RADio:MTONe:ARB:SETup:TABLe:PHASe:INITialize <Mode>

Function description: This command is used to initialize the initial phase mode in multitone modulation list, including random and fixed. In fixed mode, all tone phases in the multitone list will be set to a fixed value of 0; in random mode, all tone phases in the multitone list will be set to different random values based on random seeds. Please refer to ":RADio:MTONe:ARB:SETup:TABle:PHASe:INITialize:SEED" for relationship between tone phases in multitone modulation.

Setting format: [:SOURce]:RADio:MTONe:ARB:SETup:TABLe:PHASe:INITialize RANDom|FIXed

Query format: [:SOURce]: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: [:SOURce]:RADio:MTONe:ARB:SETup:TABLe:PHASe:INITialize FIX

Set the tone phase in multitone list to a fixed value.

Reset state: FIXed

Key path: **[Tone]** —> [Multi Tone] —> [Base Config] —> [Initial Phase]

[:SOURce]: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]:RADio:MTONe:ARB:SETup:TABLe:PHASe:INITialize:SEED RANDom|FIXed

Query format: [:SOURce]:RADio:MTONe:ARB:SETup: TABLe:PHASe:INITialize:SEED?

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: [:SOURce]:RADio:MTONe:ARB:SETup:TABLe:PHASe:INITialize:SEED FIX set the relationship between tone phases to fixed.

Reset state: FIX

Key path: **[Tone]** —> [Multi Tone] —> [Base Config] —> [Phase Rela]

[:SOURce]: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 “RADio:MTONe:ARB:SETup:TABLe”.

Setting format: [:SOURce]:RADio:MTONe:ARB:SETup:TABLe:ROW <row_index>,<pow>,<phase>,<state>

Query format: [:SOURce]:RADio:MTONe:ARB:SETup:TABLe:ROW? <row_Index>

Parameter description:

<RowIndex> row index in multitone list.

Range: 0[0, 63].

<Pow> power attenuation.

Range: 0dB[-100dB, 0dB].

<Phase> initial phase.

Range: 0deg [0deg, 359deg].

<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” this example shows that the power attenuation, phase and state in the second row of the multitone list are set to -10dB, 40deg and OFF respectively.

:RADio: MTONe:ARB:SETup:TABLe:ROW? 2 it indicates that the query index is 2, that is, the information in the third row of the multitone list includes frequency offset, power attenuation, phase and state.

Reset state: 0, 0dB, 0deg, 0

Key path: NONE

[:SOURce]:RADio:MTONe:ARB[:STATe] <State>

Function description: This command is used to set the multitone state of the signal generator. When the multitone is ON, the instructions on IQ modulation and multitone will be displayed in the main information display area on the user interface of the signal generator.

Setting format: [:SOURce]:RADio:MTONe:ARB:STATe ON|OFF|1|0

Query format: [:SOURce]: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: [:SOURce]:RADio:MTONe:ARB:STATe 1 multitone modulation ON.

Reset state: 0

Key path: **[Tone] —>** [Multi Tone] —> [Base Config] **—>** [Multi Tone ON/OFF]

[:SOURce]:RADio:TTONe:ARB:ALIGnment <Mode>

Function description: This command is used to set the alignment position of two tone signal, including left, center and right. Such setting only takes effect after the two tone modulation is turned on. Please refer to ":RADio:TTONe:ARB:STATe" for two tone modulation state.

Setting format: [:SOURce]:RADio:TTONe:ARB:ALIGnment LEFT|CENTer|RIGHt

Query format: [:SOURce]:RADio:TTONe:ARB:ALIGnment?

Parameter description:

<Mode> Discrete data. The values of alignment position of two tone signal are as follows:

LEFT	|	0: left
CENTer	|	1: center
RIGHt	|	2: right

Example: [:SOURce]:RADio:TTONe:ARB:ALIGnment RIGHt

Set the alignment of two tone signal to the right of the carrier for display.

Reset state: CENT

Key path: **[Tone]—>** [Dual Tone] **—>** [Two Tone Alignment>>]

[:SOURce]:RADio:TTONe:ARB:FSPacing <FreqSpacing>

Function description: This command is used to set the frequency spacing between tones. Such setting only takes effect after the two tone modulation is turned on. Please refer to ":RADio:TTONe:ARB:STATe" for two tone state.

Setting format: [:SOURce]:RADio:TTONe:ARB:FSPacing <val><freq unit>

Query format: [:SOURce]:RADio:TTONe:ARB:FSPacing?

Parameter description:

<FreqSpacing> frequency spacing between tones.

Range: 10MHz[0Hz, 40MHz].

Example: [:SOURce]:RADio:TTONe:ARB:FSPacing 30MHz

Set the two tone frequency spacing to 30MHz.

Reset state: 10MHz

Key path: **[Tone]—**> [Dual Tone] **—**> [Frequency Spacing]

[:SOURce]:RADio:TTONe:ARB[:STATe] <State>

Function description: This command is used to enable the two tone of the signal generator. When two tone modulation is enabled, the instructions on IQ modulation and two tone will be displayed in the main information display area of the signal generator.

Setting format: [:SOURce]:RADio:TTONe:ARB:STATe ON|OFF|1|0

Query format: [:SOURce]:RADio:TTONe:ARB:STATe?

Parameter description:

<State> Boolean data. The values of two tone modulation state are as follows:

ON | 1: Two tone modulation ON

OFF | 0: two tone modulation OFF.

Example: [:SOURce]:RADio:TTONe:ARB:STATe 1 two tone modulation ON

Reset state: 0

Key path: **[Tone]—**> [Dual Tone] **—**> [Dual tone ON/OFF]

[:SOURce]:RADio:ARB:MODE <Mode>

Function description: This command is used to set the arbitrary mode. Users may select arbitrary or sequence. In ARB mode, users may load and play any arbitrary data file in the custom format. In SEQuence mode, users may generate the waveform segment file as required and combine the waveform segment into a sequence for play. Please refer to the command ":RADio:ARB:SEQuence".

Setting format: [:SOURce]:RADio:ARB:MODE ARB|SEQuence

Query format: [:SOURce]:RADio:ARB:MODE?

Parameter description:

<Mode> Discrete data. The values of arbitrary mode are as follows:

ARB | 0: arbitrary mode

SEQuence | 1: sequence mode

Example: [:SOURce]:RADio:ARB:MODE SEQ the arbitrary operating mode is sequence.

Reset state: SEQ

Key path: [Arb] **—**> [Base Config] **—**> [Work Pattern: Arb/Seq]

[:SOURce]:RADio:ARB[:STATe] <State>

Function description: This command is used to enable the state of arbitrary waveform generator for the signal generator. When the arbitrary mode is enabled, the instructions will be displayed in the main information display area on the user interface of the signal generator.

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

Query format: [:SOURce]:RADio:ARB:STATe?

Parameter description:

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

ON | 1: Arbitrary ON

OFF | 0: Arbitrary OFF.

Example: [:SOURce]:RADio:ARB:STATe 1 arbitrary ON

Reset state: 0

Key path: [Arb] —> [Base Config] —> [Arb Seq ON/OFF]

[:SOURce]:RADio:ARB:SEQuence <FileName>,<WaveForm>,<Reps>,<Marks>

Function description: This command is used to load a waveform sequence that may consist of multiple waveform segments. The parameters include file_name, waveform, reps and M1M2M3M4, where file_name refers to the folder in which waveform segment files are saved. A folder specified by users may only be one under relative path. In this parameter, users are not entitled to specify an absolute path. For example, when a user names file_name as "D:\\USER\\SEQ", such folder cannot be created in disk D, and it will be deemed to be a bad file name. Waveform refers to the specific waveform segment file. The maximum number of waveform segment files supported by this command is 64. Reps refers to the number of loop playback of each waveform segment. The maximum number of loop playback of a waveform segment file is 65535. M1M2M3M4 refers to the marking switch of each waveform segment file. For example, if a user does not want the marks of a waveform segment to be output, select NONE; if he wants all marks of the waveform segment file to be output, select ALL.

Setting format: [:SOURce]:RADio:ARB

:SEQuence <file_name>,<waveform>,<reps>,NONE

|M1|M2|M3|M4|M1M2|M1M3|M1M4|M2M3|M2M4

|M3M4|M1M2M3|M1M2M4|M1M3M4|M2M3M4|ALL,{

<waveform2>,<reps>,NONE|M1|M2|M3|M4

|M1M2|M1M3|M1M4|M2M3|M2M4|M3M4|M1M2M3

|M1M2M4|M1M3M4|M2M3M4|ALL

Parameter description:

<FileName> string type

The folder in which waveform segment files are saved; a folder specified by users may only be one under relative path.

<WaveForm> **string** type

Name of waveform segment file; the maximum number of waveform segment files supported by this command is 64.

<Reps> integer; Number of loop playback of each waveform segment.

Range: 1[1, 65535].

<Marks> discrete data; marking switch of each waveform segment file. Refer to the command format for **specific** options.

Example: [:SOURce]:RADio:ARB:SEQuence Seq1, waveform1, 12, NONE, vaveform2, 300, M1M2

Load a waveform sequence in Seq1 folder that contains two waveform segment files waveform1 and waveform2, where the number of loop playback of waveform1 is 12, and no mark is output during the play; the number of loop playback of waveform2 is 300, and mark 1 and mark 2 of each symbol are output.

Key path: **[Arb]** —> [Base Config] —> [Add WAve Seg]

Description: For setting only.

[:SOURce]:RADio:ARB:SEQuence:CLOCk <Mode>

Function description: This command is used to set the sampling clock type of the signal generator in arbitrary mode. In arbitrary operating mode, users are only allowed to use CUSTom mode and cannot set other modes; in sequence mode, users may select CURRent, HIGH or CUSTom. Please refer to ":RADio:ARB:MODE" for the arbitrary operating mode.

Setting format: [:SOURce]:RADio:ARB:SEQuence:CLOCk CURRent|HIGH|CUSTom

Query format: [:SOURce]:RADio:ARB:SEQuence:CLOCk?

Parameter description:

<Mode> Discrete data. The values of sampling clock type in arbitrary mode are as follows:

CURRent | 0: play waveform segment files in sequence mode and based on the sampling rate of each waveform segment;

HIGH | 1: play waveform segment files in sequence mode and based on the maximum sampling rate of waveform segments;

CUSTom | 2: play waveform segment files in sequence mode and based on the current clock frequency set for the signal generator.

Example: [:SOURce]:RADio:ARB:SEQuence:CLOCk HIGH

Set the sampling clock type to HIGH.

Reset state: CURR

Key path: **[Arb]** —> [Base Config] —> [Clock Type]

[:SOURce]:RADio:ARB:SCLock:RATE <ClockRate>

Function description: This command is used to set the arbitrary signal sampling rate. The value set is valid only when the clock type is custom. Please refer to the command ":RADio:ARB:SEQuence:CLOCk" for the sampling clock type.

Setting format: [:SOURce]:RADio:ARB:SCLock:RATE <val><freq unit>

Query format: [:SOURce]:RADio:ARB:SCLock:RATE?

Parameter description:

<ClockRate> signal Q offset in I/Q.

Range: 100MHz[0.01MHz, 250MHz].

Example: [:SOURce]:RADio:ARB:SCLock:RATE 50MHz the arbitrary clock frequency is 50MHz.

Reset state: 100MHz

Key path: **[Arb]** —> [Base Config] —> [Clock Freq]

[:SOURce]:RADio:ARB:TRIGger:TYPE <Mode>

Function description: This command is used to set the trigger mode for controlling waveform play. Users may select CONTinuous, SINGle, GATE or SADVance. In single mode, it is required to wait for the completion of playing the current code sequence before the system can receive effective trigger events. In addition, CONTinuous, SINGle, GATE and SADVance all have multiple states. Please refer to ":RADio:ARB:TRIGger:TYPE:CONTinuous"

":RADio:ARB:TRIGger:TYPE:SINGle"

":RADio:ARB:TRIGger:TYPE:SADVance:TYPE" and ":RADio:ARB:TRIGger:TYPE:GATE:ACTive".

Setting format: [:SOURce]:RADio:ARB:TRIGger:TYPE CONTinuous

|SINGle|SADVance|GATE

Query format: [:SOURce]: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 | 0: Select continuous as trigger mode, and the instrument will repeat the waveform sequence after receiving an effective trigger event;

SINGle | 1: select single as trigger mode, and the instrument will play the waveform sequence once after receiving an effective trigger event;

SADVance | 2: select waveform segment as trigger mode, and the instrument will play a waveform segment after receiving an effective trigger event;

GATE | 3: Select gate as trigger mode, and the waveform sequence will be played continuously within the valid period of the gate signal.

Example: [:SOURce]:RADio:ARB:TRIGger:TYPE SING the trigger mode is single.

Reset state: CONT

Key path: **[Arb]** —> [Trigger] —> [Trig Mode]

[:SOURce]: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 continuous/single mode. Users may select FREE, TRIGger or RESet. Please refer to ":RADio:ARB:TRIGger:TYPE" for trigger mode.

Setting format: [:SOURce]:RADio:ARB:TRIGger:TYPE:CONTinuous

FREE|TRIGger|RESet

Query format: [:SOURce]: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 are as follows:

FREE | 0: Select the auto mode to automatically trigger the sequence for playing after the sequence waveform data is downloaded, and ignore all trigger events during playback.

TRIGger | 1: select trigger mode, and the system will not play current waveform sequence before receiving an effective trigger event. The system will play current waveform sequence after receiving an effective trigger event. After the sequence is played, continue to wait for an effective trigger event before replaying current waveform sequence.

RESet | 2: select real-time mode, and the system will not generate the modulation source data before receiving an effective trigger event, so no code signal is generated. When an effective trigger event is received, the system begins to generate the selected modulation source data, and then generates corresponding code data and signal. After the current source data is generated, the system will automatically restart to generate the currently set modulation source data. During the generation of modulation source data, if an effective trigger event is received, the system will immediately suspend the currently generated modulation source data and regenerate the set modulation source data from the beginning.

Example: [:SOURce]:RADio:ARB:TRIGger:TYPE:CONTinuous TRIG

Set the type for sequence file to respond to trigger signal to trigger in arbitrary continuous mode.

Reset state: FREE

Key path: **[Arb]** —> [Trigger] —> [Trig Mode] —> [Continuous]

[:SOURce]: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 FREE, TRIGger or RESet. Please refer to ":RADio:ARB:TRIGger:TYPE" for trigger mode.

Setting format: [:SOURce]:RADio:ARB:TRIGger:TYPE:SINGle

FREE|TRIGger|RESet

Query format: [:SOURce]: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:

FREE | 0: Select the auto mode to automatically trigger the sequence for playing after the sequence waveform data is downloaded, and ignore all trigger events during playback.

TRIGger | 1: select trigger mode, and the system will not play current waveform sequence before receiving an effective trigger event. The system will play current waveform sequence after receiving an effective trigger event. After the sequence is played, continue to wait for an effective trigger event before replaying current waveform sequence.

RESet | 2: select real-time mode, and the system will not generate the modulation source data before receiving an effective trigger event, so no code signal is generated. When an effective trigger event is received, the system begins to generate the selected modulation source data, and then generates corresponding code data and signal. After the current source data is generated, the system will automatically restart to generate the currently set modulation source data. During the generation of modulation source data, if an effective trigger event is received, the system will immediately suspend the currently generated modulation source data and regenerate the set modulation source data from the beginning.

Example: [:SOURce]:RADio:ARB:TRIGger:TYPE:SINGle TRIG

Set the type for sequence file to respond to trigger signal to trigger in arbitrary single mode.

Reset state: FREE

Key path: [Arb] —> [Trigger] —> [Trig Style] —> [Single]

[:SOURce]:RADio:ARB:TRIGger:TYPE:SADVance[:TYPE] <Mode>

Function description: This command is used to set the type for sequence file to respond to trigger signal in arbitrary waveform segment mode. Users may select SINGle or CONTinuous. The trigger function of waveform segment play is not for the entire waveform sequence, but for a single waveform segment in the sequence. Please refer to ":RADio:ARB:TRIGger:TYPE" for trigger mode.

Setting format: [:SOURce]:RADio:ARB:TRIGger:TYPE:SADVance:TYPE

SINGle|CONTinuous

Query format: [:SOURce]:RADio:ARB:TRIGger:TYPE:SADVance:TYPE?

Parameter description:

<Mode> Discrete data. The values of the type for sequence file to respond to trigger signal in arbitrary waveform segment mode are as follows:

SINGle | 0: single waveform segment trigger

CONTinuous | 1: continuous waveform segment trigger

Example: [:SOURce]:RADio:ARB:TRIGger:TYPE:SADVance:TYPE SING

Set the type for sequence file to respond to trigger signal to single in arbitrary waveform segment mode.

Reset state: SING

Key path: **[Arb] —>** [Trigger] **—>** [Trig Mode] **—>** [Wave Segment >>]

[:SOURce]:RADio:ARB:TRIGger:TYPE:GATE:ACTive <Mode>

Function description: This command is used to set the type for sequence file to respond to trigger signal in arbitrary gate mode. Users may select low effective or high effective. Please refer to ":RADio:ARB:TRIGger:TYPE" for trigger mode.

Setting format: [:SOURce]:RADio:ARB:TRIGger:TYPE:GATE:ACTive

LOW|HIGH

Query format: [:SOURce]: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 gate mode are as follows:

LOW | 0: low effective

HIGH | 1: high effective

Example: [:SOURce]:RADio:ARB:TRIGger:TYPE:GATE:ACTive LOW

Set the type for sequence file to respond to trigger signal to low effective in arbitrary gate mode.

Reset state: LOW

Key path: **[Arb]** —> [Trigger] —> [Trig Mode>>] —> [Gate>>]

[:SOURce]:RADio:ARB:TRIGger:SOURce <Mode>

Function description: This command is used to set the arbitrary trigger source.

Setting format: [:SOURce]:RADio:ARB:TRIGger:SOURce KEY|BUS|EXT|INT

Query format: [:SOURce]:RADio:ARB:TRIGger:SOURc e?

Parameter description:

<Mode> Discrete data. Arbitrary trigger source,

with values taken as follows:

KEY: trigger key

BUS: bus

EXT: external

INT: internal.

Example: [:SOURce]:RADio:ARB:TRIGger:SOURce BUS

To set the trigger source to bus.

Reset state: KEY

Key path: **[Arb]** —> [Trigger] —> [Trig Source]

[:SOURce]:RADio:ARB:VCO:CLOCk <Mode>

Function description: This command is used to set arbitrary sampling clock. The clock frequency is 200MHz when the sampling clock is internal. It cannot be changed. The clock frequency may be set based on the external clock frequency command when the sampling clock is external. The external clock frequency command is "RADio:ARB:EXTernal:CLOCk:RATE".

Setting format: [:SOURce]:RADio:ARB:VCO:CLCOk INTernal|EXTernal

Query format: [:SOURce]:RADio:ARB:VCO:CLOCk?

Parameter description:

<Mode> Discrete data. The values of arbitrary sampling clock are as follows:

INTernal : internal

EXTernal : external

Example: [:SOURce]:RADio:ARB:VCO:CLOCk EXTernal

To set the trigger source to bus.

Reset state: LOW

Key path: **[Arb]** —> [Trigger] —> [Samp Clock]

[:SOURce]:RADio:ARB:EXternal:CLOCk:RATE <ClockRate>

Function description: This command is used to set the external clock frequency and is valid when the sampling clock is external. Please refer to "RADio:ARB:VCO:CLOCk" for setting of sampling clock.

Setting format: [:SOURce]:RADio:ARB:EXTernal:CLCOk:RATE <val><freq unit>

Query format: [:SOURce]:RADio:ARB:EXTernal:CLOCk:RATE?

Parameter description:

<Mode> the values of arbitrary sampling clock are as follows:

Range [100Hz, 250MHz]

Example: [:SOURce]:RADio:ARB:EXTernal:CLOCk:RATE 100MHz

To set the trigger source to bus.

Reset state: 200MHz

Key path: **[Arb]** —> [Trigger] —> [External Clock Frequency]

3.3.12 MEMory Subsystem

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

:MEMory:COPY:NAME <SrcName>,<DestName>

Function description: This command is used to copy the data in one file to another file. If the source file and the destination file are not in the same folder, the file name may be the same. When copying any arbitrary file, the tag file related to the file will be copied together. It should be noted that the absolute path must be attached to the file name.

Setting format: MEMory:COPY:NAME <src_name>,<dest_name>

Parameter description:

<SrcName> string type, name of source file

<DestName> string type, name of destination file.

Example: :MEMory:COPY:NAME "c:\\data\\user\\seq1.dat","c:\\data\\user\\seq2.dat"

Copy the data in seq1.dat to seq2.dat.

Key path: **[File] —>** [Copy]

Description: For setting only.

:MEMory:DELete:NAME <FileName>

Function description: This command is used to delete user file in the signal generator.

Setting format: MEMory:DLEete:NAME <file_name>

Parameter description:

<FileName> Character string type. User file saved in the signal generator.

Example: MEMory:DELete:NAME “c:\\arb1.seq”

Delete file arb1.seq in disk C.

Key path: **NONE**

Description: For setting only.

:MEMory:MOVE <FileName>

Function description: This command is used to rename the file in the signal generator. It should be noted that the absolute path must be attached to the file name.

Setting format: MEMory:MOVE <Sourfile_name>, <Desfile_name>

Parameter description:

<FileName> Character string type. File name saved in the signal generator.

Example: MEMory:MOVE “c:\data\arb1.seq”,”c:\data\arb2.seq”

Rename file arb1.seq to arb2.seq

Description: For setting only.

:MEMory:DATA <FileName>,<#AB\n><DataBlock>

Function description: This command is used to download arbitrary data into the instrument in the way of data block through the communication interface of the signal generator, and save it in the instrument with the name of <file_name>. The command can only be used to transmit the data with the number of bytes below 1000000000, that is, the total number of bits of transmitted data is less than 10, and the data is binary data.

Setting format: MEMory:DATA <file_name>, <data_block>

Parameter description:

<FileName> Character string type. Name of arbitrary file saved in the signal generator specified by users. Users are not **entitled** to specify the absolute path.

<#AB\n> # and \n are fixed formats, where # indicates the beginning of the data, \n is the placeholder, and A and B represents the length of the data. For example, in #210\n, 3 indicates that the bytes account for 2 bits, 10 represents the total number of bytes, and <DataBlock> following 10 represents 10 bytes of data.

<DataBlock> data block to be transmitted.

Example: [:SOURce]:MEMory:DATA arb1.dat, #41024\n jklasdj…

It indicates that 1024 bytes of data are sent to the signal generator and saved in the generator with the file name of arb1.dat.

Description: For setting only.

3.3.13 ROSCillator Subsystem

The oscillator subsystem command is used to realize functions of the signal generator related to time base.

[:SOURce]:ROSCillator: REFerence <Val>

Function description: This command is used to adjust internal reference of the signal generator by setting internal calibration parameter, so as to make the frequency output more accurate. It should be noted that within 2h after starting the signal generator, the instrument should be preheated. Please do not change the reference value easily. For more detailed information, please refer to the user's manual for 1435 series signal generator.

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

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

Parameter description:

<Val> internal calibration parameter.

Range: [0, 65535].

Example: ROSCillator: REFerence 30000

Adjust the internal reference accuracy value to 30000.

3.3.14 SYSTem Subsystem

The system subsystem command is used to realize functions of the signal generator related to its performance.

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

:DIAGnostic:INFormation:CCOunt:PON

Function description: Query the accumulative startup times of the instrument

Query format: DIAGnostic:INFormation:CCOunt:PON?

Example: DIAGnostic:INFormation:CCOunt:PON? This example shows that the cumulative number of times the signal generator has been started is queried.

Description: For query only.

:DIAGnostic:INFormation:OTIMe

Function description: Query the instrument firmware date and time stamp

Query format: DIAGnostic:INFormation:OTIMe?

Example: DIAGnostic:INFormation:OTIMe? This example shows that the cumulative number of hours the signal generator has been started is queried.

Description: For query only.

:DIAGnostic:SNUM?

Function description: This command is used to read the system serial number of the signal generator.

Query format: DIAGnostic:SNUM?

Returned value: Serial number

Example: DIAGnostic:SNUM this example shows that the system serial number of the signal generator is queried.

Description: For query only.

:SYSTem:COMMunicate:GPIB:ADDRess <Address>

Function description: This command is used to set GPIB address of the signal generator, which is 19 by default. To ensure normal communication, the local GPIB address should be different from that of other devices in the same test system.

Setting format: SYSTem:COMMunicate:GPIB:ADDRess <val>

Query format: SYSTem:COMMunicate:GPIB:ADDRess?

Parameter description:

<Address> integer data, GPIB address.

Range: 19 [0, 30].

Example: SYSTem:COMMunicate:GPIB:ADDRess 19

Set GPIB address of the signal generator to 19.

Reset state: 19

Key path: **[System] —**> [GPIB Port] **—**> [Local GPIB Addr]

:SYSTem:COMMunicate:GTLocal

Function description: This command is used to switch the signal generator to local operation mode. In this mode, users can operate the buttons on the front panel of the instrument, and the indication of remote control operation mode on the instrument operation interface will disappear.

Setting format: SYSTem:COMMunicate:GTLocal

Example: SYSTem:COMMunicate:LAN:IP 172.141.114.114

:SYSTem:COMMunicate:GTLocal

Switch the signal generator from remote control state to local state.

Description: For setting only.

:SYSTem:DEVice:LANGuage <Mode>

Function description: This command is used to set the language displayed on the interface of the signal generator. At present, the instrument supports Chinese and English interface. The default interface is Chinese. After the language switch is completed, the interface can be switched in real time, but when the command is used for query, the returned value is the state value before the switch. The instrument should be restarted to make the returned value displayed correctly.

Setting format: SYSTem:DEVice:LANGuage CHINese|ENGLish

Query format: SYSTem:DEVice:LANGuage?

Parameter description

<Mode> discrete data. The values of interface language are as follows:

CHINeses	Chinese
ENGLish	English

Example: SYSTem:DEVice:LANGuage ENGLish set the interface of the instrument to English.

Reset state: Keep the current language.

Key path: [System] —> [Base Config] —> [Language/LANG]

:SYSTem:COMMunicate:LAN:IP <Address>

Function description: This command is used to set IP address of the signal generator. The parameters are expressed in dotted decimal notation.

Setting format: SYSTem:COMMunicate:LAN:IP <ipstring>

Query format: SYSTem:COMMunicate:LAN:IP?

Parameter Description

<Address> string type, network IP address expressed in dotted decimal notation

Example: SYSTem:COMMunicate:LAN:IP “172.141.114.114”

Set IP address of the signal generator to 172.141.114.114.

Key path: [System] —> [LAN Port] —> [Local Machine IP Addr]

:SYSTem:COMMunicate:LAN:SUBNet <Address>

Function description: This command is used to set subnet mask address of the signal generator. The parameters are expressed in dotted decimal notation.

Setting format: SYSTem:COMMunicate:LAN:SUBNet <ipstring>

Query format: SYSTem:COMMunicate:LAN: SUBNet?

Parameter Description

<Address> string type, network IP address expressed in dotted decimal notation

Example: SYSTem:COMMunicate:LAN: SUBNet “255.255.255.0”

Set subnet mask address of the signal generator to 255.255.255.0.

Key path: [System] —> [LAN Port] —> [Net Mask]

:SYSTem:COMMunicate:LAN:GATeway <Address>

Function description: This command is used to set network gateway address of the signal generator in external network access LAN. The parameters are expressed in dotted decimal notation.

Setting format: SYSTem:COMMunicate:LAN:GATeway <ipstring>

Query format: SYSTem:COMMunicate:LAN:GATeway?

Parameter Description

<Address> string type, network IP address expressed in dotted decimal notation

Example: SYSTem:COMMunicate:LAN: GATeway “172.141.114.254”

Set network gateway address of the signal generator to 172.141.114.254.

Key path: [System] —> [LAN Port] —> [Default Gate]

:SYSTem:ERRor[:NEXT]?

Function description: This command is used to query the errors in error queue of the signal generator. When an error is queried, it will be deleted from the error queue. If there is no error in the error queue, users will find the message: "+0, No ERROR".

Query format: SYSTem:ERRor[:NEXT]?

Returned value: <ErrorInfo>: "error code, error".

Example: SYSTem:ERRor[:NEXT]? this example shows that the errors of the signal generator are queried.

Description: For query only.

:SYSTem:PRESet:TYPE <Mode>

Function description: This command is used to set the reset state of the signal generator, including manufacturer, user and last state. In manufacturer mode, the instrument will return to the default state of the manufacture after reset. The user mode is the reset state defined by the user. After the instrument is reset, it will return to the instrument state specified by the user. The last state is the state before the instrument is reset.

Setting format: SYSTem:PRESet:TYPE NORMal|USER|LAST

Query format: SYSTem:PRESet:TYPE?

Parameter description:

<Mode> discrete data. The values of reset state are as follows:

NORMal	|	0: manufacturer
USER	|	1: user
LAST	|	2: last state

Example: SYSYem:PRESet:TYPE USER set the reset state of the signal generator to user.

Reset state: NORMal

Key path: **[System]** —> [Reset] —> [Reset type]

4. Programming Examples

4.1 Basic Operation Examples

The following examples show the basic methods for using the VISA library to realize remote control programming of the instrument, with C++ language as an example.

4.1.1 VISA Library

VISA is the generic 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 achieve remote control, it is first required to install the VISA library. VISA library packages the underlying transmission functions of VXI, GPIB, LAN and USB interfaces to make it convenient for users to call directly. 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 remote control interface, VISA library, programming language and signal generator.

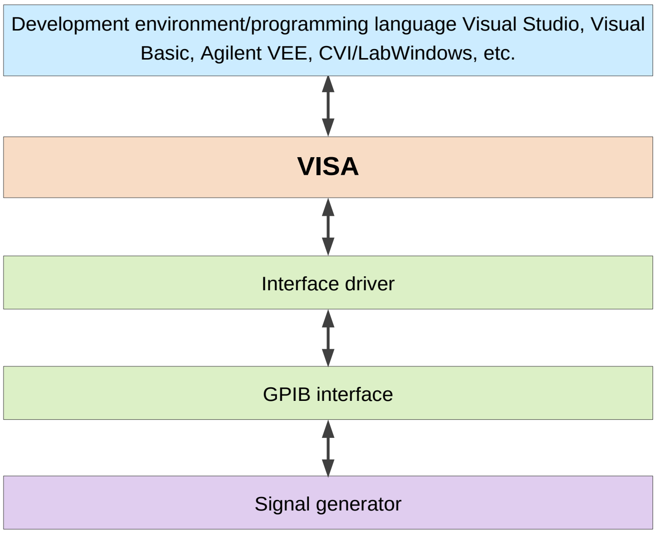

Development environment/programming language Visual Studio, Visual Basic, Agilent VEE, CVI/LabWindows, etc.
VISA
Interface driver
GPIB interface
Signal generator

Figure 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;
- Visual Studio 2010/2012 integrated development environment;
- PCI-GPIB interface card of NI or GPIB interface card of Agilent;
- VISA library of NI or Agilent;
- GPIB card;
- Network card;
- Available serial ports COM1 and COM2.

4.1.2.2 Files Included

To run an example program written in C/C++, you must include the required files in a project in VC6.0

If you use the VISA library, you must:

- Add visa32.lib to the source file;
- Add visa.h to the header file.

If you use the NI-488.2 library, you must:

- Add GPIB-32.OBJ file to the source file;
- Add windows.h file to the header file;
- Add Deci-32.h file to the header file.

For more information about the NI-488.2 library and VISA library, please refer to the websites of NI and Agilent respectively.

4.1.3 Initialize and Set Default State

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 source;

ViSession defaultRM;

Const char sourceString [VI_FIND_BUFLEN] = "GPIB0::20::INSTR";

Const sourceTimeout = 10000;
```

The constant sourceString represents the instrument descriptor, "GPIB0" represents the controller, and "20" represents the instrument connected to the controller. If it is assumed that the instrument is connected to the LAN and the IP address is "192.168.1.1", the value of the variable should be:

```
Const char sourceString [VI_FIND_BUFLEN] = "TCPIP::192.168.1.1::INSTR";
```

4.1.3.2 Initialize the Controller

```
    /***************************************************************************/
```

The following examples show how to open and establish a communication connection between the VISA library and the instrument (as specified by the instrument descriptor).

```
//Initialize the controller: open the default explorer and return the instrument handle source
/**********************************************************************/
void InitController()
{
ViStatus status;
status = viOpenDefaultRM(&defaultRM);
status = viOpen(defaultRM, sourceString, VI_NULL, VI_NULL, &source);
}
```

4.1.3.3 Initialize the Instrument

/**/

The following examples show how to initialize the default state of the instrument and empty the status register.

/**/

```
void InitDevice()
{
ViStatus status;
    long retCnt;
    status = viWrite(source, "*CLS", 4, &retCnt); //reset the status register
    status = viWrite(source, "*RST", 4, &retCnt); //reset the instrument
    status = viWrite(source, "freq 1ghz", 9, &retCnt); //set the continuous wave frequency of the signal generator to 1GHz
}
```

4.1.4 Send Setting Command

/**/

The following examples show how to set the point frequency and amplitude of 1435 series signal generator.

/**/

```
void SimpleSettings()
{
    ViStatus status;
        long retCnt;
        //Set the point frequency to 128MHz
        status = viWrite(source, "FREQENCY:CW 128MHz", 18, &retCnt);
        //Set the amplitude to -10dBm
        status = viWrite(source, "POW -10dBm", 10, &retCnt);
}
```

4.1.5 Read the State of Measuring Instrument

```
/*************************************************************************/
The following examples show how to read the set state of the instrument.
/*************************************************************************/
void ReadSettings()
{
ViStatus status;
    long retCnt;
    char rd_Buf_CW[VI_READ_BUFLEN];    // #define VI_READ_BUFLEN 20
    char rd_Buf_LVL[VI_READ_BUFLEN];

    //Query the point frequency
    status = viWrite(source, "FREQ:CW?", 8, &retCnt);
    Sleep(10);
    status = viRead(source, rd_Buf_CW, 20, &retCnt);
    //Query the amplitude
    status = viWrite(source, "POW?", 4, &retCnt);
    Sleep(10);
    status = viRead(source, rd_Buf_LVL, 20, &retCnt);
    //Print the debugging information
sprint("Cw is %s", rd_Buf_CW);
sprint("LEVel is %s", rd_Buf_ LVL);
}
```

4.1.6 Command Synchronization

```
/*************************************************************************/
The following examples illustrate the methods for command synchronization with sweep process.
/*************************************************************************/
void SweepSync()
{
    ViStatus status;
        long retCnt;
        ViEventType etype;
        ViEvent eevent;
        int stat;
        char OpcOk [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
        */
        /**********************************************************************/
        status = viWrite(source, "INIT:CONT OFF", 13, &retCnt);
        //Method 1 for waiting for the sweep to end: use *WAI
        status = viWrite(source, "ABOR;INIT:IMM;*WAI", 18, &retCnt);

        //Method 2 for waiting for the sweep to end: use *OPC?
        status = viWrite(source, "ABOR;INIT:IMM; *OPC?", 20, &retCnt);
        status = viRead(source, OpcOk, 2, &retCnt); //wait for *OPC to return “1”

        //Method 3 for waiting for the sweep to end: use *OPC
        //To use the GPIB service request, set "Disable Auto Serial Poll" to "yes"
        status = viWrite(source, "*SRE 32", 7, &retCnt);
        status = viWrite(source, "*ESE 1", 6, &retCnt); //enable service request ESR
        //Set the event enable bit and the operation is completed
        status = viEnableEvent(source, VI_EVENT_SERVICE_REQ, VI_QUEUE, VI_NULL);
        //Enable SRQ event
        status = viWrite(source, "ABOR;INIT:IMM;*OPC", 18, &retCnt);
        //Start sweep synchronously with OPC
        status = viWaitOnEvent(source, VI_EVENT_SERVICE_REQ, 10000, &etype, &eevent)
        //Wait for service request
        status = viReadSTB(source, &stat);
        status = viClose(eevent); //close the event handle
        //Disable SRQ event
        status = viDisableEvent(source, VI_EVENT_SERVICE_REQ, VI_QUEUE);
    //Main program continues…
}
```

4.2 Advanced Operation Examples

4.2.1 Set Point Frequency at LAN Interface and Query

/**/

To use the following examples correctly, you must match your host address to the IP address of the signal source. (Network design examples in this manual are implemented in VC6.0 by using WINSOCK components to establish socket.)

/**/

4.2 Advanced Operation Examples

```
#include "stdafx.h"
#include <afxsock.h>
#include <stdio.h>
#include <stdlib.h>
#ifdef _UNICODE
#define SG_IP_ADDR        L"192.168.1.199"     //IP address of the signal generator
#else
#define SG_IP_ADDR        "192.168.1.199"      //IP address of the signal generator
#endif
#define SG_SOCKET_PORT 5001                        //port number of the signal generator

void ShowMsg(PCHAR lpszText)
{
    #ifdef _UNICODE
    AfxMessageBox((CString)lpszText);
    #else
    AfxMessageBox(lpszText);
    #endif
}
void SocketTest(void)
{
    CSocket client;
    int iFlag;
    char rgcBuf[256];
    int iBufLen;
    if (!AfxSocketInit())     //initialize the network port
    {
        ShowMsg("Initialization failed");
    }
    else
    {
        iFlag = client.Create();
        if (!iFlag)
        {
            ShowMsg("Socket creation failed");
        }
        else
        {
```

```
        ShowMsg("Socket creation successful");
        iFlag = client.Connect(SG_IP_ADDR, SG_SOCKET_PORT); //Connect the network port
        if (!iFlag)
         {
            ShowMsg("Connection failed");
         }
        //Set the point frequency to 1GHz
        sprintf(rgcBuf, "%s\n", "FREQ 1GHz");
        iBufLen = (int)strlen(rgcBuf);
        iFlag = client.Send(rgcBuf, iBufLen);
        if (!iFlag)
        {
               ShowMsg("Send failed");
               }
               else
               {
                  //Query the current frequency
                  sprintf(rgcBuf, "%s\n", "FREQ?");
                  iBufLen = (int)strlen(rgcBuf);
                  iFlag = client.Send(rgcBuf, iBufLen);
                  if (!iFlag)
                 {
                    ShowMsg("Send failed");
                 }
                 else
               {
                   iFlag= client.Receive(rgcBuf, sizeof(rgcBuf), 0); //read from the network
                   if (!iFlag)
                   {
                   ShowMsg("Receive failed");
                   }
               }
       }
     }
   client.Close();
}
}
```

4.2.2 Set Point Frequency at GPIB Interface and Query

```
/**************************************************************************/
In this example, the functions of the VISA library are used to set the signal source to output point
frequency of 500MHz and power of -2dbm, query the current frequency and power, start VC6.0,
add necessary files, and input the following codes into your .cpp file
/*************************************************************************/
#include "stdafx.h"
#include "visa.h"
#include <iostream>
#include <stdlib.h>
#include <conio.h>
void main()
{
        ViSession defaultRM,vi;                   //declare a variable of type VISession
        ViStatus vistatus = 0;                //for device communication
        char buff[256];                       //declare a variable with character data stored
        vistatus = viOpenDefaultRM(&defaultRM);   //Open the GPIB task, address: 19
        vistatus=viOpen(defaultRM,"GPIB0::19::INSTR",VI_NULL,VI_NULL,&vi);
        if(vistatus)
        {
            printf("The task cannot be opened. Please recheck the device and connect \n");
            exit(0);
        }
        viPrintf(vi,"*RST\n");                //reset the signal source
        viPrintf(vi,"FREQ 500MHZ\n");         //set the point frequency to 500MHz
        viPrintf(vi,"FREQ?\n");                  //query the point frequency
        viScanf(vi,"%t",buff);                 //put the query results into an array
        printf("source CW freq is: %s\n",buff);    //display the point frequency
        viPrintf(vi,"POW   -2dBm\n");          //set the power level to -2dBm
        viPrintf(vi,"POW?\n");                //query the current power
        viScanf(vi,"%t",buff);                 //put the query results into an array
        printf("source POW is: %s\n",buff);    //display the power
        viClear(vi);
        viClose(vi);
        viClose(defaultRM);
}
```

5. Error Description

This chapter will show you how to find problems and accept after-sales service, and explain error message of the signal generator.

5.1 Errors

The signal generator records the errors during the measurement in two ways: error queue displayed on front panel operation interface and error queue in SCPI (remote control mode), which are stored and managed respectively.

5.1.1 Error Description

5.1.1.1 Local Errors

1) Error view

View via interface:

In case of any error prompt at the lower right of the signal source during use, it indicates that there is something wrong with the software or hardware of the signal source. You can basically judge the error type as per the error code, and take corresponding measures for troubleshooting.

The error display area of the signal source can only display one error prompt at a time. Since multiple errors can occur to the instrument, to see all errors, do the following:

Step 1. Click [system] and then [Machine error], and an error list window will pop up.

Step 2. The prompt will be displayed in the window.

Step 3. Use the mouse to browse the error and close the dialog window.

Step 4. Select "Clear error list" to clear historical errors.

2) Error description

If an error is detected during the measurement of the signal generator, an alarm or error will be displayed on the right side of the status indicating area (error abbreviation + detailed error description).

Table 5.2 List of local error description

Key error field	Detailed error description
Unleveled	For overpower or no power
Reference loop unlocked	The reference loop signal inside the signal generator is out of lock.
Decimal loop unlocked	The decimal loop signal inside the signal generator is out of lock.
Local oscillator loop unlocked	The local oscillator loop signal inside the signal generator is out of lock.
VCO loop unlocked	The VCO loop signal inside the signal generator is out of lock.

5.1.2.2 Program Errors

1) Error format and description

In remote control mode, errors are recorded in the error/event queue of the status reporting system, and can be queried with the command "SYSTem:ERRor?". The format is as follows:

"<Error code>, "<Error in error queue>; <"Detailed error description>"

Example:

"-110," a data breaking bounds; the input parameter is beyond the lower bound.

A negative error code defined by the SCPI standard. This type of error is not specified here.

2) Error type

Error event corresponds to only one type of error. The types of errors are classified below (8257):

- **Query error (-499 to -400):** indicating that the output queue of the instrument controls and detects a message exchange protocol error described in Chapter 6 of IEEE 488.2. At this point, the query error bit (bit2) of the event status register is set (please refer to IEEE 488.2, 6.5 for details). The data cannot be successfully read from the output queue at this time.
- **Instrument characteristic error (-399 to -300, 201 to 703, and 800 to 810):** indicating that the instrument operation is not successful, and the reason may be abnormal hardware or firmware state. Such error codes are often used self-detection of the instrument. At this point, the instrument characteristic error bit (bit3) of the event status register is set.
- **Execution error (-299 to -200):** indicating that an error is detected during the measurement of the instrument. At this point, the execution error bit (bit4) of the event status register is set.
- **Command error (-199 to -100):** indicating a syntax error detected during command parsing of the instrument, usually due to an incorrect command format. At this point, the command error bit (bit5) of the event status register is set.

5.2 Method to Obtain After-sales Services

5.2.1 Contact Us

If there is a problem with 1435 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: **800-868-7041**

Technical support: **0532-86889847 86897262**

Fax: **0532-86889056 86897258**

Website: **www.ceyear.com**

Email: **eiqd@ceyear.com**

Postal code: **266555**

Address: **No. 98, Xiangjiang Road, Qingdao Economic & Technological Development Zone, Shandong, 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 it.
2) Pack it with the original packing materials, so as to minimize possible damage.
3) Place cushions at the four corners of the outer packing carton, and place the instrument in the outer packing carton.
4) Seal the opening of the packing carton with adhesive tape and reinforce the packing carton with nylon tape.
5) Specify text like “Fragile! Do not touch! Handel with care!” and so on.
6) Please check by precision instrument.
7) Keep a copy of all shipping documents.

Note

Precautions on packing the signal generator

Using other materials 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.

Tip

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).

Annexes

Annex A Zoom Table of SCPI Classified by Subsystem

Table 1 Zoom table of 1435 SCPI commands

Index	Command	Function
1	*IDN?	Universal command
2	*RCL	Universal command
3	*RST	Universal command
4	*SAV	Universal command
5	*CLS	Universal command
6	*ESE	Universal command
7	*ESR?	Universal command
8	*STB?	Universal command
9	*TRG	Universal command
10	*TST?	Universal command
11	:OUTPut[:STATe](?)	Set RF output to ON/OFF state
12	:OUTPut:MODulation[:STATe](?)	Set the modulation to ON/OFF state
13	[:SOURce]:FREQuency[:CW|FIXed](?)	Set the output frequency of the signal generator
14	[:SOURce]:FREQuency:MODE(?)	Set the frequency generation mode
15	[:SOURce]:FREQuency:MULTiplier(?)	Set the frequency multiplier

Annex A Zoom Table of SCPI Classified by Subsystem

16	[:SOURce]:FREQuency:OFFSet(?)	Set the frequency offset
17	[:SOURce]:FREQuency:REFerence(?)	Set the relative frequency
18	[:SOURce]:FREQuency:REFerence:STATe(?)	Set the relative frequency to ON/OFF state
19	[:SOURce]:FREQuency:STEP(?)	Set the frequency step
20	[:SOURce]:FREQuency:STARt(?)	Set the step sweep start frequency
21	[:SOURce]:FREQuency:STOP(?)	Set the step sweep stop frequency
22	[:SOURce]:POWer:ALC:LEVel(?)	Set ALC level
23	[:SOURce]:POWer:ALC:SEARch(?)	Set the power search mode
24	[:SOURce]:POWer:ALC:SOURce(?)	Set the power stabilization mode
25	[:SOURce]:POWer:ALC:SOURce:EXTernal:COUPling(?)	Set the coupling coefficient of external detection
26	[:SOURce]:POWer:REFerence:STATe(?)	Set the ALC loop state
27	[:SOURce]:POWer:ATTenuation(?)	Set the power attenuation
28	[:SOURce]:POWer:ATTenuation:AUTO(?)	Set the power attenuation to ON/OFF state
29	[:SOURce]:POWer[:LEVel][:IMMediate][:AMPLitude](?)	Set the power level
30	[:SOURce]:POWer[:LEVel][:IMMediate]:OFFSet(?)	Set the power offset
31	[:SOURce]:POWer:REFerence(?)	Set the relative power
32	[:SOURce]:POWer:REFerence:STATe(?)	Set the relative power to ON/OFF state
33	[:SOURce]:POWer:STEP(?)	Set the power step

34	[:SOURce]:POWer:ALC:BANDwidth|BWIDth	Set the bandwidth of ALC loop
35	[:SOURce]:POWer:ALC:BANDwidth|BWIDth:AUTo	Set the bandwidth selection mode of ALC loop
36	[:SOURce]:POWer:SWEep[:STATe]	Set the power sweep to ON/OFF state
37	[:SOURce]:LIST:DIRection(?)	Set the list sweep direction
38	[:SOURce]:LIST:DWELl	Set the dwell time of all points
39	[:SOURce]:LIST:FREQuency	Set the list sweep frequency
40	[:SOURce]:LIST:FILL:POINts(?)	Set the list sweep points
41	[:SOURce]:LIST:FILL:STARt(?)	Set the list sweep start frequency
42	[:SOURce]:LIST:FILL:STOP(?)	Set the list sweep stop frequency
43	[:SOURce]:LIST:FILL:POWer(?)	Set the list sweep power
44	[:SOURce]:LIST:TRIGger:SOURce(?)	Set the list sweep trigger source
45	[:SOURce]:LIST:FILL:POWer(?)	Set the power offset of all list points
46	[:SOURce]:LIST:FILL:DWELl(?)	Set the dwell time of all list points
47	[:SOURce]:LIST:FILL:EXECute	Complete list filling
48	[:SOURce]:LIST:DELete	Delete list points
49	[:SOURce]:LFOutput:AMPLitude(?)	Set LF amplitude
50	[:SOURce]:LFOutput:FREQuency(?)	Set LF
51	[:SOURce]:LFOutput:RAMP(?)	Set the waveform direction of LF ramp
52	[:SOURce]:LFOutput:SHAPe(?)	Set LF waveform

Annex A Zoom Table of SCPI Classified by Subsystem

53	[:SOURce]:LFOutput:STATe(?)	Set LF to ON/OFF state
54	[:SOURce]:LFOutput:OFFSet(?)	Set LF amplitude offset
55	[:SOURce]:LFOutput:DUAL:FUNCtion:AMPLitude:PERCent?	Set the amplitude ratio of the dual function generator relative to audio 1
56	[:SOURce]:LFOutput:DUAL: FUNCtion[1]|2:FREQuency?	Set the value of frequency 1 (default) or frequency 2 of the dual function generator
57	[:SOURce]:LFOutput:DUAL:FUNCtion[1]|2:PERCent?	Set the pulse duty factor of the dual function generator relative to audio 1 (default) or audio 2
58	[:SOURce]:LFOutput:DUAL:FUNCtion: POFFset?	Set the phase offset of the dual function generator relative to audio 1
59	[:SOURce]:LFOutput:DUAL:FUNCtion:SHAPe?	Set the output waveform of the dual function generator
60	[:SOURce]:LFOutput:DUAL:FUNCtion:SHAPe:RAMP?	Set the signal output type when the output waveform of the dual function generator is ramp
61	[:SOURce]:LFOutput:FUNCtion[1]|2:FREQuency?	Set the output frequency of the function generator 1|2
62	[:SOURce]:LFOutput:FUNCtion[1]|2:PERCent?	Set the pulse duty factor of the function generator 1|2
63	[:SOURce]:LFOutput:FUNCtion[1]|2:SHAPe?	Set the output waveform of the function generator 1|2

64	[:SOURce]:LFOutput:FUNCtion[1]|2:SHAPe:RAMP?	Set the signal output type when the output waveform of the function generator 1|2 is ramp
65	[:SOURce]:LFOutput:NOISe[1]|2:TYPe?	Set the noise type of the noise generator 1|2
66	[:SOURce]:LFOutput:SWEep:FUNCtion:FREQuency:STARt?	Set the start frequency of the sweep function generator
67	[:SOURce]:LFOutput:SWEep:FUNCtion:FREQuency:STOP?	Set the stop frequency of the sweep function generator
68	[:SOURce]:LFOutput:SWEep:FUNCtion:SHAPe?	Set the output waveform of the sweep function generator
69	[:SOURce]:LFOutput:SWEep:FUNCtion:SHAPe:RAMP?	Set the signal output type when the output waveform of the sweep function generator is ramp
70	[:SOURce]:LFOutput:SWEep:FUNCtion:TIME?	Set the sweep time of the sweep function generator
71	[:SOURce]:LFOutput:SWEep:FUNCtion:TRIGger:MODE?	Set the trigger mode of the sweep function generator
72	[:SOURce]:LFOutput:SWEep:FUNCtion:TRIGger: PERiod?	Set the sweep timer period of the sweep function generator
73	[:SOURce]:LFOutput:SWEep:FUNCtion:TRIGger:TYPE?	Set the trigger type of the sweep function generator
74	[:SOURce]:SWEep:DIRection(?)	Set the step sweep direction
75	[:SOURce]:SWEep:DWELl(?)	Set the step dwell time

76	[:SOURce]:SWEep:POINts(?)	Set the step sweep points
77	[:SOURce]:SWEep:TRIGger:SOURce(?)	Set the step sweep trigger source
78	[:SOURce]:SWEep:RETRace(?)	Set the sweep to ON/OFF state
79	[:SOURce]:SWEep:STEP:TYPE(?)	Set the step sweep mode
80	[:SOURce]:SWEep:STARt:TRIGger(?)	Set the start sweep trigger type
81	[:SOURce]:SWEep:MODE(?)	Set the sweep mode
82	[:SOURce]:PULM:EXTernal:POLarity(?)	Pulse input inverted ON/OFF
83	[:SOURce]:PULM:INTernal:DELay(?)	Set the pulse delay
84	[:SOURce]:PULM:INTernal:FREQuency(?)	Set the pulse frequency
85	[:SOURce]:PULM:INTernal:PERiod(?)	Set the pulse modulation period
86	[:SOURce]:PULM:INTernal:PWIDth(?)	Set the pulse modulation width
87	[:SOURce]:PULM:SOURce(?)	Set the pulse source
88	[:SOURce]:PULM:STATe(?)	Set the pulse modulation to ON/OFF state
89	[:SOURce]:PULM:INTernal:JITTered:MODE(?)	Set the jittered mode of pulse period
90	[:SOURce]:PULM:INTernal:JITTered:PERCent (?)	Set the jittered percent of pulse frequency
91	[:SOURce]:PULM:INTernal:PTRain:DATA	Set the pulse train points
92	[:SOURce]:PULM:INTernal:PTRain:DELete	Delete any index point in the pulse train list
93	[:SOURce]:PULM:INTernal:PTRain:POINts	Query the current

		pulse train points
94	[:SOURce]:AM[1]|2[:DEPTh]:EXPonential(?)	Set the AM signal depth of Path 1 or Path 2
95	[:SOURce]:AM[1]|2[:DEPTh][:LINear](?)	Set the AM signal depth of Path 1 or Path 2 expressed as a percent
96	[:SOURce]:AM[1]|2:INTernal:FREQuency(?)	Set internal AM rate of Path 1 or Path 2
97	[:SOURce]:AM[1]|2:INTernal:RAMP(?)	Set the signal output type when the AM waveform of Path 1 or Path 2 is ramp
98	[:SOURce]:AM[1]|2:INTernal:SHAPe(?)	This command is used to set the AM signal output waveform
99	[:SOURce]:AM[1]|2:STATe(?)	Set the AM Path 1 or Path 2 of the signal generator to ON/OFF state
100	[:SOURce]:AM:MODE(?)	Set the deep AM to ON/OFF state
101	[:SOURce]:AM:SOURce(?)	Set AM input selection
102	[:SOURce]:AM:MODulation:STATe(?)	Set the AM signal output state of the signal generator
103	[:SOURce]:AM:TYPE(?)	Set the AM type
104	[:SOURce]:AM:EXTernal:COUPling(?)	Set the AM external input coupling mode
105	[:SOURce]:AM:EXTernal:PATH(?)	Set the AM external input path
106	[:SOURce]:AM[1]|2:INTernal:DUAL:FUNCtion:AMPlitude:PERCent?	Set the amplitude ratio of the dual function generator relative to audio 1
107	[:SOURce]:AM[1]|2:INTernal:DUAL:FUNCtion[1]|2:FREQuency?	Set the frequency

		of the dual function generator relative to audio 1 when the waveform of AM Path 1 or Path 2 is dual function generator
108	[:SOURce]:AM[1]|2:INTernal:DUAL:FUNCtion[1]|2:PERCent?	Set the pulse duty factor of the dual function generator relative to audio 1 when the waveform of AM Path 1 or Path 2 is dual function generator
109	[:SOURce]:AM[1]|2:INTernal:DUAL: FUNCtion:POFFset?	Set the phase offset of the dual function generator relative to audio 1 when the waveform of AM Path 1 or Path 2 is dual function generator
110	[:SOURce]:AM[1]|2:INTernal:DUAL:FUNCtion:SHAPe?	Set the output waveform of the dual function generator when the waveform of AM Path 1 or Path 2 is dual function generator
111	[:SOURce]:AM[1]|2:INTernal:DUAL:FUNCtion:SHAPe:RAMP?	Set the signal output type when the waveform of AM Path 1 or Path 2 is dual function generator and the output waveform of the generator is ramp.
112	[:SOURce]:AM[1]|2:INTernal:FUNCtion[1]|2:FREQuency?	Set the output frequency of the function generator 1|2 when the waveform of AM Path 1 or Path 2 is function generator 1|2
113	[:SOURce]:AM[1]|2:INTernal:FUNCtion[1]|2:PERCent?	Set the pulse duty factor of the function generator

		1|2 when the waveform of AM Path 1 or Path 2 is function generator 1|2
114	[:SOURce]:AM[1]|2:INTernal:FUNCtion[1]|2:SHAPe?	Set the output waveform of the function generator 1|2 when the waveform of AM Path 1 or Path 2 is function generator 1|2
115	[:SOURce]:AM[1]|2:INTernal:FUNCtion[1]|2:SHAPe:RAMP?	Set the signal output type when the waveform of AM Path 1 or Path 2 is function generator 1|2 and the output waveform of the generator 1|2 is ramp
116	[:SOURce]:AM[1]|2:INTernal:NOISe:FUNCtion[1]|2:TYPE?	Set the noise type of the noise generator 1|2 when the waveform of AM Path 1 or Path 2 is noise generator 1|2
117	[:SOURce]:AM[1]|2:INTernal:SWEep:FUNCtion:FREQuency:STARt?	Set the start frequency of the sweep generator when the waveform of AM Path 1 or Path 2 is sweep generator
118	[:SOURce]:AM[1]|2:INTernal:SWEep:FUNCtion:FREQuency:STOP?	Set the stop frequency of the sweep generator when the waveform of AM Path 1 or Path 2 is sweep generator
119	[:SOURce]:AM[1]|2:INTernal:SWEep:FUNCtion:SHAPe?	Set the sweep type of the sweep generator when the waveform of AM Path 1 or Path 2 is sweep generator

Annex A Zoom Table of SCPI Classified by Subsystem

120	[:SOURce]:AM[1]|2:INTernal:SWEep:FUNCtion:SHAPe:RAMP?	Set the signal output type when the waveform of AM Path 1 or Path 2 is sweep generator and the sweep type is ramp
121	[:SOURce]:AM[1]|2:INTernal:SWEep:FUNCtion:TIME?	Set the sweep time of the sweep generator when the waveform of AM Path 1 or Path 2 is sweep generator
122	[:SOURce]:AM[1]|2:INTernal:SWEep:FUNCtion:TRIGger:MODE?	Set the trigger mode of the sweep generator when the waveform of AM Path 1 or Path 2 is sweep generator
123	[:SOURce]:AM[1]|2:INTernal:SWEep:FUNCtion:TRIGger:PERiod?	Set the sweep timer period of the sweep generator when the waveform of AM Path 1 or Path 2 is sweep generator and the trigger type is timed trigger
124	[:SOURce]:AM[1]|2:INTernal:SWEep:FUNCtion:TRIGger:TYPE?	This command is used to set the trigger type of sweep generator when the waveform of AM Path 1 or Path 2 is sweep generator
125	[:SOURce]:FM[1]|2:DEViation(?)	Set the frequency deviation of FM Path 1 or Path 2 of the signal generator
126	[:SOURce]:FM[1]|2:INTernal:FREQuency(?)	Set the internal FM rate of FM Path 1 or Path 2 of the signal generator
127	[:SOURce]:FM[1]|2:INTernal:RAMP(?)	Set the ramp type when the FM waveform of Path 1 or Path 2 is ramp

128	[:SOURce]:FM[1]|2:INTernal:SHAPe(?)	Set the FM signal output waveform of Path 1 or Path 2
129	[:SOURce]:FM[1]|2:STATe(?)	Set the FM Path 1 or Path 2 of the signal generator to ON/OFF state
130	[:SOURce]:FM:SOURce(?)	Set the FM source
131	[:SOURce]:FM:MODulation:STATe(?)	Set the FM to ON/OFF state
132	[:SOURce]:FM:EXTernalL:COUPling (?)	Set the FM external input coupling mode
133	[:SOURce]:FM:EXTernal:PATH (?)	Set the FM external input path
134	[:SOURce]:FM[1]|2:INTernal:DUAL:FUNCtion:AMPlitude:PERCent?	Set the amplitude ratio of the dual function generator relative to audio 1 when the waveform of FM Path 1 or 2 is dual function generator
135	[:SOURce]:FM[1]|2:INTernal:DUAL:FUNCtion[1]|2:FREQuency?	Set the frequency of the dual function generator relative to audio 1 when the waveform of FM Path 1 or Path 2 is dual function generator
136	[:SOURce]:FM[1]|2:INTernal:DUAL:FUNCtion[1]|2:PERCent?	Set the pulse duty factor of the dual function generator relative to audio 1 when the waveform of FM Path 1 or Path 2 is dual function generator
137	[:SOURce]:FM[1]|2:INTernal:DUAL: FUNCtion:POFFset?	Set the phase offset of the dual function generator relative to audio 1 when the waveform of FM Path 1 or Path 2 is dual function generator

138	[:SOURce]:FM[1]|2:INTernal:DUAL:FUNCtion:SHAPe?	Set the output waveform of the dual function generator when the waveform of FM Path 1 or Path 2 is dual function generator
139	[:SOURce]:FM[1]|2:INTernal:DUAL:FUNCtion:SHAPe:RAMP?	Set the signal output type when the waveform of FM Path 1 or Path 2 is dual function generator and the output waveform of the generator is ramp
140	[:SOURce]:FM[1]|2:INTernal:FUNCtion[1]|2:FREQuency?	Set the output frequency of the function generator 1|2 when the waveform of FM Path 1 or Path 2 is function generator 1|2
141	[:SOURce]:FM[1]|2:INTernal:FUNCtion[1]|2:PERCent?	Set the pulse duty factor of the function generator 1|2 when the waveform of FM Path 1 or Path 2 is function generator 1|2
142	[:SOURce]:FM[1]|2:INTernal:FUNCtion[1]|2:SHAPe?	Set the output waveform of the function generator 1|2 when the waveform of FM Path 1 or Path 2 is function generator 1|2
143	[:SOURce]:FM[1]|2:INTernal:FUNCtion[1]|2:SHAPe:RAMP?	Set the signal output type when the waveform of FM Path 1 or Path 2 is function generator 1|2 and the output waveform of the generator 1|2 is ramp

144	[:SOURce]:FM[1]|2:INTernal:NOISe:FUNCtion[1]|2:TYPE?	Set the noise type of the noise generator 1|2 when the waveform of FM Path 1 or Path 2 is noise generator 1|2
145	[:SOURce]:FM[1]|2:INTernal:SWEep:FUNCtion:FREQuency:STARt?	Set the start frequency of the sweep generator when the waveform of FM Path 1 or Path 2 is sweep generator
146	[:SOURce]:FM[1]|2:INTernal:SWEep:FUNCtion:FREQuency:STOP?	Set the stop frequency of the sweep generator when the waveform of FM Path 1 or Path 2 is sweep generator
147	[:SOURce]:FM[1]|2:INTernal:SWEep:FUNCtion:SHAPe?	Set the sweep type of the sweep generator when the waveform of FM Path 1 or Path 2 is sweep generator
148	[:SOURce]:FM[1]|2:INTernal:SWEep:FUNCtion:SHAPe:RAMP?	Set the signal output type when the waveform of FM Path 1 or Path 2 is sweep generator and the sweep type is ramp
149	[:SOURce]:FM[1]|2:INTernal:SWEep:FUNCtion:TIME?	Set the sweep time of the sweep generator when the waveform of FM Path 1 or Path 2 is sweep generator
150	[:SOURce]:FM[1]|2:INTernal:SWEep:FUNCtion:TRIGger:MODE?	Set the trigger mode of the sweep generator when the waveform of FM Path 1 or Path 2 is sweep generator
151	[:SOURce]:FM[1]|2:INTernal:SWEep:FUNCtion:TRIGger:PERiod?	Set the sweep timer period of the sweep generator

		when the waveform of FM Path 1 or Path 2 is sweep generator and the trigger type is timed trigger
152	[:SOURce]:FM[1]|2:INTernal:SWEep:FUNCtion:TRIGger:TYPE?	This command is used to set the trigger type of sweep generator when the waveform of FM Path 1 or Path 2 is sweep generator
153	[:SOURce]:PM[1]|2:DEViation(?)	Set the phase deviation of PM Path 1 or Path 2 of the signal generator
154	[:SOURce]:PM[1]|2:INTernal:FREQuency(?)	Set the internal PM rate of PM Path 1 or Path 2 of the signal generator
155	[:SOURce]:PM[1]|2:INTernal:SHAPE:RAMP(?)	Set the direction of ramp when the waveform of PM Path 1 or Path 2 is ramp
156	[:SOURce]:PM[1]|2:INTernal:SHAPe(?)	Set the output waveform of PM Path 1 or Path 2
157	[:SOURce]:PM[1]|2:STATe(?)	Set the PM Path 1 or Path 2 of the signal generator to ON/OFF state
158	[:SOURce]:PM:SOURce(?)	Set the PM source
159	[:SOURce]:PM:MODulation:STATe(?)	Set the PM signal output state of the signal generator
160	[:SOURce]:PM:EXTernal:COUPling(?)	Set the PM external input coupling mode
161	[:SOURce]:PM:EXTernal:PATH(?)	Set the PM external input path
162	[:SOURce]:PM[1]|2:INTernal:DUAL:FUNCtion:AMPlitude:PERCent?	Set the amplitude ratio of the dual

		function generator relative to audio 1 when the waveform of PM Path 1 or 2 is dual function generator
163	[:SOURce]:PM[1]|2:INTernal:DUAL:FUNCtion[1]|2:FREQuency?	Set the frequency of the dual function generator relative to audio 1 when the waveform of PM Path 1 or Path 2 is dual function generator
164	[:SOURce]:PM[1]|2:INTernal:DUAL:FUNCtion[1]|2:PERCent?	Set the pulse duty factor of the dual function generator relative to audio 1 when the waveform of PM Path 1 or Path 2 is dual function generator
165	[:SOURce]:PM[1]|2:INTernal:DUAL: FUNCtion:POFFset?	Set the phase offset of the dual function generator relative to audio 1 when the waveform of PM Path 1 or Path 2 is dual function generator
166	[:SOURce]:PM[1]|2:INTernal:DUAL:FUNCtion:SHAPe?	Set the output waveform of the dual function generator when the waveform of PM Path 1 or Path 2 is dual function generator
167	[:SOURce]:PM[1]|2:INTernal:DUAL:FUNCtion:SHAPe:RAMP?	Set the signal output type when the waveform of PM Path 1 or Path 2 is dual function generator and the output waveform of the generator is ramp
168	[:SOURce]:PM[1]|2:INTernal:FUNCtion[1]|2:FREQuency?	Set the output frequency of the function generator 1|2 when the

		waveform of PM Path 1 or Path 2 is function generator 1|2
169	[:SOURce]:PM[1]|2:INTernal:FUNCtion[1]|2:PERCent?	Set the pulse duty factor of the function generator 1|2 when the waveform of PM Path 1 or Path 2 is function generator 1|2
170	[:SOURce]:PM[1]|2:INTernal:FUNCtion[1]|2:SHAPe?	Set the output waveform of the function generator 1|2 when the waveform of PM Path 1 or Path 2 is function generator 1|2
171	[:SOURce]:PM[1]|2:INTernal:FUNCtion[1]|2:SHAPe:RAMP?	Set the signal output type when the waveform of PM Path 1 or Path 2 is function generator 1|2 and the output waveform of the generator 1|2 is ramp
172	[:SOURce]:PM[1]|2:INTernal:NOISe:FUNCtion[1]|2:TYPE?	Set the noise type of the noise generator 1|2 when the waveform of PM Path 1 or Path 2 is noise generator 1|2
173	[:SOURce]:PM[1]|2:INTernal:SWEep:FUNCtion:FREQuency:STARt?	Set the start frequency of the sweep generator when the waveform of PM Path 1 or Path 2 is sweep generator
174	[:SOURce]:PM[1]|2:INTernal:SWEep:FUNCtion:FREQuency:STOP?	Set the stop frequency of the sweep generator when the waveform of PM Path 1 or Path 2 is sweep

		generator
175	[:SOURce]:PM[1]|2:INTernal:SWEep:FUNCtion:SHAPe?	Set the sweep type of the sweep generator when the waveform of PM Path 1 or Path 2 is sweep generator
176	[:SOURce]:PM[1]|2:INTernal:SWEep:FUNCtion:SHAPe:RAMP?	Set the signal output type when the waveform of PM Path 1 or Path 2 is sweep generator and the sweep type is ramp
177	[:SOURce]:PM[1]|2:INTernal:SWEep:FUNCtion:TIME?	Set the sweep time of the sweep generator when the waveform of PM Path 1 or Path 2 is sweep generator
178	[:SOURce]:PM[1]|2:INTernal:SWEep:FUNCtion:TRIGger:MODE?	Set the trigger mode of the sweep generator when the waveform of PM Path 1 or Path 2 is sweep generator
179	[:SOURce]:PM[1]|2:INTernal:SWEep:FUNCtion:TRIGger:PERiod?	Set the sweep timer period of the sweep generator when the waveform of PM Path 1 or Path 2 is sweep generator and the trigger type is timed trigger
180	[:SOURce]:PM[1]|2:INTernal:SWEep:FUNCtion:TRIGger:TYPE?	This command is used to set the trigger type of sweep generator when the waveform of PM Path 1 or Path 2 is sweep generator
181	[:SOURce]:DM:IQADjustment::GAIN(?)	Set internal IQ gain balance
182	[:SOURce]:DM:IQADjustment:IOFFse(?))	Set the offset of Path I

Annex A Zoom Table of SCPI Classified by Subsystem

183	[:SOURce]:DM:IQADjustment:QOFFse(?)	Set the offset of Path Q
184	[:SOURce]:DM:IQADjustment:QSKew(?)	Set the phase angle between IQ vectors
185	[:SOURce]:DM:IQADjustment[:STATe](?)	Set IQ adjustment to ON/OFF state
186	[:SOURce]:DM:MODulation:ATTenuation(?)	Set IQ signal attenuation
187	[:SOURce]:DM:MODulation:ATTenuation:AUTO(?)	Set IQ signal attenuation to ON/OFF state
188	[:SOURce]:DM:STATe(?)	Set IQ modulation to ON/OFF state
189	[:SOURce]:DM:EXTernal:BWIDth[:STATe](?)	Set external broadband I/Q input to ON/OFF state
190	[:SOURce]:DM:IQADjustment:OUTPut[:STATe](?)	Set IQ input adjustment to ON/OFF state
191	[:SOURce]:DM:IQADjustment:OUTPut:ATTen (?)	Set the attenuation of I/Q output adjustment
192	[:SOURce]:DM:IQADjustment:OUTPut:GAIN (?)	Set I/Q output adjustment gain balance
193	[:SOURce]:DM:IQADjustment:OUTPut:IOFFset(?)	Set I offset of I/Q output adjustment
194	[:SOURce]:DM:IQADjustment:OUTPut:UIOFfset(?)	Set I/offset of I/Q output adjustment
195	[:SOURce]:DM:IQADjustment:OUTPut:QOFFset(?)	Set Q offset of I/Q output adjustment
196	[:SOURce]:DM:IQADjustment:OUTPut:UQOFfset(?)	Set Q/offset of I/Q output adjustment
197	[:SOURce]:DM:IQADjustment:OUTPut:SKEW (?)	Set orthogonal offset of I/Q output adjustment
198	[:SOURce]:RADio:CUSTom:ALPHa(?)	Set the radio filter factor

199	[:SOURce]:RADio:CUSTom:DATA(?)	Set the data source of radio modulation signal
200	[:SOURce]:RADio:CUSTom:DATA:PRAM	Select stream file
201	[:SOURce]:RADio:CUSTom:FILTer(?)	Set the radio filter type
202	[:SOURce]:RADio:CUSTom:DATA:FIX4?	Set the value of the code data when fixed 4-bit code is selected as the data source
203	[:SOURce]:RADio:CUSTom:IQData	Transmit IQ data pair into the radio memory
204	[:SOURce]:RADio:CUSTom:MODulation:FSK[:DEViation](?)	Set the frequency deviation when the radio modulation type is FSK mode
205	[:SOURce]:RADio:CUSTom:MODulation:MSK:PHASe(?)	Set the phase deviation when the radio modulation type is MSK mode
206	[:SOURce]:RADio:CUSTom:MODulation[:TYPE](?)	Set the radio modulation type
207	[:SOURce]:RADio:CUSTom:MODulation:ASK:DEPTh:PERCent?	Set the modulation depth of ASK when the radio modulation type is ASK mode
208	[:SOURce]:RADio:CUSTom:MODulation:UFSK	Set the file when the radio modulation type is user FSK mode
209	[:SOURce]:RADio:CUSTom:MODulation:UIQ	Set the file when the radio modulation type is user IQ mode
210	[:SOURce]:RADio:CUSTom:SRATe(?)	Set the radio symbol rate
211	[:SOURce]:RADio:CUSTom:STATe(?)	Set the radio to ON/OFF state
212	[:SOURce]:RADio:CUSTom:POLarity[:ALL](?)	Set the radio signal

		phase polarity
213	[:SOURce]:RADio:CUSTom:DENCode(?)	Set the differential encoding to ON/OFF state
214	[:SOURce]:RADio:CUSTom:VCO:CLOCk(?)	Set the radio sampling clock type
215	[:SOURce]:RADio:CUSTom:TRIGger:EXTernal:SOURce(?)	Set the external radio trigger source
216	[:SOURce]:RADio:CUSTom:TRIGger:EXTernal:SOURce:DELay(?)	Set the external delay time of trigger source
217	[:SOURce]:RADio:CUSTom:TRIGger:EXTernal:SOURce:DELay:STATe(?)	Set the external trigger source delay to ON/OFF state
218	[:SOURce]:RADio:CUSTom:TRIGger:EXTernal:SOURce:SLOPe(?)	Set the trigger polarity of external trigger source
219	[:SOURce]:RADio:CUSTom:TRIGger:SOURce(?)	Set the radio trigger source type
220	[:SOURce]:RADio:CUSTom:TRIGger:TYPE(?)	Set the trigger mode of radio trigger source
221	[:SOURce]:RADio:CUSTom:TRIGger:TYPE:CONTinuous:TYPE (?)	Set the radio trigger type in continuous mode
222	[:SOURce]:RADio:CUSTom:TRIGger:TYPE:GATE:ACTive(?)	Set the radio trigger type in gate mode
223	[:SOURce]:RADio:MTONe:ARB:SETup	Select multitone file for loading
224	[:SOURce]:RADio:MTONe:ARB:SETup:STORe	Save the multitone file
225	[:SOURce]:RADio:MTONe:ARB: SETup:TABLe	Configure the multitone waveform sequence
226	[:SOURce]:RADio:MTONe:ARB: SETup:TABLe:FSPacing(?)	Set the multitone frequency spacing
227	[:SOURce]:RADio:MTONe:ARB: SETup:TABLe:NTONes(?)	Set the number of multitone

228	[:SOURce]:RADio:MTONe:ARB: SETup:TABLe:PHASe:INITialize(?)	Set the initial phase type of multitone
229	[:SOURce]:RADio:MTONe:ARB: SETup:TABLe:PHASeINITialize:SEED(?)	Set the relationship between tone phases
230	[:SOURce]:RADio:MTONe:ARB: SETup:TABLe:ROW(?)	Set the multitone parameters in a row of the multitone modulation list
231	[:SOURce]:RADio:MTONe:ARB[:STATe](?)	Set multitone to ON/OFF state
232	[:SOURce]:RADio:TTONe:ARB:ALIGnment(?)	Set the alignment position of two tone signal
233	[:SOURce]:RADio:TTONe:ARB:FSPacing(?)	Set the two tone frequency spacing
234	[:SOURce]:RADio:TTONe:ARB[:STATe](?)	Set two tone to ON/OFF state
235	[:SOURce]:RADio:ARB:MODE(?)	Set arbitrary mode
236	[:SOURce]:RADio:ARB[:STATe](?)	Set arbitrary to ON/OFF state
237	[:SOURce]:RADio:ARB:SEQuence	Load the arbitrary file
238	[:SOURce]:RADio:ARB:SEQuence:CLOCk (?)	Set the arbitrary clock type
239	[:SOURce]:RADio:ARB:SCLock:RATE(?)	Set the arbitrary clock frequency
240	[:SOURce]:RADio:ARB:TRIGger:TYPE(?)	Set the arbitrary trigger mode
241	[:SOURce]:RADio:ARB:TRIGger:TYPE:CONTinuous[:TYPE](?)	Set the arbitrary continuous trigger mode
242	[:SOURce]:RADio:ARB:TRIGger:TYPE:SINGle(?)	Set the arbitrary single trigger mode
243	[:SOURce]:RADio:ARB:TRIGger:TYPE:SADVance[:TYPE](?)	Set the arbitrary waveform segment trigger mode
244	[:SOURce]:RADio:ARB:TRIGger:TYPE:GATE:ACTive (?)	Set the arbitrary gate trigger mode

Annex A Zoom Table of SCPI Classified by Subsystem

245	[:SOURce]:RADio:ARB:TRIGger:SOURce (?)	Set the arbitrary trigger source
246	[:SOURce]:RADio:ARB:VCO:CLOCk(?)	Set the arbitrary trigger sampling clock
247	[:SOURce]:RADio:ARB:EXTernal:CLOCk:RATE(?)	Set the external clock frequency of arbitrary trigger
248	:MEMory:COPY:NAME	Copy a file in the signal generator
249	:MEMory:DLEete:NAME	Delete a user file
250	:MEMory:MOVE	Rename a file in the signal generator
251	:MEMory:DATA	Transmit a data file
252	:ROSCillator:ADJust:REFerence(?)	Set the internal reference of the signal generator
253	:DIAGnostic:INFormation:CCOunt:PON?	Query the accumulative startup times of the instrument
254	:DIAGnostic:INFormation:OTIMe?	Query the instrument firmware date and time stamp
255	:DIAGnostic:SNUM?	Read the system serial number of the signal generator
256	:SYSTem:COMMunicate:GPIB:ADDRess(?)	Set GPIB address of the signal generator
257	:SYSTem:COMMunicate:GTLocal	Set the signal generator to local mode

Annex B Zoom Table of Errors

Table 2 Local errors

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.