R&S®FSH20-N
Remote Control via LAN or USB
Software Manual

MANUAL ISSUE

Equipment:	R&S®FSH20-N Spectrum Analyzer
Title:	R&S®FSH20-N Spectrum Analyzer Software Manual
Manual Part #:	1176.8897.02
Revision:	C
Issue Date:	June 20, 2020

1176.8897.02 – 03

Software Manual

ROHDE&SCHWARZ

The Software Manual describes the following R&S®FSH functions

- Remote control via LAN or USB

The contents are for R&S FSH models:

- R&S FSH20-N (1314.2000.80)

Throughout this manual, products from Rohde & Schwarz are indicated without the ® symbol and the model designation, e.g. R&S®FSH is indicated as R&S FSH.

Table of Contents

Documentation Overview

The user documentation for the R&S FSH is divided as follows:

Manual

The Manual contains information for manual operation of the R&S FSH and for maintenance (service) of the R&S FSH.

It covers the following topics:

- overview of all elements of the front and rear panels
- basic information on how to set up the R&S FSH
- information on how to operate the R&S FSH in a network
- instructions on how to perform measurements
- instructions on how to set up and operate the R&S FSH in its various operating modes
- instructions on how to perform measurements with the R&S FSH
- instructions on how to work with the available software options and applications

Service Manual

The Service Manual provides information on maintenance.

It covers the following topics:

- instructions on how to perform a performance test
- instructions on how to repair the R&S FSH including a spare parts list

Software Manual for remote control

The Software Manual contains information for remote operation of the R&S FSH.

It covers the following topics:

- information about the SCPI command structure and syntax
- information on how to set up a remote connection with the R&S FSH
- information about command processing
- information about the status reporting system
- description of supported remote commands

Conventions Used in the Documentation

The following conventions are used throughout the R&S FSH Software Manual:

Typographical conventions

Convention	Description
“Graphical user interface elements”	All names of graphical user interface elements both on the screen and on the front and rear panels, such as dialog boxes, softkeys, menus, options, buttons etc., are enclosed by quotation marks.
“KEYS”	Key names are written in capital letters and enclosed by quotation marks.
Input	Input to be entered by the user is displayed in italics.
`File names, commands, program code`	File names, commands, coding samples and screen output are distinguished by their font.
"Links"	Links that you can click are displayed in blue font.
"References"	References to other parts of the documentation are enclosed by quotation marks.

Other conventions

- **Remote commands**: Remote commands may include abbreviations to simplify input. In the description of such commands, all parts that have to be entered are written in capital letters. Additional text in lower-case characters is for information only.

1 Interfaces and Protocols

The R&S FSH supports two different interfaces for remote control.

- LAN Interface: The protocol is based on TCP/IP and supports the VXI-11 standard.
- USB Interface

The connectors are located at the side of the instrument and permit a connection to a controller for remote control via a local area network (LAN) or directly via USB.

SCPI

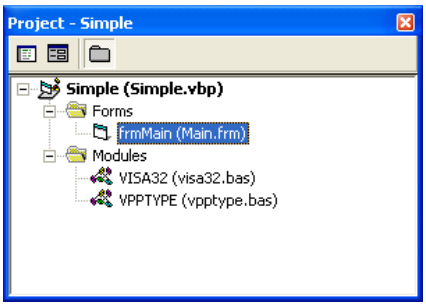
SCPI (Standard Commands for Programmable Instruments) commands - messages - are used for remote control. Commands that are not taken from the SCPI standard follow the SCPI syntax rules. The instrument supports the SCPI version 1999. The SCPI standard is based on standard IEEE 488.2 and aims at the standardization of device-specific commands, error handling and the status registers. The tutorial "Automatic Measurement Control - A tutorial on SCPI and IEEE 488.2" from John M. Pieper (R&S order number 0002.3536.00) offers detailed information on concepts and definitions of SCPI.

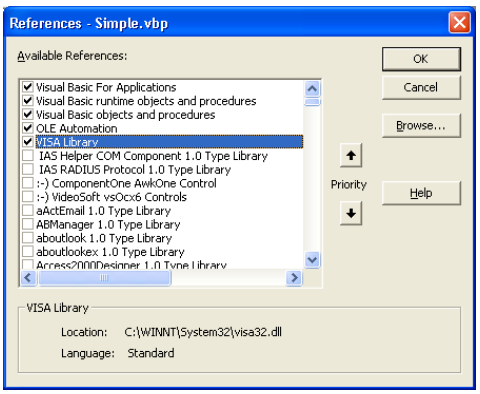
The requirements that the SCPI standard places on command syntax, error handling and configuration of the status registers are explained in detail in the following sections. Tables provide a fast overview of the bit assignment in the status registers. The tables are supplemented by a comprehensive description of the status registers.

VISA

VISA is a standardized software interface library providing input and output functions to communicate with instruments. The I/O channel (LAN or USB) is selected at initialization time by means of a channel-specific resource string. For more information about VISA refer to its user documentation.

The programming examples for remote control are all written in Microsoft® VISUAL BASIC®. Access to the VISA functions require the declaration of the functions and constants prior to their use in the project. This can be accomplished either by adding the modules VISA32.BAS and VPPTYPE.BAS or a reference to the VISA32.DLL to the project.





The modules visa32.bas and vpptype.bas can be found in the following location:

<VXIpnpPath>\WinNT\include (typically C:\VXIpnp\WinNt\include).

Resetting the R&S FSH

Manual operation is designed for maximum possible operating convenience. In contrast, the priority of remote control is the "predictability" of the device status. Therefore, control programs should always define an initial device status (e.g. with the command *RST) and then implement the required settings.

1.1 LAN Interface

To be integrated in a LAN, the instrument is equipped with a standard LAN interface, consisting of a connector, a network interface and protocols (VXI-11).

Instrument access via VXI-11 is usually achieved from high level programming platforms by using VISA as an intermediate abstraction layer. VISA encapsulates the low level VXI-11 (LAN) or USB function calls and thus makes the transport interface transparent for the user. The necessary VISA library is available as a separate product. For details contact your local R&S sales representative.

1.2 USB Interface

For remote control via the USB connection, the PC and the instrument must be connected via the USB interface. The required driver comes with the R&S InstrumentView software package and is automatically installed on the PC with the software package.

The driver adresses the instrument via the USB interface with the fix IP address 172.16.10.10.

In addition, a remote control connection via the SCPI interface requires the VISA library to be installed on the PC.

1.3 Protocols

VXI-11 Basics

The VXI-11 standard is based on the ONC-RPC protocol which in turn relies on TCP/IP as the network/transport layer. The TCP/IP network protocol and the associated network services are preconfigured. TCP/IP ensures connection-oriented communication, where the order of the exchanged messages is adhered to and interrupted links are identified. With this protocol, messages cannot be lost.

Remote control of an instrument via a network is based on standardized protocols which follow the OSI reference model (see Fig. below).

Application	SCPI
Presentation	XDR (VXI-11)
Session	ONC-RPC
Transport	TCP / UDP
Network	IP
Data Link	Ethernet/802.3
Physical	802.3/10BASE-T

Figure 1-1: Example for LAN remote control based on the OSI reference model

Based on TCP/UDP, messages between the controller and the instrument are exchanged via open network computing (ONC) - remote procedure calls (RPC). With XDR (VXI-11), legal RPC messages are known as VXI-11 standard. Based on this standard, messages are exchanged between the controller and the instrument. The messages are identical with SCPI commands. They can be organized in four groups:

- program messages (control command to the instrument)
- response messages (values returned by the instrument)
- service request (spontaneous queries of the instrument)
- low-level control messages (interface messages).

A VXI-11 link between a controller and an instrument uses three channels: core, abort and interrupt channel. Instrument control is mainly performed on the core channel (program, response and low-level control messages). The abort channel is used for immediate abort of the core channel; the interrupt channel transmits spontaneous service requests of the instrument. Link setup itself is very complex. For more details refer to the VXI-11 specification.

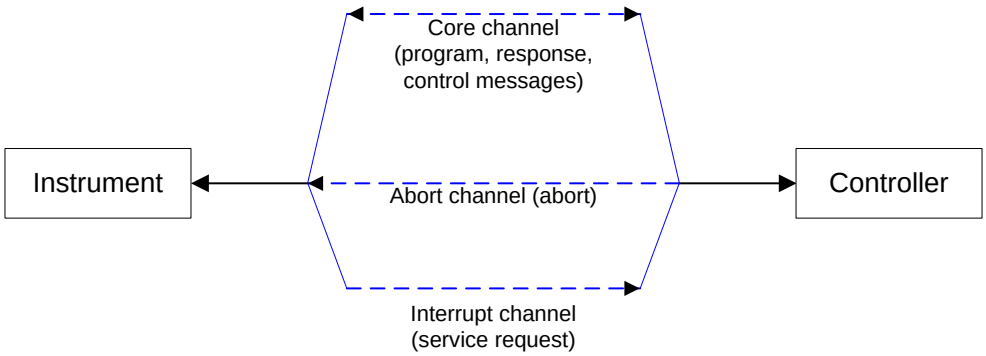


Figure 1-2: VXI-11 channels between instrument and controller

The number of controllers that can address an instrument is practically unlimited in the network. In the instrument, the individual controllers are clearly distinguished. This distinction continues up to the application level in the controller, i.e. two applications on a computer are identified by the instrument as two different controllers.

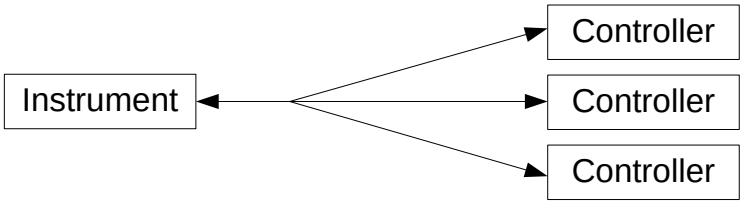


Figure 1-3: Remote control via LAN from several controllers

The controllers can lock and unlock the instrument for exclusive access. This regulates access to the instrument of several controllers.

2 Setting Up the Remote Control Connection

2.1 Preparing for Remote Control

The short and simple operating sequence below shows how to put the instrument into operation and quickly set its basic functions. The current IP address for LAN operation is shown in the SETUP – Instrument Setup Menu. In case of USB connection the IP address is fixed to 172.16.10.10.

Refer to the Quick Start Guide for instructions on how to change the IP address.

- ► Connect the instrument to the LAN or directly to the controller via USB.
- ► Switch on the instruments.
- ► Write and start the following program on the controller:

```
status = viOpenDefaultRM(defaultRM)
        'open default resource manager
status = viOpen(DefaultRM, "TCPIP::172.16.10.10", 0, 0, vi)
        'in case of USB connection
status = viopen(DefaultRM, "TCPIP::xxx.xxx.xxx.xxx", 0, 0, vi)
        'in case of a LAN connection, with xxx.xxx.xxx.xxx = IP address
cmd = "*RST;*CLS"
status = viWrite(vi, Cmd, Len(Cmd), retCount)
        'reset instrument and clear status registers
cmd = "FREQ:CENT 100MHz"
status = viWrite(vi, Cmd, Len(Cmd), retCount)
        'set center frequency to 100 MHz
cmd = "FREQ:SPAN 10MHz"
status = viWrite(vi, Cmd, Len(Cmd), retCount)
        'set span to 10 MHz
cmd = "DISP:TRAC:Y:RLEV -10dBm"
status = viWrite(vi, Cmd, Len(Cmd), retCount)
        'set reference level to -10 dBm
viclose vi
viclose default RM
```

The instrument now performs a sweep in the frequency range of 95 MHz to 105 MHz.

Changing the IP Address

In order to operate the instrument via remote control, it must be accessed via LAN (IP address) or USB (fixed IP address). If the factory-set remote control address does not fit in the network environment, it can be changed. Refer to the Quick Start Guide, chapter "Setting up a LAN or USB Connection to a PC", for instructions on how to change the IP address.

3 Instrument Model and Command Processing

The block diagram in Fig. 1-2 shows how SCPI commands are serviced in the instrument. The individual components work independently and simultaneously. They communicate with each other by means of so-called "messages".

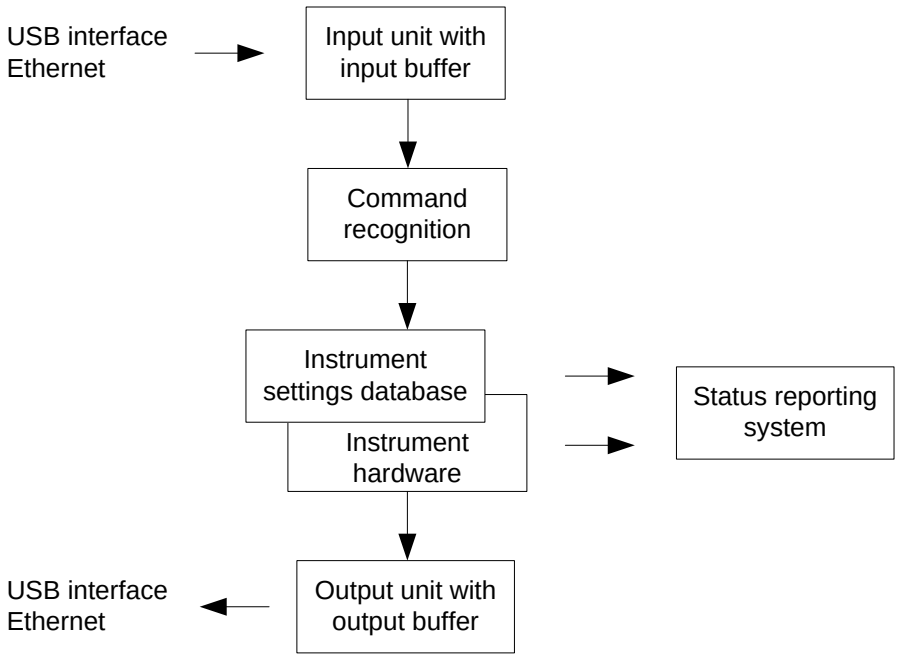


Figure 3-1: Instrument model in the case of remote control

3.1 Input Unit

The input unit receives commands character by character from the controller and collects them in the input buffer. The input unit sends a message to the command recognition as soon as the input buffer is full or as soon as it receives a delimiter, <PROGRAM MESSAGE TERMINATOR>, as defined in IEEE 488.2, or the interface message DCL.

If the input buffer is full, the traffic is stopped and the data received up to then are processed. Subsequently the traffic is continued. If, however, the buffer is not yet full when receiving the delimiter, the input unit can already receive the next command during command recognition and execution. The receipt of DCL clears the input buffer and immediately resets the command recognition.

3.2 Command Recognition

The command recognition analyses the data received from the input unit. It proceeds in the order in which it receives the data. Only DCL is serviced with priority, for example GET (Group Execute Trigger) is only executed after the commands received before. Each recognized command is immediately transferred to the internal instrument settings data base but not executed immediately.

The command recognition detects syntax errors in the commands and transfers them to the status reporting system. The rest of a program message after a syntax error is analyzed further if possible and serviced. After the syntax test, the value range of the parameter is checked, if required.

If the command recognition detects a delimiter, it passes the command to an execution unit that performs the instrument settings. In the meantime, the command recognition is ready to process new commands (overlapping execution). A DCL command is processed in the same way.

3.3 Data Base and Instrument Hardware

Here the expression "instrument hardware" denotes the part of the instrument fulfilling the actual instrument function - signal generation, measurement etc. The controller is not included. The term "data base" denotes a database that manages all the parameters and associated settings required for setting the instrument hardware.

Setting commands lead to an alteration in the data set. The data set management enters the new values (e.g. frequency) into the data set, however, only passes them on to the hardware when requested by the command recognition. This only takes place at the end of a program message.

The data are checked for compatibility with the current instrument settings before they are transmitted to the instrument hardware. If the execution is not possible, an "execution error" is signaled to the status reporting system. The corresponding settings are discarded.

Before passing on the data to the hardware, the settling bit in the STATus:OPERation register is set (refer to section "STATus:OPERation Register"). The hardware executes the settings and resets the bit again as soon as the new state has settled. This fact can be used to synchronize command servicing.

Queries induce the data set management to send the desired data to the output unit.

3.4 Status Reporting System

For detailed information refer to section "Status Reporting System".

3.5 Output Unit

The output unit collects the information requested by the controller, which it receives from the data base management. It processes it according to the SCPI rules and makes it available in the output buffer.

If the instrument is addressed as a talker without the output buffer containing data or awaiting data from the data base management, the output unit sends error message "Query UNTERMINATED" to the status reporting system. No data are sent to the controller, the controller waits until it has reached its time limit. This behavior is defined by IEEE 488.2 and SCPI.

4 SCPI Command Structure and Syntax

SCPI (Standard Commands for Programmable Instruments) describes a standard command set for programming instruments, irrespective of the type of instrument or manufacturer. The goal of the SCPI consortium is to standardize the device-specific commands to a large extent. For this purpose, a model was developed which defines the same functions inside a device or for different devices. Command systems were generated which are assigned to these functions. Thus it is possible to address the same functions with identical commands. The command systems are of a hierarchical structure.

SCPI is based on standard IEEE 488.2, i.e. it uses the same syntactic basic elements as well as the common commands defined in this standard. Part of the syntax of the device responses is defined with greater restrictions than in standard IEEE 488.2 (see section "Responses to Queries").

Remote command examples

Not all commands used in the following examples are implemented in the instrument.

4.1 Structure of a Command

The commands consist of a so-called header and, in most cases, one or more parameters. Header and parameter are separated by a "white space" (ASCII code 0 to 9, 11 to 32 decimal, e.g. blank). The headers may consist of several key words. Queries are formed by directly appending a question mark to the header.

4.1.1 Common Commands

Common commands consist of a header preceded by an asterisk "*" and one or several parameters, if any.

Examples

`*RST`	RESET, resets the device
`*ESE 253`	EVENT STATUS ENABLE, sets the bits of the event status enable register
`*ESR?`	EVENT STATUS QUERY, queries the contents of the event status register.

4.1.2 Device-Specific Commands

4.1.2.1 Hierarchy

Device-specific commands are of hierarchical structure. The different levels are represented by combined headers. Headers of the highest level (root level) have only one key word. This key word denotes a complete command system.

Example

`SENSe`
This key word denotes the SENSe command system.

For commands of lower levels, the complete path has to be specified, starting on the left with the highest level, the individual key words being separated by a colon ":".

Example

`SENSe:FREQuency:SPAN 10MHZ`
This command lies in the third level of the SENSe system. It sets the frequency span.

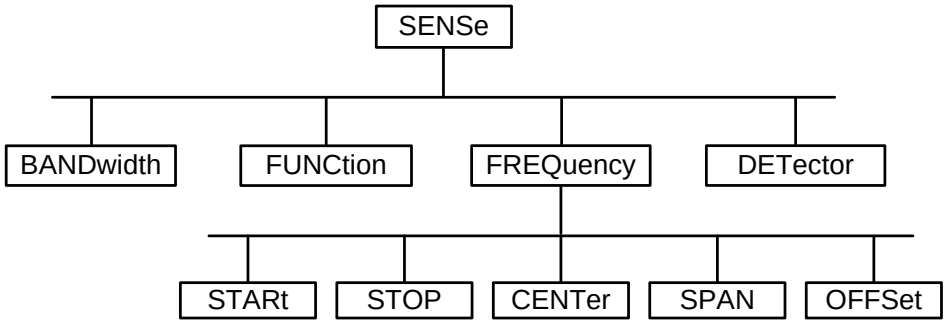


Figure 4-1: Tree structure the SCPI command systems using the SENSe system as example

4.1.2.2 Multiple Keywords

Some key words occur in several levels within one command system. Their effect depends on the structure of the command, i.e. at which position in the header of a command they are inserted.

Examples

`SOURce:FM:POLarity NORMal`
This command contains key word POLarity in the third command level. It defines the polarity between modulator and modulation signal.

`SOURce:FM:EXTernal:POLarity NORMal`
This command contains key word POLarity in the fourth command level. It defines the polarity between modulation voltage and the resulting direction of the modulation only for the external signal source indicated.

4.1.2.3 Optional Keywords

Some command systems permit certain key words to be inserted into the header or omitted. These key words are marked by square brackets in the description. The full command length must be recognized by the instrument for reasons of compatibility with the SCPI standard. Some commands are considerably shortened by these optional key words.

Example

`[SENSe]:BANDwidth[:RESolution]:AUTO`
This command couples the resolution bandwidth of the instrument to other parameters. The following command has the same effect:
`BANDwidth:AUTO`

Optional keywords with numeric suffixes

Do not omit an optional keyword if it includes a numeric suffix that is relevant for the effect of the command.

Example

`DISPlay[:WINDow<1...4>]:MAXimize <Boolean>`
Command `DISP:MAX ON` refers to window 1.

In order to refer to a window other than 1, you must include the optional `WINDow` parameter with the suffix for the required window.

`DISP:WIND2:MAX ON` refers to window 2.

4.1.2.4 Long and Short Form

The key words feature a long form and a short form. Either the short form or the long form can be entered, other abbreviations are not permitted.

Example

`STATus:QUEStionable:ENABle 1`
is equivalent to
`STAT:QUES:ENAB 1`

Upper and lower case notation of commands

Upper-case and lower-case notation only serves to distinguish the two forms in the manual, the instrument itself does not distinguish upper-case and lower-case letters.

4.1.2.5 Parameter

The parameter must be separated from the header by a "white space". If several parameters are specified in a command, they are separated by a comma ",". A few queries permit the parameters MINimum, MAXimum and DEFault to be entered. Refer to "Parameters" for a detailed description of the various parameters.

Example

```
SENSe:FREQuency:STOP? MAXimum
```
Response: `3.5E9`
This query requests the maximal value for the stop frequency.

4.1.2.6 Special Characters

| A vertical stroke in parameter definitions indicates alternative possibilities in the sense of "or". The effect of the command differs, depending on which parameter is used.

Example

```
DISPlay:FORMat SINGle | SPLit
```
If parameter `SINGle` is selected, full screen is displayed, in the case of `SPLit`, split screen is displayed.

A selection of key words with an identical effect exists for several commands. These keywords are indicated in the same line; they are separated by a vertical stroke. Only one of these keywords needs to be included in the header of the command. The effect of the command is independent of which of the keywords is used.

Example

```
SENSe:BANDwidth|BWIDth[:RESolution]
```
The two following commands with identical meaning can be created. They set the frequency of the fixed frequency signal to 1 kHz:
```
SENSe:BAND 1
SENSe:BWID 1
```

[ ] Key words in square brackets can be omitted when composing the header. The full command length must be accepted by the instrument for reasons of compatibility with the SCPI standards.

Example

```
[SENSe:]BANDwidth|BWIDth[:RESolution]
SENS:BAND:RES
```
is equivalent to
```
BAND
```

Parameters in square brackets can be incorporated optionally in the command or omitted as well.

Example

```
MMEMory:NETWork:MAP
<string>,<string>[,string>,<string>,<boolean>]
```

Entries in square brackets are optional or can be omitted.

{ } Parameters in curly brackets are optional and can be inserted once or several times, or omitted.

Example

```
SENSe:LIST:FREQuency <numeric_value>{,<numeric_value>}
```

The following are valid commands:

```
SENS:LIST:FREQ 10
SENS:LIST:FREQ 10,20
SENS:LIST:FREQ 10,20,30,40
```

4.1.2.7 Numeric Suffix

If a device features several functions or features of the same kind, e.g. inputs, the desired function can be selected by a suffix added to the command. Entries without suffix are interpreted like entries with the suffix 1. Optional keywords must be specified if they select a function with the suffix.

Example

```
SYSTem:COMMunicate:SERial2:BAUD 9600
```

This command sets the baud rate of a second serial interface.

Suffix counting

In case of remote control, suffix counting may differ from the numbers of the corresponding selection used in manual operation. SCPI prescribes that suffix counting starts with 1. Suffix 1 is the default state and used when no specific suffix is specified.

Some standards define a fixed numbering, starting with 0. With GSM, for instance, slots are counted from 0 to 7. In the case of remote control, the slots are selected with the suffixes 1 to 8. If the numbering differs in manual operation and remote control, it is indicated with the respective command.

4.1.3 Overview of Syntax Elements

The following table offers an overview of the syntax elements.

: The colon separates the key words of a command. In a program message the separating semicolon marks the uppermost command level.

; The semicolon separates two commands within a program message. It does not alter the path.

, The comma separates several parameters of a command.

? The question mark forms a query.

* The asterisk marks a common command.

" Quotation marks introduce a string and terminate it.

The hash symbol # introduces binary, octal, hexadecimal and block data.

- Binary: #B10110
- Octal: #O7612
- Hexa: #HF3A7
- Block: #21312

A "white space" (ASCII-Code 0 to 9, 11 to 32 decimal, e.g. blank) separates header and parameter.

4.2 Parameters

For most commands a parameter needs to be supplemented. The parameter has to be separated from the header by a "white space". Possible parameters are:

- Numeric values
- Special numeric values
- Boolean parameters
- Text
- Character strings
- Block data.

The type of parameter required for each command and the allowed range of values are specified in the command description.

4.2.1 Numeric Values

Numeric values can be entered in any form, i.e. with sign, decimal point and exponent. Values exceeding the resolution of the instrument are rounded up or down. The mantissa may comprise up to 255 characters, the exponent must lie inside the value range -32000 to 32000. The exponent is introduced by an "E" or "e". Entry of the exponent alone is not permissible. In the case of physical quantities, the unit can be entered. Permissible unit prefixes are G (giga), MA (mega), MOHM and MHZ are also possible), K (kilo), M (milli), U (micro) and N (nano). If the unit is missing, the basic unit is used.

Example

```
SENSe:FREQuency:STOP 1.5GHz = SENSe:FREQuency:STOP 1.5E9
```

4.2.2 Special Numeric Values

The texts MINimum, MAXimum, DEFault, UP and DOWN are interpreted as special numeric values. In case of a query, the numeric value is returned.

- MIN/MAX

  MINimum and MAXimum denote the minimum and maximum value.

- DEF

  DEFault denotes a preset value which has been stored in the EPROM. This value conforms to the default setting, as it is called by the `*RST` command

- UP/DOWN

  UP, DOWN increases or reduces the numerical value by one step. The step width can be specified via an allocated step command for each parameter which can be set via UP, DOWN.

- INF/NINF

  INFinity, Negative INFinity (NINF) Negative INFinity (NINF) represent the numerical values -9.9E37 or 9.9E37, respectively. INF and NINF are only sent as device reponses.

- NAN

  Not A Number (NAN) represents the value 9.91E37. NAN is only sent as device response. This value is not defined. Possible causes are the division of zero by zero, the subtraction of infinite from infinite and the representation of missing values.

Example:

Setting command: `SENSe:FREQuency:STOP MAXimum`

Query: `SENSe:FREQuency:STOP?, Response: 3.5E9`

4.2.3 Boolean Parameters

Boolean parameters represent two states. The ON state (logically true) is represented by ON or a numerical value unequal to 0. The OFF state (logically untrue) is represented by OFF or the numerical value 0. The numerical values are provided as response for query.

Example

Setting command: `CALCulate:MARKer:STATe ON`

Query: `CALCulate:MARKer:STATe?, Response: 1`

4.2.4 Text

Text parameters observe the syntactic rules for key words, i.e. they can be entered using a short or long form. Like any parameter, they have to be separated from the header by a white space. In the case of a query, the short form of the text is provided.

Example

Setting command: `INPut:COUPling GROund`

Query: `INPut:COUPling?, Response: GRO`

4.2.5 Strings

Strings must always be entered in quotation marks (' or ").

Example

`SYSTem:LANGuage "SCPI"` or `SYSTem:LANGuage 'SCPI'`

4.2.6 Block Data

Block data are a transmission format which is suitable for the transmission of large amounts of data. A command using a block data parameter has the following structure:

Example

```
HEADer:HEADer #45168xxxxxxxx
```

ASCII character # introduces the data block. The next number indicates how many of the following digits describe the length of the data block. In the example the 4 following digits indicate the length to be 5168 bytes. The data bytes follow. During the transmission of these data bytes all end or other control signs are ignored until all bytes are transmitted.

4.3 Structure of a Program Message

A program message may consist of one or several commands. It is terminated by the program message terminator which is the NL (New Line) charcter for LAN and USB connections.

Several commands in a program message must be separated by a semicolon ";". If the next command belongs to a different command system, the semicolon is followed by a colon. A colon ":" at the beginning of a command marks the root node of the command tree.

Example:

```
CALL InstrWrite(analyzer, "SENSe:FREQuency:CENTer
100MHz;:INPut:ATTenuation 10")
```

This program message contains two commands. The first one is part of the SENSe command system and is used to determine the center frequency of the instrument. The second one is part of the INPut command system and sets the input signal attenuation.

If the successive commands belong to the same system, having one or several levels in common, the program message can be abbreviated. For that purpose, the second command after the semicolon starts with the level that lies below the common levels (see also Fig. 1-1). The colon following the semicolon must be omitted in this case.

Example:

```
CALL InstrWrite(analyzer, "SENSe:FREQuency:STARt
1E6;:SENSe:FREQuency:STOP 1E9")
```

This program message is represented in its full length and contains two commands separated from each other by the semicolon. Both commands are part of the SENSe command system, subsystem FREQuency, i.e. they have two common levels.

When abbreviating the program message, the second command begins with the level below SENSe:FREQuency. The colon after the semicolon is omitted. The abbreviated form of the program message reads as follows:

```
CALL InstrWrite(analyzer, "SENSe:FREQuency:STARt 1E6;STOP 1E9")
```

However, a new program message always begins with the complete path.

Example:

```
CALL InstrWrite(analyzer, "SENSe:FREQuency:STARt 1E6")
CALL InstrWrite(analyzer, "SENSe:FREQuency:STOP 1E9")
```

4.4 Responses to Queries

A query is defined for each setting command unless explicitly specified otherwise. It is formed by adding a question mark to the associated setting command. According to SCPI, the responses to queries are partly subject to stricter rules than in standard IEEE 488.2.

- The requested parameter is transmitted without header.

  Example

  `INPut:COUPling?`
  Response: `DC`

- Maximum values, minimum values and all further quantities, which are requested via a special text parameter are returned as numerical values.

  Example

  `SENSe:FREQuency:STOP? MAX`
  Response: `3.5E9`

- Numerical values are output without a unit. Physical quantities are referred to the basic units or to the units set using the Unit command.

  Example

  `SENSe:FREQuency:CENTer?`
  Response: `1E6` (for 1 MHz)

- Truth values <Boolean values> are returned as 0 (for OFF) and 1 (for ON).

  Example

  `SENSe:BANDwidth:AUTO?`
  Response: `1` (for ON)

- Text (character data) is returned in a short form.

  Example

  `SYSTem:COMMunicate:SERial:CONTrol:RTS?`
  Response `STAN` (for standard)

5 Command Sequence and Command Synchronization

What has been said above makes clear that all commands can potentially be carried out overlapping. In order to prevent an overlapping execution of commands, one of the commands `*OPC`, `*OPC?` or `*WAI` must be used. All three commands cause a certain action only to be carried out after the hardware has been set. By suitable programming, the controller can be forced to wait for the respective action to occur. For more information see Table 5-1.

Table 5-1: Synchronization using *OPC, *OPC? and *WAI

Command	Action	Programming the controller
`*OPC`	Sets the Operation Complete bit in the ESR after all previous commands have been executed.	– Setting bit 0 in the ESE – Setting bit 5 in the SRE – Waiting for service request (SRQ)
`*OPC?`	Stops command processing until 1 is returned. This is only the case after the Operation Complete bit has been set in the ESR. This bit indicates that the previous setting has been completed.	Sending `*OPC?` directly after the command whose processing should be terminated before other commands can be executed.
`*WAI`	Stops further command processing until all commands sent before *WAI have been executed.	Sending `*WAI` directly after the command whose processing should be terminated before other commands are executed.

For a couple of commands the synchronization to the end of command execution is mandatory in order to obtain the desired result. The affected commands require either more than one measurement in order to accomplish the desired instrument setting (e.g. auto range functions), or they require a longer period of time for execution. If a new command is received during execution of the corresponding function this may either lead to either to an aborted measurement or to incorrect measurement data.

The following list includes the commands, for which a synchronization via `*OPC`, `*OPC?` or `*WAI` is mandatory:

Table 5-2: Commands with mandatory synchronization (overlapping commands)

Command	Purpose
`INIT`	start measurement (sweep)
`INIT:CONT OFF`	Set to single sweep
`CALC:MARK:FUNC:xx?`	All Marker function queries

6 Remote Control – Commands

The following chapters provide a detailed description of all remote control commands currently available for the R&S FSH and its firmware options.

Each section describes the commands for one of the operating modes available in the R&S FSH, beginning with the description of common commands required to operate the instrument. The structure is based on that of the operating manual.

- Common Commands on page 28
- Remote Commands of the Spectrum Analyzer on page 31
- Remote Commands of the Network Analyzer Mode on page 80

Each section is subdivided into various tasks required to perform measurements with the R&S FSH, also based on the structure of the operating manual. Some commands like those for controlling markers or configuring the frequency axis are available for all operating modes. In that case you will find a list of these commands in the corresponding section. However, a detailed description is provided only in the CAT commands section.

Availability of commands

The spectrum and network analyzer modes are implemented in the basic unit. For the other modes, the corresponding options are required.

Following the remote control commands required to perform specific measurements, you will find a description of general commands used to set up and control basic instrument functions. These commands are independent of the operating mode. Therefore they are listed separately.

- File Management on page 89
- Making and Storing Screenshots on page 96
- Configuring the Instrument on page 102
- Remote Commands of the Status Reporting System on page 128

All chapters begin with a list of commands available in the context of that chapter. Following that list you will find a detailed description of all commands.

All individual descriptions contain:

- the complete notation and syntax of the command
- the description of the effects of the command
- a list of all parameters available for that command or the type of data the command returns in case of query commands
- an example of how a program message would look like
- the *RST value
- information on SCPI conformity

An alphabetical list of all available commands is provided at the end of this manual.

6.1 Common Commands

The common commands are taken from the IEEE 488.2 (IEC 625-2) standard. A particular command has the same effect on different devices. The headers of these commands consist of an asterisk "*" followed by three letters. Some of the common commands refer to the "Status Reporting System".

List of Common Commands

***CLS**

`CLEAR STATUS` sets the status byte (STB), the standard event register (ESR) and the EVENt part of the QUEStionable and the OPERation register to zero. The command does not alter the mask and transition parts of the registers. It clears the output buffer.

***ESE**

`EVENT STATUS ENABLE` sets the event status enable register to the value indicated. The query form `*ESE?` returns the contents of the event status enable register in decimal form.

Parameter

0 to 255

***ESR?**

`STANDARD EVENT STATUS QUERY` returns the contents of the event status register in decimal form (0 to 255) and subsequently sets the register to zero.

Parameter

0 to 255

***IDN?**

IDENTIFICATION QUERY queries the instrument identification.

You can change the format of the return values with the "SYSTem:FORMat:IDENt <IDNFormat>" command.**Return values**

<InstrumentName>,<SerialNumber/Model>,<FirmwareVersion>

Example for R&S FSH:
Rohde&Schwarz,FSH4,100005/024,1.50

***IST?**

INDIVIDUAL STATUS QUERY returns the contents of the IST flag in decimal form. The IST flag is the status bit which is sent during a parallel poll.

Parameter

0 | 1

***OPC**

OPERATION COMPLETE sets bit 0 in the event status register after all preceding commands have been executed. This bit can be used to initiate a service request.

***OPT?**

OPTION IDENTIFICATION QUERY queries the options included in the instrument and returns a list of the options installed. The options are separated from each other by means of commas.

Parameter

K<number> software options

For a list of all available options and their description refer to the CD-ROM.

Example

K40, K41, K42, K45

***RST**

RESET sets the instrument to a defined default status. The command essentially corresponds to pressing the PRESET key.

***SRE**

SERVICE REQUEST ENABLE sets the service request enable register to the indicated value. Bit 6 (MSS mask bit) remains 0. This command determines under which conditions a service request is generated. The query form *SRE? reads the contents of the service request enable register in decimal form. Bit 6 is always 0.

Parameter

0 to 255

***STB?**

READ STATUS BYTE QUERY reads out the contents of the status byte in decimal form.

***TRG**

TRIGGER initiates all actions in the currently active test screen expecting a trigger event. This command corresponds to INITiate[:IMMediate].

***TST?**

SELF TEST QUERY initiates the self test of the instrument and outputs an error code in decimal form.

Parameter

0 = no error

***WAI**

WAIT TO CONTINUE permits servicing of subsequent commands only after all preceding commands have been executed and all signals have settled.

6.2 Remote Commands of the Spectrum Analyzer

This section provides a detailed description of all remote control commands required to configure and perform measurements with the spectrum analyzer. These commands are available in spectrum analyzer mode only.

Contents

6.2.1 Configuring the Horizontal Axis

The following commands configure the horizontal (frequency) axis of the active display.

List of commands

- [SENSe:]FREQuency:CENTer <Frequency> (p. 31)
- [SENSe:]FREQuency:CENTer:STEP <StepSize> (p. 32)
- [SENSe:]FREQuency:CENTer:STEP:LINK <StepSizeCoupling> (p. 32)
- [SENSe:]FREQuency:INPut:MODE <InputMode> (p. 33)
- [SENSe:]FREQuency:MODE <OperationMode> (p. 33)
- [SENSe:]FREQuency:OFFSet <FrequencyOffset> (p. 34)
- [SENSe:]FREQuency:SPAN <Span> (p. 34)
- [SENSe:]FREQuency:SPAN:AUTO <State> (p. 34)
- [SENSe:]FREQuency:SPAN:FULL (p. 35)
- [SENSe:]FREQuency:STARt <StartFrequency> (p. 35)
- [SENSe:]FREQuency:STOP <StopFrequency> (p. 35)

[SENSe:]FREQuency:CENTer <Frequency>

This command defines the center frequency of the R&S FSH.

In spectrum analyzer mode, the command also defines the measuring frequency for time domain measurements (span = 0).

Parameter

<Frequency>
Numeric value in Hz.

The range depends on the operating mode and is specified in the data sheet.

Example

```
FREQ:CENT 100MHz
```
Defines a center frequency of 100 MHz.

Characteristics

*RST value: f_{max} /2 with f_{max} = maximum frequency
SCPI: conform

[SENSe:]FREQuency:CENTer:STEP <StepSize>

This command defines the center frequency step size.

Parameter

<StepSize>
Numeric value in Hz.

The range is from 1 Hz to f_{max}.

Example

```
FREQ:CENT:STEP 120MHz
```
Defines a CF step size of 120 MHz.

Characteristics

*RST value: – (AUTO 0.1*SPAN is switched on)
SCPI: conform

[SENSe:]FREQuency:CENTer:STEP:LINK <StepSizeCoupling>

This command couples and decouples the center frequency step size to the span.

For time domain measurements, the command couples the step size to the resolution bandwidth.

Parameter

<StepSizeCoupling>

DIVTen	Couples the step size to 10% of the span
OFF	deactivates coupling (manual input)

Example

```
FREQ:CENT:STEP:LINK DIVT
```
Couples the step size to 10% of the span.

Characteristics

*RST value: DIVTen
SCPI: device-specific

[SENSe:]FREQuency:INPut:MODE <InputMode>

This command selects the frequency mode. Select the Channel frequency mode only if you want to work with channel tables. In this case, the input of the center frequency is not a frequency value, but a channel number.

Parameter

<InputMode>

FREQuency	sets the frequency input mode to frequency input (Hz)
CHANnel	sets the frequency input mode to selection of a channel

Example

```
FREQ:INP:MODE CHAN
```
Sets the frequency mode to work with channel tables.

Characteristics

*RST value: FREQ
SCPI: device-specific

[SENSe:]FREQuency:MODE <OperationMode>

This command the measurement domain (frequency or time).

In the time domain (CW and FIXed), set the frequency with:

- [SENSe:]FREQuency:CENTer <Frequency>

In the frequency domain (SWEep), set it with

- [SENSe:]FREQuency:CENTer <Frequency>
- [SENSe:]FREQuency:SPAN <Span>
- [SENSe:]FREQuency:STARt <StartFrequency>
- [SENSe:]FREQuency:STOP <StopFrequency>

Parameter

<OperationMode>

CW	time domain (span = 0)
FIXed	time domain (span = 0)
SWEep	frequency domain (span > 0

Example

```
FREQ:MODE SWE
```
Activates frequency domain measurements.

Characteristics

*RST value: SWEep
SCPI: conform

[SENSe:]FREQuency:OFFSet <FrequencyOffset>

This command defines a frequency offset.

Parameter

<FrequencyOffset>
Numeric value in the range from -100 GHz to 100 GHz.

Example

```
FREQ:OFFS 1GHZ
```
Defines a frequency offset of 1 GHz.

Characteristics

*RST value: 0 Hz
SCPI: conform

[SENSe:]FREQuency:SPAN <Span>

This command defines the frequency span.

If you set a span of 0 Hz in spectrum mode, the R&S FSH starts a measurement in the time domain.

Parameter

<Span>
Numeric value in Hz. The value range is specified in the data sheet.

Example

```
FREQ:SPAN 10MHz
```
Defines a span of 10 MHz.

Characteristics

*RST value: f_{max} with f_{max} = maximum frequency
SCPI: conform

[SENSe:]FREQuency:SPAN:AUTO <State>

This command turns the automatic calculation of the ideal span on and off.

Parameter

<State>
ON | OFF

Example

```
FREQ:SPAN:AUTO ON
```
Turns automatic span determination on and off.

Characteristics

*RST value: OFF
SCPI: device-specific

[SENSe:]FREQuency:SPAN:FULL

This command restores the full span.

This command is an event and therefore has no query and no *RST value.

Example

```
FREQ:SPAN:FULL
```
Restores full span.

Characteristics

*RST value: –
SCPI: conform

[SENSe:]FREQuency:STARt <StartFrequency>

This command defines the start frequency for measurements in the frequency domain (span > 0).

Parameter

<StartFrequency>
Numeric value in Hz.

The range depends on the operating mode and is specified in the datasheet.

Example

```
FREQ:STAR 20MHz
```
Defines a start frequency of 20 MHz.

Characteristics

*RST value: 0
SCPI: conform

[SENSe:]FREQuency:STOP <StopFrequency>

This command defines the stop frequency for measurements in the frequency domain (span > 0).

Parameter

<StopFrequency>
Numeric value in Hz.

The range depends on the operating mode and is specified in the datasheet.

Example

```
FREQ:STOP 2000MHz
```
Defines a stop frequency of 2 GHz

Characteristics

*RST value: f_{max}
SCPI: conform

6.2.2 Configuring the Vertical Axis

The following commands configure the vertical (level) axis and level parameters of the active display.

The suffix <t> at TRACe is irrelevant for these commands.

The suffix <c> at TRANsducer selects the primary or secondary transducer. The range is <1…2>.

List of commands

DISPlay[:WINDow]:TRACe<t>:Y[:SCALe]:ADJust

This command automatically scales the vertical axis for optimum display results.

This command is an event and therefore has no query and no *RST value.

Example

```
DISP:TRAC:Y:ADJ
```
Adjusts the y-axis.

Characteristics

*RST value: -
SCPI: device-specific

DISPlay[:WINDow]:TRACe<t>:Y:SPACing <Scaling>

This command selects the scaling of the vertical axis.

Parameter

<Scaling>

LOGarithmic	logarithmic scaling
LINear	linear scaling (%)

Example

```
DISP:TRAC:Y:SPAC LIN
```
Selects linear scaling of the level axis.

Characteristics

*RST value: LOGarithmic
SCPI: conform

DISPlay[:WINDow]:TRACe<t>:Y[:SCALe] <DisplayRange>

This command defines the display range of the vertical axis.

Note that you have to set a logarithmic scaling before you can use this command with `DISPlay[:WINDow]:TRACe<t>:Y:SPACing`. For a linear scale, you can not modify the display range as it is fixed.

Parameter

<DisplayRange>
numeric value in the range from 10 dB to 200 dB

Example

```
DISP:TRAC:Y 110dB
```
Sets the display range to 110 dB.

Characteristics

*RST value: 100dB
SCPI: device–specific

DISPlay[:WINDow]:TRACe<t>:Y[:SCALe]:RLEVel <ReferenceLevel>

This command defines the reference level.

With a reference level offset ≠ 0, the value range of the reference level is modified by the offset.

Parameter

<ReferenceLevel>
numeric value that sets the reference level; the unit depends on `UNIT:POWer`.

The available value range is specified in the data sheet.

Example

```
DISP:TRAC:Y:RLEV -60dBm
```
Sets the reference level to -60 dBm.

Characteristics

*RST value: –20dBm
SCPI: conform

DISPlay[:WINDow]:TRACe<t>:Y[:SCALe]:RLEVel:OFFSet <Offset>

This command defines a reference level offset.

Parameter

<Offset>
numeric value (dB) that sets the reference level offset

The available value range is from -200dB to 200dB.

Example

```
DISP:TRAC:Y:RLEV:OFFS -10dB
```

Characteristics

*RST value: 0dB
SCPI: conform

DISPlay[:WINDow]:TRACe<t>:Y[:SCALe]:RPOSition <ReferencePosition>

This command defines the position of the reference level on the display grid.

First, you have to set a logarithmic scale for the vertical axis with `DISPlay[:WINDow]:TRACe<t>:Y:SPACing`. For a linear scale, you can not modify the reference position.

Parameter

<ReferencePosition>
numeric value (%) that defines the reference position

The available value range is from 1 to 10.

Example

```
DISP:TRAC:Y:RPOS 5
```

Sets the reference position to the 5th grid line.

Characteristics

*RST value:10
SCPI: conform

INPut:ATTenuation <Attenuation>

This command defines the input attenuation.

In spectrum mode, the attenuation is coupled to the reference level. If you set the attenuation independently, the R&S FSH turns off this coupling.

The R&S FSH adjusts the reference level if it can not be set for the current RF attenuation.

Parameter

<Attenuation>
Numeric value in in the range from 0 dB to 40 dB in 5 dB steps.

Example

```
INP:ATT 30dB
```
Defines an attenuation of 30 dB and deactivates coupling to the reference level.

Characteristics

*RST value: 0 dB (AUTO is ON)
SCPI: conform

INPut:ATTenuation:MODE <AttenuationMode>

This command selects the attenuation mode.

Parameter

<AttenuationMode>

LDIStortion	Auto low distortion
LNOise	Auto low noise

Example

```
INP:ATT:MODE LNO
```
Sets the attenuation mode to Auto Low Noise.

Characteristics

*RST value: LNOise
SCPI: device-specific

INPut:ATTenuation:AUTO <State>

This command couples and decouples input attenuation to the reference level.

Parameter

<State>
ON | OFF

Example

```
INP:ATT:AUTO ON
```
Couples the attenuation set on the attenuator to the reference level.

Characteristics

*RST value: ON
SCPI: conform

INPut:GAIN:STATe <State>

This command turns the preamplifier on and off.

Parameter

<State>
ON | OFF

Example

```
INP:GAIN:STAT ON
```
Activates the preamplifier

Characteristics

*RST value: OFF
SCPI: conform

INPut:IMPedance <Impedance>

This command selects the nominal input impedance. The set impedance is taken into account in all level indications of results.

The setting 75 Ω should be selected, if the 50 Ω input impedance is transformed to a higher impedance using a 75 Ω adapter of the RAZ type (= 25 Ω in series to the input impedance of the instrument). The correction value in this case is 1.76 dB = 10 log (75Ω / 50Ω).

Parameter

<Impedance>
50 | 75

Example

```
INP:IMP 75
```
Sets the input impedance to 75 Ohm.

Characteristics

*RST value: 50 Ω
SCPI: conform

[SENSe:]CORRection:TRANsducer<c>[:STATe] <State>

This command turns a transducer factor on and off.

Before turning it on, you have to select a transducer factor with `[SENSe:]CORRection:TRANsducer<c>:SELect`.

Parameter

<State>
ON | OFF

Example

```
CORR:TRAN1 ON
```
Activates the primary transducer

Characteristics

*RST value: OFF
SCPI: device-specific

[SENSe:]CORRection:TRANsducer<c>:SELect <TransducerFactor>

This command selects a transducer factor.

If <name> does not exist yet, a new transducer factor is created.

The suffix<1...2> specifies the primary or secondary transducer.

Parameter

<Transducerfactor>
string containing the file name of the transducer factor.

If the file does not exist, the R&S FSH creates a new transducer factor.

Example

```
CORR:TRAN2:SEL 'FSH-Z38.sectrd'
```
Selects the FSH-Z38 secondary transducer factor.

Characteristics

*RST value: -
SCPI: device-specific

[SENSe:]CORRection:TRANsducer<c>:UNIT?

This command queries the unit of the current transducer factor.

Example

```
CORR:TRAN2:UNIT?
```
Queries the unit of the primary transducer.

Characteristics

*RST value: -
SCPI: device-specific

UNIT:POWer <Unit>

This command selects the unit of the vertical axis.

The availability of units depends on the operating mode and type of measurement.

Parameter

<Unit>
DBM | DBUV | DBMV | V | W | DUVM | DUAM | V_M | W_M2

Note that the availability of units depends on the operating mode.

Example

```
UNIT:POW DBUV
```
Sets the power unit to dBµV.

Characteristics

*RST value: DBM
SCPI: conform

6.2.3 Setting the Bandwidths

The following commands configure the filter bandwidths of the R&S FSH. Note that both groups of commands (BANDwidth and BWIDth) are the same.

List of commands

[SENSe:]BANDwidth|BWIDth[:RESolution] <ResBW>

This command defines the resolution bandwidth.

Parameter

<ResBW>
Numeric value in Hz.

The available value range is from 1 Hz to 3 MHz in 1 - 3 - 5 steps. In addition, you can select a 200 kHz bandwidth.

Example

```
BAND 100 kHz
```
Sets the resolution bandwidth to 100 kHz

Characteristics

*RST value: - (AUTO is set to ON)
SCPI: conform

[SENSe:]BANDwidth|BWIDth[:RESolution]:AUTO <State>

This command couples and decouples the resolution bandwidth to the span.

Parameter

<State>
ON | OFF

Example

```
BAND:AUTO OFF
```
Decouples the resolution bandwidth from the span.

Characteristics

*RST value: ON
SCPI: conform

[SENSe:]BANDwidth|BWIDth:VIDeo <VideoBW>

This command defines the video bandwidth.

Parameter

<VideoBW>
Numeric value in Hz.

The available value range is from 1 Hz to 3 MHz in 1 - 3 steps.

Example

```
BAND:VID 10kHz
```
Sets the video bandwidth to 10 kHz.

Characteristics

*RST value: - (AUTO is set to ON)
SCPI: conform

[SENSe:]BANDwidth|BWIDth:VIDeo:AUTO <State>

This command couples and decouples the video bandwidth to the resolution bandwidth.

Parameter

<State>
ON | OFF

Example

```
BAND:VID:AUTO OFF
```
Turns off video bandwidth coupling.

Characteristics

*RST value: ON
SCPI: conform

6.2.4 Performing and Triggering Measurements

The following commands control the actual measurement process, including trigger functionality.

6.2.4.1 Performing the Measurement

The following commands initialize a measurement and set up the sweep.

List of commands

ABORt

This command aborts the current measurement and resets the trigger system.

This command is an event and therefore has no query and no *RST value.

Example

```
ABOR;
INIT:IMM
```
Aborts a measurement and starts a new one.

Characteristics

RST value: –
SCPI: conform

INITiate[:IMMediate]

The command initiates a new measurement sequence.

With sweep count > 0 or average count > 0, this means a restart of the indicated number of measurements. With trace functions MAXHold, MINHold and AVERage, the previous results are reset on restarting the measurement.

In single sweep mode, synchronization to the end of the indicated number of measurements can be achieved with the command *OPC, *OPC? or *WAI. In continuous–sweep mode, synchronization to the sweep end is not possible since the overall measurement never ends.

This command is an event and therefore has no query and no *RST value.

Example

```
INIT:CONT OFF
DISP:WIND:TRAC:MODE AVER
```
Turns single sweep mode and trace averaging on.

```
INIT;*WAI
```
Starts the measurement and waits for the end of the sweep.

Characteristics

*RST value: –
SCPI: conform

INITiate:CONTinuous <SweepMode>

This command selects the sweep mode.

Parameter

<SweepMode>

ON	Continuous sweep
OFF	Single sweep

Example

```
INIT:CONT OFF
```
Turns on single sweep mode.

Characteristics

*RST value: ON
SCPI: conform

[SENSe:]SWEep:COUNt <SweepCount>

This command defines the number of sweeps included in a single sweep. It also defines the number of sweeps the R&S FSH uses to average traces or calculate maximum values.

The R&S FSH performs one sweep for sweep count 0 or 1.

Parameter

<SweepCount>
0 to 999

Example

```
SWE:COUN 64
```
Defines a sweep count of 64 sweeps.

```
INIT:CONT OFF
INIT;*WAI
```
Turns on single sweep mode, starts the sweep and waits for its end.

Characteristics

*RST value: 1
SCPI: conform

[SENSe:]SWEep:POINts?

This command queries the number of measurement points in a single sweep.

This command is a query and therefore has no *RST value.

Return value

Number of sweep points.

Example

```
SWE:POIN?
```
Returns the number of sweep points.

Characteristics

*RST value: –
SCPI: conform

[SENSe:]SWEep:TIME <SweepTime>

This command defines the sweep time.

If you set a sweep time in spectrum mode with this command, the R&S FSH decouples the sweep time from the span and the resolution and video bandwidths.

Parameter

<SweepTime>
Numeric value in seconds.

The available value range is specified in the datasheet.

Example

```
SWE:TIME 10s
```
Sets the sweep time to 10 s

Characteristics

*RST value: - (AUTO is set to ON)
SCPI: conform

[SENSe:]SWEep:TIME:AUTO <State>

This command couples and decouples the sweep time to the span and the resolution and video bandwidths.

Parameter

<State>
ON | OFF

Example

```
SWE:TIME:AUTO ON
```
Switches on the coupling to frequency span and bandwidths.

Characteristics

*RST value: ON
SCPI: conform

6.2.4.2 Triggering Measurements

The following commands set up trigger conditions if you are using a trigger for the measurement.

List of commands

- [SENSe:]SWEep:EGATe <State> (p. 47)
- [SENSe:]SWEep:EGATe:HOLDoff <GateDelay> (p. 47)
- [SENSe:]SWEep:EGATe:LENGth <GateLength> (p. 48)
- [SENSe:]SWEep:EGATe:TIME <SweepTime> (p. 48)
- TRIGger[:SEQuence]:HOLDoff[:TIME] <TriggerDelay> (p. 48)
- TRIGger[:SEQuence]:LEVel:VIDeo <TriggerLevel> (p. 49)
- TRIGger[:SEQuence]:SLOPe <TriggerSlope> (p. 49)
- TRIGger[:SEQuence]:SOURce <TriggerSource> (p. 50)

[SENSe:]SWEep:EGATe <State>

This command turns a gated trigger on and off.

Parameter

<State>
ON | OFF

Example

```
SWE:EGAT ON
```
Activates the gated trigger.

Characteristics

*RST value: off
SCPI: device-specific

[SENSe:]SWEep:EGATe:HOLDoff <GateDelay>

This command defines the length of the gate delay.

Parameter

<GateDelay>
Numeric value in the range from 0 s to 100 s.

Example

```
SWE:EGAT:HOLD 2.5
```
Sets a gate delay of 2.5 seconds.

Characteristics

*RST value: 0 s
SCPI: device-specific

[SENSe:]SWEep:EGATe:LENGth <GateLength>

This command defines the gate length.

Parameter

<GateLength>
Numeric value in the range from 10 µs to 100 s.

Example

```
SWE:EGAT:LENG 2.5
```
Sets a gate length of 2.5 seconds.

Characteristics

*RST value: 400 µs
SCPI: device-specific

[SENSe:]SWEep:EGATe:TIME <SweepTime>

This command defines the sweep time for the gated trigger.

Parameter

<SweepTime>
Numeric value in seconds.

Example

```
SWE:GATE:TIME 4ms
```
Sets a sweep time of 4 ms for the gated trigger.

Characteristics

*RST value: 400 µs
SCPI: device-specific

TRIGger[:SEQuence]:HOLDoff[:TIME] <TriggerDelay>

This command defines the length of the trigger delay.

Parameter

<TriggerDelay>
Numeric value in the range from 0 s to 100 s.

Example

```
TRIG:HOLD 500us
```
Sets the trigger delay to 500 µs.

Characteristics

*RST value: 0 s
SCPI: conform

TRIGger[:SEQuence]:LEVel:VIDeo <TriggerLevel>

This command defines the level of the video trigger.

Video trigger is available for time domain measurements (span = 0).

Parameter

<TriggerLevel>
Numeric value in the range from 0 % to 100 %.

Example

```
TRIG:LEV:VID 50PCT
```
Sets the trigger level to 50%.

Characteristics

*RST value: 50 PCT
SCPI: device–specific

TRIGger[:SEQuence]:SLOPe <TriggerSlope>

This command selects the slope of the trigger signal.

The trigger slope applies to all trigger sources.

Parameter

<TriggerSlope>
POSitive | NEGative

Example

```
TRIG:SLOP NEG
```
Characteristics

*RST value: POSitive
SCPI: conform

TRIGger[:SEQuence]:SOURce <TriggerSource>

This command selects the trigger source.

Parameter

<TriggerSource>

IMMediate	Free Run
EXTernal	External trigger
VIDeo	Video trigger

For more information see R&S FSH operating manual chapter "Setting the Sweep"

Example

```
TRIG:SOUR EXT
```
Selects the external trigger input as source of the trigger signal

Characteristics

*RST value: IMMediate
SCPI: conform

6.2.5 Working with Traces

The following commands set up the trace and the various functions associated with it, e.g. trace mathematics or the selection of the detector.

The suffix <t> is in the range {1..2} and selects the number of the trace.

List of commands

CALCulate:MATH[:EXPRession][:DEFine] <Expression>

This command defines the mathematical expression for relating traces to trace 1.

You have to activate trace mathematics with `CALCulate:MATH<t>:STATe <State>` first.

Parameter

<Expression>

(IMPLied - memory) Subtracts the trace in memory from the current trace.

(memory - IMPLied) Subtracts the current trace from the trace in memory.

Example

```
CALC:MATH (MTRACE-TRACE)
```
Selects the subtraction of the current trace from trace in the memory.

Characteristics

*RST value: –
SCPI: conform

CALCulate:MATH<t>:COPY:MEMory

This command stores the selected trace into the memory trace of the R&S FSH.

This command is an event and therefore has no query and no *RST value.

Example

```
CALC:MATH:COPY:MEM
```
Copies the trace into the memory.

Characteristics

*RST value: -
SCPI: device-specific

CALCulate:MATH:STATe <State>

This command turns trace mathematics on and off.

Parameter

<State>
ON | OFF

Example

```
CALC:MATH:STAT ON
```
Switches on the trace mathematics.

Characteristics

*RST value: OFF
SCPI: conform

DISPlay[:WINDow]:TRACe<t>[:STATe] <State>

This command turns a trace on and off.

Parameter

<State>
ON | OFF

Example

```
DISP:TRAC2 ON
```
Turns the trace on.

Characteristics

*RST value: ON for TRACe1, OFF for TRACe2
SCPI: conform

DISPlay[:WINDow]:TRACe<t>:MEMory[:STATe] <State>

This command turns the memory trace on and off.

Parameter

<State>
ON | OFF

Example

```
DISP:TRAC:MEM ON
```
Activates the memory trace.

Characteristics

*RST value: OFF
SCPI: device-specific

DISPlay[:WINDow]:TRACe<t>:MODE <TraceMode>

This command selects the trace mode.

If you are using the average, max hold or min hold trace mode, you can set the number of measurements with [SENSe:]SWEep:COUNt <SweepCount>. Note that synchronization to the end of the average count is possible only in single sweep mode.

Parameter

<TraceMode>
AVERage | MAXHold | MINHold | VIEW | WRITe

You can turn off the trace with `DISPlay[:WINDow]:TRACe<t>[:STATe] <State>`.

For more information see the operating manual, chapter "Trace Mode".

Example

```
SWE:CONT OFF
SWE:COUN 16
```
Turn on single sweep mode and sets the number of measurements to 16.

```
DISP:TRAC:MODE MAXH
```
Activates MAXHold mode for the trace.

```
INIT;*WAI
```
Starts the measurement and waits for the end of the 16 sweeps.

Characteristics

*RST value: WRITe
SCPI: device–specific

FORMat:BORDer <ByteOrder>

This command selects the format of binary data.

Parameters

<ByteOrder>

SWAPped	The least significant byte is transferred first (little endian)
NORMal	The most significant byte is transferred first (big endian)

Example

```
FORM:BORD NORM
```
Changes the byte order to normal mode

Characteristics

*RST value SWAPped
SCPI: conform

[SENSe:]DETector<t>[:FUNCtion] <Detector>

This command selects the detector.

Parameter

<Detector>
APEak | NEGative | POSitive | SAMPle | RMS

For more information see the operating manual, chapter "Detectors".

Example

```
DET POS
```
Sets the detector to "positive peak".

Characteristics

*RST value: POS
SCPI: conform

[SENSe:]DETector<t>[:FUNCtion]:AUTO <State>

This command couples and decouples the detector to the trace mode.

Parameter

<State>
ON | OFF

Example

```
DET:AUTO OFF
```
Turns off automatic detector selection.

Characteristics

*RST value: ON
SCPI: conform

TRACe[:DATA]? <Destination>

This command queries the trace data of the current measurement.

It also transfers data from a file to a particular trace.

With `FORMat[:DATA] <DataFormat>` command, you can set the data format.

Parameter

<Destination>

TRACe1	writes to or reads out trace 1
TRACe2	writes to or reads out trace 2
LIST	reads out peak list of SEM measurements

Return value

The R&S FSH returns 631 values. Each value corresponds to one pixel of a trace.

The unit depends on the measurement and the unit you have set with `UNIT:POWer <Unit>`.

Note:
If you use the auto peak detector, the command reads out positive peak values only.

Example

```
TRAC:DATA? TRACE1
```
Reads out the data for trace 1

Characteristics

*RST value: -
SCPI: conform

FORMat[:DATA] <DataFormat>

This command selects the data format that is used for transmission of trace data from the R&S FSH to the controlling computer.

Note that the command has no effect for data that you send to the R&S FSH. The R&S FSH automatically recognizes the data it receives, regardless of the format.

Parameter

<DataFormat>

ASCii	ASCii format, separated by commas
REAL,32	32-bit IEEE 754 floating-point numbers in the "definite length block format"

In REAL,32 format, a string of return values would look like:
`#42524<value 1><value 2>...<value n>`
with

#4	Number of digits of the following number of data bytes (= 4 in this example)
2524	Number of following data bytes (2524, corresponds to the 631 sweep points of the R&S FSH.
<value>	4-byte floating point value

Example

```
FORM ASC
```
Selects the ASCii data format.

Characteristics

*RST value: ASCii
SCPI: conform

6.2.6 Using Markers

- Markers and Delta Markers on page 56
- Marker Functions on page 64

6.2.6.1 Markers and Delta Markers

The following commands are for setting and controlling markers and deltamarkers.

In spectrum mode mode, the suffix <n> at CALCulate selects the trace.

The suffix <m> at MARKer is in the range {1..6} and selects the marker or deltamarker.

List of commands

CALCulate<n>:DELTamarker<m>[:STATe] <State>

This command turns delta markers on and off.

If you set the suffix at DELTamarker to 1, or use no suffix, the R&S FSH interprets this as delta marker 2 because the first marker has to be a normal marker. If more than one normal marker (2 to 6) are already active, the command turns these marker into delta markers. If no delta marker is active yet, the command activates the delta marker and positions it on the trace maximum.

Parameter

<State>
ON | OFF

Example

```
CALC:DELT3 ON
```
Turns delta marker 3 on or turn marker 3 into a delta marker.

Characteristics

RST value: OFF
SCPI: device–specific

CALCulate<n>:DELTamarker<m>:AOFF

This command turns off all active delta markers.

This command is an event and therefore has no query and no *RST value.

Example

```
CALC:DELT:AOFF
```
Turns off all delta markers.

Characteristics

RST value: –
SCPI: device–specific

CALCulate<n>:DELTamarker<m>:MAXimum[:PEAK]

This command positions a delta marker on the current trace maximum.

If necessary, the corresponding delta marker is activated first.

This command is an event and therefore has no *RST value and no query.

Example

```
CALC:DELT3:MAX
```
Moves delta marker 3 to the maximum peak.

Characteristics

RST value: –
SCPI: device–specific

CALCulate<n>:DELTamarker<m>:MAXimum:NEXT

This command positions a delta marker on the next smaller trace maximum.

If necessary, the corresponding delta marker is activated first.

This command is an event and therefore has no *RST value and no query.

Example

```
CALC:DELT2:MAX:NEXT
```
Moves delta marker 2 to the next smaller maximum peak.

Characteristics

RST value: –
SCPI: device–specific

CALCulate<n>:DELTamarker<m>:MINimum[:PEAK]

This command positions a delta marker on the current trace minimum.

If necessary, the corresponding delta marker is activated first.

This command is an event and therefore has no *RST value and no query.

Example

```
CALC:DELT3:MIN
```
Moves delta marker 3 to the trace minimum.

Characteristics

RST value: –
SCPI: device–specific

CALCulate<n>:DELTamarker<m>:X <Coordinate>

This command positions a delta marker on a particular coordinate on the horizontal axis.

Note that it is possible to place the marker outside the visible trace. In that case, this value is invalid.

If necessary, the corresponding delta marker is activated first.

Parameter

<Coordinate>
Numeric value that indicates the coordinate on the horizontal axis.

The range corresponds to the maximum span.

The unit depends on the measurement, e.g. Hz for measurements in the frequency domain and seconds for measurements in the time domain.

Example

```
CALC:DELT:MOD REL
```
Delta marker positions are relative to marker 1.

```
CALC:DELT2:X 10.7MHz
```
Positions delta marker 2 10.7 MHz to the right of marker 1.

```
CALC:DELT2:X?
CALC:DELT2:X:REL?
```
Queries the absolute and relative position of delta marker 2.

Characteristics

RST value: –
SCPI: device–specific

CALCulate<n>:DELTamarker<m>:X:RELative <Distance>

This command positions a delta marker on a position relative to the reference marker.

If necessary, the corresponding delta marker is activated first.

Parameter

<Distance>
Numeric value that defines the distance of the marker to the reference marker

The range depends on the current scaling of the horizontal axis.

The unit depends on the measurement, e.g. Hz for measurements in the frequency domain and seconds for measurements in the time domain.

Example

```
CALC:DELT3:X:REL 5 kHz
```
Sets the delta marker at a distance of 5 kHz to the reference position.

Characteristics

RST value: –
SCPI: device–specific

CALCulate<n>:DELTamarker<m>:Y?

This command queries the vertical position of a delta marker. The result is always a relative value in relation to marker 1.

If necessary, the corresponding delta marker is activated first.

To get a valid result, you have to perform a complete sweep with synchronization to the sweep end between activating the delta marker and reading out the result. This is only possible in single sweep mode.

Return value

<MarkerPosition>

In spectrum analyzer mode, the unit depends on the unit you have set and the scaling of the vertical axis.

Parameter or measuring functions	Output unit
DBM | DBPW | DBUV | DBMV | DBUA	dB (lin/log)
WATT | VOLT | AMPere	dB (lin), % (log)

Example

```
INIT:CONT OFF
CALC:DELT2 ON
```
Turns on single sweep mode and delta marker 2.

```
INIT;*WAI
```
Starts a sweep and waits for its end.

```
CALC:DELT2:Y?
```
Queries the position of delta marker 2.

Characteristics

RST value: –
SCPI: device–specific

CALCulate<n>:MARKer<m>[:STATe] <State>

This command turns markers on and off.

If you do not use a suffix at MARKer, marker 1 is selected. If one or more delta markers (2 to 6) are already active, the command turns these delta markers into normal markers.

Parameter

<State>
ON | OFF

Example

```
CALC:MARK3 ON
```
Turns on marker 3.

Characteristics

*RST value: OFF
SCPI: device–specific

CALCulate<n>:MARKer<m>:AOFF

This command turns off all active markers, delta markers and active marker measurement functions.

This command is an event and therefore has no query and no *RST value.

Example

```
CALC:MARK:AOFF
```
Switches off all markers.

Characteristics

*RST value: –
SCPI: device–specific

CALCulate<n>:MARKer<m>:MAXimum[:PEAK]

This command positions a marker on the current trace maximum.

If necessary, the corresponding marker is activated first.

This command is an event and therefore has no *RST value and no query.

Example

CALC:MARK2:MAX
Moves marker 2 to the maximum peak.

Characteristics

*RST value: –
SCPI: device–specific

CALCulate<n>:MARKer<m>:MAXimum:NEXT

This command positions a marker on the next smaller trace maximum.

If necessary, the corresponding marker is activated first.

This command is an event and therefore has no *RST value and no query.

Example

CALC:MARK2:MAX:NEXT
Moves marker 2 to the next smaller maximum peak.

Characteristics

*RST value: –
SCPI: device–specific

CALCulate<n>:MARKer<m>:MINimum[:PEAK]

This command positions a marker on the current trace minimum.

If necessary, the corresponding marker is activated first.

This command is an event and therefore has no *RST value and no query.

Example

CALC:MARK2:MIN
Moves marker 2 to the trace minimum.

Characteristics

*RST value: –
SCPI: device–specific

CALCulate<n>:MARKer<m>:X <Coordinate>

This command positions a marker on a particular coordinate on the horizontal axis.

If one or more delta markers (2 to 6) are already active, the command turns these delta markers into normal markers.

Note that it is possible to place the marker outside the visible trace. In that case, this value is invalid.

If necessary, the corresponding delta marker is activated first.

Parameter

<Coordinate>
Numeric value indicating the coordinate on the horizontal axis.

The range corresponds to the maximum span.

The unit in spectrum analyzer mode depends on the measurement, e.g. Hz for measurements in the frequency domain and seconds for measurements in the time domain.

Example

```
CALC:MARK2:X 10.7MHz
```
Positions marker 2 to frequency 10.7 MHz.

Characteristics

*RST value: –
SCPI: device–specific

CALCulate<n>:MARKer<m>:X:SLIMits[:STATe] <State>

This command turns marker search limits on and off.

If you perform a measurement in zero span, this command, this command limits the evaluation range on the trace.

The numeric suffix at MARKer is irrelevant.

Parameter

<State>
ON | OFF

Example

See `CALCulate<n>:MARKer<m>:X:SLIMits:RIGHt <Searchlimit>`.

Characteristics

*RST value: OFF
SCPI: device–specific

CALCulate<n>:MARKer<m>:X:SLIMits:LEFT <SearchLimit>

This command defines the left limit of the marker search range.

To use the command, you first have to turn on search limits with `CALCulate<n>:MARKer<m>:X:SLIMits[:STATe] <State>`.

If you perform a measurement in zero span, this command, this command limits the evaluation range on the trace.

The numeric suffix at MARKer is irrelevant.

Parameter

<SearchLimit>
Numeric value that sets the left marker search limit.

The value range corresponds to the maximum span.

The unit in spectrum analyzer mode depends on the measurement, e.g. Hz for measurements in the frequency domain and seconds for measurements in the time domain.

Example

See CALCulate<n>:MARKer<m>:X:SLIMits:RIGHt <Searchlimit>.

Characteristics

*RST value: – (is set to the left diagram border when switching on search limits)
SCPI: device–specific

CALCulate<n>:MARKer<m>:X:SLIMits:RIGHt <Searchlimit>

This command defines the right limit of the marker search range.

To use the command, you first have to turn on search limits with CALCulate<n>:MARKer<m>:X:SLIMits[:STATe] <State>.

If you perform a measurement in zero span, this command, this command limits the evaluation range on the trace.

The numeric suffix at MARKer is irrelevant.

Parameter

<SearchLimit>
Numeric value that sets the right marker search limit.

The value range corresponds to the maximum span.

The unit depends on the measurement, e.g. Hz for measurements in the frequency domain and seconds for measurements in the time domain.

Example

```
CALC:MARK:X:SLIM ON
CALC:MARK:X:SLIM:LEFT 10MHz
CALC:MARK:X:SLIM:RIGH 100MHz
```
Turns search limits on and defines a search range from 10 MHz to 100 MHz.

Characteristics

*RST value: – (is set to the right diagram border when switching on search limits)
SCPI: device–specific

CALCulate<n>:MARKer<m>:Y?

This command queries the absolute vertical position of a marker.

If necessary, the corresponding marker is activated first.

To get a valid result, you have to perform a complete sweep with synchronization to the sweep end between activating the delta marker and reading out the result. This is only possible in single sweep mode.

Return values

<MarkerPosition>
numeric value of the marker position

In spectrum mode, the unit depends on `UNIT:POWer`.

Example

```
INIT:CONT OFF
CALC:MARK2 ON
```
Turns on single sweep mode and marker 2.

```
INIT;*WAI
```
Starts a sweep and waits for the end.

```
CALC:MARK2:Y?
```
Queries the position of marker 2.

Characteristics

*RST value: –
SCPI: device–specific

6.2.6.2 Marker Functions

The following commands perform various kinds of analysis at the marker position.

The suffix <m> at MARKer is irrelevant, except where noted.

List of commands

CALCulate:MARKer<m>:COUNt:FREQuency?

This command performs a frequency measurement at the marker position and returns the result.

To get a valid result, you have to perform a complete sweep with synchronization to the sweep end to make sure that the R&S FSH actually reaches the frequency you want to measure. This is only possible in single sweep mode.

Before you can use the command, you have to turn on the frequency counter with `CALCulate:MARKer<m>:COUNt[:STATe] <State>`.

Example

See `CALCulate:MARKer<m>:COUNt[:STATe] <State>`.

Characteristics

*RST value: –
SCPI: device–specific

CALCulate:MARKer<m>:COUNt[:STATe] <State>

This command turns the frequency counter at the marker position on and off.

You can read out the result with `CALCulate:MARKer<m>:COUNt:FREQuency?`.

Frequency counting is possible only for one marker at a time. If it is activated for another marker, it is automatically deactivated for the previous marker.

To get a valid result, you have to perform a complete sweep with synchronization to the sweep end to make sure that the R&S FSH actually reaches the frequency you want to measure. This is only possible in single sweep mode.

The suffix <m> selects the marker.

Parameter

<State>
ON | OFF

Example

```
INIT:CONT OFF
CALC:MARK ON
```
Turns on single sweep mode and marker 1.

```
CALC:MARK:COUN ON
```
Turns on the frequency counter for marker 1.

```
INIT;*WAI
CALC:MARK:COUN:FREQ?
```
Performs a measurement and queries the results of the frequency counter.

Characteristics

*RST value: OFF
SCPI: device–specific

CALCulate:MARKer<m>:FREQuency:MODE <InputMode>

This command selects the marker frequency display mode.

Parameter

<InputMode>

FREQuency	sets the marker frequency mode to frequency input (Hz)
CHANnel	sets the marker frequency mode to channel input (channel number)

Example

```
CALC:MARK:FREQ:MODE FREQ
```
Selects the frequency display mode.

Characteristics

*RST value: FREQ
SCPI: device-specific

CALCulate:MARKer<m>:FUNCtion:CENTer

This command matches the center frequency to the frequency of a marker

If you use a delta marker, the R&S FSH turns it into a normal marker.

The suffix <m> selects the marker.

This command is an event and therefore has no *RST value and no query.

Example

```
CALC:MARK1:FUNC:CENT
```
Matches the center frequency to the frequency of marker 1.

Characteristics

*RST value: -
SCPI: device-specific

CALCulate:MARKer<m>:FUNCtion:DEModulation[:STATe] <State>

This command turns the audio demodulator on and off when the measurement hits a marker position.

With a span greater than 0, you can define a hold time at the marker position with `CALCulate:MARKer<m>:FUNCtion:DEModulation:HOLDoff <Time>`.

In zero span the demodulation is on permanently.

Parameter

<State>
ON | OFF

Example

```
CALC:MARK3:FUNC:DEM ON
```
Switches on the demodulation for marker 3.

Characteristics

*RST value: OFF
SCPI: device–specific

CALCulate:MARKer<m>:FUNCtion:DEModulation:HOLDoff <Time>

This command defines the hold time at the marker position for the demodulation with span > 0.

Parameter

<Time>
Numeric value in the range from 10 ms to 500 s.

Example:

```
CALC:MARK:FUNC:DEM:HOLD 3s
```
Sets a hold time of 3 seconds.

Characteristics:

*RST value: – (DEModulation is set to OFF)
SCPI: device–specific

CALCulate:MARKer<m>:FUNCtion:DEModulation:SELect <Demodulation>

This command selects the type of demodulation type for the audio demodulator.

Parameter

<Demodulation>
AM | FM

Example

```
CALC:MARK:FUNC:DEM:SEL FM
```
Selects FM demodulation.

Characteristics

*RST value: AM
SCPI: device–specific

CALCulate:MARKer<m>:FUNCtion:NDBDown <Distance>

This command defines the distance of the n dB down markers to the reference marker.

Parameter

<Distance>
Distance of the temporary markers to the reference marker in dB.

Example

See CALCulate:MARKer<m>:FUNCtion:NDBDown:STATe <State>

Characteristics

*RST value: 3 dB
SCPI: device-specific

CALCulate:MARKer<m>:FUNCtion:NDBDown:FREQuency?

This command queries the horizontal position of the n dB down markers.

Return value

<frequency1> absolute frequency of the n dB marker to the left of the reference marker in Hz

<frequency1> absolute frequency of the n dB marker to the right of the reference marker in Hz.

Example

See CALCulate:MARKer<m>:FUNCtion:NDBDown:STATe <State>

Characteristics

*RST value: -
SCPI: device-specific

CALCulate:MARKer<m>:FUNCtion:NDBDown:RESult?

This command queries the frequency spacing or bandwidth of the n dB down markers.

Return value

<Bandwidth>
Bandwidth in Hz.

Example

See CALCulate:MARKer<m>:FUNCtion:NDBDown:STATe <State>

Characteristics

*RST value: -
SCPI: device-specific

CALCulate:MARKer<m>:FUNCtion:NDBDown:STATe <State>

This command turns the n dB Down marker function on and off.

Parameter

<State>
ON | OFF

Example

```
CALC:MARK:FUNC:NDBD:STAT ON
```
Turns on the n dB marker function.

```
CALC:MARK:FUNC:NDBD 3
```
Positions two temporary markers 3 dB below a reference marker.

```
CALC:MARK:FUNC:NDBD:FREQ?
```
Queries the frequency position of the n dB Down markers; would return e.g.
`100000000,200000000`

```
CALC:MARK:FUNC:NDBD:RES?
```
Queries the measurement result; would return e.g.
`100000000`

Characteristics

*RST value: OFF
SCPI: device-specific

CALCulate:MARKer<m>:FUNCtion:NOISe[:STATe] <State>

This command turns the noise measurement for all markers on and off.

You can query the results of the noise power density at the marker position with `CALCulate:MARKer<m>:FUNCtion:NOISe:RESult?`.

Parameter

<State>
ON | OFF

Example

See `CALCulate:MARKer<m>:FUNCtion:NOISe:RESult?`.

Characteristics

*RST value: OFF
SCPI: device–specific

CALCulate:MARKer<m>:FUNCtion:NOISe:RESult?

This command queries the result of the noise measurement.

To get a valid result, you have to perform a complete sweep with synchronization to the sweep end before reading out the result. This is only possible in single sweep mode.

This command is an event and therefore has no *RST value and no query.

Return value

<NoiseLevel>

The unit depends on `UNIT:POWer`.

Example

```
INIT:CONT OFF
```
Turns on single sweep mode.

```
CALC:MARK2 ON
CALC:MARK2:FUNC:NOIS ON
```
Turns on marker 2 and assigns the noise measurement to that marker.

```
INIT;*WAI
CALC:MARK2:NOIS:RES?
```
Performs the measurement and queries the noise marker results.

Characteristics

*RST value: –
SCPI: device–specific

CALCulate:MARKer<m>:FUNCtion:REFerence

This command matches the reference level to the power level of a marker

If you use a delta marker, the R&S FSH turns it into a normal marker.

This command is an event and therefore has no *RST value and no query.

Example

```
CALC:MARK1:FUNC:REF
```
Matches the reference level to the power level of marker 1.

Characteristics

*RST value: -
SCPI: device-specific

6.2.7 Using Display Lines and Limit Lines

6.2.7.1 Display Lines

The following commands define the position of the display line.

The suffix <n> at CALCulate is irrelevant.

List of commands

- CALCulate<n>:DLINe <Amplitude> (p. 71)
- CALCulate<n>:DLINe:STATe <State> (p. 71)

CALCulate<n>:DLINe <Amplitude>

This command defines the position of a display line.

Parameter

<Amplitude>
Numeric value with a variable range and unit.

You can use any unit you want, the R&S FSH then converts the unit to the currently selected unit. If you omit a unit, the R&S FSH uses the currently selected unit.

Example

```
CALC:DLIN -20dBm
```
Sets the display line threshold to -20 dBm.

Characteristics

*RST value: – (STATe to OFF)
SCPI: device–specific

CALCulate<n>:DLINe:STATe <State>

This command turns display lines on and off.

Parameter

<State>
ON | OFF

Example

```
CALC:DLIN:STAT OFF
```
Turns on the display line.

Characteristics

*RST value: OFF
SCPI: device–specific

6.2.7.2 Limit Lines

The following commands define limit lines and perform the corresponding limit checks.

The suffix <n> at CALCulate is irrelevant.

The suffix <k> at LIMit selects the limit line and is in the range <1…2>.

List of commands

CALCulate<n>:LIMit<k>:BEEP[:STATe] <State>

This command turns the beeper that beeps if a limit line is violated on and off.

Parameter

<State>
ON | OFF

Example

```
CALC:LIM:BEEP ON
```
Activates the audio beep.

Characteristics

*RST value: OFF
SCPI: device-specific

CALCulate<n>:LIMit<k>:COMMent?

This command queries the description of a limit line.

This command is a query and therefore has no RST value.

Return value

<Comment>
String containing the description of the limit line.

Example

```
CALC:LIM:COMM?
```
Queries the description of limit line 1.

Characteristics

*RST value: -
SCPI: device-specific

CALCulate<n>:LIMit<k>:DEFine

<Name>,<Comment>,<Unit>,<Scale>,<Unit>,<x1>,<y1>[,<xn>,<yn>]

This command defines the shape of a limit line.

After you have defined the shape of the limit line you still have to activate it with CALCulate<n>:LIMit<k>:UPPer:SELect <LimitLine> before it takes effect.

Parameters

<Name>
String containing the name of the limit line.

Note: if a limit line with the same name already exists, it will be overwritten.

<Comment>
String containing a comment for the limit line.

<X-unit>
Unit of the x-axis. HZ | S | M

<Scale>
Scale of the x-axis: ABS | REL

<Y-unit>
Unit of the y-axis: DB | DBM | DBUV | DBMV | DBUVM | DBUAM | VSWR | V | W

<x1>,<xn>
Data points on the x-axis.

Note: a limit line may consist of up to 100 horizontal data points.

<y1>,<yn>
Data points on the y-axis.

Example

```
CALC:LIM:DEF 'Line','Example',HZ,ABS,DBM,10000000,-10,
10000000,0,20000000,0
```
Defines a limit line with three data points.

Characteristics

*RST value: -
SCPI: device-specific

CALCulate<n>:LIMit<k>:DELete

This command deletes a limit line.

This command is an event and therefore has no *RST value and no query.

Example

```
CALC:LIM2:DEL
```
Deletes the second limit line

Characteristics

*RST value: -
SCPI: device-specific

CALCulate<n>:LIMit<k>:FAIL?

This command queries the result of a limit check.

To get a valid result, you have to perform a complete sweep with synchronization to the sweep end before reading out the result. This is only possible in single sweep mode.

Return value

0 for PASS and 1 for FAIL

Example

```
INIT;*WAI
CALC:LIM1:FAIL?
```
Performs a measurement and queries the result of the check for limit line 1.

Characteristics

*RST value: -
SCPI: conform

CALCulate<n>:LIMit<k>:LOWer:SELect <LimitLine>

This command selects the lower limit line.

This command is an event and therefore has no *RST value and no query.

Parameter

<LimitLine>
String containing the file name of the lower limit line.

Example

```
CALC:LIM:LOW:SEL 'GSM_Lower.rellim'
```
Selects the lower limit line.

Characteristics

*RST value:
SCPI: conform

CALCulate<n>:LIMit<k>:LOWer:THReshold <Level>

This command defines the level of a lower threshold limit line.

Parameter

<Level>
Numeric value whose unit depends on the unit you have currently selected for the vertical axis.

Example

```
CALC:LIM:LOW:THR -10DBM
```
Defines a threshold of -10 dBm

Characteristics

*RST value: -
SCPI: device-specific

CALCulate<n>:LIMit<k>:STATe <State>

This command turns a limit check on and off.

You can query the result of the limit check with `CALCulate<n>:LIMit<k>:FAIL?`.

Parameter

<State>
ON | OFF

Example

```
CALC:LIM:STAT ON
```
Switches on the limit check for limit line 1.

Characteristics

*RST value: OFF
SCPI: conform

CALCulate<n>:LIMit<k>:UNIT:X?

This command queries the horizontal unit of a limit line.

This command is a query and therefore has no *RST value.

Example

```
CALC:LIM:UNIT:X?
```
Queries the x-unit of the first limit line.

Characteristics

*RST value: -
SCPI: device-specific

CALCulate<n>:LIMit<k>:UNIT[:Y]?

This command queries the vertical unit of a limit line.

This command is a query and therefore has no *RST value.

Example

```
CALC:LIM1:UNIT?
```
Queries the y-unit of the first limit line.

Characteristics

*RST value: -
SCPI: device-specific

CALCulate<n>:LIMit<k>:UPPer:SELect <LimitLine>

This command selects the upper limit line.

This command is an event and therefore has no *RST value and no query.

Parameter

<LimitLine>
string containing the file name of the upper limit line

Example

```
CALC:LIM:UPP:SEL 'GSM_Upper.rellim'
```
Selects the upper limit line.

Characteristics

*RST value: -
SCPI: conform

CALCulate<n>:LIMit<k>:UPPer:THReshold <Level>

This command defines the level of an upper threshold limit line.

Parameter

<Level>
Numeric value whose unit depends on the unit you have currently selected for the vertical axis.

Example

```
CALC:LIM:UPP:THR -10DBM
```
Defines a threshold of -10 dBm

Characteristics

*RST value: -
SCPI: device-specific

6.2.8 Configuring and Using Measurement Functions

The R&S FSH provides measurement functions that allow you to perform advanced measurements and can also be controlled remotely.

General measurement functions:

- Working with Channel Tables on page 77

6.2.8.1 Working with Channel Tables

Use the following commands to work with channel tables.

List of commands

[SENSe:]CHANnel <ChannelNumber>

This command selects the channel to be analyzed.

You have to set the frequency mode with `[SENSe:]FREQuency:INPut:MODE` to channel first.

Parameter

<ChannelNumber>
numeric value that selects the number of the channel to be analyzed

Example

See `[SENSe:]CHANnel:TABLe:SELect <ChannelTable>`.

Characteristics

*RST value:depends on the channel table
SCPI: conform

[SENSe:]CHANnel:TABLe:SELect <ChannelTable>

This command selects a channel table configured for the link direction you have selected with `[SENSe]:CHANnel:TABLe:SET <Direction>`.

Note that if you have previously selected a channel table with `[SENSe]:CHANnel:TABLe:SELect:DOWNlink <ChannelTable>` or `[SENSe]:CHANnel:TABLe:SElect:UPLink <ChannelTable>`, this command replaces that file.

Parameter

<ChannelTable>
String containing the file name of the channel table.

Example

```
CHAN:TABL:SET UP
```
Selects channel tsble selection for uplink signals.

```
CHAN:TABL:SEL 'TV China.CHNTAB'
```
Loads the channel table with the name 'TV China' for the uplink.

```
CHAN:TABL:SEL 'TV Italy.chntab'
```
or
```
CHAN:TABL:SEL:UPL 'TV Italy.chntab'
```
Both commands replace the uplink channel table 'TV China' with 'TV Italy'. Downlink channel tables are not affected by the commands sent so far.

```
FREQ:INP:MODE CHAN
CHAN 10
```
Select a particular uplink channel (e.g. #10) instead of a (center) frequency.

To select an additional downlink channel table, use either:
```
CHAN:TABL:SET DOWN
CHAN:TABL:SEL 'TV Italy.chntab'
```

or

```
CHAN:TABL:SEL:DOWN 'TV Italy.chntab'
```

Characteristics

*RST value: ''
SCPI: device-specific

[SENSe]:CHANnel:TABLe:SELect:DOWNlink <ChannelTable>

This command selects a channel table configured for downlink signals.

Parameter

<ChannelTable>
string containing the name of the channel table.

Example

See `[SENSe:]CHANnel:TABLe:SELect <ChannelTable>`.

Characteristics

*RST value: ''
SCPI: device-specific

[SENSe]:CHANnel:TABLe:SElect:UPLink <ChannelTable>

This command selects a channel table configured for uplink signals.

Parameter

<ChannelTable>
string containing the name of the channel table.

Example

See [SENSe:]CHANnel:TABLe:SELect <ChannelTable>.

Characteristics

*RST value: ' '
SCPI: device-specific

[SENSe]:CHANnel:TABLe:SET <Direction>

This command selects the link direction for measurements with channel tables.

Parameter

<Direction>

DOWN	Downlink
UP	Uplink

Example

See [SENSe:]CHANnel:TABLe:SELect <ChannelTable>.

Characteristics

*RST value: UP
SCPI: device-specific

6.3 Remote Commands of the Network Analyzer Mode

The chapter provides information on remote commands that configure and perform two-port measurements with the tracking generator. These commands are available in network analyzer mode only.

Contents

6.3.1 Configuring the Horizontal Axis

The following commands configure the horizontal axis of the active display.

List of commands

- [SENSe:]FREQuency:CENTer <Frequency> (p. 31)
- [SENSe:]FREQuency:CENTer:STEP <StepSize> (p. 32)
- [SENSe:]FREQuency:CENTer:STEP:LINK <StepSizeCoupling> (p. 32)
- [SENSe:]FREQuency:SPAN <Span> (p. 34)
- [SENSe:]FREQuency:SPAN:FULL (p. 35)
- [SENSe:]FREQuency:STARt <StartFrequency> (p. 35)
- [SENSe:]FREQuency:STOP <StopFrequency> (p. 35)

For a detailed description of the commands refer to "Configuring the Horizontal Axis" in spectrum mode.

6.3.2 Configuring the Vertical Axis

The following commands configure the level axis (y-axis) and level parameters of the active display.

The suffix <n> at DISPlay is irrelevant.

List of commands

- DISPlay<n>:MAGNitude:REFerence <RefValue> (p. 81)
- DISPlay<n>:MAGNitude:REFerence:POSition <RefPosition> (p. 81)
- DISPlay<n>:MAGNitude:Y:SCALe <DisplayRange> (p. 82)
- DISPlay<n>:MAGNitude:Y:SPACing <Scaling> (p. 82)
- INPut:ATTenuation <Attenuation> (p. 38)
- INPut:ATTenuation:AUTO <State> (p. 39)
- INPut:ATTenuation:MODE <AttenuationMode> (p. 39)
- INPut:IMPedance <Impedance> (p. 40)
- SOURce:TG:ATTenuation <TGAttenuation> (p. 83)
- UNIT:POWer <Unit> (p. 41)

For a detailed description of commands not described below refer to "Configuring the Vertical Axis" in spectrum analyzer mode.

DISPlay<n>:MAGNitude:REFerence <RefValue>

This command sets the reference value for the magnitude measurement format.

Parameter

<RefValue>
Numeric value in the range from -80 dB to 30 dB.

Example

```
DISP:MAGN:REF -10
```
Sets the reference level to -10 dB

Characteristics

*RST value: 0 dB
SCPI: device-specific

DISPlay<n>:MAGNitude:REFerence:POSition <RefPosition>

This command defines the position of the reference value on the display for the magnitude measurement format.

Each step shifts the reference position one grid line up or down.

Parameter

<RefPosition>
Numeric value in the range from 0 to 10.

Example

```
DISP:MAGN:REF:POS 5
```
Sets the reference to the center of the display (i.e. the fifth grid line from the bottom).

Characteristics

*RST value: 10
SCPI: device-specific

DISPlay<n>:MAGNitude:Y:SCALe <DisplayRange>

This command defines the display range of the vertical axis for the magnitude measurement format.

Note that you have to set a logarithmic scaling before you can use this command with `DISPlay<n>:MAGNitude:Y:SPACing <Scaling>`.

Parameter

<DisplayRange>
Numeric value in the range from 1 dB to 150 dB.

The number you enter is rounded up to the next possible display range. For example, if you enter 9, the R&S FSH automatically sets the display range to 10.

Example

```
DISP:MAGN:Y:SCAL 50 DB
```
Sets the display range of the y-axis to 50 dB

Characteristics

*RST value: 100 dB
SCPI: device-specific

DISPlay<n>:MAGNitude:Y:SPACing <Scaling>

This command selects the scaling of the vertical axis for the magnitude measurement format.

Parameter

<Scaling>

LOGarithmic	logarithmic scaling (dB)
LINear	linear scaling (%)

Example

```
DISP:MAGN:Y:SPAC LIN
```
Switches the y-axis to linear scaling

Characteristics

*RST value: LOGarithmic
SCPI: device-specific

SOURce:TG:ATTenuation <TGAttenuation>

This command sets the output level of the tracking generator.

Parameter

<TGAttenuation>
Numeric value in the range from 0 to 50 dB.

Entering an output level of, e.g., 20 dB results in an output level of -20 dBm.

Example

```
SOUR:TG:ATT 50
```
Sets the attenuation to 50 dB and therefore an output level of –50 dBm

Characteristics

*RST value: 0 dB
SCPI: device-specific

6.3.3 Setting the Bandwidths

The following commands configure the filter bandwidths of the R&S FSH. Note that both groups of commands (BANDwidth and BWIDth) are the same.

List of commands

- [SENSe:]BANDwidth|BWIDth[:RESolution] <ResBW> (p. 42)
- [SENSe:]BANDwidth|BWIDth[:RESolution]:AUTO <State> (p. 42)

For a detailed description of commands refer to "Setting the Bandwidths" in spectrum analyzer mode.

6.3.4 Performing and Triggering the Measurement

The following commands configure the sweep.

List of commands

- *WAI (p. 30)
- ABORt (p. 44)
- INITiate[:IMMediate] (p. 44)
- INITiate:CONTinuous <SweepMode> (p. 45)
- [SENSe:]SWEep:COUNt <SweepCount> (p. 45)
- [SENSe:]SWEep:POINts? (p. 46)
- [SENSe:]SWEep:TIME <SweepTime> (p. 46)
- [SENSe:]SWEep:TIME:AUTO <State> (p. 46)
- TRIGger[:SEQuence]:HOLDoff[:TIME] <TriggerDelay> (p. 48)
- TRIGger[:SEQuence]:SLOPe <TriggerSlope> (p. 49)
- TRIGger[:SEQuence]:SOURce <TriggerSource> (p. 50)

For a detailed description of commands refer to "Performing and Triggering Measurements" in spectrum analyzer mode.

6.3.5 Working with Traces

The following commands set up the trace and the various functions associated with it, e.g. the selection of the detector.

List of commands

- DISPlay[:WINDow]:TRACe<t>:MEMory[:STATe] <State> (p. 52)
- DISPlay[:WINDow]:TRACe<t>:MODE <TraceMode> (p. 52)
- FORMat[:DATA] <DataFormat> (p. 55)
- FORMat:BORDer <ByteOrder> (p. 53)
- [SENSe:]DETector<t>[:FUNCtion] <Detector> (p. 53)
- [SENSe:]DETector<t>[:FUNCtion]:AUTO <State> (p. 54)
- TRACe[:DATA]? (p. 85)

For a detailed description of commands not described here refer to "Working with Traces" in spectrum analyzer mode.

TRACe[:DATA]?

This command reads out the trace data of the current measurement.

With the `FORMat[:DATA] <DataFormat>` command, you can set the data format.

Parameter

The available parameters depend on the format of the result display.

Display of one S-parameter

TRACE1	queries the data of the currently displayed data

Return value

The R&S FSH returns 631 values. Each value corresponds to one pixel of a trace.

The result and unit depends on the measurement format.

Example

```
TRAC:DATA? TRACE1
```
Reads out the data for trace 1.

Characteristics

*RST value: -
SCPI: conform

6.3.6 Using Markers and Deltamarkers

6.3.6.1 Markers and Deltamarkers

The following commands are for setting and controlling markers and deltamarkers. If not otherwise noted, the numeric suffix <1...6> at MARKer or DELTamarker select the marker to be controlled.

The suffix <n> at CALCulate selects the measurement screen in dual trace mode and is in the range <1…2>.

The suffix <m> at MARKer selects the marker and is in the range <1…6>.

List of commands

For a detailed description of commands not described below refer to "Using Markers" in spectrum analyzer mode.

CALCulate<n>:MARKer<m>:Y?

This command queries the measurement results at the marker position.

If necessary, the corresponding marker is activated first.

To get a valid result, you have to perform a complete sweep with synchronization to the sweep end between activating the delta marker and reading out the result. This is only possible in single sweep mode.

Return value

NORMal <value> = trace value

Example

```
INIT:CONT OFF
CALC:MARK2 ON
```
Turns on single sweep mode and marker 2.

```
INIT;*WAI
CALC:MARK2:Y?
```
Performs a measurement and queries the marker position.

Characteristics

*RST value: –
SCPI: device–specific

6.3.7 Configuring the Measurement

This chapter provides information on how to configure two-port measurements with the tracking generator. The structure follows the order of the actual operation sequence used when performing a measurement:

- Calibrating the Measurement on page 87

To perform the actual measurement, use the commands described in section "Performing and Triggering the Measurement".

Commands independent of the operating mode

Note that some of the commands for configuring two-port measurements are also valid for other operating modes. If a command is available in another mode, it is indicated by the list in the respective section.

6.3.7.1 Calibrating the Measurement

The following commands query and control calibration for two-port measurements.

List of commands

- CALibration:MODE? (p. 87)
- CALibration:STATus? (p. 88)

The suffix <p> selects the measurement port. It is in the range <1…2>.

CALibration:MODE?

This command queries if the current measurement is calibrated.

This command is a query and therefore has no *RST value.

Return value

0	not calibrated
1	calibrated

Example

CAL:MODE?
Queries the calibration state.

Characteristics

*RST value: -
SCPI: device-specific

CALibration:STATus?

This command queries if the R&S FSH is fully calibrated for the current measurement.

This command is a query and therefore has no *RST value.

Return value

NORMalized	full factory or user calibration
APPRoximate	approximate calibration: measurement uncertainty must be anticipated

Example

CAL:STAT?
Queries the calibration status of the R&S FSH.

Characteristics

*RST value: -
SCPI: device-specific

6.4 File Management

The following commands perform various tasks in the context of file management.

These commands are independent from the operating mode.

List of commands

MMEMory:CATalog?

This command queries the files of the current directory.

You can select directories with `MMEMory:CDIRectory <Directory>`.

This command is a query and therefore has no *RST value.

Return value

<UsedDiskSpace>,<FreeDiskSpace>,<FileName_1>,<SizeFile_1><ModificationDateFile_1>,<ModificationTimeFile_1>,…,<FileName_n>,<SizeFile_n><ModificationDateFile_n>, <ModificationTimeFile_n>

Example

```
MMEM:CDIR '\Public\Limit Lines'
```
Opens directory 'Limit Lines'

```
MMEM:CAT?
```
Returns all files in \Public\Limit Lines

Characteristics

*RST value: -
SCPI: conform

MMEMory:CATalog:DIRectories?

This command queries the directories of the current directory.

This command is a query and therefore has no *RST value.

Return value

<UsedMemory>,<FreeMemory>,<DirName_1>,<ModificationDateDir_1>,
<ModificationTimeDir_1>,…,<DirName_n>,<ModificationDateDir_n>,
<ModificationTimeDir_n>

Example

```
MMEM:CDIR '\Public'
```
Opens directory \Public.

```
MMEM:CAT:DIR?
```
Returns all directories in the \Public directory

Characteristics

*RST value: -
SCPI: device-specific

MMEMory:CDIRectory <Directory>

This command changes the current directory.

Parameter

<Directory>
String containing the path to another directory.

Example

```
MMEM:CDIR '\Public'
```
Opens directory \Public.

Characteristics

*RST value: -
SCPI: conform

MMEMory:COPY <SourceFile>,<DestinationFile>

This command copies one or more files to another directory.

This command is an event and therefore has no *RST value and no query.

Parameter

<SourceFile>
String containing the path and file name of the source file.

<DestinationFile>
String containing the path and name of the destination file.

Example

```
MMEM:COPY
'\Public\Standards\cdmaOne.obwstd','\USB\cdmaOne.std'
```
Copies the cdmaOne standard file file to a memory stick.

Characteristics

*RST value: -
SCPI: conform

MMEMory:DATA <FileName>[,<BlockData>]

This command writes block data into a file. The delimiter must be set to EOI to obtain error-free data transfer.

When you query the contents of a file, you can save them in a file on the remote control computer.

The command is useful for reading stored settings files or trace data from the instrument or for transferring them to the instrument.

Parameter

<FileName>
String containing the path and file name.

<BlockData>
Data block with the structure

#	hash sign
<number>	length of the length information
<number>	length information of the binary data (number of bytes)
<data>	binary data with the indicated number of bytes

Example

```
MMEM:NAME '\Public\User\Testfile.txt'
```
Creates a new file called 'Testfile.txt'.

```
MMEM:DATA '\Public\User\Testfile.txt',#220Contents of the
file
```
The parameter mean:

- '\Public\...' selects the target file
- #2: hash sign and length of the length information (20 bytes = 2 digits)
- 20: indicates the number of subsequent binary data bytes
- Contents of the file: store 20 binary bytes (characters) to the file

```
MMEM:DATA? '\Public\User\Testfile.txt'
```
Transfers the contents of the file 'Testfile.txt' to the control computer.

Characteristics

*RST value: -
SCPI: conform

MMEMory:DELete <File>

This command deletes a file.

Parameter

<File>
String containing the path and file name of the file to delete.

Example

```
MMEM:DEL '\Public\Screen Shots\Screen0001.png'
```
Deletes the file Screen0001.png.

Characteristics

*RST value: -
SCPI: conform

MMEMory:FILE <File>

This command creates a file.

Parameter

<File>
String containing the file name.

Example

```
MMEM:FILE 'TEST.TXT'
```
Creates the file TEST.TXT

Characteristics

*RST value: -
SCPI: conform

MMEMory:FILE:DATE <FileName>,<Date>

This command sets the date of a file.

Parameter

<FileName>
String containing the path and file name.

<Date>
Numeric values indicating the date

You have to enter the date as comma separated values after the string with the file name. The sequence is year,month,day.

The available value range is 1980...2099, 1...12, 1...31

Example

```
MMEM:FILE:DATE '\Public\Screen
Shots\Screen0001.png',2006,04,01
```
Sets the date to April, 1st, 2006.

```
MMEM:FILE:DATE? '\Public\Screen Shots\Screen0001.png'
```
Returns the modification date of the file Screen0001.png.

Characteristics

*RST value: -
SCPI: conform

MMEMory:FILE:TIME <FileName>,<Time>

This command sets the time of a file. The sequence of entry is hour, minute, second.

Parameter

<FileName>
String containing the path and file name.

<Time>
Numeric values indicating the time.

You have to enter the time as comma separated values after the string with the file name. The sequence is hour,minute,second.

The available value range is 0...23, 0...59, 0...59.

Example

```
MMEM:FILE:TIME '\Public\Screen Shots\Screen0006.png',11,04,00
```
Sets the time to 11:04:00

Characteristics

*RST value: -
SCPI: conform

MMEMory:INIT

This command formats the indicated drive.

Note
Formatting deletes all data stored on the memory drive.

This command is an event and therefore has no *RST value and no query.

Example:

```
MMEM:INIT
```
Formats and deletes all data from the drive.

Characteristics:

*RST value: -
SCPI: conform

MMEMory:LOAD:STATe 1,<FileName>

This command loads the settings from a *.set file.

Parameter

<FileName>
String containing the path and file name.

Example

```
MMEM:LOAD:STAT 1,'\Public\Datasets\Dataset001.set'
```
Loads the settings from the file Dataset001.

Characteristics

*RST value: -
SCPI: conform

MMEMory:MDIRectory <Directory>

This command creates a new directory.

This command is an event and therefore has no *RST value and no query.

Parameter

<Directory>
String containing the path and new directory name.

Example

```
MMEM:MDIR '\Public\USER'
```
Creates the a directory called 'User'

Characteristics

*RST value: -
SCPI: device-specific

MMEMory:MOVE <SourceFile>,<NewFileName>

This command renames files, if <file_destination> contains no path. Otherwise the file is moved to the indicated path and stored under the file name specified there.

This command is an event and therefore has no *RST value and no query.

Parameter

<SourceFile>
String containing the path and file name of the source file.

<DestinationFile>
String containing the path and name of the destination file.

Example

```
MMEM:MOVE '\Public\Screen
Shots\Screen0002.png','\Public\Screen Shots\Screen0001.png'
```
Renames Screen0002.png to Screen0001.png

```
MMEM:MOVE '\Public\Screen
Shots\Screen0001.png','\Public\Test\Pic1.png'
```
Moves file Screen0006.png to the 'Test' folder and renames the file Pic1.png

Characteristics

*RST value: -
SCPI: conform

MMEMory:RDIRectory <Directory>

This command deletes the indicated directory. The directory name includes the path and may also include the drive name. The path name complies with DOS conventions.

This command is an event and therefore has no *RST value and no query.

Parameter

<Directory>
String containing the path of the directory to delete.

Example

```
MMEM:RDIR '\Public\Screen Shots\
```
Deletes the directory 'Screen Shots'.

Characteristics

*RST value: -
SCPI: device-specific

MMEMory:STORe:STATe 1,<FileName>

This command stores the current device settings in a *set file.

This command is an event and therefore has no *RST value and no query.

Parameter

1,<FileName>
String containing the path and name of the destination file.

Example

```
MMEM:STOR:STAT 1,'DATASET001.SET'
```
Saves the current device settings in the file DATASET001.SET.

Characteristics

*RST value: -
SCPI: conform

6.5 Making and Storing Screenshots

The following commands manage screenshots.

These commands are independent from the operating mode.

List of commands

HCOPy[:IMMediate]

This command makes a screenshot of the current trace and stores it on the R&S FSH internal memory.

This command is an event and therefore has no *RST value and no query.

Example

```
MMEM:NAME '\Public\Screenshots\Test.png'
```
Defines the file name of the screenshot.

```
HCOP
```
Makes and stores a screenshot of the current screen.

Characteristics

*RST value: -
SCPI: device-specific

MMEMory:NAME <FileName>

This command defines the path and file name that the R&S FSH uses for storing screenshots (see `HCOPy[:IMMediate]`). The path and file name comply with DOS conventions.

This command is an event and therefore has no *RST value and no query.

Parameter

<FileName>
String containing the file name.

Example

```
MMEM:NAME 'Public\Screeshots\Test.png'
```
Stores the screenshot under D:\Test

Characteristics

*RST value: -
SCPI: device-specific

6.6 Configuring Data Capture

The following commands configure the data capture.

These commands are independent from the operating mode.

List of commands

SYSTem:CAPTure:COUNter <Counter>

This command defines the start of the file name counter.

The counter numbers the files stored when you capture data (screenshots, datasets etc.).

Parameter

<Counter>
String containing the number with which to start numbering files.

Example

```
SYST:CAPT:COUN '0100'
```
Starts numbering files with 0100, e.g. Measurement0100.png.

Characteristics

*RST value: '0000'
SCPI: device-specific

SYSTem:CAPTure:DATaset[:STATe] <State>

This command includes or excludes datasets from the data capture.

Parameter

<State>
ON | OFF

Example

```
SYST:CAPT:DAT ON
```
Includes datasets into the data capture

Characteristics

*RST value: OFF
SCPI: device-specific

SYSTem:CAPTure:SCReen[:STATe] <State>

This command includes or excludes screenahots from the data capture.

Parameter

<State>
ON | OFF

Example

```
SYST:CAPT:SCR ON
```
Includes screenshots into the data capture

Characteristics

*RST value: ON
SCPI: device-specific

6.7 Saving Events

The following commands configure the circumstances under which the R&S FSH saves events.

These commands are independent from the operating mode.

Using the commands requires an GPS receiver and a storage device (SD card or memory stick).

List of commands

SYSTem:SOEVent:DISTance:INTerval <Distance>

This command defines a distance that you must cover before the R&S FSH saves another coordinate.

Parameter

<Distance>
Distance between one coordinate and the next.

Example

```
SYST:SOEV ON
```
Turns on saving coordinates on an event.

```
SYST:SOEV:REC:STOR USB
```
Selects an USB device as the storage device.

```
SYST:SOEV:SOUR DIST
SYST:SOEV:DIST:INT 5
```
Saves the coordinates every 5 m.

Characteristics

*RST value: 1 m
SCPI: device-specific

SYSTem:SOEVent:LIMits:MODE <Mode>

This command selects the limit check condition that must occur in order to save a coordinate.

Parameter

<Mode>

FAILonly Saves only sweeps that contain a limit check violation.

STARtonfail	Starts to save all sweeps from the moment a limit check fails.
STOPonfail	Saves all sweeps until a limit check fails.

Example

```
SYST:SOEV:SOUR LIM
SYST:SOEV:LIM:MODE FAIL
```
Saves all sweeps that contain a violation of a limit check.

Characteristics

*RST value: STAR
SCPI: device-specific

SYSTem:SOEVent:RECording:STORage <Device>

This command selects the storage device to save the coordinates to.

Parameter

<Device>

SDCard	Saves coordinates to an SD card
USB	Saves coordinates to an USB device

Example

See SYSTem:SOEVent:DISTance:INTerval <Distance>.

Characteristics

*RST value: SDCard
SCPI: device-specific

SYSTem:SOEVent:SOURce <EventType>

This command selects the type of event that triggers saving the coordinates of your current location.

Parameter

<EventType>

ALLSweeps	Saves coordinates after each sweep
LIMitsfail	Saves coordinates when a limit check has failed
TIMeinterval	Saves coordinates after a certain length of time has passed

Example

See SYSTem:SOEVent:DISTance:INTerval <Distance>.

Characteristics

*RST value: TIMeinterval
SCPI: device-specific

SYSTem:SOEVent:TIME:INTerval <Time>

This command defines a time interval that must pass before the R&S FSH saves another coordinate.

Parameter

<Time>
Time that must pass between one coordinate and the next.

Example

```
SYST:SOEV ON
```
Turns on saving coordinates on an event.

```
SYST:SOEV:REC:STOR USB
```
Selects an USB device as the storage device.

```
SYST:SOEV:SOUR TIM
SYST:SOEV:TIME:INT 5
```
Saves the coordinates every 5 seconds.

Characteristics

*RST value: 1 s
SCPI: device-specific

SYSTem:SOEVent:[:STATe] <State>

This command turns saving of your current coordinates in case of certain events on and off.

Parameter

<State>
ON | OFF

Example

See SYSTem:SOEVent:DISTance:INTerval <Distance>.

Characteristics

*RST value: OFF
SCPI: device-specific

6.8 Configuring the Instrument

The following commands configure general instrument settings.

These commands are independent from the operating mode.

Contents

6.8.1 Mode Selection

This chapter describes all commands that select the operating mode of the R&S FSH.

List of commands

- INSTrument[:SELect] <OperatingMode> (p. 102)
- INSTrument:NSELect <OperatingMode> (p. 102)

INSTrument[:SELect] <OperatingMode>

This command selects the operating mode.

Parameter

<OperatingMode>

NAN	network analyzer
SANalyzer	spectrum analyzer

Example

```
INST SAN
```

Selects spectrum analyzer mode.

Characteristics

*RST value: ACT
SCPI: conform

INSTrument:NSELect <OperatingMode>

This command selects the operating mode.

Parameter

1	spectrum analyzer
2	network analyzer

Example

```
INST:NSEL 1
```
Selects spectrum analyzer mode.

Characteristics

*RST value: 4
SCPI: conform

6.8.2 Display Configuration

This chapter describes commands to set up the display of the R&S FSH via remote control.

List of Commands

DISPlay:BRIGhtness <Brightness>

This command sets the brightness of the display backlight.

Parameter

<Brightness>
Numeric value in the range from 0 to 1

Example

```
DISP:BRIG 0.80
```
Sets the brightness of the display to 80%

Characteristics

*RST value: 0.5 (50%)
SCPI: device-specific

DISPlay:CMAP <ColorScheme>

This command sets the color scheme of the display.

Parameter

<ColorScheme>

COLor	color
BW	black & white
PF	printer friendly

Example

```
DISP:CMAP BW
```
Sets the screen colors to black and white

Characteristics

*RST value: COLor
SCPI: conform

DISPlay:CMAP:DEFault

This command sets the display to the default state.

This command is an event and therefore has no query and no *RST value.

Example

```
DISP:CMPA:DEF
```
Restores the original color scheme

Characteristics

*RST value: -
SCPI: conform

DISPlay:DATE:FORMat <DateFormat>

This command sets the display date format.

Parameter

<DateFormat>
DDMMyyyy | MMDDyyyy

Example

```
DISP:DATE:FORM DDMMyyyy
```
Characteristics

*RST value: DDMMyyyy
SCPI: device-specific

6.8.3 Audio Settings

This chapter describes all commands to control the audio functions of the R&S FSH.

List of commands

- SYSTem:AUDio:VOLume <Volume> (p. 105)
- SYSTem:BEEPer:VOLume <Volume> (p. 105)
- SYSTem:BEEPer:KEY:VOLume <Volume> (p. 105)

SYSTem:AUDio:VOLume <Volume>

This command sets the volume of the internal speaker.

Parameter

<Volume>
Numeric value in the range from 0 to 1

Example

```
SYST:AUD:VOL 0.40
```
Sets the volume to 40%

Characteristics

*RST value: 0.3 (30%)
SCPI: device-specific

SYSTem:BEEPer:VOLume <Volume>

This command sets the volume of the system beeper.

Parameter

<Volume>
Numeric value in the range from 0 to 1

Example

```
SYST:BEEP:VOL 0.50
```
Sets the volume of the beeper to 50%

Characteristics

*RST value: 0.6 (60%)
SCPI: conform

SYSTem:BEEPer:KEY:VOLume <Volume>

This command sets the volume of the keyboard click noise.

Parameter

<Volume>
Numeric value in the range from 0 to 1

Example

```
SYST:BEEP:KEY:VOL 0.10
```
Sets of keyboard clicking volume to 10%

Characteristics

*RST value: 0.3 (30%)
SCPI: conform

6.8.4 Setting up a Network Connection

This chapter describes all commands that are used if the R&S FSH is part of a network.

List of commands

SYSTem:COMMunicate:LAN:ETHernet?

This command queries the MAC address of the R&S FSH.

This command is a query and therefore has no *RST value.

Example

```
SYST:COMM:LAN:ETH?
```
Returns the MAC address

Characteristics

*RST value: -
SCPI: device-specific

SYSTem:COMMunicate:LAN:GATeway <Gateway>

This command sets the gateway in the LAN.

Parameter

<Gateway>
String containing the identifier of the gateway.

Characteristics

*RST value: -
SCPI: device-specific

SYSTem:COMMunicate:LAN:SUBMask <SubnetMask>

This command sets the subnet mask of the R&S FSH.

Parameter

<SubnetMask>
String containing the subnet mask ('x.x.x.x').

Example

```
SYST:COMM:LAN:SUBM '255.255.255.0'
```
Sets the subnet mask address to 255.255.255.0

Characteristics

*RST value: 255.255.255.0
SCPI: device-specific

SYSTem:COMMunicate:SOCKet:ADDRess <IPAddress>

This command sets the IP address of the R&S FSH.

Parameter

<IPAddress>
String containing the IP address ('x.x.x.x').

Example

```
SYST:COMM:SOCK:ADDR '172.76.68.30'
```
Sets the IP address of the R&S FSH to 172.76.68.30

Characteristics

*RST value: 172.76.68.24
SCPI: device-specific

SYSTem:COMMunicate:SOCKet:DHCP[:STATe] <State>

This command turns the Dynamic Host Configuration Protocol (DHCP) on and off.

Parameter

<State>
ON | OFF

Example

```
SYST:COMM:SOCK:DHCP ON
```
Activates DHCP.

Characteristics

*RST value: ON
SCPI: device-specific

SYSTem:COMMunicate:SOCKet:PORT <Port>

This command sets the port number for the connection.

Parameter

<Port>
Port number

Example

```
SYST:COMM:SOCK:PORT 1000
```
Sets the port number to 1000

Characteristics

*RST value: 5555
SCPI: device-specific

6.8.5 System Settings

This chapter describes all commands that define or query general system settings.

List of commands

INPut:IMPedance:PAD <MatchingPad>

This command selects the matching pad connected to the R&S FSH.

Parameter

<MatchingPad>
RAM | RAZ | HZTE

Example

```
INPut:IMP 75;PAD RAZ
```
Selects 75 Ω input impedance and the R&S RAZ as the matching pad.

Characteristics

*RST value: -
SCPI: device-specific

[SENSe:]ROSCilator:SOURce <RefSource>

This command selects the source of the frequency reference oscillator.

If you use an external reference signal, make sure to connect the signal to the Ext Ref BNC connector of the R&S FSH.

Parameter

<RefSource>

INTernal	internal reference
EXTernal	external reference

Example

```
ROSC:SOUR EXT
```
Activates external source as reference signal.

Characteristics

*RST value: -
SCPI: device-specific

SYSTem:BNC<1...2>:MODE <BNCFunction>

This command configures the BNC sockets.

The numeric suffix at BNC selects the BNC socket you want to configure.

Parameter

<BNCFunction>

REFerence	input for external reference signal (BNC1)
TRIGger	input for external trigger (BNC1)
BIAS	BIAS port (BNC1 and BNC2)
IF3	IF output (BNC2)

Example

```
SYST:BNC2:MODE TRIG
```
Sets the seconds BNC socket to trigger input.

Characteristics

*RST value: BNC 1: TRIGger, BNC 2: IF3
SCPI: device-specific

SYSTem:DATE <Date>

This command sets the date for the internal calendar.

Parameter

<Date>
Numeric value indicating the date

You have to enter the date as comma separated values after the string with the file name. The sequence is year,month,day.

The available value range is 1980...2099, 1...12, 1...31

Example

```
SYST:DATE 2000,6,1
```
Sets the date to 1/6/2000

Characteristics

*RST value: -
SCPI: conform

SYSTem:ERRor[:NEXT]?

This command queries the oldest entry in the error queue and deletes it.

This command is a query and therefore has no *RST value.

Return value

<error number>,<error description>

<error number>	number that contains information about the error negative number: error as defined in the SCPI standard positive number: error that is specific to the R&S FSH
<error description>	string containing a short error description

If the error queue is empty, the command returns 0,'no error'.

Example

```
STAT:ERR?
```
Characteristics

*RST value: -
SCPI: conform

SYSTem:ERRor:ALL?

This command queries the complete error queue.

This command is a query and therefore no *RST value.

Return value

<error number>,<error description>

<error number> number that contains information about the error

negative number: error as defined in the SCPI standard
positive number: error that is specific to the R&S FSH

<error description> string containing a short error description

The number of values depends on the length of the queue.

If the error queue is empty, the command returns 0,'no error'.

Example

```
SYST:ERR:ALL?
```

Characteristics

*RST value: -
SCPI: device-specific

SYSTem:ERRor:CODE[:NEXT]?

This command queries the code of the next error in the error queue.

This command is a query and therefore has no *RST value.

Return value

<error number>
number that contains information about the error

Negative number: error as defined in the SCPI standard
Positive number: error that is specific to the R&S FSH

If the error queue is empty, the command returns 0.

Example

```
STAT:ERR:CODE?
```

Characteristics

*RST value: -
SCPI: conform

SYSTem:ERRor:CODE:ALL?

This command queries the complete error queue.

This command is a query and therefore no *RST value.

Return value

<error number>
number that contains information about the error

Negative number: error as defined in the SCPI standard
Positive number: error that is specific to the R&S FSH

The number of values depends on the length of the queue.

If the error queue is empty, the command returns 0,'no error'.

Example

```
SYST:ERR:CODE:ALL?
```

Characteristics

*RST value: -
SCPI: device-specific

SYSTem:ERRor:COUNt?

This command queries the number of errors currently in the error queue.

This command is a query and therefore no *RST value.

Return value

<numeric_value>
number of the errors in the queue

Example

```
SYST:ERR:COUN?
```

Characteristics

*RST value: -
SCPI: device-specific

SYSTem:FORMat:IDENt <IDNFormat>

This command sets the response format to the `*IDN?` query. This function is intended for re-use of existing control programs together with the R&S FSH.

Parameter

<IDNFormat>

LEGacy	format that is compatible to R&S FSH3/6/18
NEW	format for R&S FSH4/8

Example

```
SYST:FORM:IDEN LEG
*IDN?
```

IDN would return, e.g. "Rohde&Schwarz,FSH8,101805/028,1.40"

```
SYST:FORM:IDEN NEW
*IDN?
```

IDN would return, e.g. "Rohde&Schwarz,FSH8, 1309.6000K28/101805,1.40"

Characteristics

*RST value: -
SCPI: device-specific

SYSTem:HELP:HEADers?

This command returns a list of all available remote control commands.

This command is a query and therefore no *RST value.

Example

```
SYST:HELP:HEAD?
```
Returns the syntax of all available commands.

Characteristics

*RST value: -
SCPI: conform

SYSTem:HELP:SYNTax?

This command returns the full syntax of the specified command.

This command is a query and therefore no *RST value.

Parameter

<Command>
String containing the command you want to query

Example

```
SYST:HELP:SYNT? 'SYST:ERR?'
```
Returns the full syntax. In this case: 'SYSTem:ERRor[:NEXT]'.

Characteristics

*RST value: -
SCPI: device-specific

SYSTem:LANGuage <Language>

This command sets the language of the R&S FSH user interface. You can query a list of available languages with `SYSTem:LANGuage:CATalog?`.

Parameter

<Language>
string containing the language

Example

```
SYST:LANG 'english'
```
Sets the system language to English

Characteristics

*RST value: -
SCPI: conform

SYSTem:LANGuage:CATalog?

This command lists all languages available for the user interface.

This command is a query and therefore no *RST value.

Example

```
SYST:LANG:CAT?
```

Characteristics

*RST value: -
SCPI: device-specific

SYSTem:POWer:SOURce?

This command queries the current R&S FSH power source.

This command is a query and therefore has no *RST value.

Return values

ADAP	R&S FSH is powered by the AC power supply
BATT	R&S FSH is powered by the battery

Example

```
SYST:POW:SOUR?
```

Characteristics

*RST value: -
SCPI: conform

SYSTem:POWer:STATus?

This command queries the remaining power of the battery.

This command is a query and therefore has no *RST value.

Return values

Numeric value in the range from 0 to 100 %.

Example

```
SYST:POW:STAT?
```

Example

```
SYST:POW:STAT?
```

Characteristics

*RST value: -
SCPI: conform

SYSTem:PRESet

Resets the R&S FSH to its default state or a state defined by the user, depending on SYSTem:PRESet:MODE.

This command is an event and therefore has no *RST value and no query.

Example

```
SYST:PRES
```

Characteristics

*RST value: -
SCPI: conform

SYSTem:PRESet:FACTory

This command initiates an instrument reset back to factory settings.

This command is an event and therefore has no query and no *RST value.

Example

```
SYST:PRES:FACT
```

Resets the R&S FSH to its factory settings.

Characteristics

*RST value: -
SCPI: device-specific

SYSTem:PRESet:MODE <Mode>

This command selects the preset mode.

Parameter

<Mode>

DEFault	default preset state
USER	user defined preset state

Example

```
SYST:PRES:MODE USER
```

Selects a user defined preset.

Characteristics

*RST value: -
SCPI: conform

SYSTem:PRESet:USER <Preset>

This command selects a file containing a user defined preset state.

Parameter

<Preset>
filename of the user defined preset state

Characteristics

*RST value: -
SCPI: conform

SYSTem:SHUTdown

This command turns the R&S FSH off.

This command is an event and therefore has no *RST value and no query.

Example

```
SYST:SHUT
```
Turns the R&S FSH off.

Characteristics

*RST value: -
SCPI: conform

SYSTem:TIME <Time>

This command sets the internal clock.

Parameter

<Time>
Numeric value indicating the time

You have to enter the time as comma separated values after the string with the file name. The sequence is hour,minute,second.

The available value range is 0...23, 0...59, 0...59.

Example

```
SYST:TIME 12,30,30
```
Characteristics

*RST value: -
SCPI: conform

SYSTem:TZONee <TimeShift>

This command defines a shift of the system time to select another time zone.

Parameter

<TimeShift>
Numeric vakue indicating the time shift.

You have to enter the time shift as comma separated value. The equence is hour,minute.

The available value range is 0…23, 0…59.

Example

```
SYST:TZON 01,00
```
Shifts the time an hour ahead

Characteristics

*RST value: 0,0
SCPI: device-specific

SYSTem:VERSion?

This command queries the SCPI version the remote control is based on.

This command is a query and therefore has no *RST value.

Return value

1999.0

Example

```
SYST:VERS?
```
Characteristics

*RST value: -
SCPI: conform

6.9 Status Reporting System

The status reporting system stores all information on the present operating state of the instrument, and on errors which have occurred. This information is stored in the status registers and in the error queue. The status registers and the error queue can be queried via Ethernet.

The information is of a hierarchical structure. The register status byte (STB) defined in IEEE 488.2 and its associated mask register service request enable (SRE) form the uppermost level. The STB receives its information from the standard event status register (ESR) which is also defined in IEEE 488.2 with the associated mask register standard event status enable (ESE) and registers STATus:OPERation and STATus:QUEStionable which are defined by SCPI and contain detailed information on the instrument.

The output buffer contains the messages the instrument returns to the controller. It is not part of the status reporting system but determines the value of the MAV bit in the STB.

6.9.1 Structure of an SCPI Status Register

Each standard SCPI register consists of 5 parts which each have a width of 16 bits and have different functions. The individual bits are independent of each other, i.e. each hardware status is assigned a bit number that applies to all five parts. For example, bit 0 of the STATus:OPERation register is assigned to the calibration status of the R&S FSH. Bit 15 (the most significant bit) is set to zero for all parts. Thus the contents of the register parts can be processed by the controller as positive integer.

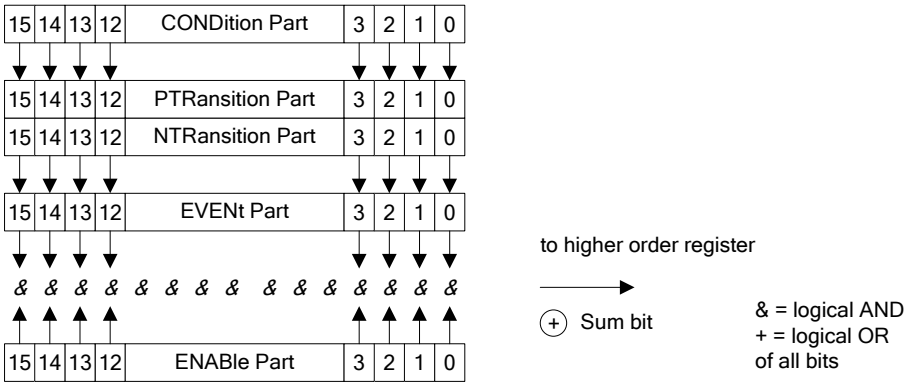


Figure 6-1: The status-register model

6.9.1.1 CONDition part

The CONDition part is directly written into by the hardware or the sum bit of the next lower register. Its contents reflects the current instrument status. This register part can only be read, but not written into or cleared. Its contents is not affected by reading.

6.9.1.2 PTRansition part

The Positive-TRansition part acts as an edge detector. When a bit of the CONDition part is changed from 0 to 1, the associated PTR bit decides whether the EVENt bit is set to 1.

PTR bit =1: the EVENt bit is set.

PTR bit =0: the EVENt bit is not set.

This part can be written into and read at will. Its contents is not affected by reading.

6.9.1.3 NTRansition part

The Negative-TRansition part also acts as an edge detector. When a bit of the CONDition part is changed from 1 to 0, the associated NTR bit decides whether the EVENt bit is set to 1.

NTR-Bit = 1: the EVENt bit is set.

NTR-Bit = 0: the EVENt bit is not set.

This part can be written into and read at will. Its contents is not affected by reading.

With these two edge register parts the user can define which state transition of the condition part (none, 0 to 1, 1 to 0 or both) is stored in the EVENt part.

6.9.1.4 EVENt part

The EVENt part indicates whether an event has occurred since the last reading, it is the "memory" of the condition part. It only indicates events passed on by the edge filters. It is permanently updated by the instrument. This part can only be read by the user. Reading the register clears it. This part is often equated with the entire register.

6.9.1.5 ENABle part

The ENABle part determines whether the associated EVENt bit contributes to the sum bit (see below). Each bit of the EVENt part is ANDed with the associated ENABle bit (symbol '&'). The results of all logical operations of this part are passed on to the sum bit via an OR function (symbol '+').

ENABle-Bit = 0: the associated EVENt bit does not contribute to the sum bit

ENABle-Bit = 1: if the associated EVENT bit is "1", the sum bit is set to "1" as well.

This part can be written into and read by the user at will. Its contents is not affected by reading.

6.9.1.6 Sum bit

As indicated above, the sum bit is obtained from the EVENt and ENABle part for each register. The result is then entered into a bit of the CONDition part of the higher-order register.

The instrument automatically generates the sum bit for each register. Thus an event, e.g. a PLL that has not locked, can lead to a service request throughout all levels of the hierarchy.

The service request enable register SRE defined in IEEE 488.2 can be taken as ENABle part of the STB if the STB is structured according to SCPI. By analogy, the ESE can be taken as the ENABle part of the ESR.

6.9.2 Overview of the Status Register

The following figure shows the status registers used by the R&S FSH.

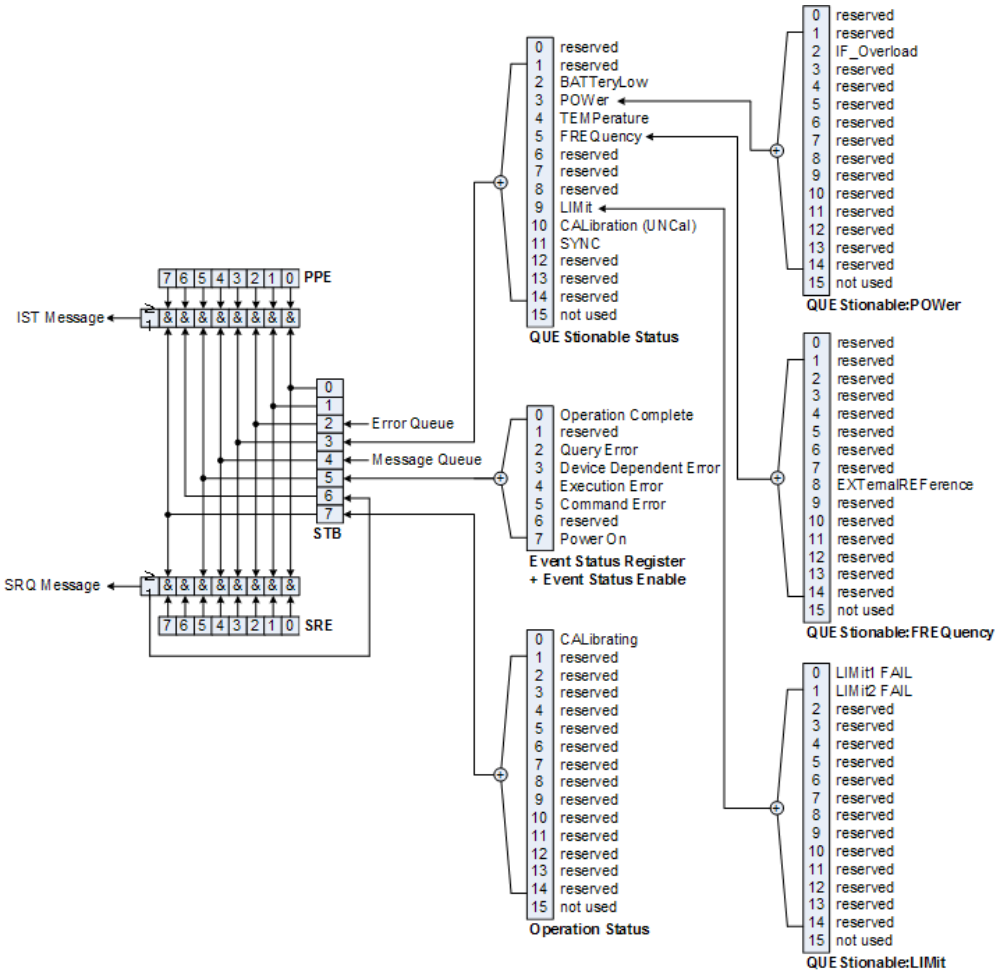


Figure 6-2: Overview of the status registers

6.9.3 Status Byte (STB) & Service Request Enable Register (SRE)

The STB is already defined in IEEE 488.2. It provides a rough overview of the instrument status by collecting the pieces of information of the lower registers. It can thus be compared with the CONDition part of an SCPI register and assumes the highest level within the SCPI hierarchy. A special feature is that bit 6 acts as the sum bit of the remaining bits of the status byte.

The STATUS BYTE is read using the command "*STB?" or a serial poll.

The STB is linked to the SRE. The latter corresponds to the ENABle part of the SCPI registers in its function. Each bit of the STB is assigned a bit in the SRE. Bit 6 of the SRE is ignored. If a bit is set in the SRE and the associated bit in the STB changes from 0 to 1, a service request (SRQ) is generated, which triggers an interrupt in the controller if this is appropriately configured and can be further processed there. The SRE can be set using the command "*SRE" and read using the command "*SRE?"

Table 6-1: Meaning of the bits used in the Status Byte

Bit No.	Meaning
0 to 1	Not used
2	**Error Queue not empty** The bit is set when an entry is made in the error queue. If this bit is enabled by the SRE, each entry of the error queue generates a service request. Thus an error can be recognized and specified in greater detail by polling the error queue. The poll provides an informative error message. This procedure is to be recommended since it considerably reduces the problems involved with remote control.
3	**QUEStionable status sum bit** The bit is set if an EVENt bit is set in the QUEStionable: status register and the associated ENABle bit is set to 1. A set bit indicates a questionable instrument status, which can be specified in greater detail by polling the QUEStionable status register.
4	**MAV bit** (message available) The bit is set if a message is available in the output buffer which can be read. This bit can be used to enable data to be automatically read from the instrument to the controller.
5	**ESB bit** Sum bit of the event status register. It is set if one of the bits in the event status register is set and enabled in the event status enable register. Setting of this bit indicates a serious error which can be specified in greater detail by polling the event status register.
6	**MSS bit** (master status summary bit) The bit is set if the instrument triggers a service request. This is the case if one of the other bits of this registers is set together with its mask bit in the service request enable register SRE.
7	**OPERation status register sum bit** The bit is set if an EVENt bit is set in the OPERation status register and the associated ENABle bit is set to 1. A set bit indicates that the instrument is just performing an action. The type of action can be determined by polling the OPERation status register.

6.9.4 Event Status Register (ESR) and Event Status Enable Register (ESE)

The ESR is defined in IEEE 488.2. It can be compared with the EVENt part of a SCPI register. The event status register can be read out using command *ESR?.

The ESE is the associated ENABle part. It can be set using the command *ESE and read using the command *ESE?.

Table 6-2: Meaning of the bits in the event status register

Bit No.	Meaning
0	**Operation Complete** This bit is set on receipt of the command *OPC exactly when all previous commands have been executed.
1	Not used
2	**Query Error** This bit is set if either the controller wants to read data from the instrument without having sent a query, or if it does not fetch requested data and sends new instructions to the instrument instead. The cause is often a query which is faulty and hence cannot be executed.
3	**Device-dependent Error** This bit is set if a device-dependent error occurs. An error message with a number between -300 and -399 or a positive error number, which denotes the error in greater detail, is entered into the error queue.
4	**Execution Error** This bit is set if a received command is syntactically correct but cannot be performed for other reasons. An error message with a number between -200 and -300, which denotes the error in greater detail, is entered into the error queue.
5	**Command Error** This bit is set if a command is received, which is undefined or syntactically incorrect. An error message with a number between -100 and -200, which denotes the error in greater detail, is entered into the error queue.
6	Not used
7	**Power On** (supply voltage on) This bit is set on switching on the instrument.

6.9.4.1 STATus:OPERation Register

In the CONDition part, this register contains information on which actions the instrument is being executing or, in the EVENt part, information on which actions the instrument has executed since the last reading. It can be read using the commands STATus:OPERation:CONDition? or STATus:OPERation[:EVENt]?.

Table 6-3: Meaning of the bits in the STATus:OPERation register

Bit No.	Meaning
0	**CALibrating** This bit is set as long as the instrument is performing a calibration.
1 to 14	Not used
15	This bit is always 0

6.9.4.2 STATus:QUEStionable Register

This register contains information about indefinite states which may occur if the unit is operated without meeting the specifications. It can be read using the commands STATus:QUEStionable: CONDition? and STATus:QUEStionable[:EVENt]?.

Table 6-4: Meaning of bits in STATus:QUEStionable register

Bit No.	Meaning
0 to 1	These bits are not used
2	**BATTERY LOW** If the instrument is running without any external power supply and the charging level of the internal battery is approximately lower than 5% this bit is set to indcate that the system will be shut down automatically in approx. 5 minutes.
3	Not used
4	**TEMPerature** This bit is set if a questionable temperature occurs.
5 to 8	Not used
9	**LIMit (device-specific)** This bit is set if a limit value is violated
10	**CALibration** The bit is set if a measurement is performed unaligned (label **UNCAL**)
11 to 14	Not used
15	This bit is always 0.

6.9.4.3 STATus:QUEStionable:FREQuency Register

This register contains information about the reference frequency. It can be read using the commands STATus:QUEStionable:LIMit:FREQuency? and STATus:QUEStionable:FREQuency[:EVENt]?.

Table 6-5: Meaning of bits in STATus:QUEStionable:FREQuency register

Bit No.	Meaning
0 to 7	Not used
8	**EXTernal REFerence** This bit is set if an external reference is used.
9 to 14	Not used
15	This bit is always 0.

6.9.4.4 STATus:QUEStionable:LIMit Register

This register contains information about the observance of limit lines. It can be read using the commands STATus:QUEStionable:LIMit:CONDition? and STATus:QUEStionable:LIMit[:EVENt]?.

Table 6-6: Meaning of bits in STATus:QUEStionable:LIMit register

Bit No.	Meaning
0	**LIMit 1 FAIL** This bit is set if limit line 1 is violated.
1	**LIMit 2 FAIL** This bit is set if limit line 2 is violated.
2 to 14	Not used
15	This bit is always 0.

6.9.4.5 STATus:QUEStionable:POWer Register

This register contains information about possible overload states. It can be read using the commands STATus:QUEStionable:POWer:CONDition? and STATus:QUEStionable:POWer[:EVENt]?.

Table 6-7: Meaning of bits in STATus:QUEStionable:POWer register

Bit No.	Meaning
0 to 1	Not used
2	**IF_Overload** This bit is set if the IF path is overloaded. 'IFOVL' is displayed.
3 to 14	Not used
15	This bit is always 0.

6.9.5 Application of the Status Reporting Systems

In order to be able to effectively use the status reporting system, the information contained there must be transmitted to the controller and further processed there. There are several methods which are represented in the following.

6.9.5.1 Service Request

Under certain circumstances, the instrument can send a service request (SRQ) to the controller. Usually this service request initiates an interrupt at the controller, to which the control program can react appropriately. As evident from Fig. 1-4, an SRQ is always initiated if one or several of bits 2, 3, 4, 5 or 7 of the status byte are set and enabled in the SRE. Each of these bits combines the information of a further register, the error queue or the output buffer. The ENABle parts of the status registers can be set so that arbitrary bits in an arbitrary status register initiate an SRQ. In order to make use of the possibilities of the service request effectively, all bits should be set to "1" in enable registers SRE and ESE.

Example

Use of the command *OPC to generate an SRQ at the end of a sweep

```
CALL InstrWrite(analyzer, "*ESE 1")
'Set bit 0 in the ESE (Operation Complete)

CALL InstrWrite(analyzer, "*SRE 32")
'Set bit 5 in the SRE (ESB)?
```

After its settings have been completed, the instrument generates an SRQ.

The SRQ is the only possibility for the instrument to become active on its own. Each controller program should set the instrument in a way that a service request is initiated in the case of malfunction. The program should react appropriately to the service request.

6.9.5.2 Serial Poll

In a serial poll, just as with command *STB, the status byte of an instrument is queried. However, the query is realized via interface messages and is thus clearly faster. The serial-poll method has already been defined in IEEE 488.1 and used to be the only standard possibility for different instruments to poll the status byte. The method also works with instruments which do not adhere to SCPI or IEEE 488.2.

The VISUAL BASIC command for executing a serial poll is IBRSP(). Serial poll is mainly used to obtain a fast overview of the state of several instruments connected to the controller.

6.9.5.3 Query by Means of Commands

Each part of any status register can be read by means of queries. The individual commands are listed in the description of the STATus Subsystem. The returned value is always a number that represents the bit pattern of the queried register. This number is evaluated by the controller program.

Queries are usually used after an SRQ in order to obtain more detailed information on the cause of the SRQ.

6.9.5.4 Error Queue Query

Each error state in the instrument leads to an entry in the error queue. The entries of the error queue are detailed plain-text error messages that can be displayed via manual operation using the setup menu or queried via remote control using the command SYSTem:ERRor?. Each call of SYSTem:ERRor? provides one entry from the error queue. If no error messages are stored there any more, the instrument responds with 0, "No error".

The error queue should be queried after every SRQ in the controller program as the entries describe the cause of an error more precisely than the status registers. Especially in the test phase of a controller program the error queue should be queried regularly since faulty commands from the controller to the instrument are recorded there as well.

6.9.6 Reset Values of the Status Reporting System

Table 6-8 contains the different commands and events causing the status reporting system to be reset. None of the commands, except *RST and SYSTem:PRESet, influences the functional instrument settings. In particular, DCL does not change the instrument settings.

Table 6-8: Resetting the status reporting system

Event	Switching on supply voltage		DCL,SDC			
	Power-On-Status-Clear		(Device Clear, Selected Device Clear)	*RST or SYSTem:PRESet	STATus:PRESet	*CLS
Effect	0	1				
Clear STB,ESR	—	yes	—	—	—	yes
Clear SRE,ESE	—	yes	—	—	—	—
Clear PPE	—	yes	—	—	—	—
Clear EVENt parts of the registers	—	yes	—	—	—	yes
Clear ENABle parts of all OPERation and QUEStionable registers; Fill ENABle parts of all other registers with "1".	—	yes	—	—	yes	—
Fill PTRansition parts with "1"; Clear NTRansition parts	—	yes	—	—	yes	—
Clear error queue	yes	yes	—	—	—	yes
Clear output buffer	yes	yes	yes	1)	1)	1)
Clear command processing and input buffer	yes	yes	yes	—	—	—

1) Every command being the first in a program message, i.e., immediately following a <PROGRAM MESSAGE TERMINATOR> clears the output buffer.

6.9.7 Remote Commands of the Status Reporting System

The following commands control the status-reporting system. *RST does not influence the status registers.

The OPERation status register contains information about the calibration status of the instrument.

The QUEStionable status register contains information about the status of the reference and local oscillator, possible overloads of the instrument and the status of limit checks and limit margins.

The commands are independent from the operating mode.

List of commands

STATus:PRESet

This command resets the edge detectors and ENABle parts of all registers to a defined value. All PTRansition parts are set to FFFFh, i.e. all transitions from 0 to 1 are detected. All NTRansition parts are set to 0, i.e. a transition from 1 to 0 in a CONDition bit is not detected. The ENABle part of the STATus:OPERation and STATus:QUEStionable registers are set to 0, i.e. all events in these registers are not passed on.

Example

```
STAT:PRES
```

Characteristics

*RST value: -
SCPI: conform

STATus:QUEue[:NEXT]

This command returns the earliest entry to the error queue and deletes it.

Positive error numbers indicate device-specific errors, negative error numbers are error messages defined by SCPI. If the error queue is empty, the error number 0, "no error", is returned. This command is identical with the command SYSTem:ERRor.

Example

```
STAT:QUE?
```

Characteristics

*RST value: –
SCPI: conform

STATus:OPERation[:EVENt]?

This command reads out the EVENt section of the OPERation register.

The command at the same time deletes the contents of the EVENt section.

Characteristics

*RST value: -
SCPI: conform

STATus:OPERation:CONDition?

This command reads out the CONDition section of the OPERation register.

The command does not delete the contents of the EVENt section.

Characteristics

*RST value: -
SCPI: conform

STATus:OPERation:ENABle <SumBit>

This command controls the ENABle part of the OPERation register.

The ENABle part allows true conditions in the EVENt part of the status register to bereported in the summary bit. If a bit is 1 in the enable register and its associated event bit transitions to true, a positive transition will occur in the summary bit reported to the next higher level.

Parameter

<SumBit>
0 to 65535

Characteristics

*RST value: -
SCPI: conform

STATus:OPERation:NTRansition <SumBit>

This command controls the Negative TRansition part of the OPERation register.

Setting a bit causes a 1 to 0 transition in the corresponding bit of the associated register. The transition also writes a 1 into the associated bit of the corresponding EVENt register.

Parameter

<SumBit>
0 to 65535

Characteristics

*RST value: -
SCPI: conform

STATus:OPERation:PTRansition <SumBit>

This command controls the Positive TRansition part of the OPERation register.

Setting a bit causes a 0 to 1 transition in the corresponding bit of the associated register. The transition also writes a 1 into the associated bit of the corresponding EVENt register.

Parameter

<SumBit>
0 to 65535

Characteristics

*RST value: -
SCPI: conform

STATus:QUEStionable[:EVENt]?
STATus:QUEStionable:FREQuency[:EVENt]?
STATus:QUEStionable:LIMit[:EVENt]?
STATus:QUEStionable:POWer[EVENt]?

This command reads out the EVENt section of the QUEStionable register.

The command at the same time deletes the contents of the EVENt section.

Characteristics

*RST value: -
SCPI: conform

STATus:QUEStionable:CONDition?
STATus:QUEStionable:FREQuency:CONDition?
STATus:QUEStionable:LIMit:CONDition?
STATus:QUEStionable:POWer:CONDition?

This command reads out the CONDition section of the QUEStionable register.

The command does not delete the contents of the EVENt section.

Characteristics

*RST value: -
SCPI: conform

STATus:QUEStionable:ENABle <SumBit>
STATus:QUEStionable:FREQuency:ENABle <SumBit>
STATus:QUEStionable:LIMit:ENABle <SumBit>
STATus:QUEStionable:POWer:ENABle <SumBit>

This command controls the ENABle part of the QUEStionable register.

The ENABle part allows true conditions in the EVENt part of the status register to bereported in the summary bit. If a bit is 1 in the enable register and its associated event bit transitions to true, a positive transition will occur in the summary bit reported to the next higher level.

Parameter

<SumBit>
0 to 65535

Characteristics

*RST value: -
SCPI: conform

STATus:QUEStionable:NTRansition <SumBit>
STATus:QUEStionable:FREQuency:NTRansition <SumBit>
STATus:QUEStionable:LIMit:NTRansition <SumBit>
STATus:QUEStionable:POWer:NTRansition <SumBit>

This command controls the Negative TRansition part of the QUEStionable register.

Setting a bit causes a 1 to 0 transition in the corresponding bit of the associated register. The transition also writes a 1 into the associated bit of the corresponding EVENt register.

Parameter

<SumBit>
0 to 65535

Example

```
STAT:QUES:NTR 65535
```

Characteristics

*RST value: -
SCPI: conform

STATus:QUEStionable:PTRansition <SumBit>
STATus:QUEStionable:FREQuency:PTRansition <SumBit>
STATus:QUEStionable:LIMit:PTRansition <SumBit>
STATus:QUEStionable:POWer:PTRansition <SumBit>

This command control the Positive TRansition part of the QUEStionable register.

Setting a bit causes a 0 to 1 transition in the corresponding bit of the associated register. The transition also writes a 1 into the associated bit of the corresponding EVENt register.

Parameter

<SumBit>
0 to 65535

Characteristics

*RST value: -
SCPI: conform

Alphabetical List of Remote Commands

Index