13 Remote Control Commands

In the following, all remote-control commands are presented in detail with their parameters and the ranges of numerical values.

For an introduction to remote control and the status registers, refer to:

- Chapter 12, "Network Operation and Remote Control", on page 702
- Chapter B.1, "Additional Basics on Remote Control", on page 1191

13.1 Conventions used in SCPI Command Descriptions

Note the following conventions used in the remote command descriptions:

- **Command usage**
  If not specified otherwise, commands can be used both for setting and for querying parameters.
  If a command can be used for setting or querying only, or if it initiates an event, the usage is stated explicitly.
- **Parameter usage**
  If not specified otherwise, a parameter can be used to set a value and it is the result of a query.
  Parameters required only for setting are indicated as **Setting parameters**.
  Parameters required only to refine a query are indicated as **Query parameters**.
  Parameters that are only returned as the result of a query are indicated as **Return values**.
- **Conformity**
  Commands that are taken from the SCPI standard are indicated as **SCPI confirmed**. All commands used by the R&S SMW follow the SCPI syntax rules.
- **Asynchronous commands**
  A command which does not automatically finish executing before the next command starts executing (overlapping command) is indicated as an **Asynchronous command**.
- **Reset values (*RST)**
  Default parameter values that are used directly after resetting the instrument (`*RST` command) are indicated as ***RST** values, if available.
- **Factory preset values**
  Default parameter values that are reset only by factory preset.
- **Default unit**
  This is the unit used for numeric values if no other unit is provided with the parameter.
- **Manual operation**
  If the result of a remote command can also be achieved in manual operation, a link to the description is inserted.

13.2 Backward Compatibility with Other Rohde & Schwarz Signal Generators

To allow you to use your existing scripts, the R&S SMW accepts also a defined subset of SCPI parameter values of other Rohde & Schwarz signal generators, like the R&S SMU. The R&S SMW accepts these values and maps them automatically to the corresponding newly introduced parameters.

Handling of parameters provided for backward compatibility with other Rohde & Schwarz signal generators

The parameter values provided for backwards compatibility are "setting only" values; the query commands return the corresponding new value.

Example:

`SOURce1:BB:ARBitrary:TRIGger:SOURce EXTernal`

The backwards compatibility parameter value `EXTernal` is accepted

`SOURce1:BB:ARBitrary:TRIGger:SOURce?`

Response: `EGT1`

The query returns the correct new value

13.3 SCPI Command Aliases for Advanced Mode with Multiple Entities

When working in Advanced mode, the R&S SMW can generate the signal of several entities. An entity is a *self-contained independent system*, consisting of a baseband source, a fading simulator, a noise generator (AWGN), and an RF part.

Additional *baseband, fading, and AWGN* SCPI commands are provided to allow consistent addressing of the entities in remote control. All these commands start with the mnemonic (keyword) **`ENTity`**.

The additional SCPI commands are not listed in this description because their syntax is straightforward: the mnemonic `ENTity` is prepended to the existing SCPI header.

Suffixes in the keywords ENTity<ch> and SOURce<hw>

You can address the same configuration with multiple entities in one of the following ways:

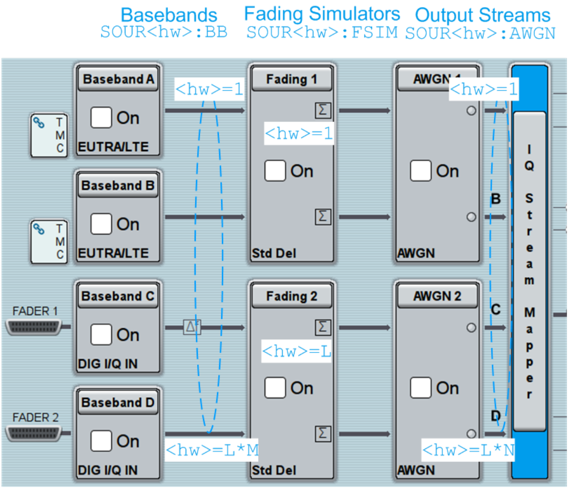
- By using the SCPI commands starting with the keyword `SOURce`
- By using the alias commands that start with the keyword `ENTity`.

You can use these ways alternatively or complementary.

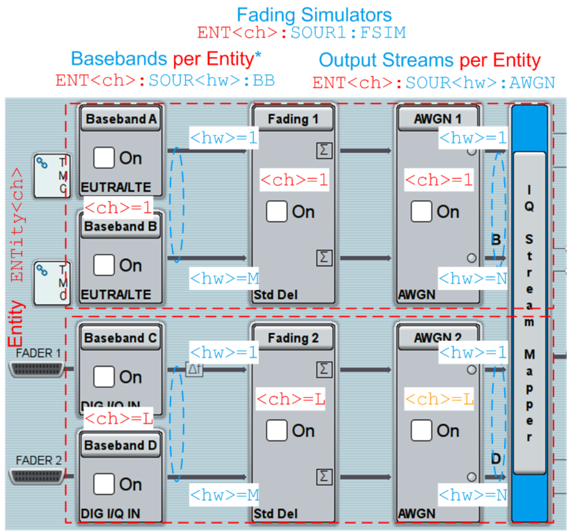
Consider the interdependency in the suffix ranges. Note also that the meaning of the keyword `SOURce<hw>` changes.

The figures in Table 13-1 highlight the differences. The figures illustrate the same LxMxN configuration but address the entities in different way.

Table 13-1: Addressing entities in remote control

<table>
<tr><th>SOURce<hw> only</th><th>ENTity<ch>:SOURce<hw> combination</th></tr>
<tr><td>


LxMxN is the short form of the used system configuration, where:

- L represents the Entity
- M the Baseband
- N the Stream
</td><td>


*) applies for:

- SOUR:BB:FOFF|POFF|PGA
- SOUR:BBIN:...
- SOUR:BB:..., if separate baseband sources are used
</td></tr>
</table>

Example:

- "System Configuration > Mode > Advanced"
- "Entities = 2", "Basebands = 2", "Streams = 2"
- "BB Source Config > Separate Source"

The command `ENTity2:SOURce1:BB:EUTRa:STATe 1` enables the generation of an EUTRA/LTE signal in the second baseband within the second entity.

It is equivalent to `SOURce3:BB:EUTRa:STATe 1`.

Logically, also the following command pairs are equivalent to each other:

- `ENTity2:SOURce1:FSIMulator:STATe` and `SOURce2:FSIMulator:STATe`
- `ENTity2:SOURce1:AWGN:STATe` and `SOURce3:AWGN:STATe`
- `ENTity2:SOURce1:BB:FOFF` and `SOURce3:BB:FOFF`
- `ENTity2:SOURce1:BBIN:DIG:SOUR` and `SOURce3:BBIN:DIG:SOUR`

You can find the corresponding commands also with the "Show SCPI command" functions in the context menu of parameters.

See Chapter 12.12, "How to Find Out the SCPI Command Corresponding to the Manual Operation via "Show SCPI Command"", on page 749.

13.4 Programming Examples

The corresponding sections of the same title provide simple programming examples for the R&S SMW. The purpose of the examples is to present **all** commands for a given task. In real applications, one would rather reduce the examples to an appropriate subset of commands.

The programming examples have been tested with a software tool which provides an environment for the development and execution of remote tests. To keep the examples as simple as possible, only the "clean" SCPI syntax elements are reported. Non-executable command lines (for example comments) start with two // characters.

At the beginning of the most remote control program, an instrument (p)reset is recommended to set the R&S SMW to a definite state. The commands `*RST` and `SYSTem:PRESet` are equivalent for this purpose. `*CLS` also resets the status registers and clears the output buffer.

In all the examples we assume that:

- A remote PC is connected to the instrument
- The remote PC and the instrument are switched on
- A connection between them is established
- The security setting "System Config > Setup > Security > SCPI over LAN" is enabled.

13.5 Common Commands

Common commands are described in the IEEE 488.2 (IEC 625-2) standard. These commands have the same effect and are employed in the same way on different devices. The headers of these commands consist of "*" followed by three letters. Many common commands are related to the Status Reporting System.

Available 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` registers to zero. The command does not alter the mask and transition parts of the registers. It clears the output buffer.

Usage: Setting only

***ESE** <Value>

Event status enable

Sets the event status enable register to the specified value. The query returns the contents of the event status enable register in decimal form.

Parameters:

<Value> Range: 0 to 255

***ESR?**

Event status read

Returns the contents of the event status register in decimal form and subsequently sets the register to zero.

Return values:

<Contents> Range: 0 to 255

Usage: Query only

***IDN?**

Identification

Returns the instrument identification.

Return values:

<ID> "Rohde&Schwarz,<device type>,<part number>/<serial number>,<firmware version>"

Example: `Rohde&Schwarz,SMW200A,`
`1412.0000K02/102030,4.00.023`

Usage: Query only

Manual operation: See "IDN String" on page 721

***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.

Return values:

<ISTflag> 0 | 1

Usage: Query only

***OPC**

Operation complete

Sets bit 0 in the event status register when all preceding commands have been executed. This bit can be used to initiate a service request. The query form writes a "1" into the output buffer as soon as all preceding commands have been executed. This is used for command synchronization.

***OPT?**

Option identification query

Queries the options included in the instrument. For a list of all available options and their description refer to the data sheet.

Return values:

<Options> The query returns a list of options. The options are returned at fixed positions in a comma-separated string. A zero is returned for options that are not installed.

Usage: Query only

Manual operation: See "OPT String" on page 721

***PRE** <Value>

Parallel poll register enable

Sets parallel poll enable register to the indicated value. The query returns the contents of the parallel poll enable register in decimal form.

Parameters:

<Value> Range: 0 to 255

***PSC** <Action>

Power on status clear

Determines whether the contents of the ENABle registers are preserved or reset when the instrument is switched on. Thus a service request can be triggered when the instrument is switched on, if the status registers ESE and SRE are suitably configured. The query reads out the contents of the "power-on-status-clear" flag.

Parameters:

<Action> 0 | 1

0
The contents of the status registers are preserved.

1
Resets the status registers.

***RCL** <Number>

Recall

Loads the instrument settings from an intermediate memory identified by the specified number. The instrument settings can be stored to this memory using the command `*SAV` with the associated number.

It also activates the instrument settings which are stored in a file and loaded using the `MMEMory:LOAD <number>, <file_name.extension>` command.

Manual operation: See "Recall Immediate x" on page 632

***RST**

Reset

Sets the instrument to a defined default status. The default settings are indicated in the description of commands.

The command is equivalent to `SYSTem:PRESet`.

Usage: Setting only

Manual operation: See "Preset" on page 624

***SAV** <Number>

Save

Stores the current instrument settings under the specified number in an intermediate memory. The settings can be recalled using the command `*RCL` with the associated number.

To transfer the stored instrument settings in a file, use the command `:MMEMory:STORe:STATe`.

Manual operation: See "Save Immediate x" on page 632

***SRE** <Contents>

Service request enable

Sets the service request enable register to the indicated value. This command determines under which conditions a service request is triggered.

Parameters:

<Contents> Contents of the service request enable register in decimal form. Bit 6 (MSS mask bit) is always 0.

Range: 0 to 255

***STB?**

Status byte query

Reads the contents of the status byte in decimal form.

Usage: Query only

***TRG**

Trigger

Triggers all actions waiting for a trigger event. In particular, *TRG generates a manual trigger signal. This common command complements the commands of the `TRIGger` subsystem.

Usage: Event

***TST?**

Self-test query

Initiates self-tests of the instrument and returns an error code

Return values:

<ErrorCode> **integer > 0 (in decimal format)**
An error occurred.
(For details see the Service Manual supplied with the instrument).

0
No errors occurred.

Usage: Query only

***WAI**

Wait to continue

Prevents servicing of the subsequent commands until all preceding commands have been executed and all signals have settled (see also command synchronization and `*OPC`).

Usage: Event

13.6 Preset Commands

The preset commands are not bundled in one subsystem. Therefore, they are listed separately in this section. In addition, a specific preset command is provided for each digital standard and for the fader. These specific commands are described in the associated subsystems.

Four presetting actions are available:

- Activating the default state of all internal instrument functions (`*RST` on page 772). Functions that concern the integration of the instrument into a measurement setup are not changed, e.g. GPIB address or reference oscillator source settings.
- Activating the preset state of the parameters related to the selected signal path (`:SOURce<hw>:PRESet` on page 774)
- Activating the preset state of all parameters that are not related to the signal path (`:DEVice:PRESet` on page 774)
- Activating the original state of delivery (factory reset, `:SYSTem:FPReset` on page 775). Only functions that are protected by a password remain unchanged as well as the passwords themselves.

:DEVice:PRESet

Presets all parameters which are not related to the signal path, including the LF generator.

Example: `DEV:PRES`
Presets all instruments settings that are not related to the signal path.

Usage: Event

Manual operation: See "Connect/Disconnect All Remote" on page 149

:SOURce<hw>:PRESet

Presets all parameters which are related to the selected signal path.

Fading simulator (if available) and the transient recorder are only preset by command `*RST`.

Example: `SOUR:PRES`
Presets all settings that are related to signal path

Usage: Event

:SYSTem:PRESet

Triggers an instrument reset. It has the same effect as:

- The PRESET key
- The `*RST` command

For an overview of the settings affected by the preset function, see Table 10-1

Example: `SYST:PRES`
All instrument settings (also the settings that are not currently active) are reset to their default values.

Usage: Setting only

:SYSTem:FPReset

Triggers an instrument reset to the original state of delivery.

Example: `SYST:FPR`
All instrument settings (also the settings that are not currently active) are reset to the factory values.

Usage: Event

Manual operation: See "Execute Factory Preset" on page 625

13.7 MMEMory Subsystem

The `MMEMory` subsystem (Mass MEMory) contains the commands for managing files and directories as well as for loading and storing complete instrument settings in files.

Mass storage location

Without any additional measures, the R&S SMW stores user files on the internal memory, or if selected, on a memory stick.

Both, the user directory `/var/user` on the internal memory or the `/var/usb` directory, can be used to **preserve** user-defined data. Any directory structure can be created.

The `/var/volatile` directory serves as a RAM drive and can be used to protect sensitive information. The data is available **temporarily**.

Default storage location

The R&S SMW stores user data in the user directory.

In the file system, user directory is always indicated as **/var/user**.

In manual control, you access this directory via the "File Manager", see Chapter 10.8, "Using the File Manager", on page 639. In remote control, you can query it with the command `:SYSTem:MMEMory:PATH:USER?`.

To query and change the default directory used for mass storage, use the command `:MMEMory:CDIRectory`.

13.7.1 File Naming Conventions

To enable files to be used in different file systems, consider the following file naming conventions:

- The *filename* can be of any length and *is case-sensitive*, i.e it is distinguished between uppercase and lowercase letters.
- All letters and numbers are permitted (numbers are, however, not permitted at the beginning of the filename).
- Avoid using special characters.
- Do not use slashes "\" and "/". These symbols are used in file paths.
- Avoid using the following filenames: `CLOCK$`, `CON`, `COM1 to COM4`, `LPT1 to LPT3`, `NUL` or `PRN`
  They are reserved by the operating system.

File extension

The file and the optional file *extension* are separated by a period sign. The R&S SMW distinguishes the files according to their extensions; each type of file is assigned a specific file content and hence a specific file extension. Refer to Chapter B.3, "Extensions for User Files", on page 1217 for an overview of the supported file extensions.

Wildcards

The two characters "*" and "?" function as "wildcards", i.e. they are used for selecting several files. The "?" character represents exactly one character, while the "*" character represents all characters up to the end of the filename. "*.*" therefore represents all files in a directory.

Filename and file path

When used in remote control commands, the parameter <filename> is specified as a string parameter with quotation marks. It can contain either the complete path including the root user directory `/var/user` and filename, or only the filename. The filename must include the file extension. The same applies for the directory `/var/volatile` and for the parameters <directory_name> and <path>.

Depending on how much information is provided, the values specified in the parameter or with the command `MMEM:CDIR` are used for the path and drive setting in the commands.

13.7.2 Accessing Files in the Default or in a Specified Directory

For better overview and easy file handling, you may not save all user files in the user directory `/var/user` but rater organize them into subdirectories.

The command syntax defines two general ways to access files with user data in a *specific* directory:

- **Change the current default directory** for mass memory storage and then directly access the files in this default directory, like stored list files, files with user data or save/recall files.
  (See Example "Store the user settings in a file in a specific directory" on page 777).
  The subsequent commands for file handling (select, delete, read out files in the directory, etc.) require only specification of the filename. File extension can be omitted; after syntax evaluation of the used command, the R&S SMW filters out the relevant files.
- Define the **absolute file path**, including the user directory `/var/user`, created subdirectories and filename (see Example "Load file with user data from a specific directory" on page 778).
  As a rule, whenever an absolute file path is determined, it overwrites a previously specified default directory.

The following example explains this rule as a principle. Exceptions of this general rule are stated in the description of the corresponding command. The Chapter 13.7.3, "Programming Examples", on page 778 explains the general working principle with the commands for mass memory storage.

The same rule applies to the `/var/volatile` directory, see Example "Working with files in the volatile memory" on page 780.

Example: Store the user settings in a file in a specific directory

This example uses the commands for storing and loading files with custom digital modulation settings. Working with the files of other subsystems or files containing the settings of a particular digital standard (e.g. `SOURce:BB:EUTRa:SETTing:...`), is analogical.

We assume that the directory `/var/user/DigMod` is existing and contains the files `SaveRecallTest.dm` and `dmSavRcl.dm`.

```
// Set the default directory first
MMEMory:CDIRectory "/var/user/DigMod"
SOURce1:BB:DM:SETTing:CATalog?
// "SaveRecallTest","dmSavRcl"

// Specify only the file name; the extenssion *.dm is assigned automatically
SOURce1:BB:DM:SETTing:STORe "dmSettings"
SOURce1:BB:DM:SETTing:DELete "dmSavRcl"
SOURce1:BB:DM:SETTing:LOAD "SaveRecallTest"
SOURce1:BB:DM:SETTing:CATalog?
// "SaveRecallTest","dmSettings"
```

Example: Load file with user data from a specific directory

This example shows how to use the custom digital modulation commands to set the data source and select a data list. Working with other list files, like control lists (`..:CLISt:..`) or lists with modulation data (`...:MLISt:...`), and the handling of data list files of other subsystems or of a particular digital standard (like `SOURce:BB:EUTRa:...:DSELect`), is analogical.

We assume that the directory `/var/user/lists` is existing and contains the files `dlist1.dm_iqd` and `myDList.dm_iqd`.

```
// Select a data list file as data source
SOURce1:BB:DM:SOURce DLISt

// Query the data list files (*.dm_iqd) in the default directory
MMEMory:CDIRectory "/var/user/lists"
SOURce1:BB:DM:DLISt:CATalog?
// "dlist1","myDList"

// Specify the complete path to select a data list file (*.dm_iqd)
// in the specific directory
MMEMory:CDIRectory
SOURce1:BB:DM:DLISt:SELect "/var/user/lists/myDList"
SOURce1:BB:DM:DLISt:DELete "/var/user/lists/dlist1"
```

13.7.3 Programming Examples

Example: Storing and loading current settings

This example shows two ways of how to store the current instrument setting in the file `settings.savrcltxt` in the directory `/var/user/savrcl`.

Before the instrument settings can be stored in a file, they have to be stored in an intermediate memory using common command `*SAV <number>`. The specified number is then used in the `:MMEMory:STORe:STATe` command.

Also, after loading a file with instrument settings with command `:MMEMory:LOAD:STATe`, these settings have to be activated with the common command `*RCL <number>`.

```
// Store the current settings in an intermediate memory with number 4
*SAV 4

// store the settings in a file in a specific directory;
// the complete path has to be specyfied
MMEMory:STORe:STATe 4,"/var/user/savrcl/settings.savrcltxt"

// store the settings in a file in the default directory;
// set the default directory; specify only the file name
```

```
MMEMory:CDIRectory  "/var/user/savrcl"
*SAV 4
MMEMory:STORe:STATe 4,"settings.savrcltxt"

// Load the stored settings in the intermediate memory 4 and activate them
MMEMory:LOAD:STATe 4,"/var/user/settings.savrcltxt"
*RCL 4
```

Example: Working with files and directories

This example shows how to list files in a directory, list the subdirectories, query the number of files in a directory, create directory, rename and delete files.

```
// Query the current default directory for mass storage,
// change the directory to the default user directory "/var/user"
// and read out the files in it
MMEMory:CDIRectory?
// "/var/user/temp"
MMEMory:CDIRectory
MMEMory:CDIRectory?
// "/var/user/"
MMEMory:CATalog?
// 1282630,8102817792,".,DIR,4096","..,DIR,4096","Log,DIR,4096",
// "settings.savrcltxt,BIN,16949","temp,DIR,4096","test,DIR,4096",
// "list.lsw,BIN,1245201"
// the directory "/var/user" contains the predefined directory "Log",
// the subdirectories "test" and "temp"
// as well as the files "settings.savrcltxt" and "list.lsw"

// query only the subdirectories of the current or specified directory
MMEMory:DCATalog? "/var/user"
// ".","..","Log","temp","test"

// query only number of subdirectories in the current or specified directory
MMEMory:DCATalog:LENGth? "/var/user"
// 5

// query number of files in the current or specified directory
MMEMory:CATalog:LENGth? "/var/user"
// 7

// Create a new directory for mass memory storage in the specified directory
MMEMory:MDIRectory "/var/user/new"

// Copy the file "settings.savrcltxt" into the new directory
MMEMory:COPY "/var/user/settings.savrcltxt","/var/user/new/settings.savrcltxt"

// Rename the file "settings.savrcltxt" into the new directory
// and read out the files in this specific directory
MMEMory:CDIRectory  "/var/user/new"
MMEMory:MOVE "settings.savrcltxt","settings_new.savrcltxt"
```

```
MMEMory:CATalog? "/var/user/new"
// 25141,8102789120,".,DIR,4096","..,DIR,4096","settings_new.savrcltxt,BIN,16949"

// Delete the "test" directory
MMEMory:RDIRectory "/var/user/test"
```

Example: Working with files in the volatile memory

This example shows how to work with files in the `/var/volatile` directory.

```
// Change the default directory for mass storage,
// read out the files, load and play a file with the ARB
MMEMory:CDIRectory "/var/volatile"
MMEMory:CDIRectory?
// "/var/volatile"
MMEMory:CATalog?
//13928,525352960,".,DIR,60","..,DIR,4096","list.lst,BIN,9772"

:SOURce1:LIST:SELect "/var/volatile/list"
:SOURce1:FREQuency:MODE LIST
:OUTPut1:STATe 1
```

13.7.4 Remote Control Commands

:MMEMory:CATalog? <path>

Returns the content of a particular directory.

Query parameters:

<path> string

String parameter to specify the directory.
If you leave out the path, the command returns the contents of the directory selected with `:MMEMory:CDIRectory`.
The path may be relative or absolute.

Return values:

<UsedDiskSpace> Byte size of all files in the directory.

<FreeDiskSpace> Remaining disk space in bytes.

<FileInfo> <NameFileN>,<SuffixFileN>,<SizeFileN>

List of files, separated by commas

<NameFileN>
Name of the file.

<SuffixFileN>
Type of the file. Possible suffixes are: ASCii, BINary, DIRectory

<SizeFileN>
Size of the file in bytes.

Example: See Example "Working with files and directories" on page 779.

Usage: Query only

Manual operation: See "Directory, File List and Filename" on page 631

:MMEMory:CATalog:LENGth? <Path>

Returns the number of files in the current or in the specified directory.

Query parameters:

<Path> string

String parameter to specify the directory. If the directory is omitted, the command queries the content of the current directory, queried with `:MMEMory:CDIRectory` command.

Return values:

<FileCount> integer

Number of files.

Example: See Example "Working with files and directories" on page 779.

Usage: Query only

:MMEMory:CDIRectory <Directory>

Changes the default directory for mass memory storage. The directory is used for all subsequent `MMEM` commands if no path is specified with them.

Parameters:

<Directory>	<directory_name> String containing the path to another directory. The path can be relative or absolute. To change to a higher directory, use two dots '..' .
Example:	See Example "Working with files and directories" on page 779.
Usage:	SCPI confirmed
Manual operation:	See "Directory, File List and Filename" on page 631

:MMEMory:COPY <SourceFile>[,<DestinationFile>]

Copies an existing file to a new file. Instead of just a file, this command can also be used to copy a complete directory together with all its files.

Setting parameters:

<SourceFile>	string String containing the path and file name of the source file
<DestinationFile>	string String containing the path and name of the target file. The path can be relative or absolute. If `<DestinationFile>` is not specified, the `<SourceFile>` is copied to the current directory, queried with the `:MMEMory:CDIRectory` command. **Note:** Existing files with the same name in the destination directory are overwritten without an error message.
Example:	See Example "Working with files and directories" on page 779.
Usage:	Setting only SCPI confirmed
Manual operation:	See "Cut, Copy&Paste and Delete" on page 640

:MMEMory:DATA <Filename>, <BinaryBlock>
:MMEMory:DATA? <Filename>

The setting command writes the block data `<BinaryBlock>` to the file identified by `<Filename>`.

Set the GPIB-bus terminator to `EOI` to ensure correct data transfer.

The query command transfers the specified file from the instrument to the GPIB-bus and then on to the controller. It is important to ensure that the intermediate memory on the controller is large enough to take the file. The setting for the GPIB-bus terminator is irrelevant.

Tip: Use this command to read/transfer stored instrument settings or waveforms directly from/to the instrument.

Parameters:

<BinaryBlock> #<number><length_entry><data>

`#`: Hash sign; always comes first in the binary block
`<number>`: the first digit indicates how many digits the subsequent length entry has
`<length_entry>`: indicates the number of subsequent bytes
`<data>`: binary block data for the specified length.
For files with a size with more than nine digits (gigabytes), the instrument allows the syntax `#(<Length>)`, where `<Length>` is the file size in decimal format.

Parameters for setting and query:

<Filename> string

String parameter to specify the name of the file.

Example:

`MMEMory:DATA '/var/user/test.txt',#15hallo`
Writes the block data to the file `test.txt`.
The digit `1` indicates a length entry of one digit; the digit `5` indicate a length of the binary data (`hallo`) in bytes.
`MMEMory:DATA? '/var/user/test.txt'`
Sends the data of the file `test.txt` from the instrument to the controller in the form of a binary block.
`Response: #15hallo`

Usage: SCPI confirmed

:MMEMory:DATA:UNPRotected <Msus>, <Data>

The **setting** command sends the I/Q data and the marker data to the file defined with the `<Msus>` parameter. The required tags are created automatically so that the file content follows the waveform file format.

The **query** reads out the I/Q data part or the marker information of the specified file.

Waveform files can also be created with the command `[:SOURce<hw>]:BB:ARBitrary:WAVeform:DATA`. In this case, the *complete content* of the waveform file must be specified, i.e. not only the I/Q or marker data but all required tags.

(see Chapter 13.18.4.4, "Waveform, Data and List Format", on page 961).

Parameters:

<Msus> <Identifier>:<file name>

Mass Storage Unit Specifier
A string that specifies whether I/Q data (`NVWFM`) or marker data (`NVMKR`) is transferred and the file name the data is stored in.
If the file with the specified name does not exist, a file is created.
Any existing content in the file is *overwritten*.
Complete file path and file extension can also be specified. If omitted, files are stored in the default directory and the extension `*.wv` is assigned to the file name.

Setting parameters:

<Data> block data

I/Q data or marker binary data, where the number of marker elements has to be equal to the number of I/Q samples
Binary block data follows the syntax:
`#<Digits><Length><I0Q0..IxQx…IN-1QN-1>` or
`#<Digits><Length><M0M1…Mx…MN-1>`

#
Indicates the start of the binary block

<Digits>
Decimal value
Gives the number of decimal digits used for the `<Length>` value

<Length>
Decimal value
Number of bytes the follow in the `<Binary data>` part

I0Q0..IxQx…IN-1QN-1
Binary data in ASCII format
`IxQx…` represents binary data (16-bit signed integer in 2's complement notation) containing the I and Q component alternately and starting with the I component. Each T and Q component consists of 2 bytes in MSB format (most significant byte first). The values of the 2 bytes in an I component and a Q component are in the range: -32768 to +32767.

M0M1…Mx…MN-1
Binary data in ASCII format
`Mx` represents one marker byte, where only the last 4 bits are used. These 4 bits are assigned to the 4 possible markers of the instrument, 1 bit per marker: Bit_0 = Marker 1, ... Bit_3 = Marker 4. One 4-bit marker element is required for every I/Q sample.

Example:

```
:MMEM:DATA:UNPR "NVWFM:/var/user/wave.wv",#220<I0Q0..IxQx>
// the binary data <I0Q0..IxQx> is added to a file named wave.wv
// and saved in the selected directory
// <I0Q0..IxQx> contains of 20 bytes, i.e. 5 I/Q samples,
// 2 bytes for each I and Q component
// <I0Q0..IxQx> is a placeholder;
// the actual ASCII values are not printable

:MMEMory:DATA:UNPRotected? "NVWFM:/var/user/wave.wv"
:MMEMory:DATA:UNPRotected "NVMKR:/var/user/wave.wv",#185*7uuf5*
// the specified marker data is added to a file named wave.wv
// in the specified directory
// used are printable values;
// used are only the last 4 bits of a byte
:MMEMory:DATA:UNPRotected? "NVMKR:/var/user/wave.wv"
// note that, the query returns binary values
SOURce1:BB:ARBitrary:WAVeform:CLOCk "/var/user/wave.wv",1.1E6
SOURce1:BB:ARBitrary:WAVeform:CLOCk? "/var/user/wave.wv"
```

:MMEMory:DCATalog? <path>

Returns the subdirectories of a particular directory.

Query parameters:

<path> String parameter to specify the directory. If the directory is omitted, the command queries the content of the current directory, queried with `:MMEMory:CDIRectory` command.

Return values:

<Catalog> <file_entry>

Names of the subdirectories separated by colons. The first two strings are related to the parent directory.

Example: See Example "Working with files and directories" on page 779.

Usage: Query only

:MMEMory:DCATalog:LENGth? [<Path>]

Returns the number of subdirectories in the current or specified directory.

Query parameters:

<Path> String parameter to specify the directory. If the directory is omitted, the command queries the contents of the current directory, to be queried with `:MMEMory:CDIRectory` command.

Return values:

<DirectoryCount> integer

Number of parent and subdirectories.

Example: See Example "Working with files and directories" on page 779.

Usage: Query only

:MMEMory:DELete <Filename>

Removes a file from the specified directory.

Setting parameters:

<Filename> string

String parameter to specify the name and directory of the file to be removed.

Example: See Example "Working with files and directories" on page 779.

Usage: Event
SCPI confirmed

Manual operation: See "Cut, Copy&Paste and Delete" on page 640

:MMEMory:LOAD:STATe <SavRclStateNumb>, <file_name>

Loads the specified file stored under the specified name in an internal memory.

After the file has been loaded, the instrument setting must be activated using an `*RCL` command.

Setting parameters:

<SavRclStateNumb> Determines to the specific <number> to be used with the `*RCL` command, e.g. `*RCL 4`.

<file_name> String parameter to specify the file name with extension `*.savrcltxt`.

Example: See Example "Storing and loading current settings" on page 778.

Usage: Setting only

Manual operation: See "Recall" on page 632

:MMEMory:MDIRectory <Directory>

Creates a subdirectory for mass memory storage in the specified directory. If no directory is specified, a subdirectory is created in the default directory. This command can also be used to create a directory tree.

Setting parameters:

<Directory> string

String parameter to specify the new directory.

Example: See Example "Working with files and directories" on page 779.

Usage: Event

Manual operation: See "Create New Directory" on page 641

:MMEMory:MOVE <SourceFile>, <DestinationFile>

Moves an existing file to a new location or, if no path is specified, renames an existing file.

Setting parameters:

<SourceFile> string

String parameter to specify the name of the file to be moved.

<DestinationFile> string

String parameters to specify the name of the new file.

Example: See Example "Working with files and directories" on page 779.

Usage: Event
SCPI confirmed

Manual operation: See "Rename " on page 640

:MMEMory:MSIS <Msis>

Defines the drive or network resource (in the case of networks) for instruments with windows operating system, using `msis` (MSIS = Mass Storage Identification String).

Note: Instruments with Linux operating system ignore this command, since Linux does not use drive letter assignment.

Usage: SCPI confirmed

:MMEMory:RDIRectory <Directory>

Removes an existing directory from the mass memory storage system. If no directory is specified, the subdirectory with the specified name is deleted in the default directory.

Setting parameters:

<Directory> string

String parameter to specify the directory to be deleted.

Example: See Example "Working with files and directories" on page 779.

Usage: Event

:MMEMory:STORe:STATe <savrcl_state_nr>, <file_name>

Stores the current instrument setting in the specified file.

The instrument setting must first be stored in an internal memory with the same number using the common command `*SAV`.

Setting parameters:

<savrcl_state_nr> Corresponds to the specific `<number>` defined with the `*SAV` command, e.g. `*SAV 4`.

<file_name>	String parameter to specify the file name with extension `*.savrcltxt`.
Example:	See Example "Storing and loading current settings" on page 778.
Usage:	Event
Manual operation:	See "Save" on page 631

:MEMory:HFRee?

Returns the used and available memory in Kb.

Return values:

<TotalPhysMemKb>	integer Total physical memory.
<ApplicMemKb>	integer Application memory.
<HeapUsedKb>	integer Used heap memory.
<HeapAvailableKb>	integer Available heap memory.
Usage:	Query only

13.8 CALibration Subsystem

The `CALibration` subsystem contains the commands needed for performing internal adjustments. This procedure is triggered by the query commands.

Suffix <hw>

Suffix	Value range	Description
`CALibration<hw>`	[1] to 2	• `CALibration[1]` = Path A (optional suffix) • `CALibration2` = Path B (mandatory suffix)

Understanding the query response

- `0`: error-free execution of the adjustments
- `1`: indicates that an error occurred; the process has been canceled

---

:CALibration:ALL[:MEASure]? [<Force>]

Starts all internal adjustments that do not need external measuring equipment.

Note: If an external frequency converter R&S SZU is connected, the internal adjustments are blocked, and the R&S SMW generates an error message.

Query parameters:

<Force> string

Return values:

<Measure> 0 | 1 | OFF | ON

Example:

```
CAL:ALL:MEAS?
//"0"
// Executes the adjustments of all instrument functions.
// When completed, it indicates that the adjustment
// has been performed successfully.
```

Usage: Query only

Manual operation: See "Adjust All" on page 1161

:CALibration:DATA:FACTory:DATE?

Queries the date of the last factory calibration.

Return values:

<Date> string

Example:

```
CAL:DATA:FACT:DATE?
// "2016-01-01"
```

Usage: Query only

Manual operation: See "Last Factory Calibration" on page 1169

:CALibration:ROSCillator[:DATA] <Data>

Sets a user-defined calibration value for the internal reference frequency.

Parameters:

<Data> integer

Range: 0 to INT_MAX
*RST: 0

Example: See `[:SOURce]:ROSCillator[:INTernal]:ADJust[:STATe]` on page 1106.

:CALibration<hw>:CONTinueonerror <State>

Continues the calibration even though an error was detected. By default adjustments are aborted on error.

Parameters:

<State> 0 | 1 | OFF | ON

*RST: n.a. (factory preset: 0)

Example:
```
CAL:CONT ON
// Continues calibration after an error
```

Manual operation: See "Continue Adjustment on Error" on page 1162

:CALibration:ALL:WARN <UncalWarning>

Generates a warning message, if the instrument is in uncalibrated state.

Parameters:

<UncalWarning> 0 | 1 | OFF | ON

*RST: 0

Example:
```
CALibration:ALL:WARN 1
```

:CALibration<hw>:LEVel:BWIDth <Bandwidth>

Parameters:

<Bandwidth> AUTO | HIGH

*RST: AUTO

:CALibration<hw>:ALL:DATE?

Queries the date of the most recently executed full adjustment.

Return values:

<Date> string

Example:
```
CAL:ALL:DATE?
// "2016-01-01"
```

Usage: Query only

Manual operation: See "Last Full Adjustment" on page 1161

:CALibration<hw>:ALL:TEMP?

Queries the temperature deviation compared to the calibration temperature.

Return values:

<Temperature> string

Example:
```
CALibration:ALL:TEMP?
// "+39.00 K"
```

Usage: Query only

Manual operation: See "Temperature Offset To Last Full Adjustment" on page 1161

:CALibration<hw>:DEBug <State>

Activates logging of the internal adjustments.

Setting parameters:

<State> 0 | 1 | OFF | ON
*RST: 0

Example:
```
CALibration:DEBug 1
```

Usage: Setting only

Manual operation: See "Log Debug Info" on page 1162

:CALibration<hw>:FMOFfset[:MEASure]?

Starts adjustment of the FM/PhiM modulator.

Return values:

<Measure> 0 | 1 | OFF | ON

Example:
```
CALibration:FMOFfset:MEASure?
// starts adjustment
// 0
// Adjustment successful
```

Usage: Query only

:CALibration<hw>:FREQuency[:MEASure]?

Performs all adjustments which affect the frequency.

Return values:

<Measure> 0 | 1 | OFF | ON

Example:
```
// Start adjustments for maximum frequency accuracy
CALibration:FREQuency:MEASure?
// 0
// Adjustments are performed successfully
```

Usage: Query only

:CALibration<hw>:IQModulator:LOCal?

Starts adjustment of the I/Q modulator for the currently set frequency and baseband gain. The I/Q modulator is adjusted with respect to carrier leakage, I/Q imbalance and quadrature.

Return values:

<Local> 0 | 1 | OFF | ON
*RST: 0

Example:
```
// Start adjustment for the I/Q modulator for the currently set frequency
CALibration:IQModulator:LOCal?
// 0
// Adjustments are performed successfully
```

Usage: Query only

Manual operation: See "Adjust I/Q Modulator Current Frequency" on page 419

:CALibration<hw>:LEVel:HACCuracy[:STATe] <State>

Activates level adjustment with high accuracy.

Parameters:

<State> 0 | 1 | OFF | ON
*RST: 0

Example:
```
CALibration:LEVel:HACCuracy:STATe 1
```

:CALibration<hw>:FREQuency:CONVerter:EXTernal?

Queries the calibration state of the external I/Q upconverter R&S SZU.

Return values:

<Success> 0 | 1 | OFF | ON
*RST: 0

Example:
```
CAL1:FREQ:CONV:EXT?
// 1
// The R&S SZU connected in path A is calibrated.
```

Usage: Query only

Manual operation: See "Adjust External Frequency Converter" on page 1162

:CALibration<hw>:LEVel:LOCal?

Starts the adjustment of the RF level.

Return values:

<Result> 0 | 1 | OFF | ON

Example:

```
// Start the adjustment
CALibration:LEVel:LOCal?
// Response: 0
// Level adjustment is performed successfully
```

Usage: Query only

Manual operation: See "Adjust Level At Current Settings" on page 438

:CALibration<hw>:LEVel:STATe <State>

Activates internal level correction.

Parameters:

<State> OFF | ON | 0 | 1

*RST: ON

Example:

```
CALibration:LEVel:STATe 1
```

:CALibration<hw>:LEVel[:MEASure]? [<Force>]

Performs all level adjustments in the selected path.

Query parameters:

<Force> "force"

*RST: force

Return values:

<Measure> 0 | 1

Example:

```
// Start adjustments for maximum level accuracy
CALibration:LEVel:MEASure?"force"
// 0
// Adjustments are performed successfully
```

Usage: Query only

:CALibration<hw>:LOSCillator:COUPling:LOCal?

Adjusts the internal LO level at the I/Q modulator automatically, when an external LO signal is fed.

Return values:

<CouplingLevel> 0 | 1 | OFF | ON

*RST: 1

Example:

```
CALibration:LOSCillator:COUPling:LOCal 1
```

Usage: Query only

Manual operation: See "Adjust LO Level at Current Frequency" on page 453

13.9 CLOCk Subsystem

The `CLOck` subsystem contains the commands for configuration of the signals at the clock input and output connectors.

:CLOCk:INPut:FREQuency?

Returns the measured frequency of the external clock signal.

Return values:

<Frequency> float

Range: 0 to max
Increment: 0.001
*RST: 0

Example: `:CLOCk:INPut:FREQuency?`

Usage: Query only

Options: R&S SMW-B10

Manual operation: See "Measured External Clock" on page 261

13.10 DIAGnostic Subsystem

The `DIAGnostic` subsystem contains the commands used for instrument diagnosis and servicing. SCPI does not define any `DIAGnostic` commands; the commands listed here are all device-specific. All `DIAGnostic` commands are query commands which are not affected by `*RST`.

The test functions are intended for services purposes.

They are thus password-protected functions. Unlock the corresponding protection level to access them, see `:SYSTem:PROTect<ch>[:STATe]`

For more information, see R&S SMW Service Manual.

Common suffixes

The following common suffixes are used in remote commands:

Suffix	Value range	Description
`DIAGnostic<hw>`	[1] | 2	Signal path

Example: Programming example

The example lists the commands required to query assemblies and test points for diagnosis purposes.

```
// Query the modules available in the instrument
// and variant and revision state of a certain assembly
DIAGnostic1:BGINfo:CATalog?
// IEC_BOARD,BNC_BOARD,BBMB,FPNL,SSYN,SYNEX,RFMB,...
DIAGnostic1:BGINfo? "SSYN"
// SSYN 1412.6667.02 05.02 100000

// Query the test points available in the instrument
// and trigger the measurement in a selected test point
DIAGnostic1:POINt:CATalog?
// ASATT6HP_M5V7,ASATT6HP_OFFSET,...
DIAGnostic1:MEASure:POINt? "BBB_CODER_BBB_US"
// -1.000000V/°C
```

:DIAGnostic<hw>:BGINfo:CATalog?

Queries the names of the assemblies available in the instrument.

Return values:

<Catalog> string

List of all assemblies; the values are separated by commas
The length of the list is variable and depends on the instrument equipment configuration.

Example: See Example "Programming example" on page 795.

Usage: Query only

:DIAGnostic<hw>:BGINfo? [<Board>]

Queries information on the modules available in the instrument, using the variant and revision state.

Query parameters:

<Board> string

Module name, as queried with the command `:DIAGnostic<hw>:BGINfo:CATalog?`.
To retrieve a complete list of all modules, omit the parameter.
The length of the list is variable and depends on the instrument equipment configuration.

Return values:

<BgInfo> <Module name> <Module stock number incl. variant> <Module revision> <Module serial number>

List of comma-separated entries, one entry per module.
Each entry for one module consists of four parts that are separated by space characters.

Example: See Example "Programming example" on page 795.

Usage: Query only

Manual operation: See "Assembly" on page 1168

:DIAGnostic<hw>:POINt:CATalog?

Queries the test points available in the instrument.

For more information, see R&S SMW Service Manual.

Return values:

<Catalog> string

List of comma-separated values, each representing a test point

Example: See Example "Programming example" on page 795.

Usage: Query only

:DIAGnostic<hw>[:MEASure]:POINt? <Name>

Triggers the voltage measurement at the specified test point and returns the measured voltage.

For more information, see R&S SMW Service Manual.

Query parameters:

<Name> <test point identifier>

Test point name, as queried with the command `:DIAGnostic<hw>:POINt:CATalog?`

Return values:

<Value> <value><unit>

Example: See Example "Programming example" on page 795.

Usage: Query only

13.11 DISPlay Subsystem

The `DISPlay` system contains the commands to set the power-save mode of the instrument.

Programming Examples

Example: Activating screen saver mode and display update

Use the following commands to switch on the screen saver of your instrument or to automatic display. These settings are particularly useful when you control the instrument remotely.

```
// Set the wait time interval and activate the screen saver
:DISPlay:PSAVe:HOLDoff 10
:DISPlay:PSAVe:STATe ON

// Disable the display of the current frequency and level values in remote control
:DISPlay:ANNotation:ALL ON
// :DISPlay:ANNotation:FREQuency ON
// :DISPlay:ANNotation:AMPLitude ON

// Enable automatic update of the display at defined time intervals
:DISPlay:UPDate ON
```

Example: Querying the dialog IDs, opening and closing dialogs

Use the following commands to query the dialog IDs of all currently open dialogs. The dialog ID is a prerequisite for opening and closing dialogs via the remote control.

The dialog ID is also required to define user key actions.

See Chapter 11.3, "Assigning Actions to the User Key", on page 678.

```
// Query the dialog IDs of all open dialogs
:DISPlay:DIALog:ID?
// CEUtraDLGenSetDlg,_,$A DlgKeyRf_Rosc

// Open and close dialogs via remote control
:DISPlay:DIALog:OPEN "CEUtraDLGenSetDlg,_,$A"
:DISPlay:DIALog:OPEN "DlgKeyRf_Rosc"
:DISPlay:DIALog:CLOSe "DlgKeyRf_Rosc"
:DISPlay:DIALog:CLOSe:ALL
```

:DISPlay:PSAVe:HOLDoff <HoldoffTimeMin>

Sets the wait time for the screen saver mode of the display.

Parameters:

<HoldoffTimeMin> integer

Range: 1 to 60
*RST: n.a. (factory preset: 10)
Default unit: minute

Example: see Example "Activating screen saver mode and display update" on page 797

Manual operation: See "Wait Time" on page 658

:DISPlay:PSAVe[:STATe] <State>

Activates the screen saver mode of the display.

We recommend that you use this mode to protect the display, if you operate the instrument in remote control.

To define the wait time, use the command `:DISPlay:PSAVe:HOLDoff`.

Parameters:

<State> 0 | 1 | OFF | ON

*RST: n.a. (factory preset: 0)

Example: See Example "Activating screen saver mode and display update" on page 797

Manual operation: See "Screen Saver" on page 658

:DISPlay:UPDate <Update>

Activates the refresh mode of the display.

Parameters:

<Update> 0 | 1 | OFF | ON

*RST: n.a. (factory preset: 1)

Example: See Example "Activating screen saver mode and display update" on page 797

Manual operation: See "Display Update is" on page 660

:DISPlay:ANNotation:AMPLitude <State>

Indicates asterisks instead of the level values in the status bar.

Parameters:

<State> 0 | 1 | OFF | ON

*RST: n.a. (factory preset: 1)

Example: See Example "Activating screen saver mode and display update" on page 797

Manual operation: See "Annotation Amplitude" on page 689

:DISPlay:ANNotation:FREQuency <State>

Indicates asterisks instead of the frequency values in the status bar.

Parameters:

<State> 0 | 1 | OFF | ON

*RST: n.a. (factory preset: 1)

Example: See Example "Activating screen saver mode and display update" on page 797

Manual operation: See "Annotation Frequency" on page 689

:DISPlay:ANNotation[:ALL] <State>

Displays asterisks instead of the level and frequency values in the status bar of the instrument.

We recommend that you use this mode if you operate the instrument in remote control.

Parameters:

<State> 0 | 1 | OFF | ON

*RST: 1

Example: See Example "Activating screen saver mode and display update" on page 797

:DISPlay:DIALog:ID?

Returns the dialog identifiers of the open dialogs in a string separated by blanks.

Return values:

<DialogIdList> <DialogID#1>< ><DialogID#2>< > ... < ><DialogID#n>

Dialog identifiers are string without blanks. Blanks are represented as `$$`.

Dialog identifiers `<DialogID>` are composed of two main parts:

`<DialogName>[<OptionalParts>]`

<DialogName>

Meaningful information, mandatory input parameter for the commands:

`:DISPlay:DIALog:OPEN` on page 800

`:DISPlay:DIALog:CLOSe` on page 800

<Optional parts>

String of `$<X>` values, where `<X>` is a character, interpreted as follows:

`$q<DialogQualifier>`: optional dialog qualifier, usually the letter `A` or `B`, as displayed in the dialog title.

`$i<Instances>`: comma-separated list of instance indexes, given in the order `h,c,s,d,g,u,0`. Default is zero; the terminating `",0"` can be omitted.

`$t<TabIds>`: comma-separated indexes or tab names; required, if a dialog is composed of several tabs.

`$x<Left>$y<Top>$h<Left>$w<Top>`: position and size; superfluous information.

Example: See Example "Querying the dialog IDs, opening and closing dialogs" on page 797

Usage: Query only

Manual operation: See "SCPI" on page 680

:DISPlay:DIALog:OPEN <DialogId>

Opens the specified dialog.

Setting parameters:

<DialogId> string

To find out the dialog identifier, use the query `:DISPlay:DIALog:ID?`.

The `<DialogName>` part of the query result is mandatory.

Example: See Example "Querying the dialog IDs, opening and closing dialogs" on page 797

Usage: Setting only

Manual operation: See "SCPI" on page 680

:DISPlay:DIALog:CLOSe <DialogId>

Closes the specified dialog.

Setting parameters:

<DialogId> string

To find out the dialog identifier, use the query `:DISPlay:DIALog:ID?`.

The `<DialogName>` part of the query result is sufficient.

Example: See Example "Querying the dialog IDs, opening and closing dialogs" on page 797

Usage: Setting only

:DISPlay:DIALog:CLOSe:ALL

Closes all open dialogs.

Example: See Example "Querying the dialog IDs, opening and closing dialogs" on page 797

Usage: Event

13.12 FORMat Subsystem

The commands in the `FORMat` subsystem determine the format of data returned by the R&S SMW to the controller. This affects all query commands that return a list of numerical data or block data, noted in the descriptions of the commands. The set data format applies to both paths.

:FORMat:BORDer <Border>

Determines the sequence of bytes within a binary block. This only affects blocks which use the IEEE754 format internally.

Parameters:

<Border> NORMal | SWAPped

NORMal
Expects/sends the *least* significant byte of each IEEE754 floating-point number first and the *most* significant byte last.

SWAPped
Expects/sends the *most* significant byte of each IEEE754 floating-point number first and the *least* significant byte last.

*RST: NORMal

Example: `FORM:BORD SWAP`
transfers the data with the most significant bit first.

:FORMat:SREGister <Format>

Determines the numeric format for responses of the status register.

Parameters:

<Format> ASCii | BINary | HEXadecimal | OCTal

ASCii
Returns the register content as a decimal number.

BINary|HEXadecimal|OCTal
Returns the register content either as a binary, hexadecimal or octal number. According to the selected format, the number starts with #B (binary), #H (hexadecimal) or #O (octal).

*RST: ASCii

Example: `FORM:SREG HEX`
returns the register content as a hexadecimal number.

:FORMat[:DATA] <Data>

Determines the data format the instrument uses to return data via the IEC/IEEE bus.

The instrument automatically detects the data format used by the controller, and assigns it accordingly. Data format determined by this SCPI command is in this case irrelevant.

Parameters:

<Data> ASCii | PACKed

ASCii
Transfers numerical data as plain text separated by commas.

PACKed
Transfers numerical data as binary block data.
The format within the binary data depends on the command.
The various binary data formats are explained in the description of the parameter types.

*RST: ASCii

Example: `FORM ASC`
transfers the data as ASCII data.

13.13 HCOPy Subsystem

The `HCOPy` subsystem contains the commands to generate and save a hard copy of the display.

To access a stored hard copy file, use the commands of the `MEMM` subsystem.

13.13.1 Programming Examples

Example: Store a hard copy of the display

The following example lists commands to configure and execute a hard copy to an automatic named file.

```
// ******************************************************************
// Hard copy settings
// ******************************************************************
:HCOPy:DEVice:LANGuage PNG
:HCOPy:FILE:NAME:AUTO:STATe 1
// defines the output format
// sets the instrument to automatically create output file names

// ******************************************************************
// Configure hard copy options, set automatic naming rules
// An automatically generated file name consists of:
// <Prefix><YYYY><MM><DD><Number>.<Format>
// ******************************************************************
:HCOPy:DEVice:LANGuage BMP
// defines output format *.bmp
:HCOPy:REGion DIALog
// selects the region to be copied
:HCOPy:FILE:AUTO:DIR "usb/HCopy"
// sets destination directory of automatic named file to "/usb/HCopy"
:HCOPy:FILE:NAME:AUTO:FILE:PREFix:STATe 1
:HCOPy:FILE:NAME:AUTO:FILE:PREFix:"hardcopy"
:HCOPy:FILE:NAME:AUTO:FILE:YEAR:STATe 1
:HCOPy:FILE:NAME:AUTO:FILE:MONTh:STATe 1
// uses automatic naming prefix
// sets automatic naming prefix to "hardcopy"
// uses automatic naming date parameters year and month

// ******************************************************************
// Execute and transfer the hard copy
// ******************************************************************
:HCOPy:EXECute
:HCOPy:DATA
// generates a hard copy
// transfers the hard copy to the remote client
:HCOPy:FILE:AUTO:FILE?
// queries the automatic file name
// response: "hardcopy1607001.bmp"
:HCOPy:FILE:AUTO:NUMBer?
// queries the number in the automatic file name
// response: "001"
:HCOPy:FILE:AUTO?
```

```
// queries the path and file name of the automatically generated file
// response: "/usb/HCopy/hardcopy1607001.bmp"
```

13.13.2 Hard Copy Settings

With the following commands, you can configure the settings of a hard copy.

:HCOPy:DATA?

Transfers the hard copy data directly as a NByte stream to the remote client.

Return values:

<Data> block data

Example: See Example "Store a hard copy of the display" on page 803

Usage: Query only

:HCOPy:IMAGe:FORMat <Format>
:HCOPy:DEVice:LANGuage <Language>

Selects the graphic format for the hard copy. You can use both commands alternatively.

Parameters:

<Language> BMP | JPG | XPM | PNG

*RST: PNG

Example: See Example "Store a hard copy of the display" on page 803

Manual operation: See "Format" on page 652

:HCOPy:REGion <Region>

Selects the area to be copied.

You can create a snapshot of the screen or an active dialog.

Parameters:

<Region> ALL | DIALog

*RST: ALL

Example: See Example "Store a hard copy of the display" on page 803

Manual operation: See "Region" on page 653

:HCOPy:FILE[:NAME] <Name>

Determines the file name and path to save the hard copy, provided automatic naming is disabled.

Note: If you have enabled automatic naming, the instrument automatically generates the file name and directory, see Chapter 13.13.3, "Automatic Naming", on page 805.

Parameters:

<Name> string

Example: See Example "Store a hard copy of the display" on page 803

Manual operation: See "File..." on page 652

:HCOPy[:EXECute]

Generates a hard copy of the current display. The output destination is a file.

Example: See Example "Store a hard copy of the display" on page 803

Usage: Event

Manual operation: See "Save" on page 652

13.13.3 Automatic Naming

Use the following commands to automatically assign a file name.

:HCOPy:FILE[:NAME]:AUTO?

Queries path and file name of the hardcopy file, if you have enabled *Automatic Naming*.

Return values:

<Auto> string

Example: See Example "Store a hard copy of the display" on page 803

Usage: Query only

:HCOPy:FILE[:NAME]:AUTO:DIRectory <Directory>

Determines the path to save the hard copy, if you have enabled *Automatic Naming*.

If the directory does not yet exist, the instrument automatically creates a new directory, using the instrument name and `/var/user/` by default.

Parameters:

<Directory> string

*RST: /var/user/

Example: See Example "Store a hard copy of the display" on page 803

Manual operation: See "Path..." on page 654

:HCOPy:FILE[:NAME]:AUTO:DIRectory:CLEar

Deletes all files with extensions `*.bmp`, `*.jpg`, `*.png` and `*.xpm` in the directory set for automatic naming.

Example: See Example "Store a hard copy of the display" on page 803

Usage: Event

Manual operation: See "Clear Path" on page 654

:HCOPy:FILE[:NAME]:AUTO:FILE?

Queries the name of the automatically named hard copy file.

An automatically generated file name consists of:

<Prefix><YYYY><MM><DD><Number>.<Format>.

You can activate each component separately, to individually design the file name.

Return values:

<File> string

Example: See Example "Store a hard copy of the display" on page 803.

Usage: Query only

:HCOPy:FILE[:NAME]:AUTO:STATe <State>

Activates automatic naming of the hard copy files.

Parameters:

<State> 0 | 1 | OFF | ON

*RST: 1

Example: See Example "Store a hard copy of the display" on page 803

Manual operation: See "Automatic Naming" on page 652

:HCOPy:FILE[:NAME]:AUTO[:FILE]:DAY:STATe <State>
:HCOPy:FILE[:NAME]:AUTO[:FILE]:MONTh:STATe <State>
:HCOPy:FILE[:NAME]:AUTO[:FILE]:YEAR:STATe <State>

Uses the date parameters (year, month or day) for the automatic naming. You can activate each of the date parameters separately.

Parameters:

<State> 0 | 1 | OFF | ON

*RST: 1

Example: See Example "Store a hard copy of the display" on page 803

Manual operation: See "Prefix, Year, Month, Day" on page 654

:HCOPy:FILE[:NAME]:AUTO[:FILE]:NUMBer?

Queries the number that is used as part of the file name for the next hard copy in automatic mode.

At the beginning, the count starts at 0. The R&S SMW searches the specified output directory for the highest number in the stored files. It increases this number by one to achieve a unique name for the new file.

The resulting auto number is appended to the resulting file name with at least three digits.

Return values:

<Number> integer

Range: 0 to 999999
*RST: 0

Example: See Example "Store a hard copy of the display" on page 803

Usage: Query only

Manual operation: See "Current Auto Number" on page 654

:HCOPy:FILE[:NAME]:AUTO[:FILE]:PREFix <Prefix>
:HCOPy:FILE[:NAME]:AUTO[:FILE]:PREFix:STATe <State>

Uses the prefix for the automatic generation of the file name, provided `PREF:STAT` is activated.

Parameters:

<State> 0 | 1 | OFF | ON

*RST: 1

Example: See Example "Store a hard copy of the display" on page 803

Manual operation: See "Prefix, Year, Month, Day" on page 654

13.14 KBOard Subsystem

The KBOard subsystem contains the commands to set a connected keyboard.

:KBOard:LAYout <Layout>

Selects the language for an external keyboard and assigns the keys acccordingly.

Parameters:

<Layout> CHINese | DANish | DUTCh | DUTBe | ENGLish | ENGUK | FINNish | FRENch | FREBe | FRECa | GERMan | ITALian | JAPanese | KORean | NORWegian | PORTuguese | RUSSian | SPANish | SWEDish | ENGUS

*RST: n.a. (factory preset: ENGLish)

Example:

```
:KBOard:LAYout US
// activates American keyboard
```

Manual operation: See "USB Keyboard > Layout" on page 658

13.15 OUTPut Subsystem

In the OUTPut subsystem, you can configure the output signals.

The LF output signal is defined with the commands of the Chapter 13.18.10, "SOURce:LFOutput Subsystem", on page 1061 system.

Common suffixes

The following common suffixes are used in remote commands:

Suffix	Value range	Description
OUTPut<hw>	[1] | 2	Signal path

:OUTPut:ALL[:STATe] <State>

Activates all RF output signals of the instrument.

Parameters:

<State> 0 | 1 | OFF | ON

*RST: n.a. (factory preset: 0)

Example: `OUTP:ALL OFF`

switches off all RF output signals.

Manual operation: See "RF On" on page 429

:OUTPut<hw>[:STATe] <State>

Activates the RF output signal.

Parameters:

<State> 0 | 1 | OFF | ON

*RST: 0

Example: `OUTP ON`

Activates the RF output.

Manual operation: See "RF On" on page 429

:OUTPut<hw>[:STATe]:PON <Pon>

Defines the state of the RF output signal when the instrument is switched on.

Parameters:

<Pon> OFF | UNCHanged

*RST: n.a. (factory preset: UNCHanged)

Example: `OUTP:PON OFF`

The RF output is deactivated when the instrument is switched on.

Manual operation: See " Power-On State " on page 660

:OUTPut<hw>:AMODe <AMode>

Sets the attenuator mode at the RF output.

Note: The setting `[:SOURce<hw>]:POWer:ATTenuation:RFOFf:MODE` **`FATTenuation`** has higher priority than the attenuator modes `FIXed` and `MANual`.

Parameters:

<AMode> FIXed | MANual | AUTO

AUTO
The attenuator adjusts the level settings automatically, within the full variation range.

FIXed
The attenuator and amplifier stages are fixed at the current position, providing level settings settings with constant output VSWR. The resulting variation range is calculated according to the position.
To use this mode, activate the ALC (see `[:SOURce<hw>]:POWer:ALC[:STATe]`).

MANual
You can set the level manually, in 10 dB steps.

*RST: AUTO

Example:
```
SOURce:POWer:ALC:STATe 1
OUTPut:AMODe FIXed
```

Manual operation: See " Mode " on page 523

:OUTPut<hw>:BLANk:LIST:STATe <State>

Activates RF output blanking.

Parameters:

<State> 0 | 1 | OFF | ON

*RST: 1

Example: `OUTP:BLAN:LIST:STAT ON`
Activates the RF output blanking.

:OUTPut<hw>:IMPedance?

Queries the impedance of the RF outputs.

Return values:

<Impedance> G1K | G50 | G10K

*RST: G50

Example: `OUTP:IMP?`
queries the impedance of RF output.
`Response: 50`
the impedance is 50 ohms

Usage: Query only

:OUTPut<hw>:AFIXed:RANGe:LOWer?
:OUTPut<hw>:AFIXed:RANGe:UPPer?

Queries the settable minimum/maximum value in mode `:OUTPut:AMODe FIXed`, i.e. when the attenuator is not being adjusted.

See `:OUTPut<hw>:AMODe` on page 809

Return values:

<Upper> float

Increment: 0.01
Default unit: dBm

Example:

```
OUTPut1:AMODe FIXed
OUTPut1:AFIXed:RANGe:UPPer?
// -27
OUTPut1:AFIXed:RANGe:LOW?
// -50
```

Usage: Query only

Manual operation: See " Attenuator Level Range " on page 437

:OUTPut<hw>:PROTection:CLEar

Resets the protective circuit after it has been tripped.

To define the output state, use the command `:OUTPut<hw>[:STATe]`.

Example: `OUTP:PROT:CLE`
Resets the protective circuit of the RF output.

Usage: Event

Manual operation: See "Overload" on page 524

:OUTPut<hw>:PROTection:STATe <State>

Attenuates the RF output signal for about 40 dB to protect external devices against internal signals.

Parameters:

<State> 0 | 1 | OFF | ON

*RST: 0

Example: `OUTP:PROT:STAT ON`
Attenuates the RF Output signal for about 40 dB.

Manual operation: See "Overload" on page 524

:OUTPut<hw>:PROTection:TRIPped?

Queries the state of the protective circuit.

Return values:

<Tripped> 0 | 1 | OFF | ON

*RST: 0

Example: `OUTP:PROT:TRIP`

queries the state of the protective circuit of the RF output.

`Response: 0`

the protective circuit has not tripped.

Usage: Query only

Manual operation: See "Overload" on page 524

13.16 SENSe, READ, INITiate and SLISt Subsystems

These subsystems contain the commands for configuring the power measurements with R&S NRP power sensor connected to the R&S SMW.

The local state is set with the `INIT` command. Switching off the local state enhances the measurement performance. Measurements results can be retrieved in local state on or off.

Sensor parameters are set with the `SENSe` commands.

To start the measurement and retrieve the result, use the `:READ<ch>[:POWer]?` command.

Suffix	Value range	Description
`SENSe<ch>`	[1] to 4	Indicates the sensor Sensor mapping: • `SENSe[1]` - default mapping for sensors connected to the SENSOR connector • `SENSe2` - sensor connected to a USB connector • `SENSe3|4` - further connected sensors to USB connectors, in the connection order Use the `:SLISt` commands to change the sensor mapping
`READ<ch>`	[1] to 4	Sensor assignment
`INIate<hw>`	[1] to 4	Sensor assignment
`ELEMent<ch>`	[1] to 25	Sensor mapping list

Programming examples

Example: Detecting and assigning a power sensor

```
SLISt:LIST?
// Response: "NRP33SN-V-900007-USB Legacy","NRP-Z211-900001-USB Legacy"
// list of automatically detected sensors

SLISt:SCAN:STATe 1
// searches for sensors connected in the LAN or via the USBTMC protocol

SLISt:LIST?
// Response:
// "NRP33SN-V-900007-USB Legacy","NRP-Z211-900001-USB Legacy",
// "NRP33SN-V-900005-USBTMC","NRP33SN-V-900011-LAN"
// list of automatically detected sensors
// the list can contain more entries

SLISt:ELEMent3:MAPPing SENS1
// maps the third sensor from the list to the first sensor channel

SLISt:SENSor:MAP "NRPS18S-100654-USB Legacy", SENS4
// maps the sensor directly to channel 4
```

Example: Performing a simple power measurement

Prerequisite: The sensor is connected to the instrument and mapped to the first sensor channel.

```
:INITiate1:CONTinuous ON
//Switches the continous power measurement on

:READ1?
// Triggers the measurement and displays the results
```

Example: Performing a power measurement with a fixed filter

Prerequisite: The sensor is connected to the instrument and mapped to the first sensor channel.

```
SENSe1:SOURce RF
//Sensor measures the power of the RF signal

SENSe1:FILTer:TYPE NSRatio
//Selects fixed noise filter mode

SENSe1:FILTer:NSRatio 0.02 DB
//Sets the maximum noise component in the result to 0.02 DB

SENSe1:FILTer:NSRatio:MTIMe 10
//Limits the settling time to 10 seconds.

:SENSe1:APERture:DEFault:STATe 0
// Deactivates the default aperture time of the sensor

:SENSe1:APERture:TIMe 10e-6
// Sets the aperture time to 10 us

SENSe1:UNIT DBM
//Selects unit dBm for the measured value

:INITiate:CONTinuous ON
//Switches the continous power measurement on

:READ?
//Triggers the measurement and displays the results
```

:SLISt[:LIST]?

Returns a list of all detected sensors in a comma-separated string.

Return values:

<SensorList> String of comma-separated entries

Each entry contains information on the sensor type, serial number and interface.
The order of the entries does not correspond to the order the sensors are displayed in the "NRP Sensor Mapping" dialog.

Example: See Example "Detecting and assigning a power sensor" on page 813.

Usage: Query only

Manual operation: See " Sensor Mapping List " on page 542

:SLISt:SCAN[:STATe] <State>

Starts the search for R&S NRP power sensors, connected in the LAN or via the USBTMC protocol.

Parameters:

<State> 0 | 1 | OFF | ON

*RST: 0

Example: See Example "Detecting and assigning a power sensor" on page 813.

Manual operation: See " Scan " on page 543

:SLISt:ELEMent<ch>:MAPPing <Mapping>

Assigns an entry from the `:SLISt[:LIST]?` to one of the four sensor channels.

Parameters:

<Mapping> SENS1 | SENSor1 | SENS2 | SENSor2 | SENS3 | SENSor3 | SENS4 | SENSor4 | UNMapped

Sensor channel.

*RST: UNMapped

Example: See Example "Detecting and assigning a power sensor" on page 813.

Manual operation: See " Sensor Mapping List " on page 542

:SLISt:SENSor:MAP <SensorId>, <Mapping>

Assigns a sensor directly to one of the sensor channels, using the sensor name and serial number.

To find out the the sensor name and ID, you can get it from the label of the R&S NRP, or using the command `:SLISt:SCAN[:STATe]`. This command detects all R&S NRP power sensors connected in the LAN or via 'USBTMC protocol.

Setting parameters:

<SensorId> string

<Mapping> enum

Example: See Example "Detecting and assigning a power sensor" on page 813.

Usage: Setting only

Manual operation: See " Sensor Mapping List " on page 542

:INITiate<hw>[:POWer]:CONTinuous <Continuous>

Switches the local state of the continuous power measurement by R&S NRP power sensors on and off. Switching off local state enhances the measurement performance during remote control.

The remote measurement is triggered with `:READ<ch>[:POWer]?`). This command also returns the measurement results. The local state is not affected, measurement results can be retrieved with local state on or off.

Parameters:

<Continuous> 0 | 1 | OFF | ON

*RST: 0

Example: `INIT1:CONT ON`

Switches local state of continuous power measurement on.

Manual operation: See " State " on page 547

:READ<ch>[:POWer]?

Triggers power measurement and displays the results. The sensor returns the result in the unit set with command `:SENSe<ch>:UNIT[:POWer]`

Certain power sensors, such as the R&S NRP-Z81, return two values, first the value of the average level and - separated by a comma - the peak value.

Note: This command does not affect the local state, i.e. you can get results with local state on or off. For long measurement times, we recommend that you use an SRQ for command synchronization (MAV bit).

Suffix:
<ch> 1..3

Return values:
<Power> float or float,float

Example:
`SENS1:UNIT DBM`
Selects unit dBm for presentation of measurement result.
`READ1?`
Queries the measurement result of the sensor.
`Response: -45.6246576745440230`
-45.6 dBm were measured at the given frequency.

Example:
R&S NRP-Z81
`READ1?`
`Response:`
`-55.62403263352178,-22.419472478812476`
-55.6 dBm is the measured average level, -22. 4 dBm is the measured peak level at the given frequency.

Usage: Query only

Manual operation: See " Level (Peak) / Level (Average) " on page 547

:SENSe<ch>:UNIT[:POWer] <Power>

Selects the unit (Watt, dBm or dBµV) of measurement result display, queried with `:READ<ch>[:POWer]?`.

Parameters:
<Power> DBM | DBUV | WATT
*RST: DBM

Example:
`SENS2:UNIT DBM`
Selects dBm as unit for the measured value returned by command `READ`.
`READ2?`
`Response: 7.34`
7.34 dBm are measured by sensor 2.

Manual operation: See " Level (Peak) / Level (Average) " on page 547

:SENSe<ch>[:POWer]:APERture:DEFault:STATe <UseDefAp>

Deactivates the default aperture time of the respective sensor.

To specify a user-defined value, use the command `:SENSe<ch>[:POWer]:APERture:TIMe` on page 818.

Parameters:

<UseDefAp> 0 | 1 | OFF | ON

*RST: 1

Example: See Example "Performing a power measurement with a fixed filter" on page 814.

Manual operation: See "Default Aperture Time" on page 549

:SENSe<ch>[:POWer]:APERture:TIMe <ApTime>

Defines the aperture time (size of the acquisition interval) for the corresponding sensor.

Parameters:

<ApTime> float

Range: depends on connected power sensor
Increment: 1E-9
*RST: depends on connected power sensor

Example: See Example "Performing a power measurement with a fixed filter" on page 814.

Manual operation: See "Aperture Time" on page 550

:SENSe<ch>[:POWer]:CORRection:SPDevice:SELect <Select>

Several S-parameter tables can be stored in a sensor. The command selects a loaded data set for S-parameter correction for the corresponding sensor.

Parameters:

<Select> float

*RST: 0

Manual operation: See " S-Parameter " on page 550

:SENSe<ch>[:POWer]:CORRection:SPDevice:STATe <State>

Activates the use of the S-parameter correction data.

Note: If you use power sensors with attenuator, the instrument automatically activates the use of S-parameter data.

Parameters:

<State> 0 | 1 | OFF | ON

*RST: 0

Example: `SENSe1:POWer:CORRection:SPDevice:STATe 1`
`// activates the use of the S-parameters correction data`

Manual operation: See " S-Parameter " on page 550

:SENSe<ch>[:POWer]:CORRection:SPDevice:LIST?

Queries the list of the S-parameter data sets that have been loaded to the power sensor.

Return values:

<List> string list

*RST: 0

Usage: Query only

Manual operation: See " S-Parameter " on page 550

:SENSe<ch>[:POWer]:DISPlay:PERManent:PRIority <Priority>

Selects average or peak power for permanent display.

Parameters:

<Priority> AVERage | PEAK

*RST: AVERage

Example: `SENS1:DISP:PERM:STAT ON`
The permanent viewer is switched on.
`SENS1:DISP:PERM:PRI AVER`
The measured average power is indicated.

Manual operation: See " Priority " on page 548

:SENSe<ch>[:POWer]:DISPlay:PERManent:STATe <State>

Activates the permanent display of the measured power level results. The instrument also indicates the sensor type, the connection, the measurement source and the offset if set.

Parameters:

<State> 0 | 1 | OFF | ON

*RST: 0

Example: `SENS1:POW:DISP:PERM:STAT ON`
The permanent viewer is switched on.

Manual operation: See " Permanent " on page 547

:SENSe<ch>[:POWer]:FILTer:LENGth:AUTO?

Queries the current filter length in filter mode `AUTO` (`:SENSe<ch>[:POWer]:FILTer:TYPE`)

Return values:

<Auto> float

Range: 1 to 65536

Example:
```
SENS1:FILT:TYPE AUTO
```
Selects auto filter.
```
SENS1:FILT:LENG:AUTO?
```
Queries the automatically set filter length.
```
Response: 1024
```

Usage: Query only

Manual operation: See "Filter Length" on page 549

:SENSe<ch>[:POWer]:FILTer:LENGth[:USER] <User>

Selects the filter length for SENS:POW:FILT:TYPE USER. As the filter length works as a multiplier for the time window, a constant filter length results in a constant measurement time.

The R&S NRP power sensors provide different resolutions for setting the filter length, depending on the used sensor type:

- Resolution = 1 for R&S NRPxx power sensors
- Resolution = 2^n for sensors of the R&S NRP-Zxx family, with n = 1 to 16

The time window is fixed to 20 ms.

Parameters:

<User> float

Range: 1 to 65536
*RST: 1

Example:
```
SENS1:FILT:TYPE USER
// Selects user filter mode
SENS1:FILT:LENG 16
// Sets a filter length of 16.
// The resulting measurement time is 640 ms (2x16x20 ms)
```

Manual operation: See "Filter Length" on page 549

:SENSe<ch>[:POWer]:FILTer:NSRatio <NSRatio>

Sets an upper limit for the relative noise content in fixed noise filter mode (`:SENSe<ch>[:POWer]:FILTer:TYPE`). This value determines the proportion of intrinsic noise in the measurement results.

Parameters:

<NSRatio> float

Range: 0.001 to 1
Increment: 0.001
*RST: 0.01

Example: See Example "Performing a power measurement with a fixed filter" on page 814.

Manual operation: See "Noise/Signal Ratio" on page 549

:SENSe<ch>[:POWer]:FILTer:NSRatio:MTIMe <MTime>

Sets an upper limit for the settling time of the auto-averaging filter in the `NSRatio` mode and thus limits the length of the filter. The filter type is set with command `:SENSe<ch>[:POWer]:FILTer:TYPE`.

Parameters:

<MTime> float

Range: 1 to 999.99
Increment: 0.01
*RST: 4

Example: See Example "Performing a power measurement with a fixed filter" on page 814.

Manual operation: See "Timeout" on page 549

:SENSe<ch>[:POWer]:FILTer:SONCe

Starts searching the optimum filter length for the current measurement conditions. You can check the result with command `:SENS1:POW:FILT:LENG:USER?` in filter mode `USER` (`:SENSe<ch>[:POWer]:FILTer:TYPE`).

Example:
`SENS1:FILT:TYPE USER`
Selects user filter mode.
`SENS1:FILT:SONC`
Activates the search for the optimum filter length.
`SENS1:FILT:LENG?`
Returns the found optimum filter length.
`Response: 128`

Usage: Event

Manual operation: See "Auto Once" on page 549

:SENSe<ch>[:POWer]:FILTer:TYPE <Type>

Selects the filter mode. The filter length is the multiplier for the time window and thus directly affects the measurement time.

Parameters:

<Type> AUTO | USER | NSRatio

AUTO
Automatically selects the filter length, depending on the measured value. The higher the power, the shorter the filter length, and vice versa.

USER
Allows you to set the filter length manually. As the filter-length takes effect as a multiplier of the measurement time, you can achieve constant measurement times.

NSRatio
Selects the filter length (averaging factor) according to the criterion that the intrinsic noise of the sensor (2 standard deviations) does not exceed the specified noise content. You can define the noise content with command `:SENSe<ch>[:POWer]:FILTer:NSRatio`.

Note: To avoid long settling times when the power is low, you can limit the averaging factor limited with the "timeout" parameter (`:SENSe<ch>[:POWer]:FILTer:NSRatio:MTIMe`).

*RST: AUTO

Example: See Example "Performing a power measurement with a fixed filter" on page 814.

Manual operation: See " Filter" on page 548

:SENSe<ch>[:POWer]:FREQuency <Frequency>

Sets the RF frequency of the signal, if signal source "USER" is selected (`:SENSe<ch>[:POWer]:SOURce`).

Parameters:

<Frequency> float

*RST: 1 GHz

Example:
```
SENS1:SOUR USER
```
Selects user-defined source.
```
SENS1:FREQ 2.44GHz
```
Sets the RF frequency of the source which is 2.44 GHz.

Manual operation: See " Frequency " on page 548

:SENSe<ch>[:POWer]:LOGGing:STATe <State>

Activates the recording of the power values, measured by a connected R&S NRP power sensor.

Parameters:

<State> 0 | 1 | OFF | ON

*RST: 0

Example: `SENS:LOGG:STAT ON`

Activates recording of the power measurement of the first sensor.

Manual operation: See "Enable Logging" on page 550

:SENSe<ch>[:POWer]:OFFSet <Offset>

Sets a level offset which is added to the measured level value after activation with command `:SENSe<ch>[:POWer]:OFFSet:STATe`. The level offset allows, e.g. to consider an attenuator in the signal path.

Parameters:

<Offset> float

Range: -100.0 to 100.0

*RST: 0

Default unit: dB

Example: `SENS1:POW:OFFS 10.0`

Sets a level offset of 10 dB

Manual operation: See " Level Offset State,Level Offset" on page 548

:SENSe<ch>[:POWer]:OFFSet:STATe <State>

Activates the addition of the level offset to the measured value. The level offset value is set with command `:SENSe<ch>[:POWer]:OFFSet`.

Parameters:

<State> 0 | 1 | OFF | ON

*RST: 0

Example: `SENS1:POW:OFFS 0.4dB`

Sets a level offset of 0.4 dB

`SENS1:POW:OFFS:STAT ON`

A level offset of 0.4 dB is added to the measured value.

Manual operation: See " Level Offset State,Level Offset" on page 548

:SENSe<ch>[:POWer]:SNUMber?

Queries the serial number of the sensor.

Return values:

<SNumber> string

Example: `SENS1:SNUM?`

Queries the serial number.

Usage: Query only

Manual operation: See " Sensor type and serial number " on page 547

:SENSe<ch>[:POWer]:SOURce <Source>

Determines the signal to be measured.

Note: When measuring the RF signal, the sensor considers the corresponding correction factor at that frequency, and uses the level setting of the instrument as reference level.

Parameters:

<Source> A | USER | RF | B

*RST: A

Example: See Example "Performing a power measurement with a fixed filter" on page 814.

Manual operation: See " Use Frequency Of" on page 548

:SENSe<ch>[:POWer]:STATus[:DEVice]?

Queries if a sensor is connected to the instrument.

Return values:

<Status> 0 | 1 | OFF | ON

*RST: 0

Example:
```
SENS1:STAT?
Response: 1
```
A sensor is connected.

Usage: Query only

Manual operation: See " State " on page 547

:SENSe<ch>[:POWer]:SVERsion?

Queries the software version of the connected R&S NRP power sensor.

Return values:

<SVersion> string

Example:
```
SENS1:POW:SVER?
```
Queries the software version of the power sensor.

Usage: Query only

Manual operation: See "Current Sensors" on page 551

:SENSe<ch>[:POWer]:TYPE?

Queries the sensor type. The type is automatically detected.

Return values:

<Type> string

Example: `SENS1:TYPE?`
Queries the type of sensor.
`Response: NRP-Z21`
The R&S NRP-Z21 sensor is used.

Usage: Query only

Manual operation: See " Sensor type and serial number " on page 547

:SENSe<ch>[:POWer]:ZERO

Performs zeroing of the sensor.

Zeroing is required after warm-up, i.e. after connecting the sensor.

Note: Switch off or disconnect the RF power source from the sensor before zeroing.

We recommend that you zero in regular intervals (at least once a day), if:

- The temperature has varied more than about 5 °C.
- The sensor has been replaced.
- You want to measure very low power.

Example: `SENS1:ZERO`
Executes zeroing.

Usage: Event

Manual operation: See " Zero " on page 547

13.17 SCONfiguration Subsystem

The `SCONfiguration` subsystem contains the commands for defining the system configuration setting, like stream mapping and used fading configuration.

Required options

See "Required options" on page 138.

Suffixes

The following suffixes are used:

Suffix	**Value range**	**Description**
`STReam<st>`	1 to 8	Available IQ streams
`CODer<ch>`, `BBMM<ch>`, `IQOutput<ch>`	1 to 2	DIG IQ connector
`FADer<ch>`	1 to 4	DIG IQ connector

Example: Distributing the I/Q streams to the output connectors (advanced configuration)

The following example lists the commands necessary to fulfill this task.

```
// ******************************************************************
// Reset the system configuration and enable a 1x4x4 MIMO fading config
// ******************************************************************
SCONfiguration:PRESet
SCONfiguration:MODE ADVanced
SCONfiguration:FADing MIMO4X4
SCONfiguration:BASeband:SOURce COUPled
SCONfiguration:APPLy

// ******************************************************************
// Map the I/Q streams to the output connectors
// ******************************************************************
SCONfiguration:OUTPut:MAPPing:STReam1:FOFFset 20E6
SCONfiguration:OUTPut:MAPPing:STReam1:POFFset 0
SCONfiguration:OUTPut:MAPPing:RF1:STReam1:STATe ON
SCONfiguration:OUTPut:MAPPing:RF2:STReam2:STATe ON
SCONfiguration:OUTPut:MAPPing:IQOutput1:STReam3:STATe ON
SCONfiguration:OUTPut:MAPPing:IQOutput2:STReam4:STATe ON
// The digital I/Q outputs are not used
SCONfiguration:OUTPut:MAPPing:BBMM1:STReam1:STATe OFF
SCONfiguration:OUTPut:MAPPing:BBMM2:STReam2:STATe OFF


// Alternatively:
// stream A&C and B&D are added and output at BBMM 1 and BBMM 2 outputs
// SCONfiguration:OUTPut:MAPPing:BBMM1:MODE ADD
// SCONfiguration:OUTPut:MAPPing:BBMM2:MODE ADD
// SCONfiguration:OUTPut:MAPPing:BBMM1:STReam1:STATe ON
// SCONfiguration:OUTPut:MAPPing:BBMM2:STReam2:STATe ON
// SCONfiguration:OUTPut:MAPPing:BBMM1:STReam3:STATe ON
// SCONfiguration:OUTPut:MAPPing:BBMM2:STReam4:STATe ON
```

Example: Connecting and configuring external instrument (advanced configuration)

The following example lists the commands necessary to fulfill this task.

```
SCONfiguration:EXTernal:DISPlay ALL
SCONfiguration:EXTernal:REMote:CLEan
SCONfiguration:EXTernal:REMote:SCAN
SCONfiguration:EXTernal:REMote:SCAN:STATe?
// 0

SCONfiguration:EXTernal:CODer1:DIRection?
// Response: IN
SCONfiguration:EXTernal:BBMM1:DIRection?
```

```
// Response: OUT
SCONfiguration:EXTernal:IQOutput1:DIRection?

SCONfiguration:EXTernal:CODer1:IQConnection:STATe?
// Response: 1
SCONfiguration:EXTernal:BBMM1:IQConnection:STATe?
// Response: 1

SCONfiguration:EXTernal:REMote:LIST?
// Response: SMBV100A,SGS_2,SGT100A,SMBV_1,SMBV_2,SMU200A (100010)
SCONfiguration:EXTernal:REMote:ADD "SGS_1","USB","100007"
SCONfiguration:EXTernal:REMote:ADD "SMU200A (100001)","LAN","10.112.11.125","2"
SCONfiguration:EXTernal:REMote:LIST?
// Response: SMBV100A,SGS_2,SGT100A,SMBV_1,SMBV_2,SMU200A (100010),SGS_1,SMU200A (100001)
SCONfiguration:EXTernal:REMote:REName "SGS_2","SGS (102030)"
SCONfiguration:EXTernal:REMote:LIST?
// Response:
// SMBV100A,SGS (102030),SGT100A,SMBV_1,SMBV_2,SMU200A (100010),SGS_1,SMU200A (100001)

// Detecting a connected external instrument and connecting it
SCONfiguration:EXTernal:BBMM1:REMote:DETect?
// Response: "SGT100A"
SCONfiguration:EXTernal:BBMM1:REMote:ISELect?
// Response: "SGT100A","A"
SCONfiguration:EXTernal:BBMM1:REMote:CONNect

// Manually defining the external instrument
SCONfiguration:EXTernal:IQOutput1:REMote:ISELect "SGS_1","A"
SCONfiguration:EXTernal:IQOutput1:REMote:CONNect

SCONfiguration:EXTernal:IQOutput1:RCONnection:STATe?
// Response: 1
SCONfiguration:EXTernal:BBMM1:RCONnection:STATe?
// Response: 1

SCONfiguration:EXTernal:CODer1:INAMe?
SCONfiguration:EXTernal:BBMM1:INAMe?
// Response: "SGT100A [A]"
SCONfiguration:EXTernal:IQOutput1:INAMe?
// Response: "SGS_1 [A]"

SCONfiguration:EXTernal:BBMM1:RF:COUPling ON
SCONfiguration:EXTernal:BBMM1:RF:FREQuency:OFFSet 0
SCONfiguration:EXTernal:BBMM1:RF:POWer:OFFSet -10
:OUTput1:STATe ON
SCONfiguration:EXTernal:BBMM1:RF:STATe?
// Response: 1

SCONfiguration:EXTernal:IQOutput1:RF:COUPling OFF
SCONfiguration:EXTernal:IQOutput1:RF:FREQuency 2143000000
```

```
SCONfiguration:EXTernal:IQOutput1:RF:POWer -20
SCONfiguration:EXTernal:IQOutput1:RF:STATe ON

SCONfiguration:EXTernal:REMote:INITialization:PREDefined:CATalog?
// Response: "external_reference_10MHz.iec"
SCONfiguration:EXTernal:IQOutput1:REMote:INITialization:PREDefined:FILE
"external_reference_10MHz"
SCONfiguration:EXTernal:IQOutput1:REMote:INITialization:FILE?
// Response: "external_reference_10MHz"

// Alternatively use an user defined initialization file
// MMEMory:CDIRectory "/var/user/RcExtInstr"
// SCONfiguration:EXTernal:REMote:INITialization:USER:CATalog?
// Response: init_seq_sgs
// SCONfiguration:EXTernal:IQOutput1:REMote:INITialization:USER:FILE "init_seq_sgs.iec"

SCONfiguration:EXTernal:ACONnect 1
*RST
SCONfiguration:EXTernal:BBMM1:INAMe?

SCONfiguration:EXTernal:REMote:PURGe
// Disabling remote control of the external instruments
SCONfiguration:EXTernal:REMote:DISConnect[:ALL]
```

Example: Configuring master-slave mode

```
// in the first (master) instrument
SOURce1:ROSCillator:SOURce INT
SCONfiguration:MULTiinstrument:MODe MAST
SCONfiguration:MULTiinstrument:STATe 1
SCONfiguration:APPLy
SOURce1:INPut:USER6:DIRection OUTP
OUTPut1:USER6:SIGNal INSinc

// in the second (slave) instrument
SOURce1:ROSCillator:SOURce EXT
SOURce1:ROSCillator:EXTernal:FREQuency 10MHZ
SCONfiguration:MULTiinstrument:MODe SLAV
SCONfiguration:MULTiinstrument:STATe 1
SOURce1:INPut:USER5:DIRection INP
SOURce1:INPut:USER5:SIGNal OUTSinc
SOURce1:INPut:USER6:DIRection OUTP
OUTPut1:USER6:SIGNal INSinc
SCONfiguration:MULTiinstrument:TRIGger:SYNChronization?
// SYNC
```

Example: Controlling the R&S SMW and external RF instruments with the same application program

If you have to remote control a test setup with R&S SMW and other instruments from a remote computer (external controller), you have to synchronize remote commands (SCPI commands) you send to the different instruments (see Example "Controlling the R&S SMW and external RF instruments with the same application program" on page 829).

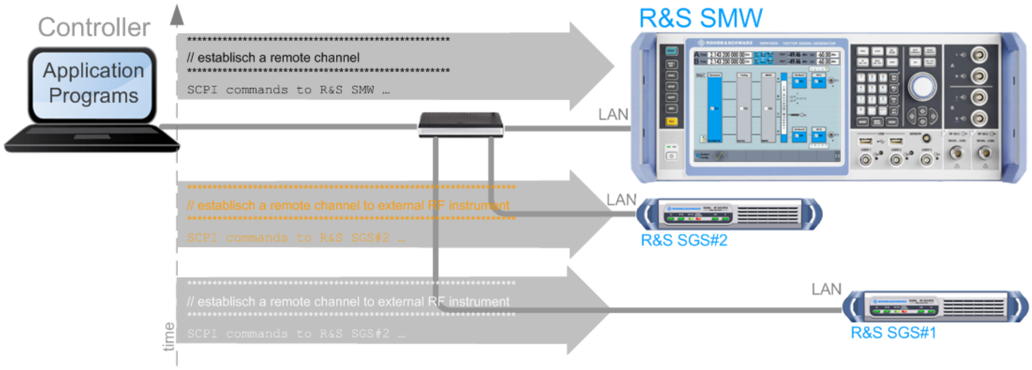


Figure 13-1: Conventional way to remote control the R&S SMW and external RF instruments

This example shows you how to use the same application program to control the external RF instruments via the R&S SMW, without opening remote sessions to these instruments.

Overview of the possible solutions

If the required remote control connections (LAN or USB) to the external instruments are established additionally to the I/Q connections, you can control these external devices from the R&S SMW in one of the following ways:

- In manual operation:
  - Use the "System Configuration > External RF and I/Q" settings and the "External Instruments > Config ...> External Instrument Configuration" settings.
    You can for example set the frequency and level of the connected instruments, as well as activate or deactivate their RF outputs.
    See Chapter 3.9.6, "How to Connect and Configure External Instruments", on page 198
  - Use the "External Instrument Configuration > Remote Control" settings to send individual SCPI commands or a sequence of SCPI commands to the connected external instruments.
- In remote operation:
  - Use the `SCONfiguration:...` remote control commands that correspond to settings in manual operation
    See Example "Connecting and configuring external instrument (advanced configuration)" on page 826
  - Use the `SCONfiguration:EXTernal:<Connector>:REMote:`**`SEND`** commands to control the connected RF instruments
    See "Your R&S SMW as a controller of external instruments" on page 830.

The Figure 13-2 shows the extent of the particular solution.

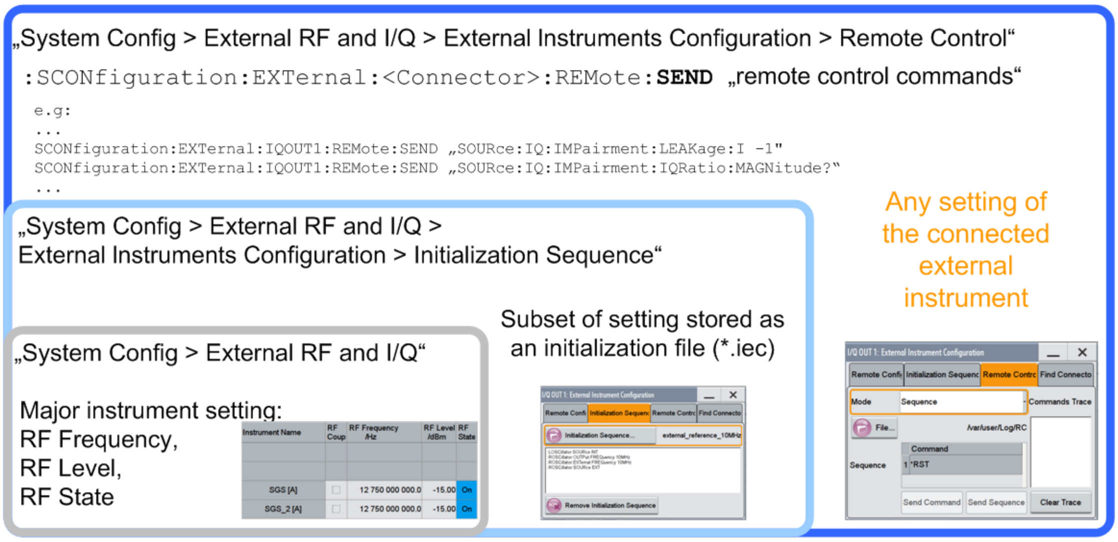


Figure 13-2: Extent and impact of the provided solutions

Your R&S SMW as a controller of external instruments

The Figure 13-3 shows a simplified version of the test setup on Figure 3-19. The figure shows a configuration example of two R&S SGS, directly controlled by an R&S SMW. The figure illustrates the principle and does not show all connections.

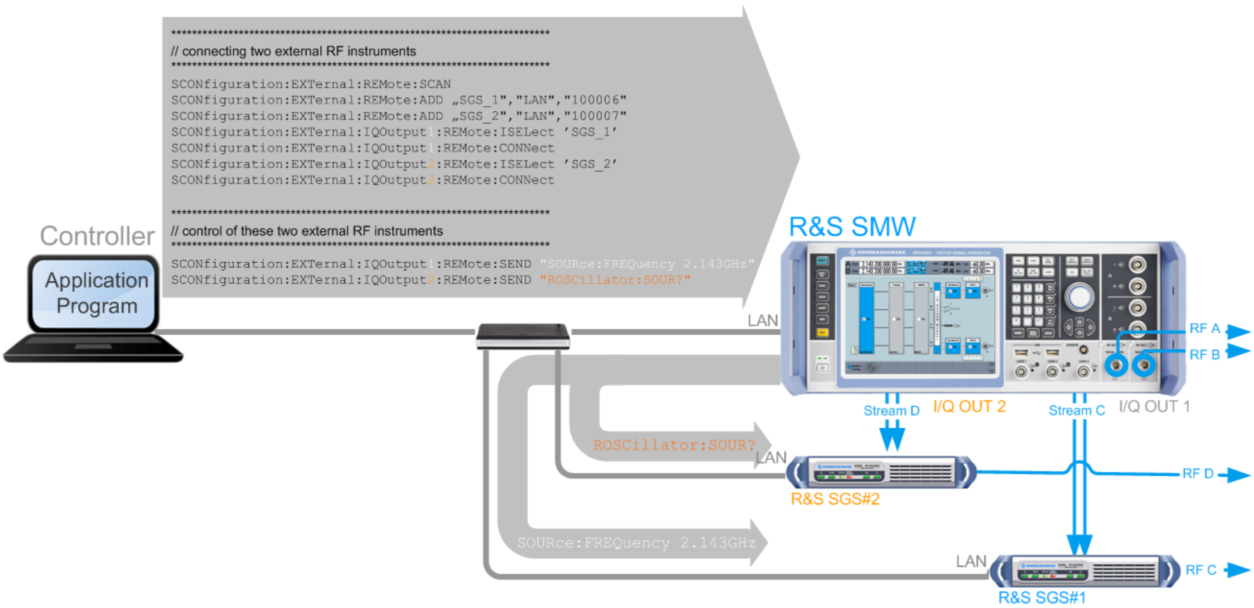


Figure 13-3: Remote control of external RF instruments via the R&S SMW

In this setup, a controller does not need to access the external instruments directly. Instead, the R&S SMW acts as a controller to these instruments.

We assume that:

- You control the R&S SMW from a remote controller

- The instruments are connected to a LAN and there is no active remote channel from the controller to the external instruments
- External RF instruments are connected to the R&S SMW, for example two R&S SGS.

To control an external instrument from the R&S SMW remotely

Perform the following general steps:

1. Connect the external RF instruments to the R&S SMW.

   See:
   - Chapter 3.9.5, "How to Cable the Instruments in MIMO Test Setups", on page 197
   - "To connect instruments to the I/Q analog interface" on page 199
2. Connect a controller and establish a remote control connection to the R&S SMW, see Chapter 12.8, "How to Set Up a Remote Control Connection", on page 732.
3. Write an application program and use the `SCONfiguration:EXTernal:<Connector>:REMote:`**`SEND`** commands to control the connected RF instruments.
   See also Example "Connecting an external instrument and sending SCPI commands to it" on page 832.

Example: Connecting an external instrument and sending SCPI commands to it

The following example lists the commands necessary to fulfill this task. This example uses the configuration made in Example "Connecting and configuring external instrument (advanced configuration)" on page 826.

We assume that an external instrument (R&S SGT) is connected to the BBMM1 connector.

```
SCONfiguration:EXTernal:DISPlay ALL
SCONfiguration:EXTernal:REMote:CLEan
SCONfiguration:EXTernal:REMote:SCAN

SCONfiguration:EXTernal:BBMM1:IQConnection:STATe?
// Response: 1

SCONfiguration:EXTernal:REMote:LIST?
// Response: SMBV100A,SGS_2,SGT100A,SMBV_1,SMBV_2,SMU200A (100010),SGS_1,SMU200A (100001)

// Detecting a connected external instrument and connecting it
SCONfiguration:EXTernal:BBMM1:REMote:DETect?
// Response: "SGT100A"
SCONfiguration:EXTernal:BBMM1:REMote:ISELect?
// Response: "SGT100A","A"
SCONfiguration:EXTernal:BBMM1:REMote:CONNect

SCONfiguration:EXTernal:BBMM1:RCONnection:STATe?
// Response: 1

SCONfiguration:EXTernal:BBMM1:INAMe?
// Response: "SGT100A [A]"

// Sending SCPI commands directly to the connected external instrument
SCONfiguration:EXTernal:BBMM1:REMote:SEND ":SOURce1:FREQuency 2.1GHz"
SCONfiguration:EXTernal:BBMM1:REMote:SEND ":SOURce1:FREQuency?"
```

Example: Generating digital multiplexed signals

The following is a general example. Not all required settings are considered.

```
// ******************************************************************
// Reset the system configuration and enable a 1x2x8 MIMO fading config
SCONfiguration:PRESet
SCONfiguration:MODE ADVanced
SCONfiguration:FADing MIMO2X8

// Enable the generation of digital multiplexed signals
SCONfiguration:OUTPut:MODE DIGM
SCONfiguration:APPLy

// ******************************************************************
// Verify the I/Q stream mapping
SCONfiguration:OUTPut:MAPPing:BBMM1:MODE?
```

```
// Response:  MULT
SCONfiguration:OUTPut:MAPPing:BBMM2:MODE?
// Response:  MULT
SCONfiguration:OUTPut:MAPPing:BBMM1:STReam1:STATe?
// Response: 1
SCONfiguration:OUTPut:MAPPing:BBMM1:STReam3:STATe?
// Response: 1
SCONfiguration:OUTPut:MAPPing:BBMM1:STReam5:STATe?
// Response: 1
SCONfiguration:OUTPut:MAPPing:BBMM1:STReam7:STATe?
// Response: 1
SCONfiguration:OUTPut:MAPPing:BBMM2:STReam2:STATe?
// Response: 1
SCONfiguration:OUTPut:MAPPing:BBMM2:STReam4:STATe?
// Response: 1
SCONfiguration:OUTPut:MAPPing:BBMM2:STReam6:STATe?
// Response: 1
SCONfiguration:OUTPut:MAPPing:BBMM2:STReam8:STATe?
// Response: 1
```


13.17.1 Multi Instrument Configuration

Option: R&S SMW-B9.

:SCONfiguration:MULTiinstrument:MODE <MSMode>

Sets if the instrument works as master or as slave.

Parameters:

<MSMode> SLAVe | MASTer

*RST: MASTer

Example: See Example "Configuring master-slave mode" on page 828

Manual operation: See "Multi Instrument Trigger" on page 610

:SCONfiguration:MULTiinstrument:STATe <TriggerState>

Activates the selected mode.

Parameters:

<TriggerState>	0 | 1 | OFF | ON *RST: 0

Example: See Example "Configuring master-slave mode" on page 828

Manual operation: See "State" on page 610

:SCONfiguration:MULTiinstrument:TRIGger:SYNChronization?

Queries if synchronization is achieved.

Return values:

<SyncSTate>	SYNC | NOSYnc *RST: NOSYnc

Example: See Example "Configuring master-slave mode" on page 828

Usage: Query only

Manual operation: See "Synchronisation State" on page 610

13.17.2 Baseband and Fading Configuration

Option: R&S SMW-B10.

:SCONfiguration:PRESet

Presets the signal routing in the baseband section and the fading configuration to the default state.

Example: see Example "Distributing the I/Q streams to the output connectors (advanced configuration)" on page 826

Usage: Event

Manual operation: See "Set to Default" on page 140

:SCONfiguration:MODE <Mode>

Switches between the operating modes.

Parameters:

<Mode> ADVanced | STANdard | REGenerator

ADVanced|STANdard
Switches between the Standard mode and Advanced mode.

REGenerator
Enables the R&S SMW to work as a radar echo generator.
The fading simulator is disabled.
See user manual R&S®SMW-K78 Radar Echo Generation.

*RST: STANdard

Example: see Example "Distributing the I/Q streams to the output connectors (advanced configuration)" on page 826

Options: REGenerator requires option R&S SMW-K78

Manual operation: See "Mode" on page 140

:SCONfiguration:OUTPut:MODE <Mode>

Defines whether an analog and digital or digital only/digital multiplexed signal is generated.

Parameters:

<Mode> DIGMux | DIGital | ALL

ALL
The generated signal is output at the analog and the digital interfaces

DIGital | DIGMux
The generated signal is output as single stream or multiplexed digital signal at the digital interfaces

*RST: ALL

Example: `SCONfiguration:OUTPut:MODE DIGMux`

Options: DIGMux|DIGital require option R&S SMW-K551

Manual operation: See "Signal Outputs" on page 140

:SCONfiguration:FADing <FadConfig>

Defines the signal routing.

:SCONfiguration:MODE	<FadConfig>
STANdard	FAAFBNone | FANFBB | FAAFBB | FAAFBA | FABFBB | FAABFBN | FANFBAB | FAABFBAB
ADVanced	MIMO1X2 | MIMO2X2 | MIMO2X3 | MIMO2X4 | MIMO3X2 | MIMO3X3 | MIMO3X4 | MIMO4X2 | MIMO4X3 | MIMO4X4 | MIMO1X8 | MIMO8X1 | MIMO2X8 | MIMO8X2 | MIMO2X1 | MIMO2X1X2 | MIMO2X2X1 | MIMO2X1X3 | MIMO2X1X4 | MIMO2X2X2 | MIMO1X3 | MIMO3X1 | MIMO1X4 | MIMO4X1 | MIMO3X1X2 | MIMO3X2X1 | MIMO4X1X2 | MIMO3X2X2 | MIMO4X2X2 | MIMO4X2X1 | SISO2X1X1 | SISO3X1X1 | SISO4X1X1 | SISO5X1X1 | SISO6X1X1 | SISO7X1X1 | SISO8X1X1 | MIMO2X2X4 | MIMO2X4X2 | MIMO2X2X3 | MIMO2X3X1 | MIMO2X3X2 | MIMO2X4X1 MIMO4X8 | MIMO8X4 | MIMO2X4X4 | MIMO2X3X3| MIMO2X3X4 | MIMO2X4X3 MIMO8X8

Parameters:

<FadConfig> FAAFBNone | FANFBB | FAAFBB | SISO2X1X1 | FAAFBA | FABFBB | FAABFBN | FANFBAB | FAABFBAB | MIMO1X2 | MIMO2X2 | MIMO2X3 | MIMO2X4 | MIMO3X2 | MIMO3X3 | MIMO3X4 | MIMO4X2 | MIMO4X3 | MIMO4X4 | MIMO1X8 | MIMO8X1 | MIMO2X8 | MIMO8X2 | MIMO2X1 | MIMO2X1X2 | MIMO2X2X1 | MIMO2X2X2 | MIMO1X3 | MIMO3X1 | MIMO1X4 | MIMO4X1 | MIMO3X1X2 | MIMO3X2X1 | MIMO4X1X2 | MIMO3X2X2 | MIMO4X2X2 | MIMO4X2X1 | SISO3X1X1 | SISO4X1X1 | SISO5X1X1 | SISO6X1X1 | SISO7X1X1 | SISO8X1X1 | MIMO2X2X4 | MIMO2X4X2 | MIMO4X8 | MIMO8X4 | MIMO2X1X3 | MIMO2X1X4 | MIMO2X2X3 | MIMO2X3X1 | MIMO2X3X2 | MIMO2X4X1 | MIMO2X4X4 | MIMO2X3X3 | MIMO2X3X4 | MIMO2X4X3 | MIMO8X8

*RST: FAAFBB

Example: See Example "Distributing the I/Q streams to the output connectors (advanced configuration)" on page 826.

Options: LxMxN configurations with L > 2 require option R&S SMW-K76
Higher order MIMO configurations require option R&S SMW-K75
MIMO8X8 requires R&S SMW-K821

Manual operation: See "Signal Routing" on page 141

:SCONfiguration:MIMO:SUBSet <Subset>

Sets the MIMO subset.

While simulating a 8x8 MIMO mode with two R&S SMW, the MIMO subset defines which fading channels from the MIMO matrix are calculated by the selected instument. The MIMO subset selected in each of the two connected instrument has to be different.

Parameters:

<Subset> SET2 | SET1

*RST: SET1

Example: In the following, we assume that the two R&S SMW are connected and configured as required. The example uses R&S SGS as RF extensions and the internal baseband signal of the first R&S SMW as trigger source for both instruments. If you use common external trigger source, consider to adapt the proposed configuration.

```
*RST
// Select 8x8 MIMO and set MIMO subset
SCONfiguration:MODE ADV
SCONfiguration:FADing MIMO8X8
SCONfiguration:MIMO:SUBSet SET1
SCONfiguration:APPLy
// the instrument generates streams A to D
// configure connectors for synchronous baseband triggering
// alternatively, use an external common trigger source
SOURce1:INPut:USER6:DIRection OUTP
OUTPut1:USER6:SIGNal MTR
// configure the fading simulator
// enable synchronization of the fading process to the baseband trigger
SOURce1:FSIMulator:RESTart:MODE BBTR
SOURce1:FSIMulator:STATe 1
// configure the baseband signal
// incl trigger settings required for the synch. baseband triggering
// if external common trigger is used,
// consider to adapt the configuration
SOURce1:BB:EUTRa:TRIGger:SEQuence ARET
SOURce1:BB:EUTRa:TRIGger:SOURce EGT1
SOURce1:BB:EUTRa:STATe 1
// set RF and level
SOURce1:FREQuency:CW 1950000000
SOURce2:FREQuency:CW 1950000000
SOURce1:POWer:POWer -50
SOURce2:POWer:POWer -50
// connect and configure the RF extensions
// for example R&S SGS
SOURce1:IQ:OUTPut:ANALog:STATe 1
SOURce2:IQ:OUTPut:ANALog:STATe 1
...
// activate the outputs
SOURce2:IQ:STATe 1
SOURce1:IQ:STATe 1
OUTPut2:STATe 1
OUTPut1:STATe 1

// save the configuration
*SAV 1
:MMEMory:STORe:STATe 1,"/var/user/8x8_MIMO_Subset1.savrcl.txt"

// transfer to file to the second instrument
// load the configuration and change the subset
```

```
SCONfiguration:MIMO:SUBSet SET2

// trigger the baseband signal generation and hence the fading process
OUTPut1:USER6:TRIGger:IMMediate
```

Options: R&S SMW-K821

Manual operation: See "Subset" on page 142

:SCONfiguration:BASeband:SOURce <SourConfig>

Determines whether coupled or separated baseband sources are used.

Parameters:

<SourConfig> SEParate | COUPled | CPENtity

SEParate
Enabled in LxMxN configurations with L ≤ 2 and M < 4

COUPled|CPENtity
Enabled in LxMxN configurations with L > 2
`CPENtity` enabled in LxMxN configurations with L > 1 and M > 1.

*RST: SEParate

Example: See Example "Distributing the I/Q streams to the output connectors (advanced configuration)" on page 826

Manual operation: See "Baseband Source Configuration" on page 143

:SCONfiguration:DUPLicate[:STReam] <DuplicateStream>

In a 3x1x1 or 4x1x1 configuration, creates a copy of each stream. Generates are a total number of 6 or 8 streams, where 4 of them can be signals with real-time data source.

Parameters:

<DuplicateStream> 0 | 1 | OFF | ON

*RST: 0

Example:

```
:SCONfiguration:APPLy
:SCONfiguration:MODE ADV
:SCONfiguration:FADing SISO3X1X1
:SCONfiguration:DUPLicate:STReam 1
:SCONfiguration:APPLy
```

Options: R&S SMW-K550

Manual operation: See "Duplicate Streams" on page 143

:SCONfiguration:APPLy

Assigns and confirms the settings.

Example: see Example "Distributing the I/Q streams to the output connectors (advanced configuration)" on page 826

Usage: Event

Manual operation: See "Apply" on page 144

13.17.3 I/Q Stream Mapping

Option: R&S SMW-B10 for commands with the keywords `BBMM<ch>` and `FADer<ch>`.

:SCONfiguration:OUTPut:MAPPing:FADer<ch>:STReam<st>:STATe? <State>
:SCONfiguration:OUTPut:MAPPing:RF<ch>:STReam<st>:STATe <State>
:SCONfiguration:OUTPut:MAPPing:IQOutput<ch>:STReam<st>:STATe <State>
:SCONfiguration:OUTPut:MAPPing:BBMM<ch>:STReam<st>:STATe <State>

Maps the I/Q output streams to the output connectors.

The stream mapping to the FADER connectors is fixed.

Parameters:

<State> 0 | 1 | OFF | ON

*RST: 0

Example: see Example "Distributing the I/Q streams to the output connectors (advanced configuration)" on page 826

Manual operation: See "Map Stream X to Connector" on page 146

:SCONfiguration:OUTPut:MAPPing:FADer<ch>:MODE? <Mode>
:SCONfiguration:OUTPut:MAPPing:RF<ch>:MODE <Mode>
:SCONfiguration:OUTPut:MAPPing:IQOutput<ch>:MODE <Mode>
:SCONfiguration:OUTPut:MAPPing:BBMM<ch>:MODE <Mode>

Enables routing of multiple streams to the same output physical connector and defines the way the streams are internally processed.

Parameters:
<Mode> SINGle | ADD | MULTiplex

ADD enabled for the RF, I/Q OUT and BBMM outputs
MULTiplex enabled for the BBMM outputs and :SCONfiguration:OUTPut:MODEDIGMux

*RST: SINGle

Example: see Example "Distributing the I/Q streams to the output connectors (advanced configuration)" on page 826
see Example "Generating digital multiplexed signals" on page 832

Options: MULTiplex requires R&S SMW-K551

Manual operation: See "Combination Mode" on page 146

:SCONfiguration:OUTPut:MAPPing:STReam<st>:FOFFset <SmFreqOffset>

Sets an absolute frequency offset and shifts streams routed to the RF, I/Q OUT and BBMM connectors in the frequency domain.

Parameters:
<SmFreqOffset> float

Range: depends on the installed options, e.g. -60E6 to +60E6 (R&S SMW-B10)
Increment: 0.01
*RST: 0

Example: See Example "Distributing the I/Q streams to the output connectors (advanced configuration)" on page 826

Manual operation: See "f_{offset}, MHz" on page 145

:SCONfiguration:OUTPut:MAPPing:STReam<st>:POFFset <SmPhasOffset>

Sets the phase offset of the corresponding stream.

Parameters:
<SmPhasOffset> float

Range: -999.99 to 999.99
Increment: 0.01
*RST: 0

Example: see Example "Distributing the I/Q streams to the output connectors (advanced configuration)" on page 826

Manual operation: See "Phase, deg" on page 146

13.17.4 External RF and I/Q Instruments

Option: R&S SMW-B10 for commands with the keywords BBMM<ch> and FADer<ch>.

:SCONfiguration:EXTernal:DISPlay <DisplayMode>

Filters the displayed connectors upon the selected criteria.

Parameters:

<DisplayMode> ALL | MAPPed | INPut | OUTPut

*RST: MAPPed

Example: see Example "Connecting and configuring external instrument (advanced configuration)" on page 826

Manual operation: See "Display" on page 149

:SCONfiguration:EXTernal:ACONnect <State>

Enables automatic detection and connection setup of connected external instruments.

Parameters:

<State> 0 | 1 | OFF | ON

*RST: 0

Example: see Example "Distributing the I/Q streams to the output connectors (advanced configuration)" on page 826

Manual operation: See "Auto Connect" on page 149

:SCONfiguration:EXTernal:REMote:CONNect[:ALL]
:SCONfiguration:EXTernal:REMote:DISConnect[:ALL]

Triggers the instrument to establish the connections to all configured external RF and I/Q instruments or to disconnect all existing connections.

Example: see Example "Connecting and configuring external instrument (advanced configuration)" on page 826

Usage: Event

Manual operation: See "Connect/Disconnect All Remote" on page 149

:SCONfiguration:EXTernal:PBEHaviour <State>

If enabled, the connection to the external instruments is retained on instrument's preset (`*RST`).

Parameters:

<State> 0 | 1 | OFF | ON

*RST: n.a. (factory preset: 0)

Example:

```
SCONfiguration:EXTernal:IQOutput1:INAMe?
// "SZU (999991) B1066"
SCONfiguration:EXTernal:IQOutput1:RCONnection:STATe?
// 1
SOURce:FREQuency:CW?
// 60000000000
SOURce1:IQ:OUTPup:ANALog:TYPE DIFFerential

SCONfiguration:EXTernal:PBEHaviour 1
*RST
SOURce:FREQuency:CW?
// 60000000000
SCONfiguration:EXTernal:IQOutput1:RCONnection:STATe?
// 1
SOURce1:IQ:OUTPup:ANALog:TYPE?
// DIFF

SCONfiguration:EXTernal:PBEHaviour 0
*RST
SOURce:FREQuency:CW?
// 1000000000
SCONfiguration:EXTernal:IQOutput1:RCONnection:STATe?
// 0
SOURce1:IQ:OUTPup:ANALog:TYPE?
// SING
```

Manual operation: See "Preset behavior: Keep connections to external instruments" on page 150

:SCONfiguration:EXTernal:CODer<ch>:DIRection?
:SCONfiguration:EXTernal:FADer<ch>:DIRection?
:SCONfiguration:EXTernal:BBMM<ch>:DIRection?
:SCONfiguration:EXTernal:IQOutput<ch>:DIRection?

Queries the connector direction.

Return values:

<Direction> NONE | IN | OUT

*RST: IN

Example: see Example "Connecting and configuring external instrument (advanced configuration)" on page 826

Usage: Query only

Manual operation: See "Direction" on page 151

:SCONfiguration:EXTernal:CODer<ch>:IQConnection:STATe?
:SCONfiguration:EXTernal:FADer<ch>:IQConnection:STATe?
:SCONfiguration:EXTernal:BBMM<ch>:IQConnection:STATe?

Queries the status of the I/Q connection of the digital interfaces.

Return values:

<IQConnState> 0 | 1 | OFF | ON

Example: See Example "Connecting and configuring external instrument (advanced configuration)" on page 826

Usage: Query only

Manual operation: See "I/Q Connection" on page 152

:SCONfiguration:EXTernal:CODer<ch>:RCONnection:STATe?
:SCONfiguration:EXTernal:FADer<ch>:RCONnection:STATe?
:SCONfiguration:EXTernal:BBMM<ch>:RCONnection:STATe?
:SCONfiguration:EXTernal:IQOutput<ch>:RCONnection:STATe?

Queries the status of the remote connection.

Return values:

<State> 0 | 1 | OFF | ON

Example: See Example "Connecting and configuring external instrument (advanced configuration)" on page 826

Usage: Query only

Manual operation: See "Remote Connection Status" on page 152

:SCONfiguration:EXTernal:CODer<ch>:INAMe?
:SCONfiguration:EXTernal:FADer<ch>:INAMe?
:SCONfiguration:EXTernal:BBMM<ch>:INAMe?
:SCONfiguration:EXTernal:IQOutput<ch>:INAMe?

Queries the name of the connected external instrument.

Return values:

<InstrName> string

Returns the name of the connected external instrument in one of the following formats:

<InstrmentName> (SerialNumber) <Path>
the instrument name, as retrieved via the DIG I/Q interface

<InstrmentName>[, <RfPath>] or <InstrmentName> (Serial-Number)
the instrument name, as defined in with the "Remote Config" settings or as defined by the command `:SCONfiguration:EXTernal:REMote:ADD`

Example: see Example "Connecting and configuring external instrument (advanced configuration)" on page 826

Usage: Query only

Manual operation: See "Instrument Name" on page 153

:SCONfiguration:EXTernal:FADer<ch>:RF:COUPling <RfCouplingState>
:SCONfiguration:EXTernal:BBMM<ch>:RF:COUPling <RfCouplingState>
:SCONfiguration:EXTernal:IQOutput<ch>:RF:COUPling <RfCouplingState>

Enables/disables coupling all major RF setting (like the frequency, level and RF state) of the external instrument to the R&S SMW.

Parameters:

<RfCouplingState> 0 | 1 | OFF | ON

*RST: 1

Example: see Example "Connecting and configuring external instrument (advanced configuration)" on page 826

Manual operation: See "RF Couple" on page 153

:SCONfiguration:EXTernal:FADer<ch>:RF:FREQuency <Frequency>
:SCONfiguration:EXTernal:BBMM<ch>:RF:FREQuency <Frequency>
:SCONfiguration:EXTernal:IQOutput<ch>:RF:FREQuency <Frequency>

In uncoupled mode, sets the RF frequency of the external instrument.

Parameters:

<Frequency> float

Range: 100E3 to 3E9
Increment: 0.01
*RST: 1E9

Example:

```
SCONfiguration:EXTernal:IQOutput1:RF:COUPling OFF
SCONfiguration:EXTernal:IQOutput1:RF:FREQuency 2143E6
// RF frequency of the external instrument is RF = 2.143 GHz
```

Manual operation: See "(Delta) RF Frequency/ RF Level" on page 154

:SCONfiguration:EXTernal:FADer<ch>:RF:FREQuency:OFFSet <FreqOffset>
:SCONfiguration:EXTernal:BBMM<ch>:RF:FREQuency:OFFSet <FreqOffset>
:SCONfiguration:EXTernal:IQOutput<ch>:RF:FREQuency:OFFSet <FreqOffset>

In coupled mode, offsets the RF frequency of the external instrument with the selected delta value.

Parameters:

<FreqOffset> float

Range: -3E9 to 3E9
Increment: 0.01
*RST: 0

Example:

```
SOURce1:FREQuency:CW 2143E6
SCONfiguration:EXTernal:IQOutput1:RF:COUPling ON
SCONfiguration:EXTernal:IQOutput1:RF:FREQuency:OFFSet 20E6
// The resulting RF frequency of the external instrument is
// RF = 2143E6 + 20E6 = 2.163 GHz
// Where both the RF frequency and the frequency offset
// are applied at the external instrument
```

Manual operation: See "(Delta) RF Frequency/ RF Level" on page 154

:SCONfiguration:EXTernal:FADer<ch>:RF:POWer <Power>
:SCONfiguration:EXTernal:BBMM<ch>:RF:POWer <Power>
:SCONfiguration:EXTernal:IQOutput<ch>:RF:POWer <Power>

In uncoupled mode, sets the RF level of the external instrument.

Parameters:

<Power> float

Range: -130 to 20
Increment: 0.01
*RST: -30

Example:

```
SOURce1:POWer:LEVel:IMMediate:AMPLitude -30
SCONfiguration:EXTernal:IQOutput1:RF:COUPling OFF
SCONfiguration:EXTernal:IQOutput1:RF:POWer -20
// RF level of the external instrument is -20 dB
```

Manual operation: See "(Delta) RF Frequency/ RF Level" on page 154

:SCONfiguration:EXTernal:FADer<ch>:RF:POWer:OFFSet <PowerOffset>
:SCONfiguration:EXTernal:BBMM<ch>:RF:POWer:OFFSet <PowerOffset>
:SCONfiguration:EXTernal:IQOutput<ch>:RF:POWer:OFFSet <PowerOffset>

In coupled mode, offsets the RF level of the external instrument with the selected delta value.

Parameters:

<PowerOffset> float

Range: -100 to 100
Increment: 0.01
*RST: 0

Example:

```
SOURce1:POWer:LEVel:IMMediate:AMPLitude -30
SCONfiguration:EXTernal:IQOutput1:RF:COUPling ON
SCONfiguration:EXTernal:IQOutput1:RF:POWer:OFFSet -20
// RF level of the external instrument is -50 dB
// Where both the RF level and the level offset
// are applied at the external instrument
```

Manual operation: See "(Delta) RF Frequency/ RF Level" on page 154

:SCONfiguration:EXTernal:FADer<ch>:RF:STATe <RemConnState>
:SCONfiguration:EXTernal:BBMM<ch>:RF:STATe <RemConnState>
:SCONfiguration:EXTernal:IQOutput<ch>:RF:STATe <RemConnState>

Queries/sets the RF output state of the connected external instrument.

Parameters:

<RemConnState> 0 | 1 | OFF | ON

Example:
```
:SCONfiguration:EXTernal:IQOutput1:RF:COUPling
ON
:OUTput1:STATe ON
:SCONfiguration:EXTernal:IQOutput1:RF:STATe?
```
Response: ON

Example:
```
:SCONfiguration:EXTernal:BBMM1:RF:COUPling OFF
:SCONfiguration:EXTernal:BBMM1:RF:STATe ON
```

Manual operation: See "RF State" on page 154

:SCONfiguration:EXTernal:REMote:SCAN

Scans the network for connected instruments, like R&S SMW, R&S SGS, R&S SMBV, R&S SMU, R&S SMATE and R&S SMJ.

Example: see Example "Connecting and configuring external instrument (advanced configuration)" on page 826

Usage: Event

Manual operation: See "Detect/Scan" on page 156

:SCONfiguration:EXTernal:REMote:SCAN:STATe?

Queries if scanning is performed or not.

To start the scanning process, use the command `:SCONfiguration:EXTernal:REMote:SCAN` on page 849.

Return values:

<ScanState> 0 | 1 | OFF | ON

1
Scanning process running

0
Not scanning

Example: See Example "Connecting and configuring external instrument (advanced configuration)" on page 826

Usage: Query only

Manual operation: See "Detect/Scan" on page 156

:SCONfiguration:EXTernal:REMote:LIST?

Lists all available instruments, found by the `:SCONfiguration:EXTernal:REMote:SCAN` command.

Return values:

<InstrNames> String

String with symbolic names and/or alias names

Example: see Example "Connecting and configuring external instrument (advanced configuration)" on page 826

Usage: Query only

Manual operation: See "Select Instrument" on page 157

:SCONfiguration:EXTernal:REMote:ADD <InstrName>, <HwChan>, <TcpIporUsbAddr>[, <RfPathNumber>]

Adds manually an external instrument to the list of available instruments.

Parameters:

<HwChan> String

Hardware channel (USB or LAN) used by the remote channel to the external instrument

Range: "LAN" to "USB"
*RST: "LAN"

<TcpIporUsbAddr> String

IP address or hostname of the connected external instrument

<RfPathNumber> String

Determines the number of RF paths the external instrument is equipped with

Range: "1" to "2"
*RST: "1"

Setting parameters:

<InstrName> String

Alias name of the instrument

Example: See Example "Connecting and configuring external instrument (advanced configuration)" on page 826

Usage: Setting only

Manual operation: See "Select Instrument" on page 157

:SCONfiguration:EXTernal:REMote:REName

Changes the symbolic name of the instrument.

Example: see Example "Connecting and configuring external instrument (advanced configuration)" on page 826

Usage: Setting only

:SCONfiguration:EXTernal:CODer<ch>:REMote:DETect?
:SCONfiguration:EXTernal:FADer<ch>:REMote:DETect?
:SCONfiguration:EXTernal:BBMM<ch>:REMote:DETect?

Searches for external instruments connected to the particular digital interfaces.

Return values:

<DetectedInstr> string

If the detection fails, the query returns `"None"`.

Example: see Example "Connecting and configuring external instrument (advanced configuration)" on page 826

Usage: Query only

Manual operation: See "Detect/Scan" on page 156

:SCONfiguration:EXTernal:CODer<ch>:REMote:ISELect <InstrName>[, <RfPath>]
:SCONfiguration:EXTernal:FADer<ch>:REMote:ISELect <InstrName>[, <RfPath>]
:SCONfiguration:EXTernal:BBMM<ch>:REMote:ISELect <InstrName>[, <RfPath>]
:SCONfiguration:EXTernal:IQOutput<ch>:REMote:ISELect <InstrName>[, <RfPath>]

Select an external instrument for the selected connector.

Parameters:

<InstrName> String

Instrument alias name, as retrieved with the command `:SCONfiguration:EXTernal:REMote:LIST?` or defined with the command `:SCONfiguration:EXTernal:REMote:ADD`.

<RfPath> String

Determines the used RF output of the external instrument.

Example: See Example "Connecting and configuring external instrument (advanced configuration)" on page 826

Manual operation: See "Select Instrument" on page 157

:SCONfiguration:EXTernal:CODer<ch>:REMote:CONNect
:SCONfiguration:EXTernal:FADer<ch>:REMote:CONNect
:SCONfiguration:EXTernal:BBMM<ch>:REMote:CONNect
:SCONfiguration:EXTernal:IQOutput<ch>:REMote:CONNect

Triggers the connection establishment.

Example: see Example "Connecting and configuring external instrument (advanced configuration)" on page 826

Usage: Event

Manual operation: See "Apply and Connect" on page 158

:SCONfiguration:EXTernal:CODer<ch>:REMote:INFO?
:SCONfiguration:EXTernal:FADer<ch>:REMote:INFO?
:SCONfiguration:EXTernal:BBMM<ch>:REMote:INFO?
:SCONfiguration:EXTernal:IQOutput<ch>:REMote:INFO?

Queries information on the external instrument.

Return values:

<Info> string

Returns information in one of the following formats:

"None"
An external instrument is not assigned

<SymbolicName>,<RemoteChannel>,<RemoteAddress>,<RF Path>
An external instrument is assigned, but not connected

<SymbolicName>,<RemoteChannel>,<RemoteAddress>,<RF Path>[,<*IDN? String>]
An external instrument is assigned and connected

Example:
```
:SCONfiguration:EXTernal:BBMM1:REMote:INFO?
"SGT (101676)","LAN","rssgt100a101676","A",
"Rohde&Schwarz,SGT100A,1419.4501k02/101676,3.1.19.4-3.18.251.99"
```

Usage: Query only

Manual operation: See "Remote Connection Status" on page 152

:SCONfiguration:EXTernal:CODer<ch>:REMote:DISConnect
:SCONfiguration:EXTernal:FADer<ch>:REMote:DISConnect
:SCONfiguration:EXTernal:BBMM<ch>:REMote:DISConnect
:SCONfiguration:EXTernal:IQOutput<ch>:REMote:DISConnect

Disconnects the selected remote connection. To disconnect all remote connections at once, use the command `:SCONfiguration:EXTernal:REMote:DISConnect[:ALL]`.

Example: See Example "Connecting and configuring external instrument (advanced configuration)" on page 826

Usage: Event

Manual operation: See "Remote Connection Status" on page 152

:SCONfiguration:EXTernal:REMote:INITialization:PREDefined:CATalog?
:SCONfiguration:EXTernal:REMote:INITialization:USER:CATalog?

Queries the names of the existing user defined/predefined initialization files.

Per default, the instrument stores user-defined files in the `/var/user/` directory. Use the command `MMEM:CDIRectory` to change the default directory to the currently used one.

Only files with extension `*.iec` are listed.

Example: See Example "Connecting and configuring external instrument (advanced configuration)" on page 826

Usage: Query only

Manual operation: See "Initialization Sequence" on page 159

:SCONfiguration:EXTernal:CODer<ch>:REMote:INITialization:USER:FILE <Filename>
:SCONfiguration:EXTernal:FADer<ch>:REMote:INITialization:USER:FILE <Filename>
:SCONfiguration:EXTernal:BBMM<ch>:REMote:INITialization:USER:FILE <Filename>
:SCONfiguration:EXTernal:IQOutput<ch>:REMote:INITialization:USER:FILE <Filename>

Loads the selected user-defined initialization file (extension `*.iec`).

Per default, the instrument stores user-defined files in the `/var/user/` directory. Use the command `MMEM:CDIRectory` to change the default directory to the currently used one.

Setting parameters:

<Filename> string

For files stored in the default directory, only the file name is required.

Example: See Example "Connecting and configuring external instrument (advanced configuration)" on page 826

Usage: Setting only

Manual operation: See "Initialization Sequence" on page 159

:SCONfiguration:EXTernal:CODer<ch>:REMote:INITialization:PREDefined:FILE <Filename>
:SCONfiguration:EXTernal:FADer<ch>:REMote:INITialization:PREDefined:FILE <Filename>
:SCONfiguration:EXTernal:BBMM<ch>:REMote:INITialization:PREDefined:FILE <Filename>
:SCONfiguration:EXTernal:IQOutput<ch>:REMote:INITialization:PREDefined:FILE <Filename>

Loads the selected predefined initialization file (extension `*.iec`).

Setting parameters:

<Filename> string

Only the file name is required

Example: See Example "Connecting and configuring external instrument (advanced configuration)" on page 826

Usage: Setting only

Manual operation: See "Initialization Sequence" on page 159

:SCONfiguration:EXTernal:CODer<ch>:REMote:INITialization:FILE?
:SCONfiguration:EXTernal:FADer<ch>:REMote:INITialization:FILE?
:SCONfiguration:EXTernal:BBMM<ch>:REMote:INITialization:FILE?
:SCONfiguration:EXTernal:IQOutput<ch>:REMote:INITialization:FILE?

Queries the currently selected initialization file.

Return values:

<Filename> string

filename with file extension (`*.iec`)

Example: see Example "Connecting and configuring external instrument (advanced configuration)" on page 826

Usage: Query only

Manual operation: See "Initialization Sequence" on page 159

:SCONfiguration:EXTernal:REMote:PURGe

Removes unused instruments from the pool of external instruments.

Example: see Example "Connecting and configuring external instrument (advanced configuration)" on page 826

Usage: Event

Manual operation: See "Purge/Clean" on page 156

:SCONfiguration:EXTernal:REMote:CLEan

Removes all instruments from the pool of external instruments.

Example: see Example "Connecting and configuring external instrument (advanced configuration)" on page 826

Usage: Event

Manual operation: See "Purge/Clean" on page 156

:SCONfiguration:EXTernal:CODer<ch>:REMote:SEND <SendScpiCommand>
:SCONfiguration:EXTernal:FADer<ch>:REMote:SEND <SendScpiCommand>

:SCONfiguration:EXTernal:BBMM<ch>:REMote:SEND <SendScpiCommand>
:SCONfiguration:EXTernal:IQOutput<ch>:REMote:SEND <SendScpiCommand>

Allows you to send SCPI commands to the RF instruments connected to the R&S SMW.

Setting parameters:

<SendScpiCommand>"<SCPI syntax>"
String containing an SCPI command (query or setting)

Example: See Chapter 13.17, "SCONfiguration Subsystem", on page 825

Usage: Setting only

Manual operation: See "Command, Send Command" on page 162

:SCONfiguration:EXTernal:IQOutput<ch>:CONNections:CHECk?

Queries the status of the required connections between the R&S SMW and the R&S SZU.

R&S SZU is connected to the R&S SMW via USB.

Return values:

<ConnectionStati> <RF_conn_status>,<I_OUT_conn_status>,<I_BAR_OUT_conn_status>,<Q_OUT_conn_status>,<Q_BAR_OUT_conn_status>,<Trigger_conn_status>

Unused
Connection not used and not required.

Failed
The physical connection is required but missing or is incorrect.

Passed
The required physical connection is up and running.

Example:

```
SCONfiguration:EXTernal:IQOutput1:REMote:ISELect "SZU (999991) B1066"
SOURce1:IQ:OUTPut:ANALog:TYPE SING
SCONfiguration:EXTernal:IQOutput1:REMote:CONNect
SCONfiguration:EXTernal:ACONnect 1
SCONfiguration:EXTernal:PBEHaviour 1

SCONfiguration:EXTernal:IQOutput1:REMote:INFO?
// "SZU (999991) B1066","USB (vendor specific)","999991","A"

SOURce1:FREQuency:CONVerter:EXTernal:CHECk:CONNections?
// Passed,Failed,Unused,Failed,Unused,Passed
// The R&S SZU is connected to the R&S SMW
// The USB and the RF A to LO In connections are correct
// There is a failure in the single-ended I/Q connection
// (between the I/Q OUT and the I/Q IN connectors)
```

Usage: Query only

Manual operation: "Check Connections Settings" on page 163

13.18 SOURce Subsystem

The SOURce subsystem contains the commands for configuring the digital and analog signals.

Common suffixes

The following common suffixes are used in remote commands:

Suffix	Value range	Description
ENTity<ch>	1 to 4	Entity in a multiple entity configuration
SOURce<hw>	[1] to 8	Baseband signals/paths ● SOURce[1] = Path A/Baseband A (optional keyword and suffix) ● SOURce2 = Path B/Baseband B (mandatory keyword and suffix) ● LF output = SOURce:LFOutput (SOURce is optional keyword)
OUTPut<ch>	[1] to 3	Available markers

You can address multiple entities configurations by using the SCPI commands starting with the keyword SOURce or the alias commands starting with the keyword ENTity.

See also Chapter 13.3, "SCPI Command Aliases for Advanced Mode with Multiple Entities", on page 767.

13.18.1 Connector Settings

This section summarizes the commands of the OUTPut and INPut subsystems, necessary to configure the local and global connectors settings. Listed are the commands for configuring the output signals and the inputs for trigger, data, and control signals.

The `...:USER<ch>:...` commands determine the global trigger threshold and input impedance values, that affect all trigger and control signal inputs. The global connector settings concern to all sweeps, digital modulations, the generation of waveforms or multi carrier signals, and all digital standards, in both paths.

Refer to Chapter 13.18.7, "SOURce:INPut Subsystem", on page 1047 for description of the commands for configuring the inputs for external modulation signals.

Required options

See Chapter 11.2.1, "Required Options", on page 661.

Example: Global connectors settings configuration

```
:SOURce1:INPut:USER1:DIRection INP
:SOURce1:INPut:USER1:SIGNal TRIG1
:SOURce1:INPut:USER3:DIRection INP
:SOURce1:INPut:USER3:SIGNal NSEGM1
:SOURce1:INPut:USER4:DIRection INP
:SOURce1:INPut:USER4:SIGNal IPULSA
:SOURce1:INPut:USER5:DIRection INP
:SOURce1:INPut:USER5:SIGNal TRIG2
:SOURce1:INPut:USER:TRIGger:LEVel 1.5
// the signal threshold at USER 1-3 is 1.5 V
// this applies to all input signal, i.e. TRIG1 and NSEGM1
:SOURce1:INPut:USER:PULM:LEVel 2
// the signal threshold at USER 4-6 is 2 V
// this applies to all input signal, i.e. IPULSA and TRIG2
:SOURce1:INPut:USER1:TRIGger:IMPedance G1K
:SOURce1:INPut:USER1:TRIGger:SLOPe POS

:SOURce1:INPut:USER2:DIRection OUTP
:OUTPut1:USER2:SIGNal MARKA2

:SOURce1:INPut:USER6:DIRection OUTP
:OUTPut1:USER6:SIGNal MTR
:OUTPut1:USER6:TRIGger:IMMediate
```

[:SOURce<hw>]:INPut:TM<ch>:DIRection <Direction>
:OUTPut<hw>:TM<ch>:DIRection <Direction>

Determines whether the connector is used as an input or an output.

Suffix:

OUTPut<hw> | SOURce<hw>	1 | 2 Determines the baseband the connectors belong to, where: `OUTPut1|SOURce1` = Baseband A/C and `OUTPut2|SOURce2` = Baseband B/D
TM<ch>	1|2|3 Determines the local connector, where: `OUTPut1:TM1|2|3=SOURce1:INPut:TM1|2|3` = TMC1/2/3 of Baseband A/C and `OUTPut2:TM1|2|3=SOURce2:INPut:TM1|2|3` = TMC4/5/6 of Baseband B/D

Parameters:

<Direction>	INPut | OUTPut *RST: OUTPut

Manual operation: See "T/M/(C) Connector Direction" on page 675

[:SOURce]:INPut:USER<ch>:DIRection <Direction>
:OUTPut:USER<ch>:DIRection <Direction>

Determines whether the connector is used as an input or an output.

Suffix:

USER<ch>	1 to 6

Parameters:

<Direction>	INPut | OUTPut | UNUSed `UNUSed` = the connector is not defined

Example: See Example "Global connectors settings configuration" on page 857

Manual operation: See "User 1 .. 6 Connector Direction" on page 671

[:SOURce<hw>]:INPut:TM<ch>:SIGNal <Signal>

Determines the control signal that is input at the selected connector.

To define the connector direction, use the command `[:SOURce<hw>]:INPut:TM<ch>:DIRection`.

Suffix:

SOURce<hw>
: 1 | 2
Determines the baseband the connectors belong to, where:
`SOURce1` = Baseband A/C and
`SOURce2` = Baseband B/D

TM<ch>
: 1|2|3
Determines the local connector, where:
`SOURce1:INPut:TM1|2|3` = TMC1/2/3 of Baseband A/C and
`SOURce2:INPut:TM1|2|3` = TMC4/5/6 of Baseband B/D

Parameters:

<Signal>
: TRIGger | CLOCk | FEEDback | DATA
`CLOCk` is available only for `TM1`
`DATA` is available only for `TM2`
`FEEDback` is available only for `TM3`

Manual operation: See "Signal" on page 675

:OUTPut<hw>:TM<ch>:SIGNal <Signal>

Determines the control signal that is output at the selected connector.

To define the connector direction, use the command `:OUTPut<hw>:TM<ch>:DIRection`.

Suffix:

OUTPut<hw>
: 1 | 2
Determines the baseband the connectors belong to, where:
`OUTPut1` = Baseband A/C or Baseband 1/3
`OUTPut2` = Baseband B/D or Baseband 2/4

TM<ch>
: 1|2|3
Determines the local connector, where:
`OUTPut1:TM1|2|3` = TMC1/2/3 of Baseband A/C or 1/3 and
`OUTPut2:TM1|2|3` = TMC4/5/6 of Baseband B/D or 2/4

Parameters:
<Signal> MARKA1 | MARKA2 | MARKA3 | SCLA | LATTA | BGATA | HOPA | CWMODA | TRIGA | MARKB1 | MARKB2 | MARKB3 | SCLB | LATTB | BGATB | HOPB | CWMODB | TRIGB | MARKC1 | MARKC2 | MARKC3 | SCLC | LATTC | BGATC | HOPC | CWMODC | TRIGC | MARKD1 | MARKD2 | MARKD3 | SCLD | LATTD | BGATD | HOPD | CWMODD | TRIGD

`MARKA1|MARKC1|MARKA2|MARKC2|MARKA3|MARKC3` = Baseband A/C Marker 1/2/3
`SCLA|SCLB|SCLC|SCLD` = Symbol Clock A/B/C/D
`LATTA|LATTB|LATTC|LATTD` = Lev Att A/B/C/D
`BGATA|BGATB|BGATC|BGATD` = Burst Gate A/B/C/D
`HOPA|HOPB|HOPC|HOPD` = HOP A/B/C/D
`CWMODA|CWMODB|CWMODC|CWMODD` = CW/Mod A/B/C/D
`TRIGA|TRIGB|TRIGC|TRIGD` = Triggered A
The character A/B/C/D in the parameter value indicates the baseband the signal is related to.

*RST: depends on TM suffix

Example:
`:SOURce2:INPut:TM2:DIRection OUTPut`
`:OUTPut2:TM2:DIRection?`
Response: `OUTPut`
`:OUTPut2:TM2:SIGNal MARKA2`

Manual operation: See "Signal" on page 675

[:SOURce]:INPut:USER<ch>:SIGNal <Signal>

Determines the control signal that is input at the selected connector.

To define the connector direction, use the command `[:SOURce]:INPut:USER<ch>:DIRection`.

Suffix:
USER<ch> 1 to 6

Parameters:
<Signal> TRIG1 | TRIG2 | NSEGM1 | NSEGM2 | IPULSA | IPULSB | NONE | FEEDback | SYNCIN

`TRIG1|TRIG2` = Global Trigger 1/2
`NSEGM1|NSEGM2` = Global Next Segment 1/2
`IPULSA|IPULSB` = Pulse In A/B, available for `USER4|5|6`
`FEEDback` = Baseband Feedback, available for `USER6`
`SYNCIN` = Baseband Sync In

Example: See Example "Global connectors settings configuration" on page 857

Options: NSEGM1|NSEGM2 require R&S SMW-B10
SYNCIN requires R&S SMW-B9

Manual operation: See "Signal" on page 671

:OUTPut:USER<ch>:SIGNal <Signal>

Sets the control signal that is output at the selected connector.

To define the connector direction, use the command `:OUTPut:USER<ch>:DIRection`.

Suffix:

USER<ch> 1 to 6

Parameters:

<Signal> MARKA1 | MARKA2 | MARKA3 | MARKB1 | MARKB2 | MARKB3 | MARKC1 | MARKC2 | MARKC3 | MARKD1 | MARKD2 | MARKD3 | SVALA | SVALB | OPULSA | OPULSB | SYNCA | VIDEOA | VIDEOB | SYNCB | NONE | MTRigger | RTRIGA | RTRIGB | SVALANegated | SVALBNegated | SYNCOUT | BERRESTOUT | BERDATENOUT | BERCLKOUT | BERDATOUT | SCLock | LATTenuation | BGATe | HOP | CWMODulation | TRIGgered

`MARK<A|B|C|D><1|2|3>` = Baseband <BB> Marker 1/2/3

`SVALA|SVALB` = Signal Valid A/B, available for `USER4|5|6`

`SVALAN|SVALBN` = Signal Valid A/B (negative), available for `USER4|5|6`

`OPULSA|OPULSB` = Pulse Out A/B, available for `USER4|5|6`

`SYNCA|SYNCB` = Sync A/B, available for `USER4|5|6`

`VIDEOA|VIDEOB` = Video A/B, available for `USER4|5|6`

`MTRigger` = Manual Trigger, available for `USER6`

`RTRIGA|RTRIGB` = REG trigger A/B, available for `USER4|5`

`SYNCOU` = Baseband Sync Out

*RST: MARKA1

Example: See Example "Global connectors settings configuration" on page 857

Options: SYNCOUT requires R&S SMW-B9

Manual operation: See "Signal" on page 671

:OUTPut:USER<ch>:TRIGger[:IMMediate]

Generates a short pulse signal and outputs it at the USER 6 connector.

This signal can serve as a common external trigger signal for triggering of several R&S SMW, see Example "Triggering several R&S SMW instruments simultaneously" on page 614.

Suffix:

USER<ch> 6

Example: See Example "Global connectors settings configuration" on page 857

Usage: Event

Manual operation: See "Execute Trigger" on page 673

[:SOURce]:INPut:USER:CLOCk:LEVel <Level>
[:SOURce]:INPut:USER:TRIGger:LEVel <Level>

Sets the threshold for any input signal at the USER1-3 connectors.

Parameters:

<Level> float

Range: 0.1 to 2
Increment: 0.1
*RST: 1

Example: See Example "Global connectors settings configuration" on page 857

Manual operation: See "Threshold USER1-3 Input" on page 673

[:SOURce]:INPut:USER:PULM:LEVel <Level>

Sets the threshold for any input signal at the USER4-6 connectors.

Parameters:

<Level> float

Range: 0.1 to 2
Increment: 0.1
*RST: 1
Default unit: V

Example: See Example "Global connectors settings configuration" on page 857

Manual operation: See "Threshold USER4-6 Input/Threshold Pulse Input" on page 514

[:SOURce]:INPut:USER:CLOCk:IMPedance <Impedance>
[:SOURce]:INPut:USER:TRIGger:IMPedance <Impedance>

Selects the input impedance for the external trigger inputs.

Parameters:

<Impedance> G1K | G50

*RST: G1K

Example: See Example "Global connectors settings configuration" on page 857

Manual operation: See "Impedance Clock/Trigger Input" on page 674

[:SOURce]:INPut:USER:CLOCk:SLOPe <Slope>

Sets the polarity of the active slope of an externally applied clock signal.

Parameters:

<Slope>	NEGative | POSitive *RST: POSitive
Manual operation:	See "Symbol Clock Slope/Clock Slope" on page 271

[:SOURce]:INPut:USER:TRIGger:SLOPe <Slope>

Sets the polarity of the active slope of an applied instrument trigger.

Parameters:

<Slope>	NEGative | POSitive *RST: POSitive
Example:	See Example "Global connectors settings configuration" on page 857
Manual operation:	See "Trigger Input Slope" on page 674

[:SOURce<hw>]:INPut:TM:CLOCk:LEVel <Level>
[:SOURce<hw>]:INPut:TM:TRIGger:LEVel <Level>

Sets the high/low threshold in volts for the trigger and clock signal inputs of the baseband section.

Suffix:

SOURce<hw>	1 | 2 Determines the baseband the connectors belong to, where: `SOURce1` = Baseband A/C and `SOURce2` = Baseband B/D

Parameters:

<Level>	float Range: 0.3 to 2 Increment: 0.1 *RST: 1
Manual operation:	See "Threshold Clock/Trigger Input" on page 677

[:SOURce<hw>]:INPut:TM:CLOCk:IMPedance <Impedance>
[:SOURce<hw>]:INPut:TM:TRIGger:IMPedance <Impedance>

Selects the input impedance for the external trigger/clock inputs.

Suffix:

SOURce<hw>	1 | 2 Determines the baseband the connectors belong to, where: `SOURce1` = Baseband A/C and `SOURce2` = Baseband B/D

Parameters:

<Impedance> G1K | G50 | G10K

G10K
Provided only for backward compatibility with other R&S signal generators.
The R&S SMW accepts this value and maps it automatically to G1K.

*RST: G1K

Manual operation: See "Impedance Clock/Trigger Input" on page 677

[:SOURce<hw>]:INPut:TM:CLOCk:SLOPe <Slope>

Sets the polarity of the active slope of an externally applied clock signal.

Suffix:

SOURce<hw> 1 | 2
Determines the baseband the connectors belong to, where:
`SOURce1` = Baseband A/C and
`SOURce2` = Baseband B/D

Parameters:

<Slope> NEGative | POSitive

*RST: POSitive

Manual operation: See "Clock Input Slope" on page 677

[:SOURce<hw>]:INPut:TM:TRIGger:SLOPe <Slope>

Sets the polarity of the active slope of an externally applied trigger signal.

Suffix:

SOURce<hw> 1 | 2
Determines the baseband the connectors belong to, where:
`SOURce1` = Baseband A/C and
`SOURce2` = Baseband B/D

Parameters:

<Slope> NEGative | POSitive

*RST: POSitive

Manual operation: See "Trigger Input Slope" on page 677

13.18.2 SOURce:BBIN Subsystem

The `SOURce:BBIN` subsystem contains the commands for setting the external digital baseband signal.

In Standard mode, the external baseband signal A can be routed to path A, path B, or both paths. The external baseband signal B can be routed to path B only.

Suffixes in the keywords ENTity<ch> and SOURce<hw>

You can address multiple entities configurations by using the SCPI commands starting with the keyword `SOURce` or the alias commands starting with the keyword `ENTity`.

Table 13-2: Value ranges of the suffixes ENTity<ch> and SOURce<hw> in advanced configuration with multiple entities

SCPI Syntax	`ENTity<ch>`	`SOURce<hw>`
`SOURce<hw>:BBIN:...`	-	1 to 4
`ENTity<ch>:SOURce<hw>:BBIN:...`	1 to 4	1 to 4

See also Chapter 13.3, "SCPI Command Aliases for Advanced Mode with Multiple Entities", on page 767.

Required options

See "Required Options" on page 170.

Example: Programming Example

The following example lists the commands necessary to apply an external digital baseband signal, to enable automatic level settings, and to monitor the signal.

An external digital signal must be applied at the corresponding connectors of the instrument.

```
// uses the external signal supplied at the CODER 1 interface
SOURce1:BBIN:DIGital:SOURce CODER1
// SOURce1:BBIN:DIGital:IQSWap:STATe 1
// set the sample rate source and query the value
SOURce1:BBIN:SRATe:SOURce DIN
SOURce1:BBIN:SRATe[:ACTual]?
SOURce1:BBIN:STATe ON
// enable automatic adjustment of the baseband input signal
SOURce1:BBIN:DIGital:ASETting:STATe ON
SOURce1:BBIN:MPERiod 10s
SOURce1:BBIN:ALEVel:EXECute
SOURce1:BBIN:CFACtor?
SOURce1:BBIN:POWer:PEAK?
SOURce1:BBIN:POWer:RMS?
// monitor the signal
SOURce1:BBIN:OLOad:STATe?
// 0, i.e. no overflow detected
SOURce1:BBIN:OLOad:HOLD:RESet
SOURce1:BBIN:OLOad:HOLD:STATe?
```

The following commands are available:

[:SOURce<hw>]:BBIN:STATe <State>

Switches the feeding of an external analog signal into the signal path on/off.

Parameters:

<State> 0 | 1 | OFF | ON

*RST: 0

Example: See Example "Programming Example" on page 865

Manual operation: See "State" on page 173

[:SOURce<hw>]:BBIN:DIGital:SOURce <DigInpSource>

Defines the connector used as an external signal source.

Parameters:

<DigInpSource> CODER1 | FADER1 | FADER2 | CODER2

*RST: CODER1

Example: see Example "Programming Example" on page 865.

Manual operation: See "Source" on page 173

[:SOURce<hw>]:BBIN:IQSWap[:STATe] <State>

If activated, swaps the I and Q channel.

Parameters:

<State> 0 | 1 | OFF | ON

*RST: 0

Example: see Example "Programming Example" on page 865

Manual operation: See "I/Q Swap" on page 173

[:SOURce<hw>]:BBIN:MODE <Mode>

Defines that a digital external signal is applied.

Parameters:

<Mode> DIGital

*RST: DIGital

Example: see Example "Programming Example" on page 865

[:SOURce<hw>]:BBIN:SRATe:SOURce <Source>

Selects whether the sample rate is estimated based on the digital input signal or is a user-defined value.

Parameters:

<Source> USER | DIN

DIN

Enabled for `[:SOURce<hw>]:BBIN:DIGital:SOURce` `CODER1|CODER2`

*RST: USER

Example: See Example "Programming Example" on page 865

Manual operation: See "Sample Rate Source" on page 174

[:SOURce<hw>]:BBIN:SRATe[:ACTual] <Actual>

Sets the sample rate of the external digital baseband signal.

Parameters:

<Actual> float

Range for `[:SOURce<hw>]:BBIN:DIGital:SOURce` `FADER1|FADER2: 100E6|200E6`

Range: 400 to 200E6
Increment: 0.001
*RST: 100E6

Example: see Example "Programming Example" on page 865

Manual operation: See "Sample Rate Value" on page 174

[:SOURce<hw>]:BBIN:DIGital:ASETting:STATe <State>

Activates automatic adjustment of the baseband input signal.

Parameters:

<State> 0 | 1 | OFF | ON

*RST: 0

Example: See Example "Programming Example" on page 865

Manual operation: See "DIG IQ Auto Setting" on page 175

[:SOURce<hw>]:BBIN:MPERiod <MPeriod>

For [:SOURce<hw>]:BBIN:DIGital:SOURceCODER1|CODER2, sets the recording duration for measuring the baseband input signal by executed [:SOURce<hw>]:BBIN:ALEVel:EXECute.

Parameters:

<MPeriod> integer

Range: 1 to 32
*RST: 2
Default unit: s

Example: see Example "Programming Example" on page 865

Manual operation: See "Measurement Period" on page 176

[:SOURce<hw>]:BBIN:ALEVel:EXECute

For [:SOURce<hw>]:BBIN:DIGital:SOURceCODER1|CODER2, starts measuring the input signal. The measurement estimates the crest factor, peak and RMS level.

Example: See Example "Programming Example" on page 865

Usage: Event

Manual operation: See "Auto Level Set" on page 176

[:SOURce<hw>]:BBIN:CFACtor <CFactor>

Enters the crest factor of the external baseband signal.

Parameters:

<CFactor> float

Range: 0 to 30
Increment: 0.01
*RST: 0
Default unit: dB

Example: See Example "Programming Example" on page 865

Manual operation: See "Crest Factor" on page 176

[:SOURce<hw>]:BBIN:POWer:PEAK <Peak>

Peak level of the external baseband signal relative to full scale of 0.5 V (in terms of dB full scale).

Parameters:

<Peak> float

Range: -60 to 3.02
Increment: 0.01
*RST: 0
Default unit: dBfs

Example: See Example "Programming Example" on page 865

Manual operation: See "Peak Level" on page 176

[:SOURce<hw>]:BBIN:POWer:RMS?

Queries the RMS level of the external digital baseband signal.

Return values:

<Rms> float

Range: -100 to 10
Increment: 0.01
*RST: 0

Example: see Example "Programming Example" on page 865

Usage: Query only

Manual operation: See "Level" on page 176

[:SOURce<hw>]:BBIN:OLOad:STATe?

Queries the current overflow state.

Return values:

<State> 0 | 1 | OFF | ON

*RST: 0

Example: see Example "Programming Example" on page 865

Usage: Query only

Manual operation: See "Signal Monitoring" on page 177

[:SOURce<hw>]:BBIN:OLOad:HOLD:STATe?

Queries an overload since the last reset for evaluating the measurement.

Return values:

<State> 0 | 1 | OFF | ON

*RST: 0

Example: See `[:SOURce<hw>]:BBIN:OLOad:STATe?` on page 869

Usage: Query only

Manual operation: See "Signal Monitoring" on page 177

[:SOURce<hw>]:BBIN:OLOad:HOLD:RESet

Reset of the Overload Hold indication.

Example: see `[:SOURce<hw>]:BBIN:OLOad:STATe?` on page 869

Usage: Event

[:SOURce<hw>]:BBIN:CDEVice?

Indicates the ID of an externally connected R&S Instrument or R&S Device.

Return values:
<CDevice> string

Example: `SOURce:BBIN:CDEVice?`
queries the connected device ID.

Usage: Query only

Manual operation: See "Connected Device" on page 173

13.18.3 Analog Modulation Subsystems

The subsystems in this section describe all commands for analog modulation of the RF signal. Divided in separate sections, you can configure amplitude modulation (AM), frequency modulation (FM) , phase modulation (PhiM) and pulse modulation (PULM).

You can perform each of the modulations either with an internally generated modulation signal or with an externally applied signal.

To configure the internal signal, use the commands listed in Chapter 13.18.10, "SOURce:LFOutput Subsystem", on page 1061.

For more information:

See Chapter 7.7, "Analog Modulations", on page 493.

13.18.3.1 SOURce:AM Subsystem

The AM subsystem contains the commands for setting the amplitude modulation and also the broadband amplitude modulation.

Activation of amplitude modulation deactivates ARB, I/Q modulation, digital modulation and all digital standards.

The following examples show some variants for generating AM signals.

Example: Creating an amplitude modulated RF signal

Using the internal LF generator, the following command sequence configures an amplitude modulated signal.

```
// Reset the instrument to start from a defined state
*RST

// Set RF frequency and amplitude
SOURce1:FREQuency:CW 6000000000
SOURce1:POWer:LEVel:IMMediate:AMPLitude -25

// Configure the modulation signal
SOURce1:LFOutput1:SHAPe SINE
SOURce1:LFOutput1:FREQuency 20000

// Configure the amplitude modulation settings and switch AM on
SOURce1:AM1:SOURce LF1
SOURce1:AM1:DEPTh 30
SOURce1:AM:RATio 40
SOURce1:AM1:STATe 1

// Switch on LF and RF signal output
SOURce1:LFOutput1:STATe 1
OUTPut1:STATe 1
```

Example: Using an external signal source

Using an external signal source, you can additionally determine whether you want to use only the AC component of the external modulation signal.

```
// Reset the instrument to start from a defined state
*RST

// Set frequency and amplitude
SOURce1:FREQuency:CW 6000000000
SOURce1:POWer:LEVel:IMMediate:AMPLitude -25

// Configure the amplitude modulation settings and switch AM on
SOURce1:AM1:SOURce EXT1
SOURce1:AM1:DEPTh 40

// Query the input sensitivity at the external modulation input
SOURce1:AM1:SENSitivity?
// Response:  40
// Since the voltage value for full modulation is 1V,
// the resulting sensitivity is precisely 50%/V.
// This value is assigned to the voltage value for full
// modulation of the input.

// select the coupling mode AC for external amplitude modulation
SOURce1:INPut:MODext:COUPling1 AC

// Switch on AM and RF signal output
SOURce1:AM1:STATe 1
OUTPut1:STATe 1
```

The following commands are available:

[:SOURce<hw>]:AM:RATio <Ratio>

Sets the deviation ratio (path#2 to path#1) in percent.

Parameters:

<Ratio> float

Range: 0 to 100
Increment: 0.01
*RST: 100

Example: See Example "Creating an amplitude modulated RF signal" on page 870.

Manual operation: See "Ratio Path2/Path1" on page 507

[:SOURce<hw>]:AM:SENSitivity?

Queries the sensitivity of the externally applied signal for amplitude modulation. The sensitivity depends on the set modulation depth.

The returned value reports the sensitivity in %/V. It is assigned to the voltage value for full modulation of the input.

Return values:

<Sensitivity> float

Range: 0 to 100

Example: See Example "Using an external signal source" on page 871.

Usage: Query only

[:SOURce<hw>]:AM<ch>:SOURce <Source>

Selects the modulation source for amplitude modulation.

Suffix:

<ch> 1 to 2
Modulation signal channel

Parameters:

<Source> LF1 | LF2 | NOISe | EXT1 | EXT2 | EXTernal | INTernal

LF1|LF2
Uses an internally generated LF signal.

EXT1|EXT2
Uses an externally supplied LF signal.

NOISe
Uses the internally generated noise signal.

INTernal
Uses the internally generated signal of LF1.

EXTernal
Uses an external LF signal (EXT1).

*RST: LF1 <AM1>; LF2 <AM2>

Example: See Example "Creating an amplitude modulated RF signal" on page 870.

Manual operation: See "Source" on page 507

[:SOURce<hw>]:AM<ch>:STATe <State>

Activates amplitude modulation.

Suffix:

<ch> 1 to 2
Modulation signal channel

Parameters:

<State> 0 | 1 | OFF | ON

*RST: 0

Example: See Example "Creating an amplitude modulated RF signal" on page 870.

Manual operation: See "State" on page 506

[:SOURce<hw>]:AM<ch>[:DEPTh] <Depth>

Sets the depth of the amplitude modulation in percent.

Suffix:

<ch> 1 to 2
Modulation signal channel

Parameters:

<Depth> float

Range: 0 to 100
Increment: 0.1

Example: See Example "Creating an amplitude modulated RF signal" on page 870.

Manual operation: See "AM Depth" on page 507

13.18.3.2 SOURce:FM Subsystem

The FM subsystem contains the commands for setting the frequency modulation.

Example: Creating a frequency modulated RF signal

Using the internal LF generator, the following command sequence configures a frequency modulated signal.

```
// Reset the instrument to start from a defined state
*RST

// Set RF frequency and amplitude
SOURce1:FREQuency:CW 6000000000
SOURce1:POWer:LEVel:IMMediate:AMPLitude -25

// Configure the modulation signal
SOURce1:LFOutput1:SHAPe SINE
SOURce1:LFOutput1:FREQuency 20000

// Configure the frequency modulation settings and switch FM on
SOURce1:FM1:SOURce LF1
SOURce1:FM1:DEViation 1000
SOURce1:FM:RATio 40
SOURce1:FM:MODe LNOise
SOURce1:FM1:STATe 1

// Switch on LF and RF signal output
SOURce1:LFOutput1:STATe 1
OUTPut1:STATe 1
```

Example: Using an external signal source

Alternatively configure the frequency modulation settings with an external modulation signal.

```
// Reset the instrument to start from a defined state
*RST

// Set RF frequency and amplitude
SOURce1:FREQuency:CW 6000000000
SOURce1:POWer:LEVel:IMMediate:AMPLitude -25

// Configure the frequency modulation settings and switch FM on
SOURce1:FM1:SOURce EXT1
SOURce1:FM1:DEViation 5000

// Query the input sensitivity at the external modulation input
:SOURCe1:FM1:SENSitivity?
```

```
// Response:  1000
// since the voltage value for full modulation is 1V,
// the resulting sensitivity is precisely 5000 Hz/V.

// Switch on FM and RF signal output
SOURce1:FM1:STATe 1
OUTPut1:STATe 1
```

The following commands are available:

[:SOURce<hw>]:FM<ch>[:DEViation] <Deviation>

Sets the modulation deviation of the frequency modulation in Hz.

Suffix:

FM<ch> — 1|2
Modulation signal channel

Parameters:

<Deviation> — float

The maximum deviation depends on the RF frequency and the selected modulation mode (see data sheet).

Range: 0 to max
Increment: 0.01
*RST: 1E3

Example: See Example "Creating a frequency modulated RF signal" on page 874.

Manual operation: See "Deviation" on page 508

[:SOURce<hw>]:FM:MODE <Mode>

Selects the mode for the frequency modulation.

Parameters:

<Mode> — NORMal | LNOise

NORMal
The maximum range for modulation bandwidth and FM deviation is available.

LNOise
Frequency modulation with phase noise and spurious characteristics close to CW mode. The range for modulation bandwidth and FM deviation is reduced (see data sheet).

*RST: NORMal

Example: See Example "Creating a frequency modulated RF signal" on page 874.

Manual operation: See "Mode" on page 508

[:SOURce<hw>]:FM:RATio <Ratio>

Sets the deviation ratio (path2 to path1) in percent.

Parameters:

<Ratio> float

Range: 0 to 100
Increment: 0.01
*RST: 100

Example: See Example "Creating a frequency modulated RF signal" on page 874.

Manual operation: See "Ratio Path2/Path1" on page 508

[:SOURce<hw>]:FM:SENSitivity?

Queries the sensitivity of the externally supplied signal for frequency modulation. The sensitivity depends on the set modulation deviation.

Return values:

<Sensitivity> float

Sensitivity in Hz/V.
It is assigned to the voltage value for full modulation of the input.

Range: 0 to max
Increment: 0.01

Example: See Example "Using an external signal source" on page 874.

Usage: Query only

[:SOURce<hw>]:FM<ch>:SOURce <Source>

Selects the modulation source for frequency modulation.

Suffix:

FM<ch> 1|2
Modulation signal channel.

Parameters:

<Source> LF1 | LF2 | NOISe | EXT1 | INTernal | EXTernal | EXT2

LF1|LF2
Uses an internally generated LF signal.

INTernal = LF2
Works like `LF1`

EXTernal
Works like `EXT1`

EXT1|EXT1
Uses an externally supplied LF signal.

NOISe
Uses the internally generated noise signal.

*RST: LF1 <FM1>; LF2 <FM2>

Example: See Example "Creating a frequency modulated RF signal" on page 874.

Manual operation: See "Source" on page 507

[:SOURce<hw>]:FM<ch>:STATe <State>

Activates frequency modulation.

Suffix:

FM<ch> 1..2
determines the modulation signal channel.

Parameters:

<State> 0 | 1 | OFF | ON

*RST: 0

Example: See Example "Creating a frequency modulated RF signal" on page 874.

Manual operation: See "State" on page 506

13.18.3.3 SOURce:PM Subsystem

The PM subsystem contains the commands for setting the phase modulation. You can configure the internal modulation source (LF generator) with the commands listed in Chapter 13.18.10, "SOURce:LFOutput Subsystem", on page 1061 .

For information about the required options, see Chapter 7.7.1, "Required Options", on page 495.

Example: Performing phase modulation

The following example shows a command sequence to perform phase modulation.

```
// Reset the instrument to start from an initial state
*RST; *CLS
```

```
// Set the RF signal frequency and level
SOURce:FREQuency:CW 4000000000
SOURce:POWer:LEVel:IMMediate:AMPLitude -25

// Configure the phase modulation settings
SOURce1:LFOutput1:SHAPe SINE
SOURce1:LFOutput1:FREQuency 1000

// Select the LF signal generated by the internal modulation generator
// or the internally generated noise signal
SOURce1:PM1:DEViation 1
SOURce1:PM1:SOURce LF1
// SOURce1:PM1:SOURce INTernal
// SOURce1:PM1:SOURce NOISe
SOURce1:PM1:RATio 40
SOURce1:PM1:MODE HBAN

// Alternatively configure the phase modulation settings for an
// external modulation source and query the input sensitivity.
SOURce1:PM1:SOURce EXT1
// SOURce1:PM1:SOURce EXTernal
SOURce1:PM1:DEViation 1
SOURce1:PM1:SENSitivity?
// Response: 1
// since the voltage value for full modulation is 1V,
// the resulting sensitivity is precisely 1RAD/V.

// Activate the signal output
SOURce1:PM1:STATe 1
OUTPut1:STATe 1
```

The following commands are available:

[:SOURce<hw>]:PM:MODE <Mode>

Selects the mode for the phase modulation.

Parameters:

<Mode> HBANdwidth | HDEViation | LNOise

HBANdwidth
Sets the maximum available bandwidth.

HDEViation
Sets the maximum range for ΦM deviation.

LNOise
Selects a phase modulation mode with phase noise and spurious characteristics close to CW mode.

*RST: HBANdwidth

Example: See Example "Performing phase modulation" on page 877.

Manual operation: See "Mode" on page 510

[:SOURce<hw>]:PM:RATio <Ratio>

Sets the deviation ratio (path2 to path1) in percent.

Parameters:

<Ratio> float

Range: 0 to 100
Increment: 0.01
*RST: 100

Example: See Example "Performing phase modulation" on page 877.

Manual operation: See "Ratio Path2/Path1" on page 509

[:SOURce<hw>]:PM:SENSitivity?

Queries the sensitivity of the externally applied signal for phase modulation.

The returned value reports the sensitivity in RAD/V. It is assigned to the voltage value for full modulation of the input.

Return values:

<Sensitivity> float

Example: See Example "Performing phase modulation" on page 877.

Usage: Query only

[:SOURce<hw>]:PM<ch>:SOURce <Source>

Selects the modulation source for phase modulation signal.

Suffix:

PM<ch> 1|2
Sets the modulation signal channel.

Parameters:

<Source> LF1 | LF2 | NOISe | EXT1 | EXT2 | INTernal | EXTernal

LF1|LF2
Uses an internally generated LF signal.

EXT1|EXT2
Uses an externally supplied LF signal.

NOISe
Uses the internally generated noise signal.

INTernal
Uses the internally generated signal of LF1.

EXTernal
Uses an external LF signal (EXT1).

*RST: LF1 <PM1>; LF2 <PM2>

Example: See Example "Performing phase modulation" on page 877.

Manual operation: See "Source" on page 507

[:SOURce<hw>]:PM<ch>:STATe <State>

Activates phase modulation.

Activation of phase modulation deactivates frequency modulation.

Suffix:

PM<ch> 1|2
Sets the modulation signal channel.

Parameters:

<State> 0 | 1 | OFF | ON

*RST: 0

Example: See Example "Performing phase modulation" on page 877.

Manual operation: See "State" on page 506

[:SOURce]:PM<ch>[:DEViation] <Deviation>

Sets the modulation deviation of the phase modulation in RAD.

Parameters:

<Deviation> float

The maximal deviation depends on the RF frequency and the selected modulation mode (see data sheet).

Range: 0 to max
Increment: 1
*RST: 1
Default unit: RAD

Example: See Example "Performing phase modulation" on page 877.

Manual operation: See "Deviation" on page 509

13.18.3.4 SOURce:PULM Subsystem

The `PULM` subsystem contains the commands for setting the pulse modulation.

Pulse Modulation Settings

With the commands described in this section, you can configure the settings for pulse modulation, select the trigger mode and determine delay times for the pulse modulation signal.

Example: Perform pulse modulation

The example shows a command sequence to perform pulse modulation.

```
// Reset the instrument to start from an initial state
*RST; *CLS

// Set the RF signal frequency and level
SOURce:FREQuency:CW 4000000000
SOURce:POWer:LEVel:IMMediate:AMPLitude -25

// Configure the pulse modulation settings
// Select the internal modulation generator,
// set trigger mode, select pulse mode, transition type
// and select the polarity of the internally generated pulse video output
SOURce:PULM:SOURce INT
SOURce:PULM:TRIGger:MODE AUTO
SOURce:PULM:MODE DOUB
SOURce:PULM:TTYPe SMO
SOURce:PULM:OUTPut:VIDeo:POLarity INVerted

// Alternatively configure the pulse modulation settings for
// external modulation source
// Select the source, set the polarity of the external signal,
// select the impedance for the external pulse modulation input/
// for the external pulse modulation trigger input
SOURce:PULM:SOURce EXT
SOURce:PULM:POLarity NORMal
SOURce:PULM:IMPedance G1K

SOURce:PULM:TRIGger:EXTernal:IMPedance G10K

// Configure the pulse generator settings
// Set pulse period, width, and delay
SOURce:PULM:PERiod 10 us
SOURce:PULM:WIDth 8 us
SOURce:PULM:DOUBle:WIDTh 0.0000012
SOURce:PULM:DOUBle:DELay 0.0000045

// Activate the signal output
```

```
SOURce:PGENerator:OUTPut:STATe 1
SOURce:PULM:STATe 1
OUTPut1:STATe 1
```

The following commands are available:

[:SOURce<hw>]:PULM:SOURce <Source>

Selects between the internal (pulse generator) or an external pulse signal for the modulation.

Parameters:

<Source> INTernal | EXTernal

*RST: INTernal

Example: See Example "Perform pulse modulation" on page 881.

Manual operation: See "Source" on page 513

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

Sets the period of the generated pulse, that means the repetition frequency of the internally generated modulation signal.

Parameters:

<Period> float

The minimum value depends on the installed options R&S SMW-K22 or R&S SMW-K23

Range: 20E-9 to 100
Increment: 5E-9
*RST: 10E-6

Example: See Example "Perform pulse modulation" on page 881.

Manual operation: See "Pulse Period" on page 515

[:SOURce<hw>]:PULM:DELay <Delay>

Sets the pulse delay.

Parameters:

<Delay>	float *RST: 1ms
Example:	See Example "Perform pulse modulation" on page 881.
Manual operation:	See "Pulse Delay" on page 516

[:SOURce<hw>]:PULM:DOUBle:DELay <Delay>

Sets the delay from the start of the first pulse to the start of the second pulse.

Parameters:

<Delay>	float *RST: 1E-6
Example:	See Example "Perform pulse modulation" on page 881.
Manual operation:	See "Double Pulse Delay" on page 516

[:SOURce<hw>]:PULM:DOUBle:WIDTh <Width>

Sets the width of the second pulse.

Parameters:

<Width>	float Increment: 5E-9
Example:	See Example "Perform pulse modulation" on page 881.
Manual operation:	See "Double Pulse Width" on page 516

[:SOURce<hw>]:PULM:POLarity <Polarity>

Sets the polarity of the externally applied modulation signal.

Parameters:

<Polarity>	NORMal | INVerted **NORMal** Suppresses the RF signal during the pulse pause. **INVerted** Suppresses the RF signal during the pulse. *RST: NORMal
Example:	See Example "Perform pulse modulation" on page 881.
Manual operation:	See "Polarity Pulse Input" on page 514

[:SOURce<hw>]:PULM:IMPedance <Impedance>

Sets the impedance for the external pulse modulation input.

Parameters:

<Impedance> G50 | G1K

*RST: G1K

Example: See Example "Perform pulse modulation" on page 881.

Manual operation: See "Impedance Pulse Input" on page 514

[:SOURce<hw>]:PULM:MODE <Mode>

Selects the mode for the pulse modulation.

Parameters:

<Mode> SINGle | DOUBle

SINGle
Generates a single pulse.

DOUBle
Generates two pulses within one pulse period.

*RST: SINGle

Example: See Example "Perform pulse modulation" on page 881.

Manual operation: See "Pulse Mode" on page 515

[:SOURce<hw>]:PULM:OUTPut:VIDeo:POLarity <Polarity>

Sets the polarity of the pulse video (modulating) signal, related to the RF (modulated) signal.

Parameters:

<Polarity> NORMal | INVerted

NORMal
the video signal follows the RF signal, that means it is high wihen RF signal is high and vice versa.

INVerted
the video signal follows in inverted mode.

*RST: NORMal

Example: See Example "Perform pulse modulation" on page 881.

Manual operation: See "Video Polarity" on page 514

[:SOURce]:PULM[:INTernal][:TRAin]:TRIGger:IMMediate

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

Activates pulse modulation.

Parameters:

<State> 0 | 1 | OFF | ON

*RST: 0

Example: See Example "Perform pulse modulation" on page 881.

Manual operation: See "State" on page 513

[:SOURce<hw>]:PULM:TRIGger:MODE <Mode>

Selects a trigger mode - auto, external, external single or external gated - for generating the modulation signal.

Parameters:

<Mode> AUTO | EXTernal | EGATe | ESINgle

*RST: AUTO

Example: See Example "Perform pulse modulation" on page 881.

Manual operation: See "Trigger Mode" on page 516

[:SOURce<hw>]:PULM:TRIGger:EXTernal:IMPedance <Impedance>

Sets the impedance for the external pulse trigger.

Parameters:

<Impedance> G50 | G10K

*RST: G50

Example: See Example "Perform pulse modulation" on page 881.

[:SOURce<hw>]:PULM:TTYPe <Source>

Sets the transition mode for the pulse signal.

Parameters:

<Source> SMOothed | FAST

SMOothed
flattens the slew rate, resulting in longer rise/fall times.

FAST
enables fast transitions with shortest rise and fall times.

*RST: FAST

Example: See Example "Perform pulse modulation" on page 881.

Manual operation: See "Transition Type" on page 514

[:SOURce<hw>]:PULM:WIDTh <Width>

Sets the width of the generated pulse, that means the pulse length. It must be at least 20ns less than the set pulse period.

Parameters:

<Width> float

Range: 20E-9 to 100
Increment: 10E-9
*RST: 2E-6

Example: See Example "Perform pulse modulation" on page 881.

Manual operation: See "Pulse Width" on page 516

[:SOURce<hw>]:PULM:DOUBle:STATe <State>

Provided for backward compatibility with former Rohde & Schwarz signal generators.

Works like the command `[:SOURce<hw>]:PULM:MODE DOUBle`.

Parameters:

<State> 0 | 1 | OFF | ON

*RST: 0

13.18.4 SOURce:BB Subsystem

This subsystem contains all commands for digital signal generation. It is divided into several subsystems which are described separately.

13.18.4.1 SOURce:BB Subsystem General Commands

The following section describes the commands for setting the frequency shift and the phase offset for the signal at the output of the "Baseband" and "BB Input" blocks.

Suffixes in the keywords ENTity<ch> and SOURce<hw>

You can address multiple entities configurations by using the SCPI commands starting with the keyword `SOURce` or the alias commands starting with the keyword `ENTity`.

Table 13-3: Value ranges of the suffixes ENTity<ch> and SOURce<hw> in advanced configuration with multiple entities

SCPI Syntax	ENTity<ch>	SOURce<hw>
SOURce<hw>:BB:...	-	1 .. 8
ENTity<ch>:SOURce<hw>:BB:...	1 .. 8	1 .. 4

See also Chapter 13.3, "SCPI Command Aliases for Advanced Mode with Multiple Entities", on page 767.

The following commands are available:

[:SOURce<hw>]:BBIN:FOFFset <FOffset>
[:SOURce<hw>]:BB:FOFFset <FOffset>

Sets a frequency offset for the internal/external baseband signal. The offset affects the generated baseband signal.

Parameters:

<FOffset> float

Range: depends on the installed options, e.g. -60 MHz to +60 MHz (R&S SMW-B10)
Increment: 0.01
*RST: 0
Default unit: Hz

Example: `BB:FOFF 2MHZ`
Sets a frequency offset of 2 MHz.

Options: BBIN:FOFFset requires R&S SMW-B10

Manual operation: See "Frequency Offset" on page 386

[:SOURce<hw>]:BB:POFFset <POffset>

Sets the relative phase offset for the selected baseband signal compared to the baseband signals of the other baseband sources (second path or external baseband).

Parameters:

<POffset>	float Range: 0 to 359.9 Increment: 0.01 *RST: 0 Default unit: DEG

Example: `BB:POFF 0.5DEG`
Sets a relative phase offset of 0.5 DEG for the baseband signal

Manual operation: See "Phase Offset" on page 386

[:SOURce<hw>]:BB:PGAin <PGain>
[:SOURce<hw>]:BBIN:PGAin <PGain>

Sets the relative gain for the internal or external baseband signal compared with the signals of the other baseband sources.

Parameters:

<PGain>	float Range: -50 to 50 Increment: 1E-3 *RST: 0 Default unit: dB

Example: `BBIN:PGA 3dB`
Sets the relative gain of 3 dB for the external baseband signal.

Manual operation: See "Gain" on page 387

[:SOURce<hw>]:BB:ROUTe <Route>
[:SOURce<hw>]:BBIN:ROUTe <Route>

Selects the signal route for the internal/external baseband signal. The internal and external signals are summed if necessary.

Parameters:

<Route>	A | B | AB

Example: `BBIN:ROUT A`
The external baseband signal is introduced into path A.

Options: R&S SMW-B10

Manual operation: See "Signal Routing" on page 170

[:SOURce<hw>]:BB:CODer:MODE <Mode>

Defines the source of the baseband generator.

Parameters:

<Mode> CODer | BBIN

CODer
Internal baseband generator

BBIN
An external digital baseband signal provided at the digital interface

*RST: CODer

Example: `SOURce1:BB:CODer:MODE BBIN`

[:SOURce<hw>]:BB:POWer:PEAK?

Queries the peak level of the baseband signal relative to full scale of 0.5 V (in terms of dB full scale).

Return values:

<Peak> float

Range: -145 to 30
Increment: 0.01
*RST: 0
Default unit: dBfs

Example: `BB:POW:PEAK`
Queries the peak level of the baseband signal.

Usage: Query only

[:SOURce<hw>]:BB:CFACtor?

Queries the crest factor of the baseband signal.

Return values:

<CFactor> float

Range: 0 to 100
Increment: 0.01
*RST: 0
Default unit: dB

Example: `SOURce1:BB:CFACtor?`
queries the crest factor of the baseband signal.

Usage: Query only

[:SOURce<hw>]:BB:POWer:RMS?

Queries the RMS level of the baseband signal relative to full scale of 0.5V (in terms of dB full scale).

Return values:

<Rms> float

Range: -145 to 30
Increment: 0.01
*RST: 0
Default unit: dBfs

Example: `BB:POW:RMS`
queries the rms level of the baseband signal.

Usage: Query only

13.18.4.2 SOURce:BB:DM Subsystem

This section list the commands of the `SOURce:BB:DM` subsystem. The commands are divided into sections, where the last one describes how to use lists for digital modulation in remote control, and all other sections describe the configuration of the digital modulation.

Suffixes in the keywords ENTity<ch> and SOURce<hw>

You can address multiple entities configurations by using the SCPI commands starting with the keyword `SOURce` or the alias commands starting with the keyword `ENTity`.

Table 13-4: Value ranges of the suffixes ENTity<ch> and SOURce<hw> in advanced configuration with multiple entities

SCPI Syntax	ENTity<ch>	SOURce<hw>
SOURce<hw>:BB:...	-	1 to 4
ENTity<ch>:SOURce<hw>:BB:...	1 to 4	1 to 4

See also Chapter 13.3, "SCPI Command Aliases for Advanced Mode with Multiple Entities", on page 767.

Required options

See Chapter 4.6.1, "Required Options", on page 264

The commands are grouped in the following sections:

Programming Examples

Example: Performing general tasks

This example shows how to enable Custom Digital Modulation with predefined settings as basis for further customization (e.g. adjusting the data source); intermediate results and configuration are stored with the save/recall function.

```
// *******************************************************************
// Reset instrument first
// *******************************************************************
*RST; *CLS

SOURce:BB:DM:PRESet
SOURce:BB:DM:STANDard W3GPp
SOURce:BB:DM:SRATe?
// 3840000
SOURce:BB:DM:CODing?
// WCDMA
SOURce:BB:DM:FORMat?
// QPSK45
SOURce:BB:DM:STATe ON
SOURce:BB:DM:SETTing:STORe "/var/user/digMod/CustDM3GPP"

// *******************************************************************
// Recall settings
// *******************************************************************
MMEM:CDIR "/var/user/digMod"
SOURce:BB:DM:SETTing:CATalog?
// CusDigMod,cdm3gpp, CustDM3GPP
SOURce:BB:DM:SETTing:DELete "cdm3gpp"
SOURce:BB:DM:SETTing:LOAD "CusDigMod"

// *******************************************************************
// Change the data source
// *******************************************************************
SOURce:BB:DM:SOURce?
// PRBS
SOURce:BB:DM:PRBS:LENGth?
// 9
SOURce:BB:DM:SOURce DLISt
// Set the default directory and query the existing data lists
MMEM:CDIR "/var/user/DLists"
SOURce:BB:DM:DLISt:CATalog?
// "DList1","DList2"
// delete a list and create a new data list
SOURce:BB:DM:DLISt:DELete "DList1"
SOURce:BB:DM:DLISt:SELect "DList2"
// copy the content of an existing data list to the new data list
SOURce:BB:DM:DLISt:COPY "DList3"
```

```
// query the content of the new data list and modify it (append data to it)
FORM ASCI
SOURce:BB:DM:DLISt:DATA? 2048,1024
// 1,1,0,0,0, ...
SOURce:BB:DM:DLISt:DATA:APPend 1,1,1,0,0,0,1,1,0,1...
SOURce:BB:DM:DLISt:SELect "DList3"
// query the free memory and nuber of bit to be utilized
SOURce:BB:DM:DLISt:FREE?
SOURce:BB:DM:DLISt:POINts?
```

Example: Adjusting clock, marker and trigger settings

The following example lists the provided commands

```
// ******************************************************************
// Clock settings
// ******************************************************************
SOURce:BB:DM:CLOCk:SOURce INTernal

// ******************************************************************
// Configure and enable standard marker signals
// ******************************************************************
SOURce:BB:DM:TRIGger:OUTPut2:MODE PULSe
SOURce:BB:DM:TRIGger:OUTPut2:PULSe:DIVider 5
SOURce:BB:DM:TRIGger:OUTPut2:PULSe:FREQuency?
SOURce:BB:DM:TRIGger:OUTPut3:MODE PATTern
SOURce:BB:DM:TRIGger:OUTPut3:PATTern #HE0F52,20
SOURce:BB:DM:TRIGger:OUTPut1:MODE RATio
SOURce:BB:DM:TRIGger:OUTPut1:ONTime 40
SOURce:BB:DM:TRIGger:OUTPut1:OFFTime 20

SOURce:BB:DM:TRIGger:OUTPut2:DELay 16

// ******************************************************************
// Configure and enable signal generation
// ******************************************************************
SOURce:BB:DM:TRIGger:SEQuence SINGle
SOURce:BB:DM:TRIGger:SLENgth 200
// the first 200 samples will be output after the next trigger event
// SOURce:BB:DM:TRIGger:SEQuence ARETrigger
// SOURce:BB:DM:TRIGger:SOURce EGT1
// external trigger signal must be provided at the connector
// configured for the External Global Trigger 1 signal
// SOURce:BB:DM:TRIGger:EXTernal:SYNChronize:OUTPut ON
// SOURce:BB:DM:TRIGger:EXTernal:DELay 200
// SOURce:BB:DM:TRIGger:EXTernal:INHibit 100

// SOURce:BB:DM:TRIGger:SOURce INTB
// the internal trigger signal from the other path must be used
// SOURce:BB:DM:TRIGger:OBASeband:DELay 25
// SOURce:BB:DM:TRIGger:OBASeband:INHibit 10
```

```
SOURce:BB:DM:TRIGger:SEQuence AAUTo
SOURce:BB:DM:TRIGger:SOURce INTernal
SOURce:BB:DM:STAT ON
SOURce:BB:DM:TRIGger:EXEC
```

Example: Enable power ramping

The following example lists the provided commands

```
SOURce:BB:DM:PRAMp:SOURce INTernal
SOURce:BB:DM:PRAM:SHAP COS
SOURce:BB:DM:PRAMp:TIME 5
SOURce:BB:DM:PRAMp:RDELay 0
SOURce:BB:DM:PRAMp:FDELay -1
SOURce:BB:DM:PRAMp:ATTenuation 10
SOURce:BB:DM:PRAMp:BBONly:STATe ON
SOURce:BB:DM:PRAMp:STATe ON
```

General Commands

[:SOURce<hw>]:BB:DM:STATe <State>

Enables/disables digital modulation. Switching on digital modulation turns off all the other digital standards in the same signal path.

Parameters:

<State> 0 | 1 | OFF | ON
*RST: 0

Example: See Example "Performing general tasks" on page 891

Manual operation: See "State" on page 266

[:SOURce<hw>]:BB:DM:PRESet

Sets the default settings for digital modulation (`*RST` values specified for the commands).

Not affected is the state set with the command `SOURce<hw>:BB:DM:STATe`

Example: See Example "Performing general tasks" on page 891

Usage: Event

Manual operation: See "Set To Default" on page 266

[:SOURce<hw>]:BB:DM:SRATe <SRate>

Sets the symbol rate in Hz/kHz/MHz or Symb/s, kSymb/s and MSymb/s.

Parameters:

<SRate> float

Range: 50 to depends on the installed options, e.g. 100E6 (R&S SMW-B10)

Increment: 1E-3

*RST: 270833.333

Example: See Example "Performing general tasks" on page 891

Manual operation: See "Symbol Rate" on page 267

[:SOURce<hw>]:BB:DM:STANdard <Standard>

Selects predefined set of settings according to the selected standard, see Table 4-11.

Parameters:

<Standard> USER | BLUetooth | DECT | ETC | GSM | GSMEdge | NADC | PDC | PHS | TETRa | W3GPp | TDSCdma | CFORward | CREVerse | WORLdspace | TFTS | APCOPH1C4fm | APCOPH1CQpsk | APCOPH2HCpm | APCOPH2HDQpsk | APCOPH2HD8PSKW | APCOPH2HD8PSKN | APCOPH1Lsm | APCOPH1Wcqpsk | CWBPsk

A query returns the value `USER` if one the following is true:

• A user-defined Custom Dig Mod setting was loaded

• One of the associated settings was changed subsequent to the selection of a standard.

*RST: GSM

Example: See Example "Performing general tasks" on page 891

Options: GSM requires R&S SMW-B10

Manual operation: See "Set according to Standard" on page 267

[:SOURce<hw>]:BB:DM:SOURce <Source>

Selects the data source, e.g. a sequence of 0 or 1, a pseudo-random sequence with different length, a pattern, a data list (`DLISt`), or external serial data (`SERial`).

Parameters:

<Source> ZERO | ONE | PRBS | PATTern | DLISt | SERial

*RST: PRBS

Example: See Example "Performing general tasks" on page 891

Options: SERial requires R&S SMW-B10

Manual operation: See "Data Source" on page 269

[:SOURce<hw>]:BB:DM:PATTern <Pattern>

Selects the data pattern for the internal data when `PATTern` is selected as the data source. The maximum length is 64 bits.

Parameters:

<Pattern> string

Range: #B0,1 to #B111...1,64
*RST: #B0,1

Example:
```
SOURce:BB:DM:SOURce PATT
SOURce:BB:DM:PATTern #B01110111010101010,17
```
Generates the user-defined sequence of 0/1 data.

[:SOURce<hw>]:BB:DM:PRBS[:LENGth] <Length>

Defines the length of the pseudo-random sequence in accordance with the following equation:

Length = (2^`Length`) - 1

Parameters:

<Length> 9 | 11 | 15 | 16 | 20 | 21 | 23 | PN9 | PN11 | PN15 | PN16 | PN20 | PN21 | PN23

*RST: 9

Example: See Example "Performing general tasks" on page 891

Manual operation: See "Data Source" on page 269

[:SOURce<hw>]:BB:DM:SMODulation:BORDer <BitOrder>

Sets the bit order for processing extern serial data.

Parameters:

<BitOrder> LSBit | MSBit

*RST: LSBit

Example:
```
SOURce1:INPut:TM1:DIRection INP
SOURce1:INPut:TM1:SIGNal CLOCk
SOURce1:INPut:TM2:DIRection INP
SOURce1:INPut:TM2:SIGNal DATA

SOURce1:BB:DM:SOURce SERial
SOURce1:BB:DM:SMODulation:BORDer LSBit
SOURce1:BB:DM:SMODulation:CDTDeviation?
SOURce1:BB:DM:SMODulation:CLOCk:SLOPe:BIT POSitive
SOURce1:BB:DM:SMODulation:THRoughput:DELay?
SOURce1:BB:DM:SMODulation:RCVState?
// Response: OPERational
```

Options: R&S SMW-B10

Manual operation: See "Bit Order" on page 272

[:SOURce<hw>]:BB:DM:SMODulation:CDTDeviation <Deviation>

Queries the timing deviations (time offset) between the clock and the data signals.

Parameters:

<Deviation> float

Range: -5E-3 to 5E-3
Increment: 1E-12
*RST: 0

Example: See [:SOURce<hw>]:BB:DM:SMODulation:BORDer on page 895.

Options: R&S SMW-B10

Manual operation: See "Clock to Data Time Deviation" on page 272

[:SOURce<hw>]:BB:DM:SMODulation:CLOCk:SLOPe:BIT <Slope>

Sets the active edge of the bit clock.

Parameters:

<Slope> NEGative | POSitive

*RST: POSitive

Example: See [:SOURce<hw>]:BB:DM:SMODulation:BORDer on page 895

Options: R&S SMW-B10

Manual operation: See "Bit Clock Slope" on page 272

[:SOURce<hw>]:BB:DM:SMODulation:THRoughput:DELay?

Queries the throughput delay from the data input to the RF output in the case of external modulation.

Return values:

<Delay> integer

Range: -100 to 100
Increment: 250E-12
*RST: 0

Example: See [:SOURce<hw>]:BB:DM:SMODulation:BORDer on page 895

Usage: Query only

Options: R&S SMW-B10

Manual operation: See "Throughput Delay" on page 272

[:SOURce<hw>]:BB:DM:SMODulation:RCVState? <RcvState>

Queries the current state of the receiver of the external data.

Parameters:

<RcvState>	OFF | OPERational | UFLow | OFLow
	*RST: OFF
Example:	See `[:SOURce<hw>]:BB:DM:SMODulation:BORDer` on page 895
Usage:	Query only
Manual operation:	See "Receiver State" on page 271

Save/Recall Settings

[:SOURce<hw>]:BB:DM:STANdard:ULISt:CATalog?
[:SOURce<hw>]:BB:DM:SETTing:CATalog?

Queries the files with digital modulation respectively user standard settings in the default directory. Listed are files with the file extension `*.dm` and `*.dm_stu`.

Refer to Chapter 13.7.2, "Accessing Files in the Default or in a Specified Directory", on page 777 for general information on file handling in the default and a specific directory.

Return values:

<Catalog>	"<filename1>,<filename2>,..."
	Returns a string of file names separated by commas.
Example:	See Example "Performing general tasks" on page 891
Usage:	Query only
Manual operation:	See "Save/Recall" on page 266

[:SOURce<hw>]:BB:DM:STANdard:ULISt:STORe <Filename>
[:SOURce<hw>]:BB:DM:SETTing:STORe <Filename>

Stores the current settings into the selected file; the file extension (`*.dm` respectively `*.dm_stu`) is assigned automatically.

Refer to Chapter 13.7.2, "Accessing Files in the Default or in a Specified Directory", on page 777 for general information on file handling in the default and a specific directory.

Setting parameters:

<Filename>	string
Example:	See Example "Performing general tasks" on page 891
Usage:	Setting only
Manual operation:	See "Save/Recall" on page 266

[:SOURce<hw>]:BB:DM:STANdard:ULISt:LOAD <Filename>
[:SOURce<hw>]:BB:DM:SETTing:LOAD <Filename>

Loads the selected file from the default or the specified directory. Loaded are files with extension `*.dm` respectively `*.dm_stu`.

Refer to Chapter 13.7.2, "Accessing Files in the Default or in a Specified Directory", on page 777 for general information on file handling in the default and a specific directory.

Setting parameters:

<Filename>	string
Example:	See Example "Performing general tasks" on page 891
Usage:	Setting only
Manual operation:	See "Save/Recall" on page 266

[:SOURce<hw>]:BB:DM:STANdard:ULISt:DELete <Filename>
[:SOURce<hw>]:BB:DM:SETTing:DELete <Filename>

Deletes the selected file from the default or specified directory. Deleted are files with the file extension `*.dm` respectively `*.dm_stu`.

Refer to Chapter 13.7.2, "Accessing Files in the Default or in a Specified Directory", on page 777 for general information on file handling in the default and a specific directory.

Setting parameters:

<Filename>	string
Example:	See Example "Performing general tasks" on page 891
Usage:	Setting only
Manual operation:	See "Save/Recall" on page 266

Filter Settings

[:SOURce<hw>]:BB:DM:FILTer:TYPE <Type>

The command selects the filter type.

When a standard is selected (`:BB:DM:STAN`), the filter type and filter parameter are set to the default value.

Parameters:

<Type>	RCOSine | COSine | GAUSs | LGAuss | CONE | COF705 | COEQualizer | COFequalizer | C2K3x | APCO25 | SPHase | RECTangle | USER | PGAuss | LPASs | DIRac | ENPShape | EWPShape | LTEFilter | LPASSEVM | APCO25Hcpm | APCO25Lsm *RST: GAUSs

Example: See [:SOURce<hw>]:BB:DM:FILTer:PARameter:SPHase on page 899

Manual operation: See "Filter" on page 276

[:SOURce<hw>]:BB:DM:FILTer:PARameter:APCO25 <Apco25>
[:SOURce<hw>]:BB:DM:FILTer:PARameter:APCO25Lsm:GAUSs <FiltParm>
[:SOURce<hw>]:BB:DM:FILTer:PARameter:APCO25Lsm:LOWPass <FiltParm>
[:SOURce<hw>]:BB:DM:FILTer:PARameter:COSine:BANDwidth <FiltParm>
[:SOURce<hw>]:BB:DM:FILTer:PARameter:COSine[:ROLLoff] <Cosine>
[:SOURce<hw>]:BB:DM:FILTer:PARameter:GAUSs <Gauss>
[:SOURce<hw>]:BB:DM:FILTer:PARameter:LPASs <LPass>
[:SOURce<hw>]:BB:DM:FILTer:PARameter:LPASSEVM <LPassEvm>
[:SOURce<hw>]:BB:DM:FILTer:PARameter:PGAuss <PGauss>
[:SOURce<hw>]:BB:DM:FILTer:PARameter:RCOSine <RCosine>
[:SOURce<hw>]:BB:DM:FILTer:PARameter:SPHase <SPhase>

Sets the filter parameter.

Filter Type	Parameter	Parameter Name	Min	Max	Increment	Default
APCO25	Roll-off factor	<Apco25>	0.05	0.99	0.01	0.2
APCO25Lsm	Cut off frequency for the lowpass/gauss filter (:LOWPass/:GAUSs)	<FiltParm>	400	25E6	1E-3	270833.333
COSinebandwidth	Bandwidth	<FiltParm>	400	25E6	1E-3	270833.333
COSinebandwidth	Roll-off factor	<Cosine>	0.05	1	0.01	0.35
GAUSs	Roll-off factor	<Gauss>	0.15	100000	0.01	0.3
LPASs	Cut-off frequency	<LPass>	0.05	2	0.01	0.5
LPASSEVM	Cut-off frequency	<LPassEvm>	0.05	2	0.01	0.5
PGAuss	Roll-off factor	<PGauss>	0.15	2.5	0.01	0.3
RCOSine	Roll-off factor	<RCosine>	0.05	1	0.01	0.35
SPHase	B x T	<SPhase>	0.15	2.5	0.01	2

Parameters:

<SPhase> float

Range: 0.15 to 2.5
Increment: 0.01
*RST: 2

Example:
```
SOURce:BB:DM:FILTe:TYPE SPHase
SOURce:BB:DM:FILTer:PARameter:SPHase 0.5
```

Manual operation: See "Filter Parameter" on page 277

[:SOURce<hw>]:DM:FILTer:PARameter <Parameter>

Sets the filter parameter of the currently selected filter type.

To set the filter type, use command `[:SOURce<hw>]:BB:DM:FILTer:TYPE` on page 898.

Parameters:

<Parameter> float

Range: 0.05 to 2.5
Increment: 0.01
*RST: 0.35

Example: see `[:SOURce<hw>]:BB:DM:FILTer:PARameter:SPHase` on page 899

Modulation and Coding Settings

[:SOURce<hw>]:BB:DM:CODing <Coding>

Selects the modulation coding.

Parameters:

<Coding> OFF | DIFF | DPHS | DGRay | GRAY | GSM | NADC | PDC | PHS | TETRa | APCO25 | PWT | TFTS | INMarsat | VDL | EDGE | APCO25FSK | ICO | CDMA2000 | WCDMA | APCO258PSK

OFF
The coding is automatically disabled if the selected modulation type is not possible with the coding that has been set

DPHS
Phase Difference

DGRay
Difference + Gray

*RST: GSM

Example: See Example "Performing general tasks" on page 891

Options: DPHS|GSM|INMarsat require R&S SMW-B10

Manual operation: See "Coding" on page 267

[:SOURce<hw>]:BB:DM:FORMat <Format>

Sets the modulation type.

When a standard is selected (`[:SOURce<hw>]:BB:DM:STANdard`), the modulation type is set to the default value.

Parameters:

<Format> ASK | BPSK | P2DBpsk | QPSK | QPSK45 | OQPSk | P4QPsk | P4DQpsk | PSK8 | P8D8psk | P8EDge | QAM16 | QAM32 | QAM64 | QAM256 | QAM1024 | MSK | FSK2 | FSK4 | USER | FSKVar | QAM128 | QEDGe | QAM16EDge | QAM32EDge | AQPSk | QAM4096

*RST: MSK

Example: See Example "Performing general tasks" on page 891

Manual operation: See "Modulation Type" on page 274

[:SOURce<hw>]:BB:DM:AQPSk:ANGLe <Angle>

For AQPSK modulation, sets the angle alpha between the point (0,0) and the I axis.

Parameters:

<Angle> float

Range: 0 to 180
Increment: 0.01
*RST: 0
Default unit: Deg

Example:
```
BB:DM:FORM AQPS
BB:DM:AQPS:ANGL 45
```

Manual operation: See "Angle Alpha" on page 275

[:SOURce<hw>]:BB:DM:ASK:DEPTh <Depth>

Sets the ASK modulation depth for modulation type ASK.

Parameters:

<Depth> float

Range: 0 to 100
Increment: 0.1
*RST: 100
Default unit: PCT

Example:
```
BB:DM:FORM ASK
BB:DM:ASK:DEPT 50 PCT
```

Manual operation: See "ASK Depth" on page 275

[:SOURce<hw>]:BB:DM:FSK:DEViation <Deviation>

Sets the frequency deviation when FSK modulation is selected.

Parameters:

<Deviation> float

The value range depends on the symbol rate.

Range: 1 to 40E6
Increment: 0.5
*RST: 135416.5

Example:

```
SOURce:BB:DM:FORMat FSK4
 SOURce:BB:DM:FSK:DEViation 1MHZ
```

Manual operation: See "FSK Deviation" on page 275

[:SOURce<hw>]:BB:DM:FSK:VARiable:SYMBol<ch0>:DEViation <Deviation>

Sets the deviation of the selected symbol for variable FSK modulation mode.

Parameters:

<Deviation> float

The value range depends on the selected symbol rate (see data sheet).

Range: -40E6 to 40E6
Increment: 0.5
Default unit: Hz

Example:

```
SOURce:BB:DM:FORMat FSKVar
SOURce:BB:DM:FSK:VARiable:TYPE FSK4
SOURce:BB:DM:FSK:VARiable:SYMBol0:DEViation 135000
```

Manual operation: See "Deviation xxxx" on page 275

[:SOURce<hw>]:BB:DM:FSK:VARiable:TYPE <Type>

The command selects the modulation type for Variable FSK.

Parameters:

<Type> FSK4 | FSK8 | FSK16

*RST: FSK4

Example: See `[:SOURce<hw>]:BB:DM:FSK:VARiable:SYMBol<ch0>:DEViation` on page 902

Manual operation: See "FSK Type" on page 275

[:SOURce<hw>]:BB:DM:SWITching:STATe <State>

Enables switching between a modulated and an unmodulated signal.

Parameters:

<State> 0 | 1 | OFF | ON

*RST: 0

Example:

```
SOURce1:BB:DM:SWITching:STATe ON
```

Manual operation: See "State Modulation <=> CW Switching" on page 276

Power Ramping

[:SOURce<hw>]:BB:DM:PRAMp:SOURce <Source>

Sets the source for the power ramp control signals.

Parameters:

<Source> INTernal

*RST: INTernal

Example: See Example "Enable power ramping" on page 893

Options: R&S SMW-B10

Manual operation: See "Source" on page 279

[:SOURce<hw>]:BB:DM:PRAMp:SHAPe <Shape>

Sets the edge shape of the ramp envelope.

Parameters:

<Shape> LINear | COSine

*RST: COSine

Example: See Example "Enable power ramping" on page 893

Options: R&S SMW-B10

Manual operation: See "Ramp Function" on page 279

[:SOURce<hw>]:BB:DM:PRAMp:TIME <Time>

Sets the power ramping rise time and fall time for a burst.

Parameters:

<Time> float

Range: 0.25 to 16
Increment: 0.01
*RST: 1
Default unit: symbol

Example: See Example "Enable power ramping" on page 893

Options: R&S SMW-B10

Manual operation: See "Ramp Time" on page 279

[:SOURce<hw>]:BB:DM:PRAMp:FDELay <FDelay>
[:SOURce<hw>]:BB:DM:PRAMp:RDELay <RDelay>

Sets the delay in the rising edge.

Parameters:

<RDelay> float

Range: 0 to 4
Increment: 0.01
*RST: 0
Default unit: symbol

Example: See Example "Enable power ramping" on page 893

Options: R&S SMW-B10

Manual operation: See "Rise Delay" on page 279

[:SOURce<hw>]:BB:DM:PRAMp:ATTenuation <Attenuation>

Sets the level attenuation for signal ranges that are flagged with level attribute *attenuated* by the control signal.

Parameters:

<Attenuation> float

Range: 0 to 50
Increment: 0.1
*RST: 15
Default unit: dB

Example: See Example "Enable power ramping" on page 893

Options: R&S SMW-B10

Manual operation: See "Attenuation" on page 279

[:SOURce<hw>]:BB:DM:PRAMp:BBONly[:STATe] <State>

Enables power ramping in the baseband only or mixed power ramping in the baseband and the RF section.

The `ON` setting is mandatory if, with power ramping active, only the baseband signal is output (I/Q outputs)

In case of two-path instruments, the `ON` setting is also mandatory if a baseband signal is applied to two RF paths (RF A and RF B).

Only then can a signal with a defined, predictable level be output.

Parameters:

<State> 0 | 1 | OFF | ON

*RST: 0

Example: See Example "Enable power ramping" on page 893

Options: R&S SMW-B10

Manual operation: See "In Baseband Only" on page 280

[:SOURce<hw>]:BB:DM:PRAMp[:STATe] <State>

Enables or disables power ramping.

Parameters:

<State> 0 | 1 | OFF | ON

*RST: 0

Example: See Example "Enable power ramping" on page 893

Options: R&S SMW-B10

Manual operation: See "State" on page 278

Trigger Settings

[:SOURce<hw>]:BB:DM[:TRIGger]:SEQuence <Sequence>

Selects the trigger mode. For detailed description of the trigger modes, refer to "Impact of the Trigger Modes on the Signal Generation" on page 239.

Parameters:

<Sequence> AUTO | RETRigger | AAUTo | ARETrigger | SINGle

*RST: AUTO

Example: See Example "Adjusting clock, marker and trigger settings" on page 892

Manual operation: See "Trigger Mode" on page 255

[:SOURce<hw>]:BB:DM:TRIGger:SOURce <Source>

Selects the trigger signal source and determines the way the triggering is executed. Provided are:

- Internal triggering by a command (`INTernal`)
- External trigger signal via one of the local or global connectors
  - `EGT1|EGT2`: External global trigger
  - `EGC1|EGC2`: External global clock
  - `ELTRigger`: External local trigger
  - `ELCLock`: External local clock
- Internal triggering by a signal from the other basebands (`INTA|INTB`)
- In master-slave mode, the external baseband synchronization signal (`BBSY`)
- `OBASeband|BEXTernal|EXTernal`: Setting only
  Provided only for backward compatibility with other Rohde & Schwarz signal generators.
  The R&S SMW accepts these values and maps them automatically as follows:
  `EXTernal = EGT1`, `BEXTernal = EGT2`, `OBASeband = INTA or INTB` (depending on the current baseband)

Parameters:

<Source> INTB | INTernal | OBASeband | EGT1 | EGT2 | EGC1 | EGC2 | ELTRigger | INTA | ELCLock | BEXTernal | EXTernal | BBSY

*RST: INTernal

Example: See Example "Adjusting clock, marker and trigger settings" on page 892

Options: ELTRigger|ELCLock require R&S SMW-B10
BBSY require R&S SMW-B9

Manual operation: See "Trigger Source" on page 256

[:SOURce<hw>]:BB:DM:TRIGger:SLENgth <SLength>

Defines the length of the signal sequence to be output in the `SINGle` trigger mode.

Parameters:

<SLength> integer

Range: 1 to 4294967295
*RST: 1000
Default unit: symbol

Example: See Example "Adjusting clock, marker and trigger settings" on page 892

Manual operation: See "Trigger Signal Duration" on page 255

[:SOURce<hw>]:BB:DM:TRIGger:RMODe?

Queries the status of signal generation.

Return values:

<RMode> STOP | RUN

Example:

```
SOURce1:BB:DM:TRIGger:SOURce ELTRigger
SOURce1:BB:DM:TRIGger:SEQuence ARETrigger
SOURce1:BB:DM:TRIGger:RMODe?
Response: RUN
```

Usage: Query only

Manual operation: See "Running/Stopped" on page 255

[:SOURce<hw>]:BB:DM:TRIGger:EXTernal:SYNChronize:OUTPut <Output>

Enables signal output synchronous to the trigger event.

Parameters:

<Output> 0 | 1 | OFF | ON

*RST: 1

Example: See Example "Adjusting clock, marker and trigger settings" on page 892

Manual operation: See "Sync. Output to External Trigger/Sync. Output to Trigger" on page 256

[:SOURce<hw>]:BB:DM:TRIGger:ARM:EXECute

Stops signal generation; a subsequent internal or external trigger event restart signal generation.

Example: See also Example "Adjusting clock, marker and trigger settings" on page 892

Usage: Event

Manual operation: See "Arm" on page 255

[:SOURce<hw>]:BB:DM:TRIGger:EXECute

Executes a trigger.

Example: See Example "Adjusting clock, marker and trigger settings" on page 892

Usage: Event

Manual operation: See "Execute Trigger" on page 255

[:SOURce<hw>]:BB:DM:TRIGger:OBASeband:DELay <Delay>

Specifies the trigger delay (expressed as a number of symbols) for triggering by the trigger signal from the other path.

Parameters:

<Delay> float

Range: 0 to 2147483647
Increment: 0.01
*RST: 0

Example: See Example "Adjusting clock, marker and trigger settings" on page 892

Manual operation: See "(Specified) External Delay/(Specified) Trigger Delay" on page 257

[:SOURce<hw>]:BB:DM:TRIGger:OBASeband:RDELay?

Queries the time a trigger event form the other path is delayed.

Return values:

<ObResTimeDelSec> float

Range: 0 to 688
Increment: 0.25E-9
*RST: 0

Example: See Example "Specifying delay and inhibit values in time units" on page 922

Usage: Query only

Manual operation: See "Actual Trigger Delay/Actual External Delay" on page 258

[:SOURce<hw>]:BB:DM:TRIGger:OBASeband:TDELay <ObasTimeDelay>

Specifies the trigger delay for triggering by the signal from the other path.

Parameters:

<ObasTimeDelay> float

Range: 0 to 7929.170398682
Increment: 250E-12
*RST: 0
Default unit: s

Example: See Example "Specifying delay and inhibit values in time units" on page 922

Manual operation: See "(Specified) External Delay/(Specified) Trigger Delay" on page 257

[:SOURce<hw>]:BB:DM:TRIGger:OBASeband:INHibit <Inhibit>

Specifies the number of symbols by which a restart is inhibited. This command applies only for triggering by the second path.

Parameters:

<Inhibit> integer

Range: 0 to 67108863
*RST: 0
Default unit: symbol

Example: See Example "Adjusting clock, marker and trigger settings" on page 892

Manual operation: See "External / Trigger Inhibit" on page 257

[:SOURce<hw>]:BB:DM:TRIGger:DELay:UNIT <TrigDelUnit>

Determines the units the trigger delay is expressed in.

Parameters:

<TrigDelUnit> SAMPle | TIME

*RST: SAMPle

Example: See Example "Specifying delay and inhibit values in time units" on page 922

Manual operation: See "(External) Delay Unit" on page 257

[:SOURce<hw>]:BB:DM:TRIGger[:EXTernal]:DELay <Delay>

Specifies the trigger delay.

Parameters:

<Delay> float

Range: 0 to 2147483647
Increment: 0.01
*RST: 0
Default unit: symbol

Example: See Example "Adjusting clock, marker and trigger settings" on page 892

Manual operation: See "(Specified) External Delay/(Specified) Trigger Delay" on page 257

[:SOURce<hw>]:BB:DM:TRIGger[:EXTernal]:TDELay <ExtTimeDelay>

Specifies the trigger delay for external triggering. The value affects all external trigger signals.

Parameters:

<ExtTimeDelay> float

Range: 0 to 7929.170398682
Increment: 0.25E-9
*RST: 0
Default unit: s

Example: See Example "Specifying delay and inhibit values in time units" on page 922

Manual operation: See "(Specified) External Delay/(Specified) Trigger Delay" on page 257

[:SOURce<hw>]:BB:DM:TRIGger[:EXTernal]:RDELay?

Queries the time (in seconds) an external trigger event is delayed for.

Return values:

<ResTimeDelaySec> float

Range: 0 to 688
Increment: 0.25E-9
*RST: 0

Example: See Example "Specifying delay and inhibit values in time units" on page 922

Usage: Query only

Manual operation: See "Actual Trigger Delay/Actual External Delay" on page 258

[:SOURce<hw>]:BB:DM:TRIGger[:EXTernal]:INHibit <Inhibit>

Specifies the number of symbols by which a restart is inhibited.

Parameters:

<Inhibit> integer

Range: 0 to 21.47*symbRate
*RST: 0
Default unit: symbol

Example: See Example "Adjusting clock, marker and trigger settings" on page 892

Manual operation: See "External / Trigger Inhibit" on page 257

Marker Settings

[:SOURce<hw>]:BB:DM:TRIGger:OUTPut<ch>:MODE <Mode>

Defines the signal for the selected marker output.

For detailed description of the regular marker modes, refer to "Marker Modes" on page 235.

Parameters:

<Mode> CLISt | PULSe | PATTern | RATio

CLISt
A marker signal that is defined in the selected control list is generated.

*RST: RATio

Example: See Example "Adjusting clock, marker and trigger settings" on page 892

Manual operation: See "Marker Mode" on page 259

[:SOURce<hw>]:BB:DM:TRIGger:OUTPut<ch>:ONTime <OnTime>
[:SOURce<hw>]:BB:DM:TRIGger:OUTPut<ch>:OFFTime <OffTime>

Sets the number of symbols in a period (ON time + OFF time) for marker `RATio`.

*) If R&S SMW-B9 is installed, the minimum marker duration depends on the sample/symbol rate.

See "Marker Minimum Duration" on page 236.

Parameters:

<OffTime> integer

Range: 1 (R&S SMW-B10) / 16* (R&S SMW-B9) to 16777215
*RST: 1
Default unit: symbol

Example: See Example "Adjusting clock, marker and trigger settings" on page 892

Manual operation: See "Marker Mode" on page 259

[:SOURce<hw>]:BB:DM:TRIGger:OUTPut<ch>:PATTern <Pattern>

Defines the bit pattern used to generate the marker signal.

Parameters:

<Pattern> integer

0 = marker off, 1 = marker on

Range: #B0,1 to #B111...1,32
*RST: #B,1

Example: See Example "Adjusting clock, marker and trigger settings" on page 892

Manual operation: See "Marker Mode" on page 259

[:SOURce<hw>]:BB:DM:TRIGger:OUTPut<ch>:PULSe:DIVider <Divider>

Sets the divider for pulse marker mode (`PULSe`).

*) If R&S SMW-B9 is installed, the minimum marker duration depends on the sample/symbol rate.

See "Marker Minimum Duration" on page 236.

Parameters:

<Divider> integer

Range: 2 (R&S SMW-B10) / 32* (R&S SMW-B9) to 1024
*RST: 2

Example: See Example "Adjusting clock, marker and trigger settings" on page 892

Manual operation: See "Marker Mode" on page 259

[:SOURce<hw>]:BB:DM:TRIGger:OUTPut<ch>:PULSe:FREQuency?

Queries the pulse frequency of the pulsed marker signal `PULSe`.

Return values:

<Frequency> float

Example: See Example "Adjusting clock, marker and trigger settings" on page 892

Usage: Query only

Manual operation: See "Marker Mode" on page 259

[:SOURce<hw>]:BB:DM:TRIGger:OUTPut<ch>:DELay <Delay>

Defines the delay between the signal on the marker outputs and the start of the signal, expressed in terms of symbols.

Parameters:

<Delay> float

Range: 0 to 16777215
Increment: 0.001
*RST: 0

Example: See Example "Adjusting clock, marker and trigger settings" on page 892

Manual operation: See "Marker x Delay" on page 260

Clock Settings

[:SOURce<hw>]:BB:DM:CLOCk:MODE <Mode>

Sets the type of externally supplied clock.

Parameters:

<Mode> SYMBol

*RST: SYMBol

Example: See Example "Adjusting clock, marker and trigger settings" on page 892

Options: R&S SMW-B10

Manual operation: See "Clock Mode" on page 261

[:SOURce<hw>]:BB:DM:CLOCk:SOURce <Source>

Selects the clock source:

- `INTernal`: Internal clock reference
- `ELCLock`: External local clock
- `EXTernal = ELCLock`: Setting only
  Provided for backward compatibility with other Rohde & Schwarz signal generators

Parameters:

<Source> INTernal | ELCLock | EXTernal

*RST: INTernal

Example: See Example "Adjusting clock, marker and trigger settings" on page 892

Options: ELCLock requires R&S SMW-B10

Manual operation: See "Clock Source" on page 260

Handling List Files

[:SOURce<hw>]:BB:DM:CLISt:CATalog?
[:SOURce<hw>]:BB:DM:FLISt:CATalog?
[:SOURce<hw>]:BB:DM:MLISt:CATalog?
[:SOURce<hw>]:BB:DM:DLISt:CATalog?

Reads out the list files present in the default directory (see `:MMEMory:CDIRectory`).

List type	Command	File extension
Data list	`...:DLISt...`	`*.dm_iqd`
Control list	`...:CLISt...`	`*.dm_iqc`
User filter files	`...:FLISt...`	`*.vaf`
User mapping lists	`...:MLISt...`	`*.vam`

Refer to Chapter 13.7.2, "Accessing Files in the Default or in a Specified Directory", on page 777 for general information on file handling in the default and a specific directory.

The numeric suffix on `SOURce<[1]|2>` is ignored.

Return values:

<Catalog> "<filename1>,<filename2>,..."

Returns a string of file names separated by commas

Example: See Example "Performing general tasks" on page 891

Usage: Query only

Manual operation: See "Data Source" on page 269

[:SOURce<hw>]:BB:DM:CLISt:SELect <Filename>
[:SOURce<hw>]:BB:DM:FLISt:SELect <Filename>
[:SOURce<hw>]:BB:DM:MLISt:SELect <Filename>
[:SOURce<hw>]:BB:DM:DLISt:SELect <Select>

Selects the specified list file from the default directory (see `:MMEMory:CDIRectory`) or in the directory specified with the absolute file path.

If a list with the specified name does not yet exist, it is created. The file extension can be omitted.

Refer to Chapter 13.7.2, "Accessing Files in the Default or in a Specified Directory", on page 777 for general information on file handling in the default and a specific directory.

List type	Command	File extension
Data list	`...:DLISt...`	`*.dm_iqd`
Control list	`...:CLISt...`	`*.dm_iqc`
User standard	`...:ULISt...`	`*.dm_stu`

List type	Command	File extension
User filter files	`...:FLISt...`	`*.vaf`
User mapping lists	`...:MLISt...`	`*.vam`

Parameters:
<Select> <list name>

Example: See Example "Performing general tasks" on page 891

Manual operation: See "Data Source" on page 269

[:SOURce<hw>]:BB:DM:CLISt:DELete <Filename>
[:SOURce<hw>]:BB:DM:FLISt:DELete <Filename>
[:SOURce<hw>]:BB:DM:MLISt:DELete <Filename>
[:SOURce<hw>]:BB:DM:DLISt:DELete <Filename>

Deletes the specified list from the default directory (see `:MMEMory:CDIRectory`) or from the directory specified with the absolute file path.

Refer to Chapter 13.7.2, "Accessing Files in the Default or in a Specified Directory", on page 777 for general information on file handling in the default and a specific directory.

List type	Command	File extension
Data list	`...:DLISt...`	`*.dm_iqd`
Control list	`...:CLISt...`	`*.dm_iqc`
User standard	`...:ULISt...`	`*.dm_stu`
User filter files	`...:FLISt...`	`*.vaf`
User mapping lists	`...:MLISt...`	`*.vam`

Setting parameters:
<Filename> string

Example: See Example "Performing general tasks" on page 891

Usage: Setting only

Manual operation: See "Select Data List" on page 270

[:SOURce<hw>]:BB:DM:CLISt:FREE?
[:SOURce<hw>]:BB:DM:FLISt:FREE?
[:SOURce<hw>]:BB:DM:MLISt:FREE?
[:SOURce<hw>]:BB:DM:DLISt:FREE?

Queries the list free memory.

List type	Command	File extension
Data list	`...:DLISt...`	`*.dm_iqd`
Control list	`...:CLISt...`	`*.dm_iqc`
User filter files	`...:FLISt...`	`*.vaf`
User mapping lists	`...:MLISt...`	`*.vam`

Return values:

<Free> integer

Range: 0 to INT_MAX
*RST: 0

Example: See Example "Performing general tasks" on page 891

Usage: Query only

[:SOURce<hw>]:BB:DM:CLISt:POINts?

Queries the number of lines (2 bytes) in the currently selected list.

Return values:

<Points> integer

Range: 0 to INT_MAX
*RST: 0

Example:
```
SOURce:BB:DM:CLISt:SELect "c_list"
SOURce:BB:DM:CLISt:POINts?
```
Response: `20` (the control list consists of 20 lines)

Usage: Query only

Options: R&S SMW-B10

[:SOURce<hw>]:BB:DM:DLISt:POINts <Points>

Defines the number of bits in the selected data list to be utilized. When a list is being filled with block data, this data is only ever sent in multiples of 8 bits. However the exact number of bits to be exploited can be set to a different figure. The superfluous bits in the list are then ignored.

Parameters:

<Points> integer

Range: 0 to INT_MAX
*RST: 0

Example:
```
SOURce:BB:DM:DLISt:POINts 234
```
Defines the number of bits in the data list to be utilized as 234 bits. If the list was filled with block data, at least the last 6 bits are ignored.

[:SOURce<hw>]:BB:DM:FLISt:POINts?
[:SOURce<hw>]:BB:DM:MLISt:POINts?

Queries the user modulation mapping/user filter list length.

Return values:

<Points> integer

Range: max
*RST: 0

Example:
```
BB:DM:FORM USER
BB:DM:MLIS:POIN?
```

Usage: Query only

[:SOURce<hw>]:BB:DM:CLISt:COPY <Filename>
[:SOURce<hw>]:BB:DM:DLISt:COPY <Filename>

Copies the selected data list (`*.dm_iqd`)/ control list (`*.dm_iqc`) as a new list with name specified by `<Filename>`. If a list with the specified name exists, it is overwritten. If it does not yet exist, it is created.

The source file has to be available in the default directory (see `:MMEMory:CDIRectory`).

Refer to Chapter 13.7.2, "Accessing Files in the Default or in a Specified Directory", on page 777 for general information on file handling in the default and a specific directory.

Setting parameters:

<Filename> string

Example: See Example "Performing general tasks" on page 891

Usage: Setting only

Manual operation: See "Select Data List" on page 270

[:SOURce<hw>]:BB:DM:CLISt:DATA <Data>

Sends the data to the currently selected control list. If the list already contains data, it is *overwritten*. This command only writes data into the data section of the file.

The values for the control signals are sent, arranged in an 8-bit value as defined in Table 13-5.

Table 13-5: Contents of a control lists

Signal	Order	Decimal value of bits
Marker 1 Marker 2 Marker 3 Marker 4	LSB	1 2 4 8
Burst	LSB	16

Signal	Order	Decimal value of bits
LevAtt1	LSB	32
CWMod	LSB	64
Hop	MSB	128

The data can also be sent as a binary block, each binary block being a 2-byte value in which the 16 bits represent the binary values (16-bit unsigned integer, 2 bytes, LSB first).

When binary data transmission is in use, use the command `:SYSTem:COMMunicate:GPIB:LTERminator EOI` to set the termination character mode to 'EOI control data message only' so that a random `LF` in the data sequence is not interpreted as `End`, thereby prematurely terminating the data transmission. The command `...LTER STAN` resets the mode.

According to the specifications, the byte sequence is defined as 'most significant byte first'.

Tip: Control lists are created in binary format. You may however need the control list in an ASCII format, e.g for creating a waveform file with R&S WinIQSIM2. Refer to the example in "How to Create a Control List Using Tag File Format" on page 979 for description on how to create a control list file in ASCII format manually.

`*RST` has no effect on data lists.

Setting parameters:
<Data> string

Example:
```
MMEMory:CDIrectory "/var/user/clists"
SOURce1:BB:DM:CLISt:SELect "clist_marker4"
SOURce1:BB:DM:CLISt:DATA
0,0,0,0,8,8,8,0,0,0,0...
```
Enters the control values in list `clist_marker4`. In the example, only ramps for Marker 4 are set.

Usage: Setting only

Options: R&S SMW-B10

Manual operation: See "Select Ramp to Edit" on page 283

[:SOURce<hw>]:BB:DM:DLISt:DATA <Data>
[:SOURce<hw>]:BB:DM:DLISt:DATA? [<Start>[, <Count>]]

The **Setting** command sends the bit data to the selected data list. Any existing content in the list is *overwritten*. This command only writes data into the data section of the file.

Data can be sent as block data in binary or packet format (`:FORMat ASCii | PACKed`), each byte being interpreted as 8 data bits.

When binary data transmission is in use, use the command `:SYSTem:COMMunicate:GPIB:LTERminator EOI` to set the termination character mode to 'EOI control data message only' so that a random `LF` in the data sequence is not interpreted as `End`, thereby prematurely terminating the data transmission. The command `...LTER STAN` resets the mode.

According to the specifications, the byte sequence is defined as 'most significant byte first'.

The **query** reads out the data part of the list file. If the query is expanded by using the two parameters `<Start>` and `<Count>`, the list is read out in smaller sections. Without the parameters the total length is always read out starting from address 1.

`*RST` has no effect on data lists.

Parameters:

<Data> integer

bit data

Query parameters:

<Start> integer

Range: 1 to 2147483647

<Count> integer

Range: 1 to 2147483647

Example: See Example "Performing general tasks" on page 891

[:SOURce<hw>]:BB:DM:DLISt:DATA:APPend <Bits>

Appends the bit data onto the end of the existing data in the selected data list. Existing content in the data list is not overwritten. Hence, you can create long data lists piecemeal.

The command cannot be used with an empty data list, like for example data lists that has just been created. Use the command `[:SOURce<hw>]:BB:DM:DLISt:DATA` first and enter modulation data in the list.

`*RST` has no effect on data lists.

Setting parameters:

<Bits> 0 | 1 {,0 | 1 } | block data

Example: See Example "Performing general tasks" on page 891

Usage: Setting only

[:SOURce<hw>]:BB:DM:CLISt:TAG?
[:SOURce<hw>]:BB:DM:DLISt:TAG?

Queries the content of the specified tag in the selected file.

Return values:

<Tag> <control list>,<tag name>

Refer to "Tag Description" on page 962 for description of the available tag formats.

Example: `SOURce1:BB:DM:DLISt:TAG? "D_list","date"`

Queries the `Date` tag in list `D_list`.

Usage: Query only

13.18.4.3 SOURce:BB:ARBitrary Subsystem

This section list the commands of the `SOURce:BB:ARBitrary` subsystem.

Suffixes in the keywords ENTity<ch> and SOURce<hw>

You can address multiple entities configurations by using the SCPI commands starting with the keyword `SOURce` or the alias commands starting with the keyword `ENTity`.

Table 13-6: Value ranges of the suffixes ENTity<ch> and SOURce<hw> in advanced configuration with multiple entities

SCPI Syntax	ENTity<ch>	SOURce<hw>
SOURce<hw>:BB:...	-	1 .. 8
ENTity<ch>:SOURce<hw>:BB:...	1	1 .. 8
	2 .. 4	1 .. 4

See also Chapter 13.3, "SCPI Command Aliases for Advanced Mode with Multiple Entities", on page 767.

Required options

See Chapter 4.7.1, "Required Options", on page 300.

The commands are grouped in the following sections:

Programming Examples

Example: Creating test signals

The following example lists commands necessary to configure the different test signals.

```
// ******************************************************************
// Reset instrument first
// ******************************************************************
*RST; *CLS
SOURce:BB:ARBitrary:PRESet

// ******************************************************************
// Create test signals
// ******************************************************************
SOURce:BB:ARBitrary:TSIGnal:CIQ:I -0.5
SOURce:BB:ARBitrary:TSIGnal:CIQ:Q -0.33
SOURce:BB:ARBitrary:TSIGnal:CIQ:CREate

SOURce:BB:ARBitrary:TSIGnal:SINE:FREQuency 1MHz
SOURce:BB:ARBitrary:TSIGnal:SINE:SAMPles 100
SOURce:BB:ARBitrary:TSIGnal:SINE:PHASe -90
SOURce:BB:ARBitrary:TSIGnal:SINE:CREate:NAMed "/var/user/ARBtestSignals/sineTest"

SOURce:BB:ARBitrary:TSIGnal:RECTangle:FREQuency 100KHz
SOURce:BB:ARBitrary:TSIGnal:RECTangle:SAMPles 1000
SOURce:BB:ARBitrary:TSIGnal:RECTangle:AMPLitude 0.5
SOURce:BB:ARBitrary:TSIGnal:RECTangle:OFFSet -0.3
SOURce:BB:ARBitrary:TSIGnal:RECTangle:CREate:NAMed "/var/user/ARBtestSignals/rectTest"

SOURce:ARBitrary:STATe ON
```

Example: Managing waveform files

The following example lists the commands provided for handling of waveform files.

```
// ******************************************************************
// Reset instrument first
// ******************************************************************
*RST; *CLS
SOURce:BB:ARBitrary:PRESet

// ******************************************************************
// Set the default directory and list the available waveform files
// ******************************************************************
MMEM:CDIR "/var/user/ARBtestSigs"
SOURce:BB:ARBitrary:WAVeform:CATalog?
// sineTest,rectTest,ciqTestSignal,waveformTest,test2
SOURce:BB:ARBitrary:WAVeform:CATalog:LENGth?
// 5
```

```
// SOURce:BB:ARBitrary:WAVeform:FREE?

// ******************************************************************
// Select a waveform and querry information
// ******************************************************************
// SOURce:BB:ARBitrary:WAVeform:DELete "/var/user/ARBtestSigs/test2"
SOURce:BB:ARBitrary:WAVeform:SELect "/var/user/ARBtestSigs/wvTest"
SOURce:BB:ARBitrary:WAVeform:POINts?
// 100
SOURce:BB:ARBitrary:WAVeform:TAG? "TYPE"
// "SMU-WV"
SOURce:BB:ARBitrary:WAVeform:TAG? "COMMENT"
// "Waveform for test purposes"
// alternatively: query the comment tag of the current waveform file
// SOURce:BB:ARBitrary:WAVeform:DATA? "comment"
// "Waveform for test purposes"

// to query the date tag of a specific waveform file
// SOURce:BB:ARBitrary:WAVeform:DATA? "/var/user/ARBtestSigs/waveformTest","date"
// #2192014-04-15;16:19:30

// ******************************************************************
// Clock settings
// ******************************************************************
SOURce:BB:ARBitrary:CLOCk:SOURce INTernal
SOURce:BB:ARBitrary:CLOCk?
// 100000000
// or alternativelly use SOURce:BB:ARBitrary:WAVeform:TAG? "CLOCK"

// ******************************************************************
// Configure and enable standard marker signals
// ******************************************************************
SOURce:BB:ARBitrary:TRIGger:OUTPut1:MODE RESTart
SOURce:BB:ARBitrary:TRIGger:OUTPut2:MODE PULSe
SOURce:BB:ARBitrary:TRIGger:OUTPut2:PULSe:DIVider 5
SOURce:BB:ARBitrary:TRIGger:OUTPut2:PULSe:FREQuency?
// 20000000
SOURce:BB:ARBitrary:TRIGger:OUTPut3:MODE PATTern
SOURce:BB:ARBitrary:TRIGger:OUTPut3:PATTern #HE0F52,20
SOURce:BB:ARBitrary:TRIGger:OUTPut1:MODE RATio
SOURce:BB:ARBitrary:TRIGger:OUTPut1:ONTime 40
SOURce:BB:ARBitrary:TRIGger:OUTPut1:OFFTime 20

SOURce:BB:ARBitrary:TRIGger:OUTPut2:DELay 16

// ******************************************************************
// Configure and enable signal triggering; start ARB generator
// ******************************************************************
SOURce:BB:ARBitrary:TRIGger:SEQuence SINGle
SOURce:BB:ARBitrary:TRIGger:SLENgth 200
```

```
// the first 200 samples of the current waveform will be output after
// the next trigger event
// SOURce:BB:ARBitrary:TRIGger:SEQuence ARETrigger
// SOURce:BB:ARBitrary:TRIGger:SOURce EGT1
// external trigger signal must be provided at the USER connector
// SOURce:BB:ARBitrary:TRIGger:EXTernal:SYNChronize:OUTPut ON
// SOURce:BB:ARBitrary:TRIGger:EXTernal:DELay 200
// SOURce:BB:ARBitrary:TRIGger:EXTernal:INHibit 100

// SOURce:BB:ARBitrary:TRIGger:SOURce INTB
// the internal trigger signal from the other path must be used
// SOURce:BB:ARBitrary:TRIGger:OBASeband:DELay 25
// SOURce:BB:ARBitrary:TRIGger:OBASeband:INHibit 10

SOURce:BB:ARBitrary:TRIGger:SEQuence AAUTo
SOURce:BB:ARBitrary:TRIGger:SOURce INTernal
SOURce:BB:ARBitrary:STAT ON
SOURce:BB:ARBitrary:TRIGger:EXEC
```

Example: Specifying delay and inhibit values in time units

The following example lists the commands necessary to configure the instrument as described in "Specifying delay and inhibit values in time units" on page 243.

```
SOURce1:BB:ARBitrary:CLOCk 1000000
SOURce1:BB:ARBitrary:TRIGger:SEQuence AAUT
SOURce1:BB:ARBitrary:TRIGger:SOURce EGT1
SOURce1:BB:ARBitrary:TRIGger:DELay:UNIT SAMP
SOURce1:BB:ARBitrary:TRIGger:EXTernal:DELay 100
SOURce1:BB:ARBitrary:TRIGger:EXTernal:RDELay?
// Response: 100

SOURce1:BB:ARBitrary:TRIGger:DELay:UNIT TIME
SOURce1:BB:ARBitrary:TRIGger:EXTernal:TDELay 0.00001
SOURce1:BB:ARBitrary:TRIGger:EXTernal:RDELay?
// Response: 0.00001

SOURce1:BB:ARBitrary:TRIGger:DELay:UNIT SAMP
SOURce1:BB:ARBitrary:TRIGger:EXTernal:DELay 10
```

Example: Creating a multi-segment waveform

The following example lists the commands necessary to create a multi-segment waveform.

```
// ******************************************************************
// Reset instrument first
// ******************************************************************
*RST; *CLS
// SOURce:BB:ARBitrary:PRESet
```

```
// ******************************************************************
// Set the default directory and list the available waveform files
// ******************************************************************
MMEM:CDIR "/var/user/ARB/multi_segment"
SOURce:BB:ARBitrary:WAVeform:CATalog:LENGth?
// 4
SOURce:BB:ARBitrary:WAVeform:CATalog?
// Seg_0, Seg_1, Seg_2, Seg_3
// ******************************************************************
// List the available configuration files and select/create file
// ******************************************************************
SOURce:BB:ARBitrary:WSEGment:CONFigure:CATalog?
// multi_segment,ms_waveform
// the directory contains the configuration files multi_segment.inf_mswv
and ms_waveform.inf_mswv
SOURce:BB:ARBitrary:WSEGment:CONFigure:DELete "ms_waveform.inf_mswv"
SOURce:BB:ARBitrary:WSEGment:CONFigure:SELect
"/var/user/ARB/multi_segment/config.inf_mswv"
// creates new empty configuration file config.inf_mswv

// ******************************************************************
// Append waveforms to the multi segment sequence
// ******************************************************************
SOURce:BB:ARBitrary:WSEGment:CONFigure:SEGMent:APPend "Seg_0"
// Waveform Seg_0.wv will be the first segment of a
// multi segment waveform created with configuration file config.inf_mswv
SOURce:BB:ARBitrary:WSEGment:CONFigure:SEGMent:APPend "Seg_1"
SOURce:BB:ARBitrary:WSEGment:CONFigure:SEGMent:APPend "Seg_2"
SOURce:BB:ARBitrary:WSEGment:CONFigure:SEGMent:APPend "Seg_3"

SOURce:BB:ARBitrary:WSEGment:CONFigure:BLANk:APPend 1000,100000000
// adds a blank segment with 1000 samples and 100 MHz clock rate

SOURce:BB:ARBitrary:WSEGment:CONFigure:LEVel:MODE ERMS
SOURce:BB:ARBitrary:WSEGment:CONFigure:CLOCk:MODE HIGHest
// SOURce:BB:ARBitrary:WSEGment:CONFigure:CLOCk:MODE USER
// SOURce:BB:ARBitrary:WSEGment:CONFigure:CLOCk 30000000
SOURce:BB:ARBitrary:WSEGment:CONFigure:MARKer:MODE TAKE

SOURce:BB:ARBitrary:WSEGment:CONFigure:SELect
"/var/user/ARB/multi_segment/config.inf_mswv"
SOURce:BB:ARBitrary:WSEGment:CONFigure:COMMent "Multi Segment File"
SOURce:BB:ARBitrary:WSEGment:CONFigure:OFILe "ms_0to3"
SOURce:BB:ARBitrary:WSEGment:CONFigure:SEGMent:CATalog?
// /var/user/ARB/multi_segment/Seg_0.wv,/var/user/ARB/multi_segment/Seg_1.wv,
// /var/user/ARB/multi_segment/Seg_2.wv,/var/user/ARB/multi_segment/Seg_3.wv

// ******************************************************************
// Create and load the waveforms into the ARB generator; the ARB is activated
// the first segment is output deppending on the trigger settings
```

```
// ******************************************************************
// SOURce:BB:ARBitrary:WSEGment:CREate "/var/user/ARB/multi_segment/config.inf_mswv"
SOURce:BB:ARBitrary:WSEGment:CLOad "/var/user/ARB/multi_segment/config.inf_mswv"

SOURce:BB:ARBitrary:TRIGger:SEQuence AAUTo
SOURce:BB:ARBitrary:TRIGger:SOURce INTernal
SOURce:BB:ARBitrary:STAT ON
SOURce:BB:ARBitrary:TRIGger:EXEC
```

Example: Configuring the output order of the segments

The following example lists the commands necessary to trigger the output of the segments in desired playback order. The example lists only the relevant commands.

We assume, that the multi-segment sequence `ms_0to3` composed of four segments, Seg_0, Seg_1, Seg_2 and Seg_3 is created and loaded in the ARB (see Example "Creating a multi-segment waveform" on page 922). The required output order of the segments is Seg_0, Seg_3 [2], Seg_2.

```
// ******************************************************************
// Select a multi segment waveform
// ******************************************************************
MMEM:CDIR "/var/user/ARB/multi_segment"
SOURce:BB:ARBitrary:WSEGment:CONFigure:CATalog?
// config
SOURce:BB:ARBitrary:WSEGment:CONFigure:SELect "config"
SOURce:BB:ARBitrary:WSEGment:CONFigure:OFILe?
// "/var/user/ARB/multi_segment/ms_0to3"
SOURce:BB:ARBitrary:WSEGment:CLOad "/var/user/ARB/multi_segment/config.inf_mswv"

// ******************************************************************
// Adjust trigger settings and enable the ARB
// ******************************************************************
SOURce:BB:ARBitrary:TRIGger:SEQuence AAUTo
SOURce:BB:ARBitrary:TRIGger:SOURce INTernal
SOURce:BB:ARBitrary:STATe ON

// ******************************************************************
// Select the next segment trigger mode and source
// ******************************************************************
SOURce:BB:ARBitrary:TRIGger:SMODe NEXT
// SOURce:BB:ARBitrary:TRIGger:SMODe NSEam
SOURce:BB:ARBitrary:WSEGment:NEXT:SOURce INTernal

SOURce:BB:ARBitrary:WSEGment?
// 0
SOURce:BB:ARBitrary:WSEGment:NAME?
// "/var/user/ARB/multi_segment/Seg_0.wv"
// Seg_0 is output continuously

// ******************************************************************
```

```
// Trigger a switch over to the next segment
// ******************************************************************
SOURce:BB:ARBitrary:WSEGment:NEXT 3
// stops Seg_0 and starts immediatly Seg_3; Seg_3 is output continuously
SOURce:BB:ARBitrary:TRIGger:EXECute
//restarts Seg_3
SOURce:BB:ARBitrary:WSEGment:NEXT 2
SOURce:BB:ARBitrary:WSEGment:NEXT 0

// ******************************************************************
// Scrolling trough the segments, i.e. shwitch over in incremental order
// ******************************************************************
SOURce:BB:ARBitrary:WSEGment:NEXT:EXECute
SOURce:BB:ARBitrary:WSEGment?
// 1
SOURce:BB:ARBitrary:WSEGment:NEXT:EXECute
SOURce:BB:ARBitrary:WSEGment?
// 2
```

Refer to "To apply a manual segment switch to the output segments in the required order" on page 346 for description of the steps necessary to achieve a similar task via manual operation of the instrument.

Example: Using the ARB sequencer

The following example lists the commands necessary to configure a play list. The example lists only the relevant commands.

We assume, that the multi-segment sequence `ms_0to3` composed of four segments, Seg_0, Seg_1, Seg_2 and Seg_3 is created and loaded in the ARB (see Example "Creating a multi-segment waveform" on page 922). The required output order of the segments is Seg_0, Seg_3 [2], Seg_2.

The ARB Sequencer mode requires waveform files with equal clock rate!

```
// ******************************************************************
// Select a multi segment waveform composed from waveforms with equal clock rates
// ******************************************************************
MMEM:CDIR "/var/user/ARB/multi_segment"
SOURce:BB:ARBitrary:WSEGment:CONFigure:CATalog?
// config
SOURce:BB:ARBitrary:WSEGment:CONFigure:SELect "config"
SOURce:BB:ARBitrary:WSEGment:CLOad "/var/user/ARB/multi_segment/config.inf_mswv"
SOURce:BB:ARBitrary:WSEGment:CONFigure:OFILe?
// "/var/user/ARB/multi_segment/ms_0to3.wv"

// ******************************************************************
// Select a sequencing list (*.wvs)
```

```
// *******************************************************************
SOURce:BB:ARBitrary:WSEGment:SEQuence:SELect "play_list"
SOURce:BB:ARBitrary:WSEGment:SEQuence:APPend ON,0,1,NEXT
// adds the segment number 0 as a new (first) segment in the sequencing list
// this segment is activ and will be repeated once followed by the next segment
SOURce:BB:ARBitrary:WSEGment:SEQuence:APPend ON,3,2,NEXT
SOURce:BB:ARBitrary:WSEGment:SEQuence:APPend ON,2,1,SEG0

SOURce:BB:ARBitrary:TRIGger:SMODe SEQuencer
SOURce:BB:ARBitrary:TRIGger:SEQuence AAUTo
SOURce:BB:ARBitrary:TRIGger:SOURce INTernal
SOURce:BB:ARBitrary:STAT ON
SOURce:BB:ARBitrary:TRIGger:EXEC
```

Refer to "To apply a manual segment switch to the output segments in the required order" on page 346 for description of the steps necessary to achieve a similar task via manual operation of the instrument.

Example: Adding extra marker signals

The following example lists the commands necessary to add two marker signals, a sequence restart marker and a segment restart marker, to the multi-segment sequence. The example lists only the relevant commands.

```
// *******************************************************************
// Select the multi segment file and the corresponding configuration list
// *******************************************************************
MMEM:CDIR "/var/lists/ARB/multi_segment"
SOURce:BB:ARBitrary:WSEGment:CONFigure:CATalog?
// config
SOURce:BB:ARBitrary:WSEGment:CONFigure:SELect "config"
SOURce:BB:ARBitrary:WSEGment:CLOad "/var/user/ARB/multi_segment/config.inf_mswv"
SOURce:BB:ARBitrary:WSEGment:CONFigure:OFILe?
// "/var/user/ARB/multi_segment/ms_0to3.wv"

// *******************************************************************
// Enable restart markers on marker trace 1 and 2
// *******************************************************************
SOURce:BB:ARBitrary:WSEGment:CONFigure:MARKer:MODE IGNore
SOURce:BB:ARBitrary:WSEGment:CONFigure:MARKer:FSEGment MRK1
SOURce:BB:ARBitrary:WSEGment:CONFigure:MARKer:ESEGment MRK2

SOURce:BB:ARBitrary:TRIGger:OUTPut1:MODE UNCHanged
SOURce:BB:ARBitrary:TRIGger:OUTPut2:MODE UNCHanged
```

Example: Generating a multi carrier signal

The following example lists the commands necessary to generate a multi carrier signal.

```
// ******************************************************************
// Generating a multi carrier signal
// ******************************************************************
// Load a standardized 3GPP downlink test model, e.g. "Test Model 1 16 Channels"
SOURce1:BB:W3GPp:SETTing:TMODel:BSTation "Test_Model_1_16channels"'
// Confirm that the standardized 3GPP downlink test model is currently selected
SOURce1:BB:W3GPp:SETTing:TMODel:BSTation?
// "Test_Model_1_16channels"
// Enable the generation of 3GPP FDD signal
SOURce1:BB:W3GPp:STATe ON
// Generate a 3GPP FDD ARB waveform file with name "3gpp_arb"
// store the waveform in the default directory ("/var/user")
SOURce1:BB:W3GPp:WAVeform:CREate "3gpp_arb"

// ******************************************************************
// Configure a multi carrier scenario with 4 carriers
// and carrier spacing of 5 MHz
// Activate the carriers
// ******************************************************************
SOURce1:BB:ARBitrary:MCARrier:CARRier:COUNt 4
SOURce1:BB:ARBitrary:MCARrier:CARRier:SPACing 5 MHz
SOURce1:BB:ARBitrary:MCARrier:CARRier1:STATe ON
SOURce1:BB:ARBitrary:MCARrier:CARRier2:STATe ON
SOURce1:BB:ARBitrary:MCARrier:CARRier3:STATe ON
SOURce1:BB:ARBitrary:MCARrier:CARRier4:STATe ON

// ******************************************************************
// Select and load the waveform file "3gpp_arb" to all 4 carriers
// (the file "3gpp_arb" is in default directory)
// ******************************************************************
SOURce1:BB:ARBitrary:MCARrier:CARRier1:FILE "3gpp_arb"
SOURce1:BB:ARBitrary:MCARrier:CARRier2:FILE "3gpp_arb"
SOURce1:BB:ARBitrary:MCARrier:CARRier3:FILE "3gpp_arb"
SOURce1:BB:ARBitrary:MCARrier:CARRier4:FILE "3gpp_arb"

// ******************************************************************
// Trigger the signal caulation and load the waveform in the ARB generator
// ******************************************************************
// Define the file name of the multi carrier output file, e.g. "3gpp_mc"
SOURce1:BB:ARBitrary:MCARrier:OFILe "3gpp_mc"
// Create and load the multi carrier waveform file in the ARB generator
SOURce1:BB:ARBitrary:MCARrier:CLOad
// Alternatively: create the multi carrier waveform and
// load it subsequently in the ARB generator
// SOURce1:BB:ARBitrary:MCARrier:CREate
// SOURce1:BB:ARBitrary:WAVeform:SELect"3gpp_mc"

// Activate the ARB generator
SOURce1:BB:ARBitrary:STATe ON
```

```
// *******************************************************************
// Use the save/recall function to store the settings
// *******************************************************************
// Query available settings files in a specified directory
MMEM:CDIR "/var/user/waveform"
SOURce1:BB:ARBitrary:MCARrier:SETTing:CATalog?
// mcar1, mcar2
// the directory contains the settings files mcar1.arb_multcarr and mcar2.arb_multcarr
SOURce1:BB:ARBitrary:MCARrier:SETTing:STORe "3gpp_mc"
```

General Commands

[:SOURce<hw>]:BB:ARBitrary:PRESet

Sets all ARB generator parameters to their default values.

Example: See "Programming Examples" on page 920

Usage: Event

Manual operation: See "Set To Default" on page 308

[:SOURce<hw>]:BB:ARBitrary:STATe <State>

Enables the ARB generator.

A waveform must be selected before the ARB generator is activated.

Parameters:

<State> 0 | 1 | OFF | ON

*RST: 0

Example: See "Programming Examples" on page 920

Manual operation: See "State" on page 307

Test Signal Commands

[:SOURce<hw>]:BB:ARBitrary:TSIGnal:CIQ:I <I>
[:SOURce<hw>]:BB:ARBitrary:TSIGnal:CIQ:Q <TSig>

Sets the value for the I and Q component of the test signal

Parameters:

<TSig> float

Range: -1 to 1
Increment: 0.001
*RST: 0
Default unit: FS

Example: See Example "Creating test signals" on page 920

Manual operation: See "I Value, Q Value" on page 315

[:SOURce<hw>]:BB:ARBitrary:TSIGnal:RECTangle:AMPLitude <Amplitude>

Sets the digital amplitude of the rectangular wave.

Parameters:

<Amplitude> float

Range: 0 to 1
Increment: 0.001
*RST: 0.800
Default unit: FS

Example: see Example "Creating test signals" on page 920

Manual operation: See "Amplitude" on page 313

[:SOURce<hw>]:BB:ARBitrary:TSIGnal:RECTangle:FREQuency <Frequency>

Sets the frequency of the test signal.

Parameters:

<Frequency> float

Range: 100 to depends on the installed options
Increment: 0.01
*RST: 1000
Default unit: Hz

Example: See Example "Creating test signals" on page 920.

Manual operation: See "Frequency" on page 313

[:SOURce<hw>]:BB:ARBitrary:TSIGnal:RECTangle:OFFSet <Offset>

Sets the DC component.

Parameters:

<Offset> float

Range: -1 to 1
Increment: 0.001
*RST: 0
Default unit: FS

Example: see Example "Creating test signals" on page 920

Manual operation: See "Offset DC" on page 314

[:SOURce<hw>]:BB:ARBitrary:TSIGnal:RECTangle:SAMPles <Samples>

Sets the number of sample values required for the rectangular signal per period.

Parameters:

<Samples> integer

Range: 4 to 1000
*RST: 100

Example: See Example "Creating test signals" on page 920

Manual operation: See "Samples per Period" on page 313

[:SOURce<hw>]:BB:ARBitrary:TSIGnal:SINE:FREQuency <Frequency>

Sets the frequency of the simple sinusoidal test signal.

Parameters:

<Frequency> float

Range: 100 to depends on the installed options
Increment: 0.01
*RST: 1000
Default unit: Hz

Example: See Example "Creating test signals" on page 920.

Manual operation: See "Frequency" on page 311

[:SOURce<hw>]:BB:ARBitrary:TSIGnal:SINE:PHASe <Phase>

Sets the phase offset of the sine wave on the Q channel relative to the sine wave on the I channel.

Parameters:

<Phase> float

Range: -180 to 180
Increment: 0.01
*RST: 90
Default unit: DEG

Example: see Example "Creating test signals" on page 920

Manual operation: See "Phase Offset Q" on page 312

[:SOURce<hw>]:BB:ARBitrary:TSIGnal:SINE:SAMPles <Samples>

Sets the sample rate for the sine signal in samples per period.

The resulting clock rate must not exceed the maximum ARB clock rate (see data sheet).

The maximum value is automatically restricted by reference to the set frequency and has to fulfill the rule *Frequency * Samples <= ARB clock rate*.

Parameters:

<Samples> integer

Range: 4 to 1000
*RST: 100

Example: see Example "Creating test signals" on page 920

Manual operation: See "Samples per Period" on page 312

[:SOURce<hw>]:BB:ARBitrary:TSIGnal:CIQ:CREate:NAMed <Filename>
[:SOURce<hw>]:BB:ARBitrary:TSIGnal:RECTangle:CREate:NAMed <Filename>
[:SOURce<hw>]:BB:ARBitrary:TSIGnal:SINE:CREate:NAMed <Filename>

Generates a signal and saves it to a waveform file.

Setting parameters:

<Filename> string

Example: See Example "Creating test signals" on page 920

Usage: Setting only

Manual operation: See "Generate Signal HD" on page 312

[:SOURce<hw>]:BB:ARBitrary:TSIGnal:CIQ:CREate
[:SOURce<hw>]:BB:ARBitrary:TSIGnal:RECTangle:CREate
[:SOURce<hw>]:BB:ARBitrary:TSIGnal:SINE:CREate

Generates a signal and uses it as output straight away.

Example: see Example "Creating test signals" on page 920

Usage: Event

Manual operation: See "Generate Signal RAM" on page 312

Waveform Commands

The following rule applies for all commands described in this section.

By default, the waveform files are saved in the default directory of the instrument, that is the `/var/user/` directory or the directory specified with the command `:MMEMory:CDIRectory`. To access the waveform files in this default directory, only the file name is required, without the path and the file extension (`*.wv`).

However, to access waveform files located in a directory different to the default one, the complete file path and file name are required.

[:SOURce<hw>]:BB:ARBitrary:WAVeform:CATalog?

Reads out the files extension `*.wv` in the default directory.

Return values:

<Catalog> string

Returns a list of the file names separated by commas

Example: see Example "Managing waveform files" on page 920

Usage: Query only

[:SOURce<hw>]:BB:ARBitrary:WAVeform:CATalog:LENGth?

Reads out the files with extension `*.wv` in the default directory and returns the number of waveform files in this directory. The default directory is set using command `MMEM:CDIRectory`.

Return values:

<Length>	integer Number of waveform files in default directory Range: 0 to INT_MAX *RST: 0
Example:	see Example "Managing waveform files" on page 920
Usage:	Query only

[:SOURce<hw>]:BB:ARBitrary:WAVeform:SELect <Filename>

Selects an existing waveform file, i.e. file with extension `*.wv`.

Parameters:

<Filename>	string
Example:	See Example "Managing waveform files" on page 920
Manual operation:	See "Load Waveform/File" on page 308

[:SOURce<hw>]:BB:ARBitrary:WAVeform:DELete <Filename>

Deletes the specified waveform file. If the file is not on the default path, the path must be specified at the same time. The file extension may be omitted. Only files with the file extension `*.wv` are deleted.

Setting parameters:

<Filename>	string
Example:	See Example "Managing waveform files" on page 920
Usage:	Setting only

[:SOURce<hw>]:BB:ARBitrary:WAVeform:FREE?

Queries the free disk space on the default path of the instrument's hard disk.

Return values:

<Free>	integer Range: 0 to INT_MAX *RST: 1
Example:	See Example "Managing waveform files" on page 920
Usage:	Query only

[:SOURce<hw>]:BB:ARBitrary:WAVeform:POINts?

Queries the number of samples (the number of I/Q values pairs) in the selected waveform file.

Return values:

<Points> <waveform filename>

Range: 0 to 1000
*RST: 1

Example: see Example "Managing waveform files" on page 920

Usage: Query only

[:SOURce<hw>]:BB:ARBitrary:WAVeform:DATA <Filename>, <Data>
[:SOURce<hw>]:BB:ARBitrary:WAVeform:DATA? [<Filename>,]<Tag>

The **setting** command writes the binary block data `<data>` to the file identified by `<filename>`. The *complete content* of the waveform file (i.e. including all tags) must be specified; the complete content is transmitted as binary data block.

I/Q data and the marker data can also be written to a file with the command `:MMEMory:DATA:UNPRotected`.

Tip: To ensure trouble-free data transmission, set the GPIB delimiter to EOI.

The **query** command retrieves the content of the specified tag of the currently selected waveform file or the waveform file specified with the `<filename>`.

See also "Tag Description" on page 962.

Setting parameters:

<Data> block data

Binary block data with the following syntax:
`#<Digits><Length><Binary data>`

#
Indicates the start of the binary block

<Digits>
Decimal value
Gives the number of decimal digits used for the `<Length>` value

<Length>
Decimal value
Number of bytes the follow in the `<Binary data>` part

<Binary data>
Binary data in ASCII format

Query parameters:

<Tag> 'comment' | 'copyright' | 'date' | 'lacpfilter' | 'marker name' | 'poweroffset'

Parameters for setting and query:

<Filename> string

Specifies the name of the waveform file in that the binary data is copied

Example: **Query**

See Example "Managing waveform files" on page 920

Example: **Setting**

```
SOURce:BB:ARB:WAVeform:DATA "/var/user/test1.wv",#220<binary data>
// Writes the binary block data <binary data> to file test1.wv
// <binary data> contains 20 bytes
// <binary data> is a placeholder; the actual ASCII values are not printa
```

[:SOURce<hw>]:BB:ARBitrary:WAVeform:TAG?

Queries the content of the specified tag of the selected waveform file (see also Chapter 13.18.4.4, "Waveform, Data and List Format", on page 961).

Return values:

<Tag> 'comment' | 'copyright' | 'date' | 'lacpfilter' | 'marker name' | 'poweroffset' | 'samples'

Example: see Example "Managing waveform files" on page 920

Usage: Query only

[:SOURce<hw>]:BB:ARBitrary:CLOCk <Clock>

Sets the clock rate.

If you load a waveform, the clock rate is determined as defined with the waveform tag `{CLOCK: frequency}`. This command subsequently changes the clock rate; see data sheet for value range.

In the case of an external clock source, the clock of the external source must be entered with this command.

Parameters:

<Clock> float

Range: depends on the installed options, e.g. 400 Hz to 200 MHz (R&S SMW-B10)

Increment: 0.001

Default unit: Hz

Example: see Example "Managing waveform files" on page 920

Manual operation: See "Clock Frequency" on page 309

[:SOURce<hw>]:BB:ARBitrary:COUPled:STATe <State>

In an instrument configuration with Coupled sources, selects that all basebands use the same waveform.

Parameters:

<State> 0 | 1 | OFF | ON

*RST: 0

Example:

```
SCONfiguration:PRESet
SCONfiguration:MODE ADV
SCONfiguration:FADing SISO4X1X1
SCONfiguration:BASeband:SOURce COUP
SCONfiguration:APPLy

ENTity1:SOURce1:BB:ARBitrary:COUPled:STATe 0
ENTity1:SOURce1:BB:ARBitrary:WAVeform:SELect "/var/user/lte"
ENTity1:SOURce2:BB:ARBitrary:WAVeform:SELect "/var/user/lte"
ENTity1:SOURce3:BB:ARBitrary:WAVeform:SELect "/var/user/3gpp"
ENTity1:SOURce4:BB:ARBitrary:WAVeform:SELect "/var/user/gsm"
ENTity1:SOURce4:BB:ARBitrary:COUPled:TRIGger:DELay:OFFSet 100
SOURce1:BB:ARBitrary:STATe 1
```

Manual operation: See "Use one Waveform for all basebands" on page 308

[:SOURce<hw>]:BB:ARBitrary:COUPled:TRIGger:DELay:OFFSet <Offset>

Sets a time delay to delay the waveform processing of a particular baseband.

Parameters:

<Offset> float

Range: 0 to 2147483647/clockrate
Increment: 250E-12
*RST: 0

Example: see [:SOURce<hw>]:BB:ARBitrary:COUPled:STATe on page 934

Manual operation: See "Delay" on page 309

[:SOURce<hw>]:BB:ARBitrary:WAVeform:CLOCk <Filename>, <Clock>

Appends information on the ARB clock rate to specified waveform file. The file must contain I/Q and/or marker data and have been created with the command :MMEMory:DATA:UNPRotected on page 783.

Parameters:

<Filename> string

Complete file path and file name with file extension (*.wv).
If the file is in the default directory, the file path can be omitted.

Setting parameters:

<Clock> float

Range: 400 to 100E6
Increment: 1E-3
*RST: 1E6

Example: see :MMEMory:DATA:UNPRotected on page 783

Multi-Segment Commands

Required options

See Chapter 4.8.1, "Required Options", on page 324.

[:SOURce<hw>]:BB:ARBitrary:WSEGment?

Queries the index of the currently processed segment.

Return values:

<WSegment> integer

Range: 0 to 1023
*RST: 0

Example: see Example "Configuring the output order of the segments" on page 924

Usage: Query only

Manual operation: See "Current Segment/Current Segment Index" on page 343

[:SOURce<hw>]:BB:ARBitrary:WSEGment:NAME?

Queries the name of the waveform of the currently output segment of the multi-segment waveform.

Return values:

<Name> string

Example: See Example "Configuring the output order of the segments" on page 924

Usage: Query only

Manual operation: See "Current Segment/Current Segment Index" on page 343

[:SOURce<hw>]:BB:ARBitrary:WSEGment:NEXT <Next>

Selects the segment to be output.

Parameters:

<Next> integer

Range: 0 to 1023
*RST: 0

Example: see Example "Configuring the output order of the segments" on page 924

Manual operation: See "Segment" on page 343

[:SOURce<hw>]:BB:ARBitrary:WSEGment:NEXT:EXECute

Triggers manually switchover to the subsequent segment in the multi-segment file.

This command is disabled, if a sequencing play list is enabled.

Example: See Example "Configuring the output order of the segments" on page 924

Usage: Event

Manual operation: See "Execute Next Segment" on page 345

[:SOURce<hw>]:BB:ARBitrary:WSEGment:NEXT:SOURce <Source>

Selects the next segment source.

Parameters:

<Source> INTernal | NSEGM1 | NSEGM2

*RST: INTernal

Example: see Example "Configuring the output order of the segments" on page 924

Options: NSEGM1|NSEGM2 require R&S SMW-B10

Manual operation: See "Next Segment Source" on page 344

[:SOURce<hw>]:BB:ARBitrary:WSEGment:LMODe <LevelMode>

Sets how the segments are leveled.

Parameters:

<LevelMode> HIGHest | UNCHanged

*RST: HIGHest

Example: `:SOURce1:BB:ARBitrary:WSEGment:LMODe HIGHest`

Manual operation: See "Level Mode" on page 344

[:SOURce<hw>]:BB:ARBitrary:WSEGment:SEQuence:SELect <Filename>

Selects the sequencing list (files with extension `*.wvs`)

Parameters:

<Filename> string

Example: see Example "Using the ARB sequencer" on page 925

Manual operation: See "New/Load Sequencing List" on page 339

[:SOURce<hw>]:BB:ARBitrary:WSEGment:SEQuence:APPend <State>, <Segment>, <Count>, <Next>

Appends a new segment to the selected sequencing play list.

Setting parameters:

<State> ON | OFF

Activates/deactivates the appended segment

*RST: ON

<Segment> integer

Indicates the number of the segment as in the multi-segment waveform file

Range: 0 to SegmentCount - 1

<Count> integer

Defines how many times this segment is repeated

Range: 1 to 1048575

<Next> NEXT | BLANk | ENDLess | SEG0 | SEG1 | ... | SEG31 | 0...maxSegment

Determines the action after completing the current segment, like for instance which segment is processed after the processing of the current one is finished.

Example: See Example "Using the ARB sequencer" on page 925

Usage: Setting only

Manual operation: See "Append" on page 341

[:SOURce<hw>]:BB:ARBitrary:WSEGment:CREate <FilenameInput>

Creates a multi-segment waveform (`*.wv`) using the current settings of the specified *configuration file* (`*.inf_mswv`).

Setting parameters:

<FilenameInput> Absolute file path, file name of the configuration file and file extension (*.inf_mswv)

Example: See Example "Creating a multi-segment waveform" on page 922

Usage: Setting only

Manual operation: See "Save List/Save List As..." on page 332

[:SOURce<hw>]:BB:ARBitrary:WSEGment:CLOad <FilenameInput>

Creates a multi-segment waveform using the current entries of the specified *configuration file* (`*.inf_mswv`).

The ARB generator is activated, the new multi-segment waveform (`*.wv`) is loaded and the first segment is output in accordance to the trigger settings.

Setting parameters:

<FilenameInput> string

Absolute file path, file name of the configuration file and file extension (`*.inf_mswv`)

Example: See Example "Creating a multi-segment waveform" on page 922

Usage: Setting only

Manual operation: See "Save List/Save List As..." on page 332

[:SOURce<hw>]:BB:ARBitrary:WSEGment:CONFigure:MARKer:MODE <Mode>

Defines the way the marker information within the separate segments is processed.

Parameters:

<Mode> IGNore | TAKE

*RST: TAKE

Example: see Example "Adding extra marker signals" on page 926

Manual operation: See "Segment Marker" on page 336

[:SOURce<hw>]:BB:ARBitrary:WSEGment:CONFigure:MARKer:ESEGment <Mode>

[:SOURce<hw>]:BB:ARBitrary:WSEGment:CONFigure:MARKer:FSEGment <Mode>

Enables/disables the generation of an additional marker restart signal at the beginning of the first segment (`FSEGment`) or at the beginning of each segment (`ESEGment`).

If additional marker generation is enabled, the existing marker signals in the individual segment waveform files are not considered.

Parameters:

<Mode> OFF | MRK1 | MRK2 | MRK3 | MRK4

*RST: OFF

Example: see Example "Adding extra marker signals" on page 926

Manual operation: See "Sequence Restart" on page 337

[:SOURce<hw>]:BB:ARBitrary:TRIGger:SMODe <SMode>

Selects the extended trigger mode for multi segment waveforms.

Parameters:

<SMode> SAME | NEXT | SEQuencer | NSEam

NSEam = Next Segment Seamless

*RST: NEXT

Example: see Example "Configuring the output order of the segments" on page 924

Manual operation: See "Next Segment Mode" on page 343

Multi-Segment Sequencing Commands

Required options

See Chapter 4.8.1, "Required Options", on page 324.

[:SOURce<hw>]:BB:ARBitrary:WSEGment:CONFigure:BLANk:APPend <SampCount>, <Frequency>

Adds a blank segment to the multi-segment file.

Setting parameters:

<SampCount> float

Specifies the number of samples.

Range: 512 to 1E7
Increment: 1
*RST: 1000

<Frequency> float

Determines the clock rate.

Range: 400 Hz to depends on the installed options
Increment: 0.001
*RST: 1E8

Example: See Example "Creating a multi-segment waveform" on page 922

Usage: Setting only

Manual operation: See "Blank Segment" on page 334

[:SOURce<hw>]:BB:ARBitrary:WSEGment:CONFigure:CATalog?

Queries the available configuration files in the default directory. See also Chapter 4.8.2.3, "File Concept", on page 326.

Return values:

<Catalog> string

Example: see Example "Creating a multi-segment waveform" on page 922

Usage: Query only

Manual operation: See "Load List" on page 332

[:SOURce<hw>]:BB:ARBitrary:WSEGment:CONFigure:CLOCk <Clock>

Defines the clock rate used for multi-segment waveform output if the clock mode is `USER`.

Parameters:

<Clock> float

Increment: 1E-3
*RST: max SampleRate

Example: See Example "Creating a multi-segment waveform" on page 922

Manual operation: See "User Clock Rate" on page 336

[:SOURce<hw>]:BB:ARBitrary:WSEGment:CONFigure:CLOCk:MODE <Mode>

Selects the clock rate mode for the multi segment waveform. Use the command `[:SOURce<hw>]:BB:ARBitrary:WSEGment:CONFigure:CLOCk` to define the clock in clock mode user.

Parameters:

<Mode> UNCHanged | HIGHest | USER

*RST: UNCHanged

Example: See Example "Creating a multi-segment waveform" on page 922

Manual operation: See "Clock" on page 336

[:SOURce<hw>]:BB:ARBitrary:WSEGment:CONFigure:COMMent <Comment>

Enters a comment for the selected configuration file.

Parameters:

<Comment> string

Example: See Example "Creating a multi-segment waveform" on page 922

Manual operation: See "Comment" on page 332

[:SOURce<hw>]:BB:ARBitrary:WSEGment:CONFigure:DELete <Filename>

Deletes the selected configuration file.

Setting parameters:

<Filename> string

Example: see Example "Creating a multi-segment waveform" on page 922

Usage: Setting only

Manual operation: See "Multi Segment Table, Append/Delete/Shift Seg." on page 334

[:SOURce<hw>]:BB:ARBitrary:WSEGment:CONFigure:LEVel[:MODE] <Mode>

Selects the level mode, unchanged or equal RMS, for the multi-segment waveform.

Parameters:

<Mode> UNCHanged | ERMS

*RST: UNCHanged

Example: See Example "Creating a multi-segment waveform" on page 922

Manual operation: See "Level" on page 336

[:SOURce<hw>]:BB:ARBitrary:WSEGment:CONFigure:OFILe <OFile>

Defines the file name of the output multi-segment waveform.

Parameters:

<OFile> string

Example: See Example "Creating a multi-segment waveform" on page 922

Manual operation: See "Save List/Save List As..." on page 332

[:SOURce<hw>]:BB:ARBitrary:WSEGment:CONFigure:SEGMent:APPend <Waveform>

Appends the specified waveform to the configuration file.

Setting parameters:

<Waveform> string

Example: see Example "Creating a multi-segment waveform" on page 922

Usage: Setting only

Manual operation: See "Multi Segment Table, Append/Delete/Shift Seg." on page 334

[:SOURce<hw>]:BB:ARBitrary:WSEGment:CONFigure:SEGMent:CATalog?

Queries the segments of the currently selected configuration file.

Return values:

<Catalog> string

Example: see Example "Creating a multi-segment waveform" on page 922

Usage: Query only

Manual operation: See "Multi Segment Table, Append/Delete/Shift Seg." on page 334

[:SOURce<hw>]:BB:ARBitrary:WSEGment:CONFigure:SELect <Filename>

Selects a configuration file from the default directory. If a configuration file with the specified name does not yet exist, it is created. The file extension `*.inf_mswv` may be omitted.

Parameters:

<Filename>	string
Example:	see Example "Creating a multi-segment waveform" on page 922
Manual operation:	See "New List" on page 332

Multi-Carrier Commands

Required options

See Chapter 4.9.1, "Required Options", on page 355.

[:SOURce<hw>]:BB:ARBitrary:MCARrier:PRESet

Sets all the multi-carrier parameters to their default values.

Example:	`SOURce1:BB:ARBitrary:MCARrier:PRESet`
Usage:	Event
Manual operation:	See "Set to Default" on page 358

[:SOURce<hw>]:BB:ARBitrary:MCARrier:SETTing:CATalog?

Queries the available settings files in the specified default directory. Only files with the file extension `*.arb_multcarr` are listed.

Return values:

<Catalog>	string
Example:	See Example "Generating a multi carrier signal" on page 926
Usage:	Query only
Manual operation:	See "Save/Recall Frame" on page 359

[:SOURce<hw>]:BB:ARBitrary:MCARrier:SETTing:LOAD <Filename>

Loads the selected file from the default or the specified directory. Loaded are files with extension `*.arb_multcarr`.

Setting parameters:

<Filename>	"<filename>" Filename or complete file path; file extension can be omitted
Example:	See Example "Generating a multi carrier signal" on page 926.
Usage:	Setting only
Manual operation:	See "Save/Recall Frame" on page 359

[:SOURce<hw>]:BB:ARBitrary:MCARrier:SETTing:STORe <Filename>

Stores the current settings into the selected file; the file extension (`*.arb_multcarr`) is assigned automatically.

Setting parameters:

<Filename> string

Filename or complete file path

Example: See Example "Generating a multi carrier signal" on page 926

Usage: Setting only

Manual operation: See "Save/Recall Frame" on page 359

[:SOURce<hw>]:BB:ARBitrary:MCARrier:CARRier:COUNt <Count>

Sets the number of carriers in the ARB multi-carrier waveform.

Parameters:

<Count> integer

Range: 1 to 512
*RST: 1

Example: See Example "Generating a multi carrier signal" on page 926

Manual operation: See "Number of Carriers" on page 359

[:SOURce<hw>]:BB:ARBitrary:MCARrier:CARRier:MODE <Mode>

The command sets the carrier frequency mode.

Parameters:

<Mode> EQUidistant | ARBitrary

*RST: EQUidistant

Example: `BB:ARB:MCAR:CARR:MODE EQU`
Sets an equidistant carrier spacing. The carrier frequency cannot be set.

Manual operation: See "Mode" on page 359

[:SOURce<hw>]:BB:ARBitrary:MCARrier:CARRier:SPACing <Spacing>

Sets the frequency spacing between adjacent carriers of the multi-carrier waveform (see Chapter 4.9.2.1, "Defining the Carrier Frequency", on page 356).

Parameters:

<Spacing> float

Range: 0.0 to depends on the installed options, e.g. 120E6 (R&S SMW-B10)
Increment: 0.01
*RST: 0
Default unit: Hz

Example: See Example "Generating a multi carrier signal" on page 926

Manual operation: See "Carrier Spacing" on page 360

[:SOURce<hw>]:BB:ARBitrary:MCARrier:SAMPles?

Queries the resulting file size.

Return values:

<Samples> integer

Range: 0 to INT_MAX
*RST: 0
Default unit: samples

Example: `SOURce1:BB:ARBitrary:MCARrier:SAMPles?`
Queries the file size of the currently calculated multi-carrier waveform.

Usage: Query only

Manual operation: See "File Size" on page 364

[:SOURce<hw>]:BB:ARBitrary:MCARrier:TIME <Time>

Sets the user-defined signal period.

Parameters:

<Time> float

Range: 0 to 1E9
Increment: 1E-9
*RST: 0
Default unit: s

Example: `SOURce1:BB:ARBitrary:MCARrier:MODE USER`
Selects signal period mode user.
`SOURce1:BB:ARBitrary:MCARrier:TIME 10`
Sets a signal period of 10 seconds

Manual operation: See "Signal Period" on page 361

[:SOURce<hw>]:BB:ARBitrary:MCARrier:CFACtor:MODE <Mode>

Sets the mode for optimizing the crest factor by calculating the carrier phases.

Parameters:

<Mode>	OFF | MIN | MAX *RST: OFF
Example:	`SOURce1:BB:ARBitrary:MCARrier:CFACtor:MODE OFF` Switches off automatic crest factor optimization.
Manual operation:	See "Crest Factor Mode" on page 360

[:SOURce<hw>]:BB:ARBitrary:MCARrier:CLIPping:CFACtor <CFactor>

Sets the value of the desired crest factor, if baseband clipping is enabled.

A target crest factor above the crest factor of the unclipped multicarrier signal has no effect.

Parameters:

<CFactor>	float Range: -50 to 50 Increment: 0.01 *RST: 50 Default unit: dB
Example:	see `[:SOURce<hw>]:BB:ARBitrary:MCARrier:CLIPping[:STATe]` on page 946
Manual operation:	See "Target Crest Factor" on page 360

[:SOURce<hw>]:BB:ARBitrary:MCARrier:CLIPping:CUToff <Cutoff>

Sets the cutoff frequency of the final low pass filter, if baseband clipping is enabled.

Parameters:

<Cutoff>	float Range: 0 to 50E6 Increment: 0.01 *RST: 50E6 Default unit: MHz
Example:	`[:SOURce<hw>]:BB:ARBitrary:MCARrier:CLIPping[:STATe]` on page 946
Manual operation:	See "Filter Cutoff Frequency" on page 360

[:SOURce<hw>]:BB:ARBitrary:MCARrier:CLIPping[:STATe] <State>

Switches baseband clipping on and off.

Parameters:

<State>	0 | 1 | OFF | ON *RST: 0

Example:
```
SOURce1:BB:ARBitrary:MCARrier:CLIPping:STATe ON
SOURce1:BB:ARBitrary:MCARrier:CLIPping:CFACtor
37
SOURce1:BB:ARBitrary:MCARrier:CLIPping:CUToff
50
```

Manual operation: See "Clipping" on page 360

[:SOURce<hw>]:BB:ARBitrary:MCARrier:TIME:MODE <Mode>

Selects the mode for calculating the resulting signal period of the multi-carrier waveform. The resulting period is always calculated for all carriers in the carrier table irrespective of their state (ON/OFF).

Parameters:

<Mode> USER | LONG | SHORt | LCM

*RST: LONG

Example: `BB:ARB:MCAR:TIME:MODE LONG`
The resulting signal period is defined by the longest I/Q file in the carrier table. Shorter I/Q files are periodically repeated.

Manual operation: See "Signal Period Mode" on page 361

[:SOURce<hw>]:BB:ARBitrary:MCARrier:POWer:REFerence <Reference>

Defines the way the individual carriers in a composed multi carrier signal are leveled.

Parameters:

<Reference> RMS | PEAK

*RST: RMS

Manual operation: See "Power Reference" on page 361

[:SOURce<hw>]:BB:ARBitrary:MCARrier:OFILe <OFile>

Defines the output file name for the multi-carrier waveform (file extension `*.wv`).

This file name is required to calculate the waveform with the commands `[:SOURce<hw>]:BB:ARBitrary:MCARrier:CLOad` or `[:SOURce<hw>]:BB:ARBitrary:MCARrier:CREate`.

Parameters:

<OFile> string

Example: See Example "Generating a multi carrier signal" on page 926

Manual operation: See "File" on page 364

[:SOURce<hw>]:BB:ARBitrary:MCARrier:CLOad

Creates a multi-carrier waveform using the current entries of the carrier table and activates the ARB generator.

Use the command `[:SOURce<hw>]:BB:ARBitrary:MCARrier:OFILe` to define the multi-carrier waveform file name. The file extension is `*.wv`.

Example: See Example "Generating a multi carrier signal" on page 926

Usage: Event

Manual operation: See "Create/Create and Load" on page 364

[:SOURce<hw>]:BB:ARBitrary:MCARrier:CLOCk?

Queries the resulting sample rate at which the multi-carrier waveform is output by the arbitrary waveform generator. The output clock rate depends on the number of carriers, carrier spacing, and input sample rate of the leftmost or rightmost carriers.

Return values:

<Clock> float

Range: 400 to Max
Increment: 1E-3

Example: `BB:ARB:MCAR:CLOC?`
Queries the ARB multi-carrier output clock rate.

Usage: Query only

Manual operation: See "Clock Rate" on page 364

[:SOURce<hw>]:BB:ARBitrary:MCARrier:CREate

Creates a multi-carrier waveform using the current settings of the carrier table.

Use the command `[:SOURce<hw>]:BB:ARBitrary:MCARrier:OFILe` to define the multi-carrier waveform file name. The file extension is `*.wv`.

Example: See Example "Generating a multi carrier signal" on page 926

Usage: Event

Manual operation: See "Create/Create and Load" on page 364

[:SOURce<hw>]:BB:ARBitrary:MCARrier:CARRier<ch>:STATe <State>

Enables/diasbled the selected carrier.

Parameters:

<State> 0 | 1 | OFF | ON

*RST: 0

Example: See Example "Generating a multi carrier signal" on page 926

Manual operation: See "State" on page 366

[:SOURce<hw>]:BB:ARBitrary:MCARrier:CARRier<ch>:FREQuency <Frequency>

Sets or indicates the carrier frequency, depending on the selected carrier frequency mode.

Parameters:

<Frequency> integer

Range: depends on the installed options, e.g. -60 MHz to +60 MHz (R&S SMW-B10)

*RST: 0

Example:
```
SOURce1:BB:ARBitrary:MCARrier:CARRier:MODE ARB
SOURce1:BB:ARBitrary:MCARrier:CARRier1:
FREQuency 5.0
```
Sets 5.0 MHz carrier frequency.

Manual operation: See "Carrier Freq [MHz]" on page 366

[:SOURce<hw>]:BB:ARBitrary:MCARrier:CARRier<ch>:PHASe <Phase>

Sets the start phase of the selected carrier.

Parameters:

<Phase> float

Range: 0 to 359.99

Increment: 0.01

*RST: 0

Default unit: DEG

Example:
```
SOURce1:BB:ARBitrary:MCARrier:CARRier19:PHASe
90
```
Sets a start phase of 90° for carrier 15.

Manual operation: See "Phase" on page 366

[:SOURce<hw>]:BB:ARBitrary:MCARrier:CARRier<ch>:POWer <Power>

Sets the gain of the selected carrier.

Parameters:

<Power> float

Range: -80 to 0

Increment: 0.01

*RST: 0

Default unit: dB

Example:
```
SOURce1:BB:ARBitrary:MCARrier:CARRier15:POWer
-50
```
Sets the power of carrier 15 to -50 dB.

Manual operation: See "Gain" on page 366

[:SOURce<hw>]:BB:ARBitrary:MCARrier:CARRier<ch>:DELay <Delay>

Sets the start delay of the selected carrier.

Parameters:

<Delay> float

Range: 0 to 1
Increment: 1E-9
*RST: 0
Default unit: s

Example: `BB:ARB:MCAR:CARR15:DEL 5us`
sets a start delay of 50 us for carrier 15.

Manual operation: See "Delay" on page 366

[:SOURce<hw>]:BB:ARBitrary:MCARrier:CARRier<ch>:FILE <File>

Selects the file with I/Q data to be modulated onto the selected carrier.

Parameters:

<File> <file name>

Example: See Example "Generating a multi carrier signal" on page 926

Manual operation: See "File" on page 366

[:SOURce<hw>]:BB:ARBitrary:MCARrier:CARRier<ch>:CONFlict?

Queries carrier conflicts. A conflict arises when the carriers overlap.

Return values:

<Conflict> 0 | 1 | OFF | ON

0
No conflict
*RST: 0

Example: `BB:ARB:MCAR:CARR:CONF?`

Usage: Query only

Manual operation: See "!!!" on page 367

[:SOURce<hw>]:BB:ARBitrary:MCARrier:EDIT:CARRier:STATe <State>

Switches all the carriers in the selected carrier range on or off.

Parameters:

<State> 0 | 1 | OFF | ON

*RST: 1

Example: `BB:ARB:MCAR:EDIT:CARR:STAT ON`

Sets all the carriers in the carrier range to ON.

Manual operation: See "Carrier State" on page 368

[:SOURce<hw>]:BB:ARBitrary:MCARrier:EDIT:CARRier:STARt <Start>
[:SOURce<hw>]:BB:ARBitrary:MCARrier:EDIT:CARRier:STOP <Stop>

Selects the last carrier in the carrier range to which the settings shall apply.

Parameters:

<Stop> integer

Range: 0 to 511
*RST: 0

Example: `BB:ARB:MCAR:EDIT:CARR:STOP 4`

The carrier range stops at carrier 4.

Manual operation: See "Carrier Start/Start" on page 368

[:SOURce<hw>]:BB:ARBitrary:MCARrier:EDIT:CARRier:POWer:STEP <Step>

Sets the step width by which the starting power of the carriers in the defined carrier range is incremented.

Parameters:

<Step> float

Range: -80 to 80
Increment: 0.01
*RST: 0
Default unit: dB

Example: `BB:ARB:MCAR:EDIT:CARR:POW -80dB`

Sets a power of -80 dB for the carriers in the carrier range.

`BB:ARB:MCAR:EDIT:CARR:POW:STEP 1 dB`

The power is incremented by 1dB for each carrier.
That is, the first carrier has -80 dB, the second -79 dB, and so on.

Manual operation: See "Gain Step" on page 368

[:SOURce<hw>]:BB:ARBitrary:MCARrier:EDIT:CARRier:POWer[:STARt] <Start>

Sets the power for the individual carriers in the defined carrier range.

Parameters:

<Start> float

Range: -80 to 0
Increment: 0.01
*RST: 0
Default unit: dB

Example: `BB:ARB:MCAR:EDIT:CARR:POW -50 dB`
sets the power of the carriers in the carrier range to -50 dB.

Manual operation: See "Gain Start" on page 368

[:SOURce<hw>]:BB:ARBitrary:MCARrier:EDIT:CARRier:PHASe:STEP <Step>

Sets the step width by which the start phases of the carriers in the defined carrier range is incremented.

Parameters:

<Step> float

Range: -359.99 to 359.99
Increment: 0.01
*RST: 0
Default unit: DEG

Example: `BB:ARB:MCAR:EDIT:CARR:PHAS 90 DEG`
Sets a start phase of 90° for the carriers in the carrier range.
`BB:ARB:MCAR:EDIT:CARR:PHAS:STEP 1 DEG`
The start phase is incremented by 1° for each carrier. That is, the first carrier has a start phase of 90°, the second a start phase of 91°, and so on.

[:SOURce<hw>]:BB:ARBitrary:MCARrier:EDIT:CARRier:PHASe[:STARt] <Start>

Sets the start phase for the individual carriers in the defined carrier range.

Parameters:

<Start> float

Range: 0 to 359.99
Increment: 0.01
*RST: 0
Default unit: DEG

Example: `BB:ARB:MCAR:EDIT:CARR:PHAS 90 DEG`
sets a start phase of 90° for the carriers in the carrier range.

Manual operation: See "Phase Start" on page 368

[:SOURce<hw>]:BB:ARBitrary:MCARrier:EDIT:CARRier:DELay:STEP <Step>

Sets the step width by which the start delays of the carriers in the defined carrier range is incremented.

Parameters:

<Step> float

Range: -1 to 1
Increment: 1E-9
*RST: 0
Default unit: s

Example: `BB:ARB:MCAR:EDIT:CARR:DEL 5 us`
Sets a start delay of 5 us for the carriers in the carrier range.
`BB:ARB:MCAR:EDIT:CARR:DEL:STEP 1 us`
The start delay is incremented by 1us for each carrier. That is, the first carrier has a start delay of 5 us, the second a start delay of 6 us, and so on.

Manual operation: See "Delay Step" on page 369

[:SOURce<hw>]:BB:ARBitrary:MCARrier:EDIT:CARRier:DELay[:STARt] <Start>

Sets the start delay for the individual carriers in the defined carrier range.

Parameters:

<Start> float

Range: 0 to 1
Increment: 1E-9
*RST: 0
Default unit: s

Example: `BB:ARB:MCAR:EDIT:CARR:DEL 5us`
Sets a start delay of 5 us for the carriers in the carrier range.

Manual operation: See "Delay Start" on page 369

[:SOURce<hw>]:BB:ARBitrary:MCARrier:EDIT:CARRier:FILE <File>

Selects the input file. The data of the file are modulated onto the carriers in the defined carrier range.

Parameters:

<File> string

Example: `BB:ARB:MCAR:EDIT:CARR:FILE "/var/user/temp/IQ_wcdma"`
selects input file `IQ_wcdma`.

Manual operation: See "Input Waveform File" on page 369

[:SOURce<hw>]:BB:ARBitrary:MCARrier:EDIT:CARRier:EXECute

Adopts the settings for the selected carrier range.

Example:
```
BB:ARB:MCAR:EDIT:CARR:STAR 4
BB:ARB:MCAR:EDIT:CARR:STOP 20
BB:ARB:MCAR:EDIT:CARR:STAT ON
BB:ARB:MCAR:EDIT:CARR:EXEC
```

Usage: Event

Manual operation: See "Apply Assistant Settings" on page 369

Trigger Commands

[:SOURce<hw>]:BB:ARBitrary[:TRIGger]:SEQuence <Sequence>

The command selects the trigger mode.

Parameters:

<Sequence> AUTO | RETRigger | AAUTo | ARETrigger | SINGle

*RST: AUTO

Example: See Example "Managing waveform files" on page 920

Manual operation: See "Trigger Mode" on page 255

[:SOURce<hw>]:BB:ARBitrary:TRIGger:SOURce <Source>

Selects the trigger signal source and determines the way the triggering is executed. Provided are:

- Internal triggering by a command (`INTernal`)
- External trigger signal via one of the local or global connectors
  - `EGT1|EGT2`: External global trigger
  - `EGC1|EGC2`: External global clock
  - `ELTRigger`: External local trigger
  - `ELCLock`: External local clock
- Internal triggering by a signal from the other basebands (`INTA|INTB`)
- In master-slave mode, the external baseband synchronization signal (`BBSY`)
- `OBASeband|BEXTernal|EXTernal`: Setting only
  Provided only for backward compatibility with other Rohde & Schwarz signal generators.
  The R&S SMW accepts these values and maps them automatically as follows:
  `EXTernal = EGT1`, `BEXTernal = EGT2`, `OBASeband = INTA or INTB` (depending on the current baseband)

Parameters:

<Source> INTB | INTernal | OBASeband | EGT1 | EGT2 | EGC1 | EGC2 | ELTRigger | INTA | ELCLock | BEXTernal | EXTernal | BBSY

*RST: INTernal

Example: See Example "Managing waveform files" on page 920

Options: ELTRigger|ELCLock require R&S SMW-B10
BBSY require R&S SMW-B9

Manual operation: See "Trigger Source" on page 256

[:SOURce<hw>]:BB:ARBitrary:TRIGger:RMODe?

Queries the status of waveform output.

Return values:

<RMode> STOP | RUN

Example: see Example "Managing waveform files" on page 920

Usage: Query only

Manual operation: See "Running/Stopped" on page 255

[:SOURce<hw>]:BB:ARBitrary:TRIGger:SLENgth <SLength>

Defines the length of the signal sequence that is output in the `SINGle` trigger mode.

Parameters:

<SLength> integer

Maximun value dependents on the selected units `[:SOURce<hw>]:BB:ARBitrary:TRIGger:SLUNit` as follows:

`SAMPle`: Max = 2^{32}-1

`SEQuence`: Max = 1000

Range: 1 to dynamic
*RST: 1

Example: See Example "Managing waveform files" on page 920

[:SOURce<hw>]:BB:ARBitrary:TRIGger:SLUNit <SLUnit>

Defines the unit for the entry of the length of the signal sequence to be output in the `Single` trigger mode.

Parameters:

<SLUnit> SEQuence | SAMPle

*RST: SEQuence

Example: See Example "Managing waveform files" on page 920

Manual operation: See "Trigger Signal Duration" on page 255

[:SOURce<hw>]:BB:ARBitrary:TRIGger:ARM:EXECute

Stops (arms) waveform output.

Example: See Example "Managing waveform files" on page 920

Usage: Event

Manual operation: See "Arm" on page 255

[:SOURce<hw>]:BB:ARBitrary:TRIGger:EXECute

Executes an internal trigger event.

Example: See Example "Managing waveform files" on page 920

Usage: Event

Manual operation: See "Execute Trigger" on page 255

[:SOURce<hw>]:BB:ARBitrary:TRIGger[:EXTernal]:SYNChronize:OUTPut <Output>

Enables signal output synchronous to the trigger event.

Parameters:

<Output> 0 | 1 | OFF | ON

*RST: 1

Example: See Example "Managing waveform files" on page 920

Manual operation: See "Sync. Output to External Trigger/Sync. Output to Trigger" on page 256

[:SOURce<hw>]:BB:ARBitrary:TRIGger:OBASeband:DELay <Delay>

Delays the trigger event compared to the trigger event in the other basebands.

Parameters:

<Delay> float

Range: 0 to 2147483647
Increment: 0.01
*RST: 0

Manual operation: See "(Specified) External Delay/(Specified) Trigger Delay" on page 257

[:SOURce<hw>]:BB:ARBitrary:TRIGger:OBASeband:RDELay?

Queries the time a trigger event form the other path is delayed.

Return values:

<ResTimeDelaySec> float

Range: 0 to 688
Increment: 0.25E-9
*RST: 0
Default unit: s

Example: See Example "Specifying delay and inhibit values in time units" on page 922

Usage: Query only

Manual operation: See "Actual Trigger Delay/Actual External Delay" on page 258

[:SOURce<hw>]:BB:ARBitrary:TRIGger:OBASeband:TDELay <ObasTimeDelay>

Specifies the trigger delay for triggering by the signal from the other path.

Parameters:

<ObasTimeDelay> float

Range: 0 to 2147.483647
Increment: 0.25E-9
*RST: 0
Default unit: s

Example: See Example "Specifying delay and inhibit values in time units" on page 922

Manual operation: See "(Specified) External Delay/(Specified) Trigger Delay" on page 257

[:SOURce<hw>]:BB:ARBitrary:TRIGger:OBASeband:INHibit <Inhibit>

For triggering via the other path, specifies the number of samples by which a restart is inhibited.

Parameters:

<Inhibit> integer

Range: 0 to 67108863
*RST: 0
Default unit: sample

Example: See Example "Managing waveform files" on page 920

Manual operation: See "External / Trigger Inhibit" on page 257

[:SOURce<hw>]:BB:ARBitrary:TRIGger:DELay:UNIT <TrigDelUnit>

Sets the units the trigger delay is expressed in.

Parameters:

<TrigDelUnit> SAMPle | TIME

*RST: SAMPle

Example: See Example "Specifying delay and inhibit values in time units" on page 922

Manual operation: See "(External) Delay Unit" on page 257

[:SOURce<hw>]:BB:ARBitrary:TRIGger[:EXTernal]:DELay <Delay>

Specifies the trigger delay.

Parameters:

<Delay> float

Range: 0 to 2147483647
Increment: 0.01
*RST: 0

Example: see Example "Managing waveform files" on page 920

Manual operation: See "(Specified) External Delay/(Specified) Trigger Delay" on page 257

[:SOURce<hw>]:BB:ARBitrary:TRIGger[:EXTernal]:TDELay <ExtTimeDelay>

Specifies the trigger delay for external triggering. The value affects all external trigger signals.

Parameters:

<ExtTimeDelay> float

Range: 0 to 2147483647/clock rate
Increment: 1E-9
*RST: 0
Default unit: s

Example: See Example "Specifying delay and inhibit values in time units" on page 922

Manual operation: See "(Specified) External Delay/(Specified) Trigger Delay" on page 257

[:SOURce<hw>]:BB:ARBitrary:TRIGger[:EXTernal]:RDELay?

Queries the time (in seconds) an external trigger event is delayed for.

Return values:

<ResTimeDelaySec> float

Range: 0 to 688
Increment: 0.25E-9
*RST: 0

Example: See Example "Specifying delay and inhibit values in time units" on page 922

Usage: Query only

Manual operation: See "Actual Trigger Delay/Actual External Delay" on page 258

[:SOURce<hw>]:BB:ARBitrary:TRIGger[:EXTernal]:INHibit <Inhibit>

Specifies the number of symbols by which a restart is inhibited.

Parameters:

<Inhibit> integer

Range: 0 to 21.47*clockRate
*RST: 0

Example: see Example "Managing waveform files" on page 920

Manual operation: See "External / Trigger Inhibit" on page 257

Marker Commands

[:SOURce<hw>]:BB:ARBitrary:TRIGger:OUTPut<ch>:MODE <Mode>

Defines the signal for the selected marker output.

For detailed description of the regular marker modes, refer to "Marker Modes" on page 235.

Parameters:

<Mode> UNCHanged | RESTart | PULSe | PATTern | RATio

UNCHanged
A marker signal as defined in the waveform file (tag `'marker mode x'`) is generated.

*RST: UNCHanged

Example: see Example "Managing waveform files" on page 920

Manual operation: See "Marker Mode" on page 315

[:SOURce<hw>]:BB:ARBitrary:TRIGger:OUTPut<ch>:ONTime <OnTime>
[:SOURce<hw>]:BB:ARBitrary:TRIGger:OUTPut<ch>:OFFTime <OffTime>

Sets the number of samples in the ON and OFF periods.

*) If R&S SMW-B9 is installed, the minimum marker duration depends on the sample/symbol rate.

See "Marker Minimum Duration" on page 236.

Parameters:

<OffTime> integer

Range: 1 (R&S SMW-B10) / 16* (R&S SMW-B9) to 14913079

*RST: 1

Example: see Example "Managing waveform files" on page 920

Manual operation: See "Marker Mode" on page 315

[:SOURce<hw>]:BB:ARBitrary:TRIGger:OUTPut<ch>:PATTern <Pattern>

Defines the bit pattern used to generate the marker signal.

Parameters:

<Pattern> <64 bits pattern>

0 = marker off, 1 = marker on

*RST: #H2,2

Example: See Example "Managing waveform files" on page 920

Manual operation: See "Marker Mode" on page 315

[:SOURce<hw>]:BB:ARBitrary:TRIGger:OUTPut<ch>:PULSe:DIVider <Divider>

Sets the divider for the pulsed marker signal.

*) If R&S SMW-B9 is installed, the minimum marker duration depends on the sample/symbol rate.

See "Marker Minimum Duration" on page 236.

Parameters:

<Divider> integer

Range: 2 (R&S SMW-B10) / 32* (R&S SMW-B9) to 1024
*RST: 2

Example: See Example "Managing waveform files" on page 920

Manual operation: See "Marker Mode" on page 315

[:SOURce<hw>]:BB:ARBitrary:TRIGger:OUTPut<ch>:PULSe:FREQuency?

Queries the pulse frequency of the pulsed marker signal. The pulse frequency is derived by dividing the symbol rate by the divider.

Return values:

<Frequency> float

Increment: 0.001

Example: see Example "Managing waveform files" on page 920

Usage: Query only

Manual operation: See "Marker Mode" on page 315

[:SOURce<hw>]:BB:ARBitrary:TRIGger:OUTPut<ch>:DELay <Delay>

Defines the delay between the signal on the marker outputs and the start of the signals.

Parameters:

<Delay> float

Range: 0 to depends on other values
Increment: 0.001
*RST: 0
Default unit: Symbol

Example: see Example "Managing waveform files" on page 920

Manual operation: See "Marker x Delay" on page 260

Clock Commands

[:SOURce<hw>]:BB:ARBitrary:CLOCk:SOURce <Source>

Selects the clock source:

- INTernal: Internal clock reference
- ELCLock: External local clock
- EXTernal = ELCLock: Setting only
  Provided for backward compatibility with other Rohde & Schwarz signal generators

Parameters:

<Source>	INTernal | ELCLock | EXTernal
	*RST: INTernal
Example:	see Example "Managing waveform files" on page 920
Options:	ELCLock requires R&S SMW-B10
Manual operation:	See "Clock Source" on page 260

[:SOURce<hw>]:BB:ARBitrary:CLOCk:MODE <Mode>

Enters the type of externally supplied clock.

Parameters:

<Mode>	SAMPle
	*RST: SAMPle
Example:	See Example "Managing waveform files" on page 920
Options:	R&S SMW-B10
Manual operation:	See "Clock Mode" on page 261

13.18.4.4 Waveform, Data and List Format

The R&S SMW uses a simple tag-oriented format for externally or internally generated waveforms, data and control lists. Files with this format can be transmitted via the GPIB interface from an external computer to the instrument and vice versa. These files are transmitted as a binary data block, using SCPI command [:SOURce<hw>]:BB:ARBitrary:WAVeform:DATA, [:SOURce<hw>]:BB:DM:DLISt:DATA, and [:SOURce<hw>]:BB:DM:CLISt:DATA.

Tag general format

Tags are self-contained information units, enclosed in braces { }. Their general format is {Name: Data} or {Name-Length: Data}. The colon separates the name part and the data part. The colon can be followed by a space for the sake of legibility.

- The **Name** identifies the tag. It is always expressed in capital letters.
- The **Data** is tag-specific, and usually it is in ASCII plain text.

- The **Length** specifies the number of bytes in a `WAVEFORM` tag, `DATA LIST` tag, or `EMPTYTAG`
  Length is an ASCII integer value, defining the number of bytes from the colon **:** to the end brace `}`

Rules

Each waveform file must begin with the `TYPE` tag. The sequence of the remaining tags is arbitrary. For each tag, an indication shows whether it must be included in the file concerned (mandatory) or may be included (optional).

Unknown tags are not analyzed by the R&S SMW; they are left unchanged and saved without an error message for a possible further read back.

R&S SMU waveforms can also be loaded on the instrument, where they are converted internally into an R&S SMW waveform.

In all examples of file contents listed in this section, the tags have been separated by line breaks for better reading.

Tag Description

This section describes the mandatory `TYPE` tag followed by description of all other tags in an alphabetical order. Most tags are valid for all three file types. If a tag is valid only for a single file type, e.g. only for a waveform, this fact is indicated in the description.

{TYPE: magic, xxxxxxxx}

(mandatory, must be the first tag in the file)

Identifies the file as a valid R&S SMW file. It must be present and must be the first in the waveform. If a file of the same name exists on the target medium, it is overwritten.

Setting parameters:

magic
: Designates the file type and has the following values:

  SMU-WV
  A valid R&S SMW waveform.

  SMU-MWV
  A valid R&S SMW multi-segment waveform.

  SMU-DL
  A valid R&S SMW data list.

  SMU-CL
  A valid R&S SMW control list.

xxxxxxxx
: Is an ASCII-coded checksum of the data part of the `WAVEFORM` tag in the file. This value is always 0 for data lists and control lists.
  The checksum for waveforms is used for detecting transmission errors. If the `TYPE` tag contains 0 or a non-numeric value for the checksum, it is ignored.
  It is calculated in accordance with the algorithm given bellow, where 'start' is a pointer to the first byte after the '#' character in the `WAVEFORM` tag and 'length' is the number of bytes between 'start' and the closing curly bracket (excluding the latter; 'length' must be divisible by 4 without a remainder):

```
UINT32 checksum(void *start, UINT32 length)
{
UINT32 i, result = 0xA50F74FF;
for(i=0; i < length/4; i++)
result = result ^ ((UINT32 *)start)[i];
return(result);
}
```

Example:

```
{TYPE: SMU-WV,106656}
BB:ARB:WAV:TAG? 'TYPE'
```

Queries the content of the `TYPE` tag.

```
Response: 'SMU-WV,106656'
```

This is a valid waveform.

{CLOCK: frequency}

(mandatory for waveforms)

The tag specifies the clock frequency at which the waveform has to be output, in Hz (on multi-segment waveforms this tag contains the maximal clock of all segments).

A query of `ARB:CLOCk?` after loading the waveform returns the value set using the `CLOCK` tag. This value can later be altered with the command `ARB:CLOCk?`.

Example: `{CLOCK: 54000000}`
`BB:ARB:WAV:TAG? 'CLOCK'`
Queries the content of the `CLOCK` tag.
`Response: 54000000`
The clock frequency is set to 54 MHz.

Usage: Setting only

{COMMENT: string}

The tag contains a plain text ASCII string of arbitrary length. The string is not analyzed in the R&S SMW. It is used to describe the file. The string is allowed to contain all printable ASCII characters except the closing curly bracket.

Example: `{COMMENT: File with data for 3GPP enhanced channels}`
`BB:ARB:WAV:TAG? 'COMMENT'`
Queries the content of the `COMMENT` tag of the selected waveform file.
`Response: 'File with data for 3GPP enhanced channels'`
The comment on the waveform reads "File with data for 3GPP enhanced channels".

Usage: Setting only

{COPYRIGHT: string}

The tag contains an ASCII string of arbitrary length. The string is not analyzed in the R&S SMW. It is used to store copyright information about the file content.

Example: `{COPYRIGHT: Rohde&Schwarz}`
`BB:ARB:WAV:TAG? 'COPYRIGHT'`
Queries the content of the `COPYRIGHT` tag of the selected waveform file.
`Response: 'Rohde&Schwarz'`
Copyright resides with Rohde & Schwarz.

Usage: Setting only

{DATA BITLENGTH: BitLength}

(mandatory for data lists)

The tag defines the length of the data field in the DATA LIST tag in bits in ASCII format.

Example:
`{DATA BITLENGTH: 444}`
`BB:DM:DLIS:SEL "/var/user/dl"`
`BB:DM:DLIS:TAG? "dl","DATA BITLENGTH"`
Queries the content of the `DATA BITLENGTH` tag of the selected data list file.
`Response: '444'`
The data list is 444 bits long.

Usage: Setting only

{DATA LIST-Length: #d0d1...dx...dN-1…}

(mandatory for data lists)

The tag contains the actual bit sequence of the data list in binary format.

Setting parameters:

Length
Defines the number of bytes in the `DATA LIST` tag in ASCII Format (see `{WAVEFORM-Length: #I0Q0I1Q1...IxQx...IN-1QN-1…}` for details).

dx
Data bits in binary format (8-bit unsigned characters, MSB first).

Example:
`{DATA LIST-17: #d0d1...dx...d127}`
16 bytes containing 128 data bits, first bit is the MS bit of the first byte.

Usage: Setting only

{DATE: yyyy-mm-dd;hh:mm:ss}

(optional)

The tag contains the date and time at which the file was created. The year must be expressed as four digits. The instrument does not analyze this tag.

Example:
`{DATE: 2009-04-02;14:32:12}`
`BB:ARB:WAV:TAG? 'DATE'`
Queries the content of the `DATE` tag of the selected waveform file.
`Response: '2009-04-02;14:32:12'`
The waveform was created on April 2, 2009 at 14 hrs 32 min

Usage: Setting only

{EMPTYTAG-Length: #EmptySequence}

(mandatory in automatically generated one and multi-segment waveforms)

This tag is empty, i.e. contains no data, and is used as placeholder.

Setting parameters:

Length
: An ASCII integer value that specifies the number of bytes in the `EMPTYTAG`, i.e. defines the number of bytes from the colon `:` to the end brace `}`
Note: If you change the content of a waveform file, change also the `{EMPTYTAG-Length}` value. For example, if you add a tag or add bytes to a tag, reduce the length by the number of newly introduced bytes.

EmptySequence
: An empty sequence containing blanks only. The number of used blanks is calculated as the difference between the hex addresses of the `{WAVEFORM}` tag and the hash sign `#` in the `{EMPTYTAG}`.
The `{WAVEFORM}` tag always starts at hex address #4000.

Example:

```
{TYPE:SMU-WV, 837236424}
{COPYRIGHT:2003 Rohde&Schwarz SMU}
{DATE:2012-07-11;14:38:01}
{SAMPLES:80000}
{CLOCK:8666666.666666666}
{VECTOR MAX:1.000000038569158}
{LEVEL OFFS:3.333553817875577e-07,0}
{MARKER LIST 1:0:1;1:0;1249:0}
{MARKER LIST 2:0:1;1:0;1249:0}
{MARKER LIST 3:0:1;1:0;1249:0}
{MARKER LIST 4:0:1;1:0;1249:0}
{EMPTYTAG-15947:# ...}
{WAVEFORM-320017:#IQIQIQ...}
```

The example waveform file contains 436 (0x1b4) bytes before the `#` sign in the `EMPTYTAG`; the hex address of the `#` sign is *0x1b5*. The `{WAVEFORM}` starts at *0x4000*. The `EMPTYTAG` contains *15946* blanks and has a length of (15946+1) bytes.

Usage: Setting only

{CONTROL LENGTH: ControlLength}

(optional / recommended for marker and control lists)

The tag specifies the length of *all* control or marker list in ASCII format.

The control length influences the way the marker and control lists are processed, in particular the way traces are repeated; see Figure 13-4.

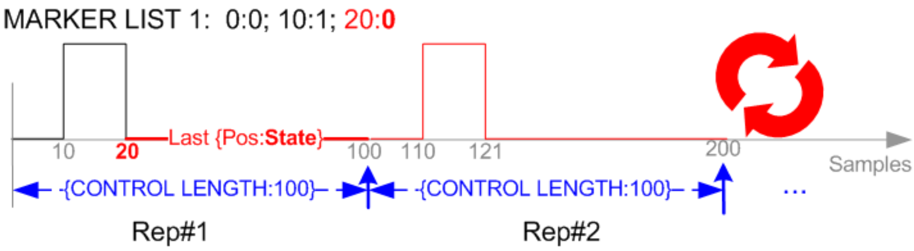


Figure 13-4: Example: Processing of MARKER TRACE if CONTROL LENGHT is specified

If the `CONTROL LENGTH` tag is not used, the marker and control list length are determined by the last position, that is the last {**Pos**:State} couple, defined in the particular `[TRACE] LIST` tag; see Figure 13-5.

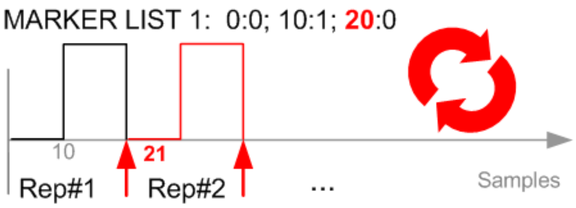


Figure 13-5: Example: Processing of MARKER TRACE if CONTROL LENGHT is not used

To maintain marker signals and waveform data synchronized, set the `CONTROL LENGTH` to be equal to the number of samples specified with the tag SAMPLES.

See also "How to Define Periodically Repeating Traces" on page 974.

Example: `{CONTROL LENGTH: 500}`
`SOURce:BB:ARBitary:CLISt:TAG? 'CONTROL LENGTH'`
Queries the length of the control list.
`Response: 500`

Manual operation: See "Total List Length" on page 284

{LEVEL OFFS: RMSOffset_dB,PeakOffset_dB}

(recommended for waveforms)

The tag determines the level of the ARB signal in the waveform file. The offset levels define the offset of RMS and peak value relative to the 16-bit full scale modulation (-32767 to + 32767) = 0 dB.

Setting parameters:

RMSOffset_dB
: Defines the RMS level offset of the signal relative to full scale ARB signal in the `WAVEFORM` tag. The offset is defined in ASCII float format. The value is always positive.
A 3 dB value indicates that the RMS level of the signal is 3 dBs below the full scale.
full scale = max. amplitude of vector of I/Q samples = $|S_{IQ}|$max = sqrt(I^2+Q^2)max = 0 dB

PeakOffset_dB — Defines the peak level offset of the signal relative to full scale for the ARB signal in the `WAVEFORM` tag. The offset is defined in ASCII float format.
The value usually equals 0 dB as usually the I/Q samples (signed 16-bit integer values) are modulated to full scale: Full scale = 0 dB = max. amplitude of vector of I/Q samples = $|S_{IQ}|$ max = $\mathrm{sqrt}(I^2+Q^2)$max = $(2^{15})-1$ = 32767.
A positive `PeakOffset_dB` value indicates that a headroom to full scale is provided when generating the waveform. A negative PeakOffset_dB value indicates that overrange is likely for some samples, i.e. clipping might occur.
The crest factor can be calculated from the two values as follows:
`Crest Factor = |PeakOffset_dB - RMSOffset_dB|`

Example:
`{LEVEL OFFS: 3.45,2}`
`BB:ARB:WAV:TAG? 'LEVEL OFFS'`
Queries the content of the `LEVEL OFFS` tag of the selected waveform file.
`Response: 3.45,2`
The level of the waveform is below full scale, clipping does not occur.

Usage: Setting only

{SAMPLES: Samples}

(recommended for waveforms)

The tag contains the number of I/Q samples in the waveform in ASCII format.

On multi-segment waveforms, this tag contains the total I/Q samples of all segments.

Example:
`{SAMPLES: 1000}`
`BB:ARB:WAV:TAG? 'SAMPLES'`
Queries the content of the `SAMPLES` tag of the selected waveform file.
`Response: 1000`
The waveform contains 1000 I/Q samples.

Usage: Setting only

See also "How to Define Periodically Repeating Traces" on page 974.

{[TRACE] LIST [#]: Pos0:State0; Pos1:State1; ...PosN-1:StateN-1}

(mandatory for control lists / optional for waveforms)

The tag contains the data for the marker and control signals in the control list or the marker signals of ARB waveforms.

Traces are processed different, depending on the selected CONTROL LENGTH.

See also "How to Define Periodically Repeating Traces" on page 974.

Setting parameters:

[TRACE] MARKER | BURST | LEVATT | CW MODE | HOP | MAP
Name of the marker or control signal.
For ARB waveforms, it is only meaningful to define marker signals; in the ARB multi-segment waveforms these tags are ignored!

[#] 1 .. 3
Sets the marker or control trace number; supported is only LEVATT LIST 1.

Pos Specifies in ASCII format the position (i.e. sample number or data value), with effect from which the binary `State` of the marker or of the control signal changes.

State 0 | 1
Specifies the binary state of the marker or of the control signal from `Pos`$_x$ to `Pos`$_{x+1}$ exclusive in ASCII format.

Example: `{MARKER LIST 1: 0:0;10:1;20:0;30:1}`
`BB:DM:CLIS:TAG? 'MARKER LIST 1'`
Queries the content of the `MARKER LIST 1` tag of the selected control list file.
`Response: '0:0;10:1;20:0;30:1'`
The marker setting for samples 0 to 9 = 0 (low), for 10 to 19 = 1 (high) and for 20 to 29 = 0. From sample 30 onward the marker setting = 1.

Example: `{LEVATT LIST 1: 0:0;10:1;20:0;30:1}`
`BB:DM:CLIS:TAG? 'LEVATT LIST 1'`
Queries the content of the `LEVATT LIST 1` tag of the selected control list file.
`Response: '0:0;10:1;20:0;30:1'`
Level attenuation applies to data values 10 to 19 (high) and from data value 30 onward.

Usage: Setting only

Manual operation: See "Select Ramp to Edit" on page 283

{WAVEFORM-Length: #I0Q0I1Q1...IxQx...IN-1QN-1…}

(mandatory for waveforms)

The tag contains the actual waveform data or multi-segment waveform data (I/Q stream). Refer to Chapter 4.8, "Generating Multi-Segment Waveform Files", on page 324 for background information description of the multi-segment waveform function.

Setting parameters:

Length
: Specifies the number of bytes in a `WAVEFORM` tag and is calculated as follows:
`Length` = `Number of I/Q pairs * 4` (2 bytes per I and 2 bytes per Q value) + `1 byte` (the length of the `#`)

IxQx
: `IxQx`… represents binary data (16-bit signed integer in 2's complement notation) containing the I and Q component alternately and starting with the I component. Each component consists of 2 bytes in Little endian format representation, i.e least significant byte (LSB) first.
The values of the 2 bytes in an I component and a Q component are in the range 0x0 to 0xFFFF (-32767 to +32767). This value is transferred to the D/A converter.
This tag is also used to store multi-segment waveforms. The I/Q streams of the individual waveforms are directly concatenated to one collectively waveform I/Q stream.
The number of segments and the start offset and length of the individual segments inside the total waveform I/Q stream is determined by the additional tags MWV_SEGMENT_COUNT, MWV_SEGMENT_START, and MWV_SEGMENT_LENGTH.
Further `MWV_SEGMENT_...` tags are also available, for example for level and clock information.

Example: **One segment waveform**
{WAVEFORM-401:#I_0,Q_0,I_1,Q_1,I_2,Q_2,...I_{99},Q_{99}}
100 I/Q pairs with 4 bytes each are transmitted - none multi-segment

Example: **Multi-segment waveform**
{WAVEFORM-1201:
#$I_{0,Seg0}$,$Q_{0,Seg0}$,$I_{1,Seg0}$,$Q_{1,Seg0}$,...$I_{99,Seg0}$,$Q_{99,Seg0}$,$I_{0,Seg1}$,$Q_{0,Seg1}$,$I_{1,Seg1}$,
$Q_{1,Seg1}$,... $I_{199,Seg1}$,$Q_{199,Seg1}$}
2 segments: segment 0 with 100 I/Q pairs; segment 1 with 200 I/Q pairs. Each I/Q pair consists of 2*16 bit = 4 bytes
The data is transmitted using SCPI command `[:SOURce<hw>]:BB:ARBitrary:WAVeform:DATA`.

Usage: Setting only

{MWV_SEGMENT_COUNT: NumOfSeg}

(mandatory for multi-segment waveforms)

The tag contains the number of segments in the multi-segment waveform in ASCII integer format.

Example: `{MWV_SEGMENT_COUNT: 2}`
Multi-segment waveform with 2 segments

Usage: Setting only

{MWV_SEGMENT_LENGTH: SamplesSeg0, SamplesSeg1, …, SamplesSegN-1}

(mandatory for multi-segment waveforms)

The tag contains a list of I/Q sample lengths for every segment in the multi-segment waveform in ASCII integer format.

Example: `{MWV_SEGMENT_LENGTH: 100,200}`
2 segments: 100 samples in segment 0 and 200 samples in segment 1.

Usage: Setting only

{MWV_SEGMENT_START: SampleStartOffsetSeg0, SampleStartOffsetSeg1, …, SampleStartOffsetSegN-1}

(mandatory for multi-segment waveforms)

The tag contains a list of I/Q sample start offsets for every segment in the multi-segment waveform in ASCII integer format.

Example: `{MWV_SEGMENT_START: 0,100}`
2 segments with 100 samples in segment 0 and 200 samples in segment 1.
The start offset of first segment is 0 samples, start offset of next segment 1 is the sample length of segment 0 = 100 samples.

Usage: Setting only

{MWV_SEGMENT_CLOCK_MODE: Mode}

(mandatory for multi-segment waveforms)

The tag contains a string in ASCII format which supplies the clock rate mode, that was used for calculation of the multi-segment output waveform (see also "Clock" on page 336).

The tag `CLOCK` contains always the highest clock rate of all segments. The tag `MWV_SEGMENT_CLOCK` contains the clock rates of the individual segments.

Setting parameters:

Mode

UNCHANGED
The segments may have different clock rates; each segment is output with the clock rate defined in its waveform file.

HIGHEST
All segments are output at the highest available clock rate.

USER
All segments are output at the clock rate defined by the user.
Note: Only upsampling is allowed, no downsampling!

Example: `{MWV_SEGMENT_CLOCK_MODE: UNCHANGED}`

Usage: Setting only

{MWV_SEGMENT_CLOCK: ClockSeg0, ClockSeg1, …, ClockSegN-1}

(mandatory for multi-segment waveforms)

The tag contains a list of clock frequencies for every segment in the multi-segment waveform in ASCII floating point format.

Example: `{MWV_SEGMENT_CLOCK: 100e6,80e6}`
2 segments: clock of segment 0 is 100 MHz, clock of segment 1 is 80 MHz.
Note: If the segments have different clock frequencies, there are some restrictions on signal output, i.e. seamless switching between segments is only possible, if all segments have the same clock frequency. Software resampling (upsampling) can be used to bring all segments to the same clock.

Usage: Setting only

**{MWV_SEGMENT_LEVEL_OFFS:
RMSOffs_dBSg0,PeakOffs_dBSg0, …, RMSOffs_dBSgN-1, PeakOffs_dBSgN-1}**

(mandatory for multi-segment waveforms)

The tag contains a list of level pairs in ASCII floating point format, one pair for every segment in the multi-segment waveform. The first value of a level pair defines the rms offset and the second value the peak offset relative to the 16-bit full scale modulation (-32767; + 32767) = 0 dB. The meaning of one level value pair is the same as in the LEVEL OFFS tag for normal waveforms.

Example: `{MWV_SEGMENT_LEVEL_OFFS: 3.0,0.0,6.0,0.0}`
2 segments: RMS level of segment 0 is 3 dB below full scale;
RMS level of segment 1 is 6dB below full scale.
Peak level of both segments is 0 dB full scale.

Usage: Setting only

**{MWV_SEGMENT_FILES:
“FileNameSeg0.wv”, “FileNameSeg1.wv”, …, “FileNameSegN-1.wv”}**

(optional for multi-segment waveforms)

The tag contains a list of file names for every segment in the multi-segment waveform in ASCII format.

Example: `{MWV_SEGMENT_FILES: "d:\waveforms\sine.wv", "d:\waveforms\rect.wv"”}`

Usage: Setting only

{MWV_SEGMENTx_COMMENT: text}

(optional for multi-segment waveforms)

The tag contains a user comment for a specific segment `x = [0…NumOfSeg-1]` in the multi-segment waveform in ASCII format.

Example: `{MWV_SEGMENT1_FILES: segment 1 contains a QPSK signal.}`

Usage: Setting only

{CONTROL LIST WIDTH4–Length: #m0m1…mx…mM-1}

(optional for waveforms and multi-segment waveforms)

The tag contains a binary marker element stream, which is output synchronously to the I/Q sample sequence. One marker element `mx` consists of 4 bit, which are assigned to the 4 possible marker lines of the instrument (1 bit per marker line). One 4-bit marker element is required for every I/Q sample in the `WAVEFORM` tag. Hence, the number of marker elements `m` should be equal to the number of I/Q samples. The `CONTROL LENGTH` tag has to contain the number of all marker elements `m`.

MSB 7	Byte						LSB 1
Marker element m_x (synchronous to I/Q Sample x)				Marker element m_{x+1} (synchronous to I/Q Sample x+1)			
Marker 4	Marker 3	Marker 2	Marker 1	Marker 4	Marker 3	Marker 2	Marker 1

Figure 13-6: Marker element in 4-bit binary format bit order

For standard waveforms, the `MARKER LIST x` tags are a more compact way to define markers, but in principle this `CONTROL LIST WIDTH4` format can also be used instead of the `MARKER LIST x` tags.

For multi-segment waveforms, this `CONTROL LIST WIDTH4` format is required for marker definition. The binary marker streams of the individual segments are directly concatenated (without any gap) to one collectively marker stream.

Setting parameters:

Length — Defines the number of bytes in the `CONTROL LIST WIDTH4` tag in ASCII Format and is calculated as follows:
Length = Size of "#" (1 byte) + Number of marker elements m_x ** (4 bit) / (8 bits/byte)*
The value is rounded up for byte alignment.

mx — Marker element in 4-bit binary format.

Example: {CONTROL LIST WIDTH4-51: #$m_0m_1…m_x…m_{99}$}
100 marker elements, each marker element with 4 bits

Usage: Setting only

How to Define Periodically Repeating Traces

If a marker trace is required that marks for example each frame start, it is sufficient to define the trace ones and repeat it over the length of a waveform. This is useful if you describe a long waveform and a periodical marker is required.

The following examples use marker traces; control lists are processed in the same way.

To define periodical marker trace

The waveform in the example on Figure 13-7 consists of 3 frames, each frame is 100-samples long. The waveform is processed continuously ("Trigger Mode > Auto").

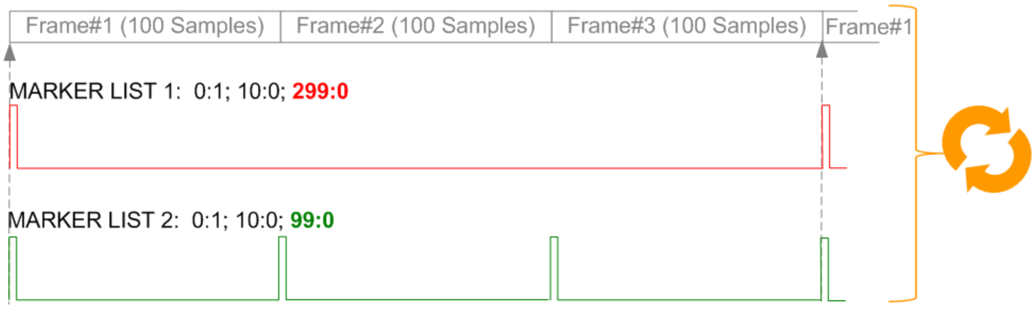


Figure 13-7: Example: Defining marker traces (CONTROL LENGTH tag is not used)

To define a restart marker and a frame start marker, use the following tags:

1. The waveform is 300 samples long, i.e. set {SAMPLES: 300}.
2. Set two [TRACE] LIST tags:
   - For Marker 1 that acts as a restart marker:
     `{MARKER LIST 1: 0:1; 10:0; 299:0}`
   - For Marker 2 that marks each frame start:
     `{MARKER LIST 2: 0:1; 10:0; 99:0}`
3. Do not use the CONTROL LENGTH tag.

   The length of the repeated patterns is determined by the last sample number in the `[TRACE] LIST`, that is the last {**Pos:**State} .

Example: How the CONTROL LENGTH tag influences the processing of the traces

For the example on Figure 13-7, use the same marker traces and set the `CONTROL LENGTH` tag, e.g. `{CONTROL LENGTH: 150}`.

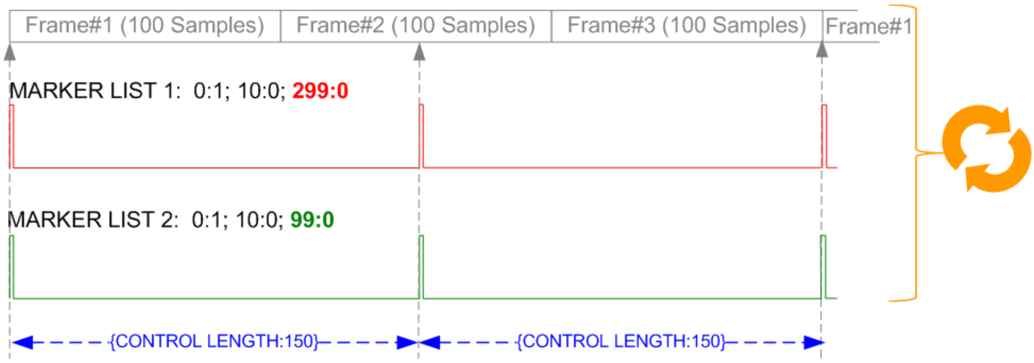


Figure 13-8: Example: Processing of control signals if the CONTROL LENGTH tag is used

The length of **all** control signals is determined by the `CONTROL LENGTH`. Observe how the marker traces are processed. In this example, both marker traces are repeated each 150 samples.

How to Manually Create a Waveform Using Tag File Format

As described in Chapter 4.7.2.1, "Waveform Files Sources", on page 301, you can generate waveform files internally, with the built-in function, and externally. For description on how to generate waveform files internally, see Chapter 4.7.4.2, "How to Create Waveform Files with the Built-In "Generate Waveform File" Function", on page 317. This section provides an example on how to create a waveform externally. The waveform file is created manually; the tag-oriented file format is used.

The provided example uses a sine function in the I channel and a cosine function in the Q channel, each with 20 points. The example uses a short program written in the programming language C to calculate the sine and cosine values (see Example "C-program for creating a waveform file" on page 978). They are stored in the file `SICO.txt`. The decimal values in `SICO.txt` are normalized such that they are between –1.0 and +1.0. The data is converted into binary format. The appropriate mandatory tags are added and the data is packed into the `WAVEFORM` tag. As result, the waveform file `SICO.wv` is generated.

This example follows the general principle of creating of a waveform manually, using the tag file format. The Figure 13-9 illustrates this general workflow.

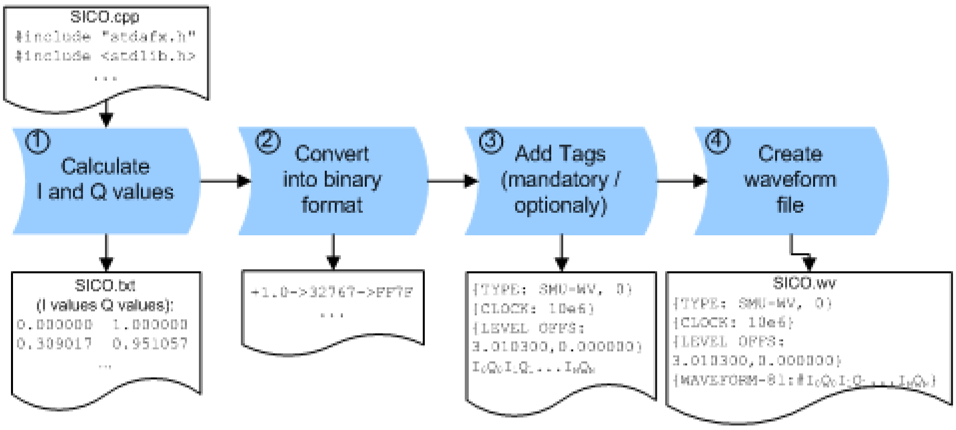


Figure 13-9: Principle of creating a waveform manually

The following steps outline how to create the waveform file `SICO.wv`:

1. Calculate the sine and cosine values, e.g. use the `SICO.cpp` program.

   The result is stored in the file `SICO.txt`.

```
 0.000000  1.000000
 0.309017  0.951057
 0.587785  0.809017
 0.809017  0.587785
 0.951057  0.309017
 1.000000 -0.000000
 0.951057 -0.309017
 0.809017 -0.587785
 0.587785 -0.809017
 0.309017 -0.951057
-0.000000 -1.000000
-0.309017 -0.951056
-0.587785 -0.809017
-0.809017 -0.587785
-0.951056 -0.309017
-1.000000  0.000000
-0.951056  0.309017
-0.809017  0.587785
-0.587785  0.809017
-0.309017  0.951057
```

   Figure 13-10: Contents of SICO.txt: first column Sine (I), second column Cosine (Q)

2. Convert the values from the file `SICO.txt` into binary format consisting of 16-bit signed integer numbers. The numeric range between –1.0 and +1.0 corresponds to the modulation range of the waveform 16-bit D/A converter of -32767 to +32767.

+1.0 ->	32767 ->	= 0x7FFF
0.0 ->	0 ->	= 0x0000
-1.0 ->	-32767 ->	= 0x8001

The Figure 13-11 shows the calculation and conversion steps. The highlighted columns contain the resulting I and Q values represented in Little endian format.

Sample n	deg = 360°/20 * n	I = sin(deg)	$I_{quant,dec}$ = I * FS = I * (2^{15}-1)	$I_{quant,hex}$	$I_{quant,hex}$ (Little endian waveform file representation)	Q = cos(deg)	$Q_{quant,dec}$ = I * FS = I * (2^{15}-1)	$Q_{quant,hex}$	$Q_{quant,hex}$ (Little endian waveform file representation)
0	0	0.000000	0	0000	0000 I_0	1.000000	32767	7FFF	FF7F Q_0
1	18	0.309017	10126	278E	8E27	0.951057	31163	79BB	BB79
2	36	0.587785	19260	4B3C	3C4B	0.809017	26509	678D	8D67
3	54	0.809017	26509	678D	8D67	0.587785	19260	4B3C	3C4B
4	72	0.951057	31163	79BB	BB79	0.309017	10126	278E	8E27
5	90	1.000000	32767	7FFF	FF7F	0.000000	0	0000	0000
6	108	0.951057	31163	79BB	BB79	-0.309017	-10126	D872	72D8
7	126	0.809017	26509	678D	8D67	-0.587785	-19260	B4C4	C4B4
8	144	0.587785	19260	4B3C	3C4B	-0.809017	-26509	9873	7398
9	162	0.309017	10126	278E	8E27	-0.951057	-31163	8645	4586
10	180	0.000000	0	0000	0000	-1.000000	-32767	8001	0180
11	198	-0.309017	-10126	D872	72D8	-0.951057	-31163	8645	4586
12	216	-0.587785	-19260	B4C4	C4B4	-0.809017	-26509	9873	7398
13	234	-0.809017	-26509	9873	7398	-0.587785	-19260	B4C4	C4B4
14	252	-0.951057	-31163	8645	4586	-0.309017	-10126	D872	72D8
15	270	-1.000000	-32767	8001	0180	0.000000	0	0000	0000
16	288	-0.951057	-31163	8645	4586	0.309017	10126	278E	8E27
17	306	-0.809017	-26509	9873	7398	0.587785	19260	4B3C	3C4B
18	324	-0.587785	-19260	B4C4	C4B4	0.809017	26509	678D	8D67
19	342	-0.309017	-10126	D872	72D8	0.951057	31163	79BB	BB79

Figure 13-11: I and Q values calculation and conversion

3. Use an ASCII editor which is able to handle binary data. Create and add the following mandatory tags before this binary data set can be further processed:
   - TYPE
   - CLOCK
   - LEVEL OFFS

   An example of the `SICO.wv` file contents could be:

   ```
   {TYPE: SMU-WV, 0}{CLOCK: 10e6}{LEVEL OFFS: 3.010300,0.000000}
   0000FF7F8E27BB79 ... 72D8BB79
   ```

   To simplify the example, the checksum is set to 0, i.e. the instrument does not evaluate a checksum.

   Tip: The tags `TYPE`, `CLOCK`, `LEVEL OFFS` and `WAVEFORM` are mandatory for each waveform. All other tags are optional and can be inserted after the `TYPE` tag in arbitrary order.

4. Pack the binary data into a WAVEFORM tag with the described structure.

   ```
   {WAVEFORM-Length: #I0Q0I1Q1I2Q2 ... InQn}
   ```

   - Calculate the `Length`
     Length = Number of I/Q pairs * 4 + 1 = 20*4 + 1 = 81 bytes
   - Place the string `{WAVEFORM-81:#` at the beginning of the data set
   - Place the symbol `}` at the end of the data set

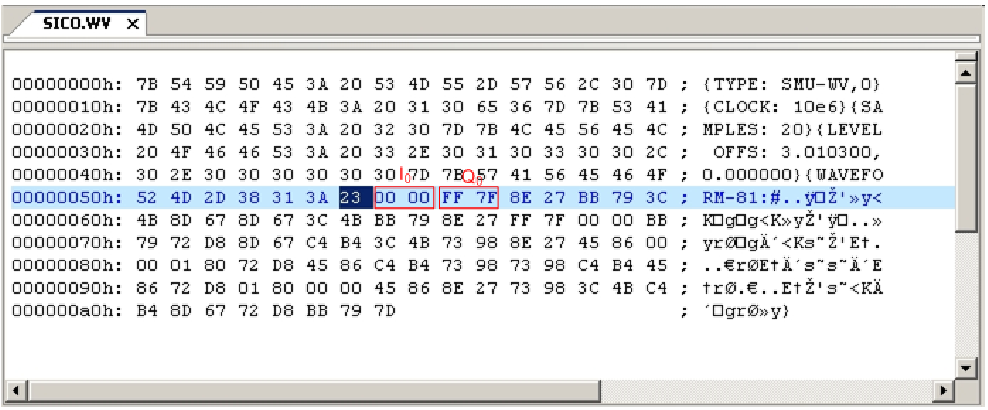
   The contents of the waveform file `SICO.wv` for 20 I/Q pairs is now ready for operation and reads:

   ```
   {TYPE: SMU-WV,0}
   {CLOCK: 10e6}
   {LEVEL OFFS: 3.010300,0.000000}
   ```

`{WAVEFORM-81:#I0Q0I1Q1...InQn}`

Note: There is no readable representation for binary values in this document. This is why we use the sequence `I0Q0I1Q1...InQn` to characterize the binary code in the present example.

The following figure shows this waveform in a data editor.



Example: C-program for creating a waveform file

C-program `SICO.cpp` for creating the file `SICO.txt` containing 20 sine and cosine pairs, converting them into binary data and creating the waveform file `SICO.wv`.

```
// SICO.cpp
// Defines the entry point for the console application

#include "stdafx.h"
#include <stdlib.h>
#include <stdio.h>
#include <math.h>

int _tmain(int argc, _TCHAR* argv[])
{
    const unsigned int samples = 20;
    const float pi = 3.141592654f;
    int i;

// SICO.txt
// Creating the file SICO.txt containing 20 sine and cosine pairs
    float grad, rad;
    FILE *fp;
    fp = fopen("SICO.txt", "w");
    if (fp == 0)
        return;
    for (i=0; i<samples; i++)
    {
        grad = (360.0f / (float)samples) * (float)i;
        rad = grad * (pi / 180.0f);
        fprintf(fp, "%f %f\n", sin(rad), cos(rad));
    }
```

```
    fclose(fp);

// SICO.wv
// Generating a binary data set from the I/Q pairs in the file SICO.txt
// and storing the result to file SICO.wv
    FILE *fp_sour, *fp_dest;
    float i_float, q_float;
    unsigned short i_usint, q_usint;
    fp_sour = fopen("SICO.TXT", "rt");
    if (fp_sour == 0)
        return -1;
    fp_dest = fopen("SICO.WV", "wb");
    if (fp_dest == 0)
    {
        fclose(fp_sour);
        return -1;
    }
    // Write required tags to waveform file
    fprintf(fp_dest, "{TYPE: SMU-WV,0}");
    fprintf(fp_dest, "{CLOCK: 10e6}");
    fprintf(fp_dest, "{SAMPLES: %d}", samples);
    // RMS, Peak
    fprintf(fp_dest, "{LEVEL OFFS: %f,%f}", -1.0f * 20.0f * log10(1.0f/sqrt(2.0f)), 0.0f);
    fprintf(fp_dest, "{WAVEFORM-%d:#", (samples * 4) + 1);
    for (i=0; i<samples; i++)
    {
      // Read I/Q pair from ASCII file
         if (fscanf(fp_sour, "%f %f", &i_float, &q_float) == EOF)
            break;
        // Convert I/Q pair to unsigned short
        i_usint = (unsigned short)floor((i_float * 32767.0) + 0.5);
        q_usint = (unsigned short)floor((q_float * 32767.0) + 0.5);
        // Write converted I/Q pair to waveform file
        fwrite(&i_usint, 2, 1, fp_dest);
        fwrite(&q_usint, 2, 1, fp_dest);
    }
    fprintf(fp_dest, "}");
    fclose(fp_dest);
    fclose(fp_sour);
return 0;
}
```

How to Create a Control List Using Tag File Format

The R&S SMW provides the following ways to create a file containing control signals:

- to use the dedicated "Control Data Editor" and create a file in ASCII format and with extension `*.dm_iqc`, see "To create a control list in ASCII format manually" on page 286
- To use the tag-oriented format and create a control list file, see "To create a control list using tag file format" on page 980

- To use SCPI commands and create a file in binary format, see "To create a control list in binary format" on page 980

To create a control list using tag file format

To create an ASCII control list file directly, use the provided tag commands.

1. Use a hex data editor and create the **mandatory** tags:
   - TYPE
   - [TRACE] LIST

   The `[TRACE] LIST` tag defines the individual markers or control traces in a combined `{Pos:State}` way within the control list period (`CONTROL LENGTH`).

2. Use a hex data editor and create the **recommended** tag CONTROL LENGTH.

   This tag defines the *periodicity* of the total control list

3. Add the required optional tags.
   They can be inserted after the `TYPE` tag in arbitrary order.

   An example of the control list file contents could be:
   ```
   {TYPE:SMU-CL}{COPYRIGHT:Rohde&Schwarz}
   {DATE:2012-06-11;15:00:09}{HOP LIST:0:0;498:1;506:0}
   {CW MODE LIST:0:0;380:1}{LEVATT LIST 3:0:0;464:1}
   {BURST LIST:0:0;122:1;270:0;582:1;924:0}
   {MARKER LIST 4:0:0;706:1;764:0}
   {MARKER LIST 3:0:0;530:1;633:0}
   {MARKER LIST 2:0:0;350:1;457:0}
   {MARKER LIST 1:0:0;108:1;160:0}
   {CONTROL LENGTH:1000}
   ```
   The Figure 4-14 shows the representation of the created control list in the "Control Data Editor".
   Compare the displayed ramp values of "Marker 1" and the "Total List Length" with the values in the corresponding tags.
   Note: In the provided example, the tags have been separated by line breaks for better reading.
   See also Example "How to assign and activate control signals from a control list" on page 981.

To create a control list in binary format

Generation of a control list in binary format is not necessary but possible.

► Use the commands `BB:DM:CLISt:...` to generate a control list in binary format (see "Handling List Files" on page 913 and the example in `[:SOURce<hw>]:BB:DM:CLISt:DATA` on page 916).

  See also Example "How to assign and activate control signals from a control list" on page 981.

Example: How to assign and activate control signals from a control list

Note: Irrespectively on the way they are created, generated control lists are not automatically used.

We assume, that a control list `clist.dm_iqc` containing information on marker 2, burst gate and level attenuation control signals is created and stored in the directory `/var/user/temp/`.

The following example shows how to enable the R&S SMW to:

- Use the control list for a particular marker output, e.g. the "Custom Digital Modulation > Marker 2".
- Use the Burst Gate and Level Attenuation control signals as defined in a control list.

```
MMEM:CDIRectory "/var/user/temp"
SOURce1:BB:DM:CLISt:CATalog?
// Response: clist
SOURce1:BB:DM:CLISt:SELect "clist"
SOURce1:BB:DM:TRIGger:OUTPut2:MODE CLISt

SOURce1:BB:DM:PRAMp:SOURce INTernal
```

How to Create a Data List Using Tag File Format

The R&S SMW provides the following ways to create a data list file:

- To use the dedicated "Data List Editor" and create a file with extension `*.dm_iqd`, see "To create data lists manually" on page 288.
- To use the tag-oriented format and create a data list file, see "To create a data list file using tag file format" on page 981.
- To use SCPI commands and create a file in binary format, see "To create a data list in binary format" on page 982.

To create a data list file using tag file format

► Use a hex data editor and create the mandatory tags: `{TYPE}`, `{DATA BITLENGTH}` and `{DATA LIST}`
Consider the tag syntax and rules.

The following is an example of the data list file content. The tags are separated by line breaks for better reading. The text in brackets is short explanation.

For details, see the tag description:
- TYPE
- {DATA BITLENGTH}
- {DATA LIST-Length}

```
{TYPE:SMU-DL}{COPYRIGHT:Rohde&Schwarz}
{DATE:201-06-11;15:00:09}
{DATA BITLENGTH: 8}
{DATA LIST-2: #d0d1...d7}
```

(1 byte containing 8 data bits `d0` to `d7` in binary format, where `d0` is the MSB)

See also:
- Figure 4-15 for representation of the created data list in the "Data List Editor".

- Example "How to assign and activate a data list" on page 982

To create a data list in binary format

► Use the commands `BB:DM:DLISt:...`, see "Handling List Files" on page 913.

```
MMEM:CDIRectory "/var/user"
// create a new data list file
SOURce1:BB:DM:DLISt:SELect "dl_new"
// append data to the data list and query the content
:FORMat ASCii
SOURce1:BB:DM:DLISt:DATA:APPend 0,1,1,1,0,1,0,1
SOURce1:BB:DM:DLISt:DATA?
// Response: 0,1,1,1,0,1,0,1
```

See also Example "How to assign and activate a data list" on page 982.

Example: How to assign and activate a data list

Note: Irrespectively on the way they are created, generated data lists are not automatically used.

We assume, that a data list `dl.dm_iqd` is created and stored in the directory `/var/user/`.

The following example shows how to enable the R&S SMW to use this data list as data source for the custom digital modulation.

```
MMEM:CDIRectory "/var/user"
SOURce1:BB:DM:DLISt:CATalog?
// Response: dl
SOURce1:BB:DM:DLISt:SELect "dl"
```

Editing Waveform Files, Data and Control Lists

You can edit the internally and externally crated waveform files, data and control lists. The waveform, data and control lists files contain binary and ASCII data.

Consider the following rules while editing files with binary data.

Rules for editing binary data (waveforms, data and control lists)

- **Use hex data editor**
  Always use a hex data editor to edit files containing binary data.
  Editing of binary data file with a text editor, even if you only change the ASCII part of the file, corrupts the file.
- **Adapt the length information in the `{EMPTYTAG}`**
  If you change the content of a waveform file, change also the `{EMPTYTAG-Length}` value.
  For example, if you add a tag or add bytes to an existing tag, reduce the length information by the number of newly introduced bytes.

13.18.4.5 SOURce:BB:MCCW Subsystem

This subsystem contains the commands for setting the Multi-Carrier CW signals.

The generation of multi-carrier CW signals requires an instrument equipped with the software options R&S SMW-K61.

Common Suffixes

The following common suffixes are used in remote commands:

Suffix	Value range	Description
`ENTity<ch>`	1 to 4	Entity in a multiple entity configuration
`SOURce<ch>`	[1] to 4	Available baseband signals
`OUTPut<ch>`	1 to 3	Available markers

You can address multiple entities configurations by using the SCPI commands starting with the keyword `SOURce` or the alias commands starting with the keyword `ENTity`.

See also Chapter 13.3, "SCPI Command Aliases for Advanced Mode with Multiple Entities", on page 767.

Required options

See Chapter 4.10.1, "Required Options", on page 373

Programming Examples

Generating a multi-carrier signal to test the frequency response of a DUT

```
// ******************************************************************
// Generating a multi-carrier signal to test the frequency response of a DUT
// ******************************************************************

// ******************************************************************
// Reset the instrument first
// ******************************************************************
*RST; *CLS

// ******************************************************************
// Configuring and enabling the multi-carrier signal
// ******************************************************************
SOURce:BB:MCCW:CARRier:COUNt 81
SOURce:BB:MCCW:CARRier:SPACing 1E6
// Set the number of carriers and their distance
// Further settings stay in default state (for example trigger settings)
```

```
SOURce:BB:MCCW:STATe ON
// Enable signal generation

OUTPut ON
// Enable the signal output

// ******************************************************************
// Storing the current settings
// ******************************************************************
:MMEMory:MDIR '/var/user/savrcl'
// Create a new directory "savrcl"
*SAV 4
:MMEMory:STORe:STATe 4,"/var/user/savrcl/mccw_SSB.savrcltxt"
// Store the current settings in an intermediate memory with number 4
// Store the settings file in the specified directory; the complete path
// and filename has to be specified
```

To visualize the configured signal, switch to local mode with the command `&NREN` and open the "Multi-carrier Continuous Wave > Carrier Graph".

Generating a multi-carrier signal to test the image rejection of an SSB filter

This example generates a multi-carrier single sideband signal as shown in Figure 13-12.

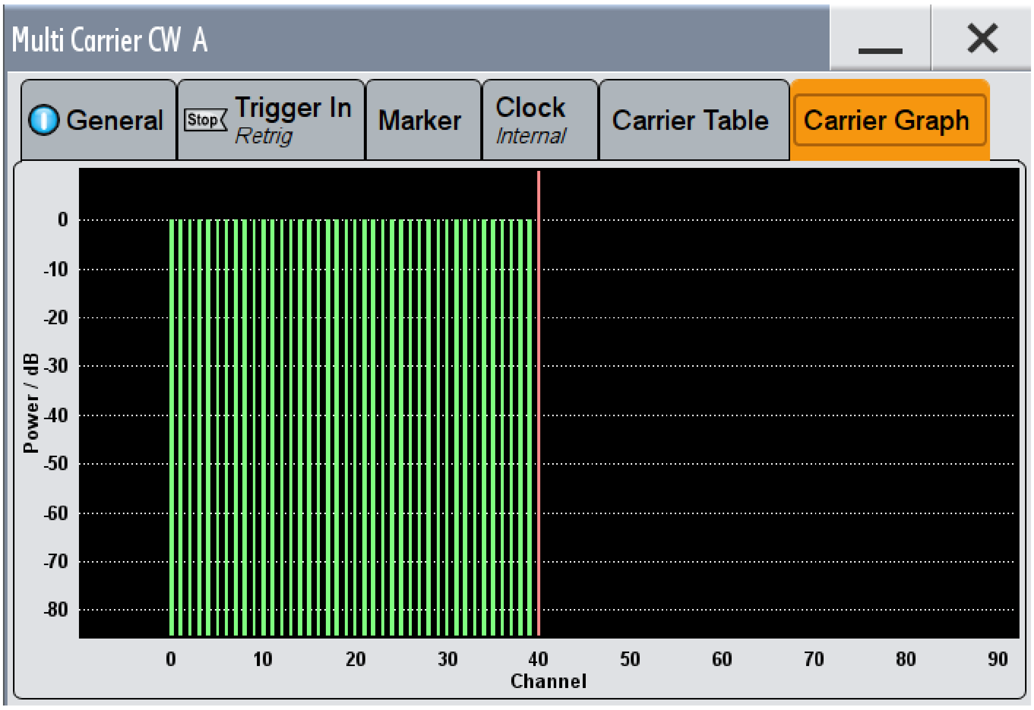


Figure 13-12: Multi-carrier signal to test the image rejection of an SSB filter

```
// ******************************************************************
// Generating a multi-carrier signal to test the image rejection of an SSB filter
// ******************************************************************

// ******************************************************************
// Reset the instrument first
// ******************************************************************
*RST; *CLS

// ******************************************************************
// Configuring and enabling the multi-carrier signal
// ******************************************************************
SOURce:BB:MCCW:CARRier:COUNt 81
SOURce:BB:MCCW:CARRier:SPACing 1E6
// Set the number of carriers and their distance
// Further settings stay in default state (for example trigger settings)

SOURce:BB:MCCW:EDIT:CARRier:STARt 40
SOURce:BB:MCCW:EDIT:CARRier:STOP 80
SOURce:BB:MCCW:EDIT:CARRier:STATe OFF
SOURce:BB:MCCW:EDIT:CARRier:EXECute
// Configure the carrier table for the single sideband signal:
// Set the carriers no. 40 to 80 to "OFF"

SOURce:BB:MCCW:STATe ON
// Enable signal generation

OUTPut ON
// Enable signal output
```

Generating a composed multi-carrier signal

This example generates a multi-carrier signal as shown in Figure 13-13.

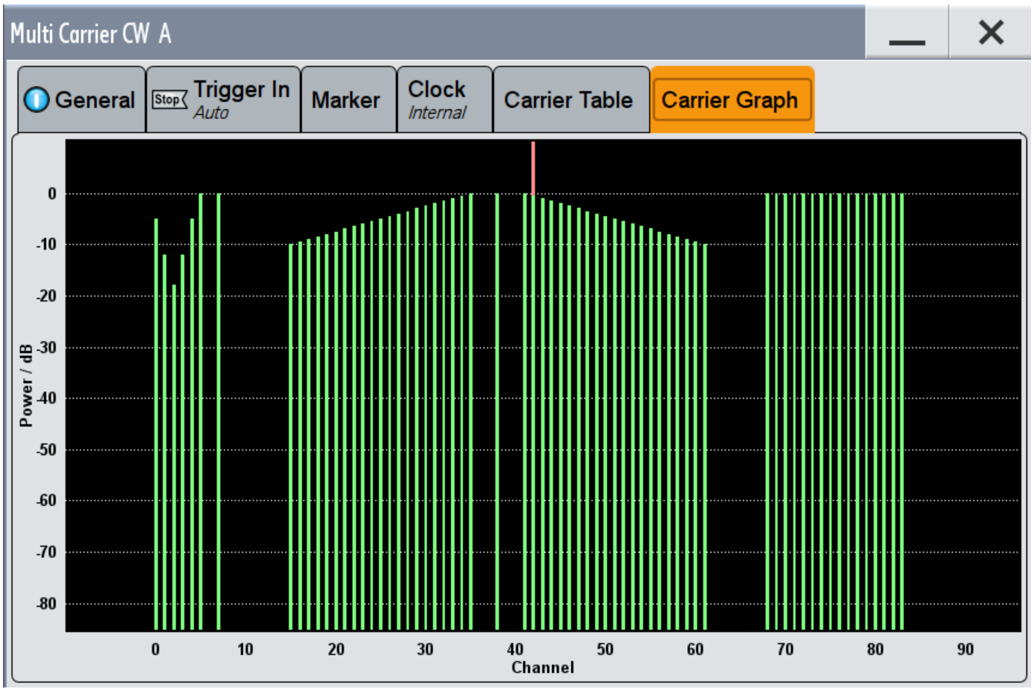


Figure 13-13: Composed multi-carrier signal

```
// ******************************************************************
// Generating a composed multi-carrier signal
// ******************************************************************
// ******************************************************************
// Reset the instrument first
// ******************************************************************
*RST; *CLS

// ******************************************************************
// Configuring the multi-carrier signal
// ******************************************************************
SOURce:BB:MCCW:CARRier:COUNt 100
SOURce:BB:MCCW:CARRier:SPACing 500000
// Setup 100 carriers with a spacing of 500kHz

SOURce:BB:MCCW:CLOCk?
// Return value 128000000
SOURce:BB:MCCW:CFACtor:MODE SLOW
SOURce:BB:MCCW:CFACtor 3
SOURce:BB:MCCW:CFACtor:ACTual?
// Return value 3

// ******************************************************************
// Adjust the settings of a group of carriers
// ******************************************************************
SOURce:BB:MCCW:CARRier:LIST:POWer -6,-12,-18,-12,-6,0
```

```
// Set the power levels for carriers no. 0 to 5 to -10dB,-15dB,-20dB,-15dB,-10dB,0dB
SOURce:BB:MCCW:CARRier:LIST:STATe ON,ON,ON,ON,ON,ON,OFF,ON
// Set the state of the carriers no. 0 to 7
SOURce:BB:MCCW:EDIT:CARRier:STARt 15
SOURce:BB:MCCW:EDIT:CARRier:STOP 35
SOURce:BB:MCCW:EDIT:CARRier:STATe ON
SOURce:BB:MCCW:EDIT:CARRier:POWer:STAR -10
SOURce:BB:MCCW:EDIT:CARRier:POWer:STEP 0.5
SOURce:BB:MCCW:EDIT:CARRier:EXECute
// Activates the carriers no. 15 to 35
// Increase the power in 0.5dB steps starting with a power level of -10dB
// Apply the settings

SOURce:BB:MCCW:CARRier:POWer 38,0
SOURce:BB:MCCW:CARRier:STATe 38,1
// Set a power level of 0dB for carrier no. 38 and activates it

SOURce:BB:MCCW:EDIT:CARRier:STARt 41
SOURce:BB:MCCW:EDIT:CARRier:STOP 61
SOURce:BB:MCCW:EDIT:CARRier:STATe ON
SOURce:BB:MCCW:EDIT:CARRier:POWer:STARt 0
SOURce:BB:MCCW:EDIT:CARRier:POWer:STEP -0.5
SOURce:BB:MCCW:EDIT:CARRier:EXECute
// Activates the carriers no. 41 to 61
// Decrease the power in 0.5dB steps starting with a power level of 0dB
// Apply the settings

SOURce:BB:MCCW:EDIT:CARRier:STARt 70
SOURce:BB:MCCW:EDIT:CARRier:STOP 85
SOURce:BB:MCCW:EDIT:CARRier:STATe ON
SOURce:BB:MCCW:EDIT:CARRier:EXECute
// Activates the carriers no. 70 to 85
// Apply the settings

// ******************************************************************
// Configuring the clock settings
// ******************************************************************
SOURce:BB:MCCW:CLOCk:SOURce INTernal

// ******************************************************************
// Configuring and enabling marker signals
// ******************************************************************
SOURce:BB:MCCW:TRIGger:OUTPut1:MODE RESTart
SOURce:BB:MCCW:TRIGger:OUTPut2:MODE PULSe
SOURce:BB:MCCW:TRIGger:OUTPut2:PULSe:DIVider 5
SOURce:BB:MCCW:TRIGger:OUTPut2:PULSe:FREQuency?
SOURce:BB:MCCW:TRIGger:OUTPut3:MODE PATTern
SOURce:BB:MCCW:TRIGger:OUTPut3:PATTern #HE0F52,20
// SOURce:BB:MCCW:TRIGger:OUTPut1:MODE RATio
// SOURce:BB:MCCW:TRIGger:OUTPut1:ONTime 40
```

```
// SOURce:BB:MCCW:TRIGger:OUTPut1:OFFTime 20
SOURce:BB:MCCW:TRIGger:OUTPut2:DELay 16

// ******************************************************************
// Configuring and enabling signal triggering
// ******************************************************************
SOURce:BB:MCCW:TRIGger:SEQuence SINGle
SOURce:BB:MCCW:TRIGger:SLENgth 200
// The first 200 samples of the current waveform will be output after
// the next trigger event
SOURce:BB:MCCW:TRIGger:SEQuence ARETrigger
SOURce:BB:MCCW:TRIGger:SOURce INTernal
// SOURce:BB:MCCW:TRIGger:SOURce EGT1
// An external global trigger signal must be provided at the connector
// configured for the External Global Trigger 1 signal
// SOURce:BB:MCCW:TRIGger:EXTernal:SYNChronize:OUTPut ON
// SOURce:BB:MCCW:TRIGger:EXTernal:DELay 200
// SOURce:BB:MCCW:TRIGger:EXTernal:INHibit 100

// SOURce:BB:MCCW:TRIGger:SOURce OBASeband
// The internal trigger signal from the other path must be used
// SOURce:BB:MCCW:TRIGger:OBASeband:DELay 25
// SOURce:BB:MCCW:TRIGger:OBASeband:INHibit 10

// ******************************************************************
// Applying the settings and enabling signal generation
// ******************************************************************

*TRG
SOURce:BB:MCCW:STATe ON
// Stop the internal trigger manually
SOURce:BB:MCCW:TRIGger:ARM:EXECute
// Execute manual internal trigger, i.e. restarting signal generation
SOURce:BB:MCCW:TRIGger:EXECute
SOURce:BB:MCCW:TRIGger:RMODe?
// RUN
```

General Settings and Carrier Setup Settings

[:SOURce<hw>]:BB:MCCW:PRESet

Sets all multi carrier signal parameters to their default values.

Example: See "Programming Examples" on page 983

Usage: Event

Manual operation: See "Set to Default" on page 375

[:SOURce<hw>]:BB:MCCW:STATe <State>

Enables/disables the multi carrier CW signal.

Parameters:

<State> 0 | 1 | OFF | ON
*RST: 0

Example: See "Programming Examples" on page 983

Manual operation: See "State" on page 374

[:SOURce<hw>]:BB:MCCW:CARRier:COUNt <Count>

Sets the number of carriers in the multi carrier CW signal.

Parameters:

<Count> integer
Range: 1 to 160001
*RST: 64

Example: see "Programming Examples" on page 983

Manual operation: See "Number of Carriers" on page 375

[:SOURce<hw>]:BB:MCCW:CARRier:SPACing <Spacing>

Sets the carrier spacing.

Parameters:

<Spacing> float

Value range depends on the available bandwidth and the number of carriers, see "Cross-reference between total bandwidth, carrier spacing, and number of carriers" on page 375.

Range: 0 to depends on the installed options, e.g. 120E6 (R&S SMW-B10)
Increment: 0.01
*RST: 10E3

Example: See "Programming Examples" on page 983

Manual operation: See "Carrier Spacing" on page 375

[:SOURce<hw>]:BB:MCCW:CLOCk?

Queries the output clock rate. The output clock rate depends on the number of carriers and on the selected carrier spacing.

Return values:

<Clock> float

Range: 0 to Max
Increment: 1E-3
*RST: 0

Example: See "Programming Examples" on page 983

Usage: Query only

Manual operation: See "Clock Frequency" on page 375

[:SOURce<hw>]:BB:MCCW:CFACtor:MODE <Mode>

Sets the mode by which automatic settings minimize the crest factor or hold it at a chosen value.

Parameters:

<Mode> OFF | CHIRp | SLOW

SLOW
corresponds to the manual control "Target Crest"

*RST: CHIRp

Example: See "Programming Examples" on page 983

Manual operation: See "Optimize Crest Factor Mode" on page 375

[:SOURce<hw>]:BB:MCCW:CFACtor <CFactor>

Sets the desired crest factor, if the optimization mode target crest factor is used.

Parameters:

<CFactor> float

Range: 0 to 30
Increment: 0.01
*RST: 3

Example: See "Programming Examples" on page 983

Manual operation: See "Desired Crest Factor" on page 376

[:SOURce<hw>]:BB:MCCW:CFACtor:ACTual?

Queries the actual Crest Factor for optimization mode target crest.

Return values:

<Actual> float

Range: 0 to 100
Increment: 0.01
*RST: 3

Example: See "Programming Examples" on page 983

Usage: Query only

[:SOURce<hw>]:BB:MCCW:CARRier:STATe <CarrierIndex>, <State>

Switches the selected carrier on or off.

Parameters:

<CarrierIndex> integer

Range: 0 to lastCarrier

<State> 0 | 1 | OFF | ON

*RST: 0

Example: See "Programming Examples" on page 983

Manual operation: See "Carrier Table" on page 378

[:SOURce<hw>]:BB:MCCW:CARRier:LIST:STATe <Stat0[,Stat1..]>
[:SOURce<hw>]:BB:MCCW:CARRier:LIST:STATe? <Start>, <Count>

Switches the carrier on or off with the aid of a value list.

The first value in the list is assigned to the carrier with index 0, the second value to the carrier with index 1, etc. The maximum length corresponds to the maximum number of multi carriers. There is no need to enter all the values every time. Values not set by the value list are set with the default values provided they have already been explicitly set by a previous command. If this is the case, the values continue to apply until overwritten.

Setting parameters:

<Stat0[,Stat1..]> ON | OFF

Query parameters:

<Start> integer

start carrier index

Range: 0 to lastCarrier

<Count> integer

number of carriers in the carrier range, starting from the `<Start>` carrier

Range: 1 to lastCarrier

Return values:

<State[,State..]> select

Example: See "Programming Examples" on page 983

Manual operation: See "Carrier Table" on page 378

[:SOURce<hw>]:BB:MCCW:CARRier:PHASe <CarrierIndex>, <Phase>

For disabled optimization of the crest factor, sets the start phase of the selected carrier.

Parameters:

<CarrierIndex> integer

Range: 0 to lastCarrier

<Phase> float

Range: 0 to 359.99
Increment: 0.01
*RST: 0
Default unit: DEG

Example: see "Programming Examples" on page 983

Manual operation: See "Carrier Table" on page 378

[:SOURce<hw>]:BB:MCCW:CARRier:LIST:PHASe <Phas0[,Phas1..]>
[:SOURce<hw>]:BB:MCCW:CARRier:LIST:PHASe? [<Start>[, <Count>]]

Sets the start phase of the carrier with the aid of a value list.

Setting parameters:

<Phas0[,Phas1..]> float

Range: 0 to 360
Increment: 0.01
*RST: 0
Default unit: DEG

Query parameters:

<Start> integer

start carrier index

Range: 0 to lastCarrier

<Count> integer

number of carriers in the carrier range, starting from the `<Start>` carrier

Range: 1 to max

Return values:

<Phas[,Phas..]> float

Example: see "Programming Examples" on page 983

Manual operation: See "Carrier Table" on page 378

[:SOURce<hw>]:BB:MCCW:CARRier:POWer <CarrierIndex>, <Power>

Sets the power of the selected carrier.

Parameters:

<CarrierIndex> integer

Range: 0 to lastCarrier

<Power> float

Range: -80 to 0
Increment: 0.01
*RST: 0

Example: see "Programming Examples" on page 983

Manual operation: See "Carrier Table" on page 378

[:SOURce<hw>]:BB:MCCW:CARRier:LIST:POWer <Pow0[,Pow1..]>
[:SOURce<hw>]:BB:MCCW:CARRier:LIST:POWer? <Start>, <Count>

Sets the power of the carrier with the aid of a value list.

Setting parameters:

<Pow0[,Pow1..]> float

Increment: 0.01
*RST: 0 dB

Query parameters:

<Start> integer

start carrier index

Range: 0 to lastCarrier

<Count> integer

number of carriers in the carrier range, starting from the `<Start>` carrier

Range: 1 to lastCarrier

Return values:

<Pow[,Pow..]> float

Example: see "Programming Examples" on page 983

Manual operation: See "Carrier Table" on page 378

[:SOURce<hw>]:BB:MCCW:EDIT:CARRier:PHASe[:STARt] <Start>
[:SOURce<hw>]:BB:MCCW:EDIT:CARRier:POWer[:STARt] <Start>

Sets the power/pahse for the starting carrier. The power of the remaining carriers is stepped up or down by the power specified with the `[:SOURce<hw>]:BB:MCCW:EDIT:CARRier:POWer:STEP` command.

Parameters:

<Start> float

Range: -80 to 0
Increment: 0.01
*RST: 0

Example: see "Programming Examples" on page 983

Manual operation: See "Power Start" on page 377

[:SOURce<hw>]:BB:MCCW:EDIT:CARRier:PHASe:STEP <Step>

For disabled optimization of the crest factor, sets the step width by which the start phase of the carriers in the defined carrier range is incremented.

Parameters:

<Step> float

Range: -359.99 to 359.99
Increment: 0.01
*RST: 0

Example: See "Programming Examples" on page 983

Manual operation: See "Phase Step" on page 378

[:SOURce<hw>]:BB:MCCW:EDIT:CARRier:POWer:STEP <Step>

Sets the step width by which the starting power of the carriers in the defined carrier range is incremented.

Parameters:

<Step> float

Range: -80 to 80
Increment: 0.01
*RST: 0

Example: See "Programming Examples" on page 983

Manual operation: See "Power Step" on page 378

[:SOURce<hw>]:BB:MCCW:EDIT:CARRier:STARt <Start>
[:SOURce<hw>]:BB:MCCW:EDIT:CARRier:STOP <Stop>

Defines the first/last carrier in the carrier range to which joint configuration applies.

Parameters:

<Stop> integer

Range: 0 to 8191
*RST: 0

Example: See "Programming Examples" on page 983

Manual operation: See "Carrier Start/Stop" on page 377

[:SOURce<hw>]:BB:MCCW:EDIT:CARRier:STATe <State>

Switches all the carriers in the selected carrier range on or off.

Parameters:

<State> 0 | 1 | OFF | ON

*RST: 1

Example: See "Programming Examples" on page 983

Manual operation: See "Carrier State" on page 377

[:SOURce<hw>]:BB:MCCW:EDIT:CARRier:EXECute

Adopts the settings for the carrier range.

Example: See "Programming Examples" on page 983

Usage: Event

Manual operation: See "Accept" on page 378

Trigger Settings

[:SOURce<hw>]:BB:MCCW:TRIGger:ARM:EXECute

Stops signal generation.

Example: See "Programming Examples" on page 983

Usage: Event

Manual operation: See "Arm" on page 255

[:SOURce<hw>]:BB:MCCW:TRIGger:EXECute

Executes a trigger.

Example: See "Programming Examples" on page 983

Usage: Event

Manual operation: See "Execute Trigger" on page 255

[:SOURce<hw>]:BB:MCCW:TRIGger:EXTernal:SYNChronize:OUTPut <Output>

Enables signal output synchronous to the trigger event.

Parameters:

<Output> 0 | 1 | OFF | ON

*RST: 1

Example: See "Programming Examples" on page 983

Manual operation: See "Sync. Output to External Trigger/Sync. Output to Trigger" on page 256

[:SOURce<hw>]:BB:MCCW:TRIGger:OBASeband:DELay <Delay>

Specifies the trigger delay for external triggering.

Parameters:

<Delay> float

Range: 0 to 2147483647
Increment: 0.01
*RST: 0
Default unit: samples

Example: See "Programming Examples" on page 983

Manual operation: See "(Specified) External Delay/(Specified) Trigger Delay" on page 257

[:SOURce<hw>]:BB:MCCW:TRIGger:OBASeband:INHibit <Inhibit>

Specifies the number of samples by which a restart is inhibited following a trigger event from the other path.

Parameters:

<Inhibit> integer

Range: 0 to 67108863
*RST: 0

Example: See "Programming Examples" on page 983

Manual operation: See "External / Trigger Inhibit" on page 257

[:SOURce<hw>]:BB:MCCW:TRIGger:RMODe?

Queries the status of signal generation for all trigger mode, if multi-carrier CW generation is on.

Return values:

<RMode> STOP | RUN

Example: See "Programming Examples" on page 983

Usage: Query only

Manual operation: See "Running/Stopped" on page 255

[:SOURce<hw>]:BB:MCCW:TRIGger:SLENgth <SLength>

Defines the length of the signal sequence to be output in the "Single" trigger mode.

Parameters:

<SLength> integer

Range: 1 to 1000
*RST: 1
Default unit: samples

Example: See "Programming Examples" on page 983

Manual operation: See "Trigger Signal Duration" on page 255

[:SOURce<hw>]:BB:MCCW:TRIGger:SOURce <Source>

Selects the trigger signal source and determines the way the triggering is executed. Provided are:

- Internal triggering by a command (`INTernal`)
- External trigger signal via one of the local or global connectors
  - `EGT1|EGT2`: External global trigger
  - `EGC1|EGC2`: External global clock
  - `ELTRigger`: External local trigger
  - `ELCLock`: External local clock
- Internal triggering by a signal from the other basebands (`INTA|INTB`)
- In master-slave mode, the external baseband synchronization signal (`BBSY`)
- `OBASeband|BEXTernal|EXTernal`: Setting only

Provided only for backward compatibility with other Rohde & Schwarz signal generators.
The R&S SMW accepts these values and maps them automatically as follows:
`EXTernal = EGT1, BEXTernal = EGT2, OBASeband = INTA or INTB`
(depending on the current baseband)

Parameters:

<Source>	INTB | INTernal | OBASeband | EGT1 | EGT2 | EGC1 | EGC2 | ELTRigger | INTA | ELCLock | BEXTernal | EXTernal | BBSY *RST: INTernal

Example: See "Programming Examples" on page 983

Options: ELTRigger|ELCLock require R&S SMW-B10
BBSY require R&S SMW-B9

Manual operation: See "Trigger Source" on page 256

[:SOURce<hw>]:BB:MCCW:TRIGger[:EXTernal]:DELay <Delay>

Specifies the trigger delay.

Parameters:

<Delay>	float Range: 0 to 2147483647 Increment: 0.01 *RST: 0 Default unit: samples

Example: See "Programming Examples" on page 983

Manual operation: See "(Specified) External Delay/(Specified) Trigger Delay" on page 257

[:SOURce<hw>]:BB:MCCW:TRIGger[:EXTernal]:INHibit <Inhibit>

Specifies the number of samples by which a restart is inhibited following an external trigger event.

Parameters:

<Inhibit>	integer Range: 0 to 21.47*clockRate *RST: 0

Example: See "Programming Examples" on page 983

Manual operation: See "External / Trigger Inhibit" on page 257

[:SOURce<hw>]:BB:MCCW[:TRIGger]:SEQuence <Sequence>

Selects the trigger mode. For detailed description of the trigger modes, refer to "Impact of the Trigger Modes on the Signal Generation" on page 239.

Parameters:

<Sequence> AUTO | RETRigger | AAUTo | ARETrigger | SINGle

*RST: AUTO

Example: See "Programming Examples" on page 983

Manual operation: See "Trigger Mode" on page 255

Marker Settings

[:SOURce<hw>]:BB:MCCW:TRIGger:OUTPut<ch>:MODE <Mode>

Defines the signal for the selected marker output. For detailed description of the regular marker modes, refer to "Marker Modes" on page 235.

Parameters:

<Mode> RESTart | PULSe | PATTern | RATio

*RST: RESTart

Example: See "Programming Examples" on page 983

Manual operation: See "Marker Mode" on page 259

[:SOURce<hw>]:BB:MCCW:TRIGger:OUTPut<ch>:ONTime <OnTime>
[:SOURce<hw>]:BB:MCCW:TRIGger:OUTPut<ch>:OFFTime <OffTime>

Sets the number of samples in the off period of the corresponding marker signal.

*) If R&S SMW-B9 is installed, the minimum marker duration depends on the sample/symbol rate.

See "Marker Minimum Duration" on page 236.

Parameters:

<OffTime> integer

Range: 1 (R&S SMW-B10) / 16* (R&S SMW-B9) to 16777215

*RST: 1

Example: see "Programming Examples" on page 983

Manual operation: See "Marker Mode" on page 259

[:SOURce<hw>]:BB:MCCW:TRIGger:OUTPut<ch>:PATTern <Pattern>

Defines the bit pattern used to generate the marker signal.

Parameters:

<Pattern>	<32 bit pattern> 0 = marker off, 1 = marker on *RST: 0
Example:	See "Programming Examples" on page 983
Manual operation:	See "Marker Mode" on page 259

[:SOURce<hw>]:BB:MCCW:TRIGger:OUTPut<ch>:PULSe:DIVider <Divider>

Sets the divider for pulse marker mode.

*) If R&S SMW-B9 is installed, the minimum marker duration depends on the sample/symbol rate.

See "Marker Minimum Duration" on page 236.

Parameters:

<Divider>	integer Range: 2 (R&S SMW-B10) / 32* (R&S SMW-B9) to 1024 *RST: 2
Example:	See "Programming Examples" on page 983
Manual operation:	See "Marker Mode" on page 259

[:SOURce<hw>]:BB:MCCW:TRIGger:OUTPut<ch>:PULSe:FREQuency?

Queries the pulse frequency of the pulsed marker signal.

Return values:

<Frequency>	float Increment: 1E-3
Example:	see "Programming Examples" on page 983
Usage:	Query only
Manual operation:	See "Marker Mode" on page 259

[:SOURce<hw>]:BB:MCCW:TRIGger:OUTPut<ch>:DELay <Delay>

Defines the delay between the signal on the marker outputs and the start of the signals, expressed as number of samples.

Parameters:

<Delay> float

Range: 0 to 16777215
Increment: 0.001
*RST: 0

Example: See "Programming Examples" on page 983

Manual operation: See "Marker x Delay" on page 260

Clock Settings

[:SOURce<hw>]:BB:MCCW:CLOCk:SOURce <Source>

Selects the clock source:

- `INTernal`: Internal clock reference
- `ELCLock`: External local clock
- `EXTernal = ELCLock`: Setting only
  Provided for backward compatibility with other Rohde & Schwarz signal generators

Parameters:

<Source> INTernal | ELCLock | EXTernal

*RST: INTernal

Example: see "Programming Examples" on page 983

Options: ELCLock requires R&S SMW-B10

Manual operation: See "Clock Source" on page 260

[:SOURce<hw>]:BB:MCCW:CLOCk:MODE <Mode>

Sets the type of externally supplied clock.

Parameters:

<Mode> SAMPle

*RST: SAMPle

Example: See "Programming Examples" on page 983

Options: R&S SMW-B10

Manual operation: See "Clock Mode" on page 261

13.18.4.6 SOURce:AWGN Subsystem

The `SOURce:AWGN` subsystem contains the commands for setting the noise generator.

Suffixes in the keywords ENTity<ch> and SOURce<hw>

You can address multiple entities configurations by using the SCPI commands starting with the keyword `SOURce` or the alias commands starting with the keyword `ENTity`.

Table 13-7: Value ranges of the suffixes ENTity<ch> and SOURce<hw> in advanced configuration with multiple entities

SCPI Syntax	ENTity<ch>	SOURce<hw>
SOURce<hw>:AWGN:...	-	1 .. 8
ENTity<ch>:SOURce<hw>:AWGN:...	1 .. 8	1 .. 4

The meaning of the numeric suffix to `SOURce` depends on:

- The presence of the keyword `ENTity`
- The selected `:SCONfiguration:MODE`
- The enabled MIMO configuration (`:SCONfiguration:FADing`)
- The state of the `[:SOURce<hw>]:AWGN:CMODe[:STATe]`

See the example in Chapter 13.3, "SCPI Command Aliases for Advanced Mode with Multiple Entities", on page 767.

Required options

See Chapter 5.2.1, "Required Options", on page 389.

Programming Examples

Example: Enabling/disabling the AWGN generator and the streams

The following example shows the difference between working with coupled and uncoupled streams.

```
SCONfiguration:MODE ADVanced
SCONFiguration:FADing MIMO4X4

// coupled mode; the AWGN generator and all four streams are enabled
SOURce1:AWGN:CMODe:STATe ON
SOURce1:AWGN:STATe ON

// uncoupled mode; the streams are configured individually
SOURce1:AWGN:CMODe:STATe OFF
SOURce1:AWGN:STATe?
// Response: 1
SOURce2:AWGN:STATe?
// Response: 0
SOURce3:AWGN:STATe?
// Response: 0
SOURce4:AWGN:STATe?
// Response: 0
```

```
// stream B, C and D are disabled
SOURce4:AWGN:STATe ON
```

Example: Generating a pure noise signal

The following example generates a pure noise signal with specified bandwidth and noise level.

```
SCONfiguration:MODE STANdard
SOURce1:AWGN:MODE ONLY
SOURce1:AWGN:BWIDth 3840000
SOURce1:AWGN:BWIDth:RATio 2
SOURce1:AWGN:STATe ON
SOURce1:AWGN:BWIDth:NOISe?
// Response: 7680000
SOURce1:AWGN:DISP:MODE RFA
SOURce1:AWGN:POWer:NOISe -80
// SOURce1:POWer:LEVel:IMMediate:AMPlitude?
// Response: -80
SOURce1:AWGN:POWer:NOISe:TOTal?
```

Example: Generating a CW interferer signal

The following example generates a CW interferer signal in stream B with specified target frequency and level.

```
SCONfiguration:MODE ADVanced
SCONFiguration:FADing MIMO2X2
SOURce:AWGN:CMODe:STATe OFF

SOURce2:AWGN:MODE CW
SOURce2:AWGN:FREQuency:TARGet 20000000
SOURce2:AWGN:STATe ON
SOURce2:AWGN:FREQuency:RESult?

SOURce2:AWGN:POWer:RMODe CARRier
SOURce2:AWGN:CNRatio 10
SOURce2:AWGN:POWer:CARRier -80
// the Level display indicates the PEP of the carrier
// SOURce2:POWer:LEVel:IMMediate:AMPlitude?
// Response: -80
// Querry the power of the interfering signal
SOURce2:AWGN:POWer:NOISe?
// Response: -90
SOURce2:AWGN:POWer:SUM?
// Response: -79.5860731484178
SOURce2:AWGN:POWer:SUM:PEP?
// Response: -80
```

Example: Generating an additive noise signal

The following example shows how to adjust the AWGN settings to generate a signal with the following characteristics:

- The carrier signal is an uplink EUTRA/LTE signal with:
  – "Channel Bandwidth = 1.4 GHz", i.e. "Occupied Bandwidth = 1.080 MHz"
  – "RF Frequency = 1.950 GHz"
  – "RF Level = -76 dBm"
- Required is an SNR of 12.7 dB

```
SCONfiguration:MODE STANdard
SOURce1:FREQuency:CW 1950000000
SOURce1:POWer:LEVel:IMMediate:AMPlitude -76

SOURce1:AWGN:MODE ADD
// set the system bandwidht to the occupied bandwidth
SOURce1:AWGN:BWIDth 1080000
SOURce1:AWGN:BWIDth:RATio 1.5
SOURce1:AWGN:STATe ON
SOURce1:AWGN:BWIDth:NOISe?
// Response: 1620000

SOURce1:AWGN:DISP:MODE RFA
SOURce1:AWGN:POWer:MODE CN
SOURce1:AWGN:POWer:RMODe CARRier
SOURce1:AWGN:BRATe?
//  Response: 100000
SOURce1:AWGN:CNRatio 12.7
SOURce1:AWGN:ENRatio?
// Response: 23.0342375548695
SOURce1:AWGN:POWer:CARRier?
// Response: -76
// the Level display indicates the PEP of the carrier

// Querry the resulting noise power, in the system and total bandwidht
SOURce1:AWGN:POWer:NOISe?
// Response: -88.7
SOURce1:AWGN:POWer:NOISe:TOTal?
// Response: -88.7
// Querry the carrier+noise power and PEP
SOURce1:AWGN:POWer:SUM?
// Response: -75.7728170942726
SOURce1:AWGN:POWer:SUM:PEP?
// Response: -76
```

[:SOURce<hw>]:AWGN:CMODe[:STATe] <State>

Couples the configuration of all streams.

Parameters:

<State>	0 | 1 | OFF | ON *RST: 1
Example:	See Example "Enabling/disabling the AWGN generator and the streams" on page 1002
Manual operation:	See "Coupled Mode" on page 396

[:SOURce<hw>]:AWGN:STATe <State>

Activates or deactivates the AWGN generator or the corresponding stream.

See also "Suffixes in the keywords ENTity<ch> and SOURce<hw>" on page 1002.

Parameters:

<State>	0 | 1 | OFF | ON *RST: 0
Example:	See Example "Enabling/disabling the AWGN generator and the streams" on page 1002
Manual operation:	See "State" on page 396

[:SOURce<hw>]:AWGN:MODE <Mode>

Determines how the interfering signal is generated.

Parameters:

<Mode> ONLY | ADD | CW

ADD
The AWGN noise signal is added to the baseband signal.

ONLY
The pure AWGN noise signal is modulated to the carrier. The connection to the baseband is interrupted

CW
The sine interfering signal is added to the baseband signal.

*RST: ADD

Example: See Example "Generating an additive noise signal" on page 1004

Manual operation: See "Mode" on page 396

[:SOURce<hw>]:AWGN:BWIDth <BWidth>

Sets the system bandwidth.

Parameters:

<BWidth> float

Range: 1000 to 80E6
Increment: 100
*RST: 3.84E6

Example: See Example "Generating an additive noise signal" on page 1004

Manual operation: See "System Bandwidth" on page 397

[:SOURce<hw>]:AWGN:BWIDth:RATio <Ratio>

Sets the ratio of minimum real noise bandwidth to system bandwidth, see also "Signal and noise parameters" on page 390.

Parameters:

<Ratio> float

Range: 1 to Max
Increment: 0.1
*RST: 1

Example: see Example "Generating an additive noise signal" on page 1004

Manual operation: See "Minimum Noise/System Bandwidth Ratio" on page 397

[:SOURce<hw>]:AWGN:BWIDth:NOISe?

Queries the real noise bandwidth.

Return values:

<Noise> float

Range: 0 to 200E6
Increment: 100
*RST: 0

Example: See Example "Generating an additive noise signal" on page 1004

Usage: Query only

Manual operation: See "Noise Bandwidth" on page 403

[:SOURce<hw>]:AWGN:DISP:MODE <Mode>

Sets the output to that the AWGN settings are related.

Parameters:

<Mode> RFA | RFB | IQOUT1 | IQOUT2 | BBMM1 | BBMM2 | FADER1 | FADER2 | FADER3 | FADER4

*RST: RF

Example: See Example "Generating an additive noise signal" on page 1004

Manual operation: See "Show Powers for Output" on page 399

[:SOURce<hw>]:AWGN:POWer:MODE <Mode>

Selects the mode for setting the noise level.

Parameters:

<Mode> CN | SN | EN

*RST: SN

Example: see Example "Generating an additive noise signal" on page 1004

Manual operation: See "Set Noise Power Via" on page 399

[:SOURce<hw>]:AWGN:POWer:RMODe <RMode>

Determines whether the carrier or the noise level is kept constant when the C/N value or Eb/N0 value is changed.

Parameters:

<RMode> CARRier | NOISe

*RST: CARRier

Example: See Example "Generating an additive noise signal" on page 1004

Manual operation: See "Reference Mode" on page 399

[:SOURce<hw>]:AWGN:BRATe <BRate>

Sets the bit rate used for calculation of bit energy to noise power ratio.

Valid units are bps, kbps and mabps as well as b/s, kb/s and mab/s.

Parameters:

<BRate> float

Range: 400 to depends on the installed options
Increment: 0.001
*RST: 100000

Example: see Example "Generating an additive noise signal" on page 1004

Manual operation: See "Bit Rate" on page 400

[:SOURce<hw>]:AWGN:CNRatio <CnRatio>

Sets the carrier/interferer ratio.

Parameters:

<CnRatio> float

Range: -50 to 45
Increment: 0.01
*RST: 0

Example: See Example "Generating an additive noise signal" on page 1004

Manual operation: See "Carrier/Noise Ratio, Signal/Noise Ratio" on page 400

[:SOURce<hw>]:AWGN:ENRatio <EnRatio>

Sets the ratio of bit energy to noise power density.

Parameters:

<EnRatio> float

Range: -50 to depends on the installed options
Increment: 0.01
*RST: 15.84
Default unit: dB

Example: See Example "Generating an additive noise signal" on page 1004

Manual operation: See "E_b/N_0" on page 400

[:SOURce<hw>]:AWGN:POWer:CARRier <Carrier>

Sets the carrier power.

Parameters:

<Carrier> float

Increment: 0.01
*RST: 0

Example: see Example "Generating an additive noise signal" on page 1004

Manual operation: See "Carrier Power, Signal Power" on page 401

[:SOURce<hw>]:AWGN:POWer:NOISe <Noise>

Sets the power of the noise signal in the system respectively total bandwidth.

Parameters:

<Noise> float

Increment: 0.01

Example: see Example "Generating an additive noise signal" on page 1004

Manual operation: See "Noise Power (System Bandwidth), Interferer Power" on page 401

[:SOURce<hw>]:AWGN:POWer:NOISe:TOTal?

Queries the noise level in the total bandwidth.

Return values:

<Total> float

Range: -145 to 20
Increment: 0.01
*RST: -30

Example: see Example "Generating an additive noise signal" on page 1004

Usage: Query only

Manual operation: See "Noise Power (Total Bandwidth)" on page 402

[:SOURce<hw>]:AWGN:POWer:SUM?

Queries the overall power of the noise/interferer signal plus useful signal

Return values:

<Sum> float

Range: -145 to 20
Increment: 0.01
*RST: 0

Example: see Example "Generating an additive noise signal" on page 1004

Usage: Query only

Manual operation: See "Carrier + Noise Power, Signal + Noise Power (System Bandwidth), Carrier + Interferer Power, Signal + Interferer Power " on page 403

[:SOURce<hw>]:AWGN:POWer:SUM:PEP?

Queries the peak envelope power of the overall signal comprised of noise signal plus useful signal.

Return values:

<Pep> float

Range: -145 to 20
Increment: 0.01
*RST: 0

Example: see Example "Generating an additive noise signal" on page 1004

Usage: Query only

Manual operation: See "Carrier + Noise PEP, Signal + Noise PEP (Total Bandwidth), Carrier + Interferer PEP, Signal + Interferer PEP" on page 403

[:SOURce<hw>]:AWGN:FREQuency:TARGet <Target>

Sets the desired frequency of the sine wave.

Parameters:

<Target> float

Range: -40E6 to 40E6
Increment: 0.01
*RST: 0

Example: see Example "Generating a CW interferer signal" on page 1003

Manual operation: See "Target CW Frequency Offset" on page 397

[:SOURce<hw>]:AWGN:FREQuency:RESult?

Queries the actual frequency of the sine wave.

Return values:

<Result> float

Range: -40E6 to 40E6
Increment: 0.01
*RST: 0

Example: see Example "Generating a CW interferer signal" on page 1003

Usage: Query only

Manual operation: See "Resulting CW Frequency Offset" on page 397

13.18.4.7 SOURce:BB:IMPairment Subsystem

This subsystem contains the commands for the analog and digital I/Q impairments.

Suffixes

The following suffixes are used:

Suffix	Value range	Description
RF<ch>, BBMM<ch>, IQOutput<ch>	1 to 2	DIG IQ connector
FADer<ch>	1 to 4	DIG IQ connector

Required options

See Chapter 5.3.1, "Required Options", on page 406.

[:SOURce]:BB:IMPairment:BBMM<ch>:DELay <Delay>
[:SOURce]:BB:IMPairment:FADer<ch>:DELay <Delay>
[:SOURce]:BB:IMPairment:IQOutput<ch>:DELay <Delay>
[:SOURce]:BB:IMPairment:RF<ch>:DELay <Delay>

Defines the time delay of both I and Q vectors between the marker signal at the marker outputs relative to the signal generation start.

A positive value means that the I and Q vectors delay relative to the marker/trigger and vice versa.

Table 13-8: Value range

Output	Min	Max	Resolution
`RF<ch>`	0	10E-6	1E-12
`FADer<ch>`	0	500E-9	1E-12
`IQOutput<ch>`	-500E-9	500E-9	1E-12
`BBMM<ch>`	-500E-9	500E-9	1E-12

Parameters:

<Delay> float

Range: 0 to 10E-6
Increment: 1E-12
*RST: 0

Example: `SOURce:BB:IMPairment:RF1:DELay 32.0E-9`

Manual operation: See "IQ Delay" on page 412

[:SOURce<hw>]:IQ:IMPairment:IQRatio[:MAGNitude] <Magnitude>
[:SOURce]:BB:IMPairment:BBMM<ch>:IQRatio[:MAGNitude] <IqRatio>
[:SOURce]:BB:IMPairment:FADer<ch>:IQRatio[:MAGNitude] <IqRatio>
[:SOURce]:BB:IMPairment:IQOutput<ch>:IQRatio[:MAGNitude] <IqRatio>
[:SOURce]:BB:IMPairment:RF<ch>:IQRatio[:MAGNitude] <IqRatio>

Sets the ratio of I modulation to Q modulation (amplification imbalance) of the corresponding digital channel.

Table 13-9: Value range

Impairments	Min [dB]	Max [dB]	Resolution
Digital	-4	4	0.0001
Analog	-1	1	0.0001

Parameters:

<IqRatio> float

The setting value can be either in dB or %. An input in percent is rounded to the closest valid value in dB.

Range: -4 to 4
Increment: 1E-4
*RST: 0
Default unit: dB | PCT (setting command) / dB (result value)

Example:

```
SOURce:BB:IMPairment:RF1:IQRatio:MAGNitude 10 PCT
SOURce:BB:IMPairment:RF1:IQRatio:MAGNitude?
// 0.848
// the value is returned in dB
```

Manual operation: See "Gain Imbalance" on page 411

[:SOURce<hw>]:IQ:IMPairment:LEAKage:I <I>
[:SOURce<hw>]:IQ:IMPairment:LEAKage:Q <Q>
[:SOURce]:BB:IMPairment:BBMM<ch>:LEAKage:I <I>
[:SOURce]:BB:IMPairment:BBMM<ch>:LEAKage:Q <Q>
[:SOURce]:BB:IMPairment:FADer<ch>:LEAKage:I <I>
[:SOURce]:BB:IMPairment:FADer<ch>:LEAKage:Q <Q>
[:SOURce]:BB:IMPairment:IQOutput<ch>:LEAKage:I <I>
[:SOURce]:BB:IMPairment:IQOutput<ch>:LEAKage:Q <Q>
[:SOURce]:BB:IMPairment:RF<ch>:LEAKage:I <I>
[:SOURce]:BB:IMPairment:RF<ch>:LEAKage:Q <Q>

Determines the leakage amplitude of the I or Q signal component of the corresponding stream

Parameters:

<Q> float

Range: -10 to 10
Increment: 0.01
*RST: 0

Example: `SOURce:BB:IMPairment:RF1:LEAKage:Q 4 PCT`
Sets the leakage for the Q-component to 4 percent.

Manual operation: See "I/Q Offset" on page 411

[:SOURce<hw>]:IQ:IMPairment:QUADrature[:ANGLe] <Angle>
[:SOURce]:BB:IMPairment:BBMM<ch>:QUADrature[:ANGLe] <Angle>
[:SOURce]:BB:IMPairment:FADer<ch>:QUADrature[:ANGLe] <Angle>
[:SOURce]:BB:IMPairment:IQOutput<ch>:QUADrature[:ANGLe] <Angle>
[:SOURce]:BB:IMPairment:RF<ch>:QUADrature[:ANGLe] <Angle>

Sets a quadrature offset (phase angle) between the I and Q vectors deviating from the ideal 90 degrees.

A positive quadrature offset results in a phase angle greater than 90 degrees.

Table 13-10: Value range

Impairments	Min [dB]	Max [dB]	Resolution
Digital	-30	30	0.01
Analog	-10	10	0.01

Parameters:

<Angle> float

Range: -30 to 30
Increment: 0.01
*RST: 0
Default unit: DEG

Example: `SOURce:BB:IMPairment:RF1:QUADrature:ANGLe -5`
Sets the quadrature offset to -5 degrees.

Manual operation: See "Quadrature Offset" on page 412

[:SOURce]:BB:IMPairment:BBMM<ch>:SKEW <Skew>
[:SOURce]:BB:IMPairment:FADer<ch>:SKEW <Skew>
[:SOURce]:BB:IMPairment:IQOutput<ch>:SKEW <Skew>
[:SOURce]:BB:IMPairment:RF<ch>:SKEW <Skew>

Sets a delay between the Q vector and the I vector of the corresponding stream.

Parameters:

<Skew> float

Range: -500E-9 to 500E-9
Increment: 1E-12
*RST: 0

Manual operation: See "Skew" on page 412

[:SOURce<hw>]:IQ:IMPairment[:STATe] <State>
[:SOURce]:BB:IMPairment:BBMM<ch>:STATe <State>
[:SOURce]:BB:IMPairment:FADer<ch>:STATe <State>
[:SOURce]:BB:IMPairment:IQOutput<ch>:STATe <State>
[:SOURce]:BB:IMPairment:RF<ch>:STATe <State>

Activates the impairment or correction values `LEAKage`, `QUADrature` and `IQRatio` for the corresponding stream.

Parameters:

<State> 0 | 1 | OFF | ON

*RST: 0

Example: `SOURce1:BB:IMPairment:RF1:STATe ON`
Activates digital impairment.

Manual operation: See "State" on page 411

[:SOURce<hw>]:BB:IMPairment:OPTimization:MODE <Mode>

Sets the optimization mode.

Option: R&S SMW-K544

The value selected here is used also as optimization mode for the user-defined frequency response corrections, and vice versa.

See user manual R&S®SMW-K544 User-Defined Frequency Response Correction.

Parameters:

<Mode> FAST | QHIGh | QHTable

FAST
Optimization by compensation for I/Q skew.

QHTable
Enabled if R&S SZU is connected to the R&S SMW and the connection is active.
Improved optimization by maintained speed.

QHIGh
Optimization by compensation for I/Q skew and frequency response correction.
This mode interrupts the RF signal. Do not use it in combination with the uninterrupted level settings and strictly monotone modes RF level modes (see `[:SOURce<hw>]:POWer:LBEHaviour` on page 1097)

*RST: FAST (R&S SMW-B10) / QHIGh (R&S SMW-B9) / QHTable (if R&S SZU connected to R&S SMW and the connection is active)

Example:

```
SOURce1:IQ:SOURce BASeband
SOURce1:BB:IMPairment:OPTimization:MODE FAST
```

Manual operation: See "Optimization Mode" on page 421

13.18.4.8 SOURce:BB:GRAPhics Subsystem

This subsystem contains the commands used to setup the graphical display.

Required options

See Chapter 8.1.1, "Required Options", on page 559

[:SOURce<hw>]:BB:GRAPhics:MODE <Mode>

Selects the graphics mode of the graphical signal display.

Parameters:

<Mode>	IQ | VECTor | CCDF | PSPectrum | CONStellation | EYEI | EYEQ *RST: IQ
Example:	`BB:GRAP:MODE VECT` Selects the vector diagram graphics mode.
Manual operation:	See "Mode" on page 570

[:SOURce<hw>]:BB:GRAPhics:SRATe:MODE <Mode>

Sets how the time resolution of the signal is determined. Maximum resolution corresponds to a diagram covering the entire signal bandwidth. The higher the resolution is, the shorter the length of the displayed signal segment will be for the specified recording depth.

Parameters:

<Mode>	AUTO | FULL | USER *RST: AUTO
Example:	`BB:GRAP:SRAT:MODE FULL` Sets the sample rate mode.
Manual operation:	See "Sample Rate Mode" on page 571

[:SOURce<hw>]:BB:GRAPhics:SRATe:USER <User>

(Enabled for `BB:GRAPH:SRAT:MODE USER`)

Selects the signal bandwidth for the diagram. The setting range moves between the minimum and maximum bandwidth which is possible for the selected graphical signal display. The selection is made graphically by moving the pointer.

Parameters:

<User>	float Range: 0.01 to 100 Increment: 0.01 *RST: 10 Default unit: PCT
Example:	`BB:GRAP:SRAT:USER 20` sets the sample rate factor.

Example: `BB:GRAP:SRAT:MODE USER`
sets the sample rate mode.

Manual operation: See "Sample Rate" on page 571

[:SOURce]:BB:GRAPhics:CLOSe

Closes all graphical signal displays.

Usage: Event

Manual operation: See "Remove" on page 571

[:SOURce]:BB:GRAPhics:ADD <Size>

Adds a graphical signal display (according to the current `MODE`, `SOURce`, `SRATe:*` and `TRIGger:*` settings).

Setting parameters:
<Size> MAXimized | MINimized

Usage: Setting only

Manual operation: See "Add" on page 571

[:SOURce]:BB:GRAPhics:SOURce <Source>

Defines the signal acquisition point, that is the location in the signal flow where the displayed signal is tapped from.

The available acquisition points depend on the selected system configuration.

Parameters:
<Source> STRA | STRB | STRC | STRD | STRE | STRF | STRG | STRH | BBA | BBB | BBC | BBD | BBE | BBF | BBG | BBH | RFA | RFB | RFC | RFD | IQO1 | IQO2 | DO1 | DO2 | BBIA | BBIB

STRA|STRB|STRC|STRD|STRE|STRF|STRG|STRH
Streams (A ... H); input stream of the "IQ Stream Mapper"

BBA|BBB|BBC|BBD|BBE|BBF|BBG|BBH
Baseband signals (A ... H)

BBIA|BBIB
Digital baseband input signals

RFA|RFB|RFC|RFD
RF signals (A ... D)

IQO1|IQO2
Analog I/Q output signals

DO1|DO2
Digital I/Q output signals; outputs of the "IQ Stream Mapper"

*RST: STRA

Manual operation: See "Source" on page 570

Location and use of the source types is shown in Chapter 8.1.2.2, "Signal Acquisition Points", on page 565.

[:SOURce]:BB:GRAPhics:SOURce:MUX <Mode>

In :SCONfiguration:OUTPut:MODE DIGMux mode, select which of the multiplexed streams is displayed.

Parameters:

<Mode> STRA | STRB | STRC | STRD | STRE | STRF | STRG | STRH

*RST: STRA

Example:

```
SCONfiguration:OUTPut:MODE DIGM
SCONfiguration:APPLy
SOURce1:BB:GRAPhics:MODE PSP
SOURce1:BB:GRAPhics:FFTLen LEN2048
SOURce1:BB:GRAPhics:SOURce DO1
SOURce1:BB:GRAPhics:SOURce:MUX STRA
SOURce1:BB:GRAPhics:TRIGger:SOURce SOFT
SOURce1:BB:GRAPhics:SRATe:MODE AUTO
SOURce1:BB:GRAPhics:ADD MIN
```

Manual operation: See "Mux Stream" on page 570

[:SOURce]:BB:GRAPhics:FFTLen <Mode>

Sets the FFT size.

Parameters:

<Mode> LEN256 | LEN512 | LEN1024 | LEN2048 | LEN4096 | LEN8192

*RST: LEN2048

Example: `SOURce:BB:GRAPhics:FFTLen LEN2048`

Manual operation: See "FFT Length" on page 570

[:SOURce]:BB:GRAPhics:FFTFscale <State>

Defines the normalization of the power values in the power spectrum diagram.

Parameters:

<State> 0 | 1 | OFF | ON

1
Normalized power in dBFS

0
Shows power distribution in dB/Hz

*RST: 0

Example:

```
SOURce:BB:GRAPhics:FFTFscale PSPectrum
SOURce:BB:GRAPhics:FFTFscale 1
// the power spectrum shows the normalized power in dBFS
```

Manual operation: See "Full Scale (dBFS)" on page 570

[:SOURce<hw>]:BB:GRAPhics:TRIGger:SOURce <Source>

Defines the trigger for the starting time of the graphic recording.

Parameters:

<Source> SOFTware | MARKer

*RST: SOFTware

Example: `SOURce1:BB:GRAPhics:TRIGger:SOURce MARKer`
Sets the trigger source.

Manual operation: See "Trigger Source" on page 570

13.18.4.9 SOURce:BB:MEASurement Subsystem

This subsystem contains the commands for measuring the power values of the digital baseband signal.

Required options

See Chapter 8.2.1, "Required Options", on page 589

Example: Querying information on burst baseband signals using the multi-gated acquisition

The following example lists the commands necessary to start baseband power measurements and retrieve measurements results. We assume, that the instrument is configured as described in "To perform baseband power measurements on bursted signals using the multi-gated acquisition" on page 604.

```
// configure and enable baseband power measurements
:SOURce:BB:MEASurement:POWer:SOURce BBA
:SOURce:BB:MEASurement:POWer:OUTPut RFA
:SOURce:BB:MEASurement:POWer:ACQuisition MGATed
:SOURce:BB:MEASurement:POWer:GSOurce MGATed
:SOURce:BB:MEASurement:POWer:DURation 0.1
:SOURce:BB:MEASurement:POWer:RMODe SINGle
:SOURce:BB:MEASurement:POWer:ADD
:SOURce:BB:MEASurement:POWer:EXECute

// query status information on the initiated measurement
:SOURce:BB:MEASurement:POWer:PROGress?
// Response: 23
// the initiated measurement is in progress
:SOURce:BB:MEASurement:POWer:RState?
// Response: 1
:SOURce:BB:MEASurement:POWer:PROGress?
// Response: 100
:SOURce:BB:MEASurement:POWer:RState?
```

```
// Response: 0
// performed is one single measurment

// query baseband power values
:SOURce:BB:MEASurement:POWer:PEAK?
// Response: -24.0601192007,-24.0601192007,-24.0601192007,-24.0601192007
:SOURce:BB:MEASurement:POWer:RMS?
// Response: -35.0006514336062,-35.0006514336062,-35.0006514336062,-35.0006514336062
// performed are four sub-measurements


// stop and remove the measurement
:SOURce:BB:MEASurement:POWer:INDex 1
:SOURce:BB:MEASurement:POWer:ABORt
:SOURce:BB:MEASurement:POWer:DELete
```

[:SOURce]:BB:MEASurement:POWer:SOURce <Source>

Defines the measurement signal source.

Parameters:

<Source> BBA | BBB | BBC | BBD | BBINA | BBINB | FADINPA | FADINPB | FADINPC | FADINPD | FADOUTA | FADOUTB | FADOUTC | FADOUTD | AWGNA | AWGNB | AWGNC | AWGND | STREAMA | STREAMB | STREAMC | STREAMD

*RST: BBA

Example: See Example "Querying information on burst baseband signals using the multi-gated acquisition" on page 1019

Manual operation: See "Source/Output" on page 595

[:SOURce]:BB:MEASurement:POWer:OUTPut <Output>

Defines the output point the measurement are performed at.

Parameters:

<Output> RFA | RFB | IQOUT1 | IQOUT2 | BBMM1 | BBMM2

*RST: RFA

Example: See Example "Querying information on burst baseband signals using the multi-gated acquisition" on page 1019

Manual operation: See "Source/Output" on page 595

[:SOURce]:BB:MEASurement:POWer:ACQuisition <Acquistion>

Sets the acquisition method.

Parameters:

<Acquistion> NOMinal | CONTinuous | GATed | MGATed

*RST: CONTinuous

Example: See Example "Querying information on burst baseband signals using the multi-gated acquisition" on page 1019

Manual operation: See "Acquisition" on page 595

[:SOURce]:BB:MEASurement:POWer:GSOurce <GateSource>

Determines the marker signal defining the signal part to be evaluated. The available values depend on the selected acquisition (`[:SOURce]:BB:MEASurement:POWer:ACQuisition`).

Parameters:

<GateSource> NONE | MARK1 | MARK2 | MARK3 | MGATed

NONE
Default value for nominal and continuous acquisition.

MARK1|MARK2|MARK3
Marker signal as defined in the baseband

MGATed
Reserved for multi gated acquisition

*RST: NONE

Example: See Example "Querying information on burst baseband signals using the multi-gated acquisition" on page 1019

Manual operation: See "Gate Source" on page 595

[:SOURce]:BB:MEASurement:POWer:RMODe <RunMode>

Determines whether a single or a continuous measurement is executed.

Parameters:

<RunMode> SINGle | AUTO

*RST: AUTO

Example: See Example "Querying information on burst baseband signals using the multi-gated acquisition" on page 1019

Manual operation: See "Run Mode" on page 596

[:SOURce]:BB:MEASurement:POWer:ADD

Adds a measurement.

Example: See Example "Querying information on burst baseband signals using the multi-gated acquisition" on page 1019

Usage: Event

Manual operation: See "Add, Change, Remove, Start, Abort" on page 597

[:SOURce]:BB:MEASurement:POWer:PEAK?

Queries the peak power of the baseband signal at the measurement point determined with the command `[:SOURce]:BB:MEASurement:POWer:OUTPut`.

Return values:

<PeakPower> <Peak_SubMes#1>,<Peak_SubMes#2>,...

Returns the peak power of the measured signal or if a multi-gated acquisition is used, a string of measured values with one value per performed submeasurement

Range: -145 to 30

Example: See Example "Querying information on burst baseband signals using the multi-gated acquisition" on page 1019

Usage: Query only

Manual operation: See "Level/PEP" on page 596

[:SOURce]:BB:MEASurement:POWer:RMS?

Queries the RMS power of the baseband signal at the measurement point determined with the command `[:SOURce]:BB:MEASurement:POWer:OUTPut`.

Return values:

<RmsPower> <Power_SubMes#1>,<Level_SubMes#2>,...

Returns the power of the measured signal or if a multi-gated acquisition is used, a string of measured values with one value per performed submeasurement

Range: -145 to 30

Example: See Example "Querying information on burst baseband signals using the multi-gated acquisition" on page 1019

Usage: Query only

Manual operation: See "Level/PEP" on page 596

[:SOURce]:BB:MEASurement:POWer:RSTate?

Queries the state (running/stopped) of the current measurement.

Return values:

<RunState> 0 | 1 | OFF | ON

*RST: 0

Example: See Example "Querying information on burst baseband signals using the multi-gated acquisition" on page 1019

Usage: Query only

Manual operation: See "Running/Stopped" on page 597

[:SOURce]:BB:MEASurement:POWer:PROGress?

Queries the status of the initiated measurement. The query returns a value that indicates the task progress in percent.

Return values:

<Progress> float

Range: 0 to 100
*RST: 0

Example: See Example "Querying information on burst baseband signals using the multi-gated acquisition" on page 1019

Usage: Query only

Manual operation: See "BB Powers Summary List" on page 595

[:SOURce]:BB:MEASurement:POWer:DURation <Duration>

Sets the measurement's time of a single measurement.

Parameters:

<Duration> float

Range: 1E-3 to 5400
Increment: 1E-3
*RST: 1

Example: see Example "Querying information on burst baseband signals using the multi-gated acquisition" on page 1019

Manual operation: See "Sweep Time/Duration" on page 596

[:SOURce]:BB:MEASurement:POWer:INDex <MeasIndex>

Selects the measurement index the subsequent settings apply to, for example changing, starting or removing form the list of measurements.

Parameters:

<MeasIndex> integer

Range: 1 to dynamic
*RST: 1

Example: see Example "Querying information on burst baseband signals using the multi-gated acquisition" on page 1019

Manual operation: See "Add, Change, Remove, Start, Abort" on page 597

[:SOURce]:BB:MEASurement:POWer:CHANge

Triggers the instrument to adopt the changed measurement configuration.

Example: See Example "Querying information on burst baseband signals using the multi-gated acquisition" on page 1019

Usage: Event

Manual operation: See "Add, Change, Remove, Start, Abort" on page 597

[:SOURce]:BB:MEASurement:POWer:EXECute

Example: Triggers the instrument to perform the configured measurement.

Example: See Example "Querying information on burst baseband signals using the multi-gated acquisition" on page 1019

Usage: Event

Manual operation: See "Add, Change, Remove, Start, Abort" on page 597

[:SOURce]:BB:MEASurement:POWer:ABORt

Stops the current measurement.

Example: See Example "Querying information on burst baseband signals using the multi-gated acquisition" on page 1019

Usage: Event

Manual operation: See "Add, Change, Remove, Start, Abort" on page 597

[:SOURce]:BB:MEASurement:POWer:DELete

Removes the selected measurement from the list.

Example: See Example "Querying information on burst baseband signals using the multi-gated acquisition" on page 1019

Usage: Event

Manual operation: See "Add, Change, Remove, Start, Abort" on page 597

13.18.4.10 SOURce:BB:PROGress Subsystem General Commands

In the R&S SMW, some calculation processes take longer time. While operating the instrument manually, you can observe the status of an initiated process by the busy indicator. The following commands fulfill the same task in the remote control operation.

Example: Querying the status of the Create Waveform file process

The following is an example on how to use these commands to retrieve information about how many percent of the initiated process are completed.

```
:SCONfiguration:MODE ADVanced
:SCONfiguration:FADing MIMO4x4
:SCONfiguration:BASeband:SOURce COUPled
:SCONfiguration:APPLy
:SOURce1:BB:EUTRa:SETTing:TMOD:DL "E-TM1_1__15MHz"
:SOURce1:BB:EUTRa:SLENgth 100
```

```
:SOURce1:BB:PROGress:MCODer?
// 100 (task compleated)
:SOURce1:BB:EUTRa:STATe ON
:BB:PROGress:MCODer?
// 67 (task in progress)
:SOURce1:BB:EUTRa:WAVeform:CREate
:SOURce1:BB:PROGress:MCODer?
// 25 (task in progress)
// :SOURce1:BB:PROGress:MCODer:DM:FILTer?
// 100
```


[:SOURce<hw>]:BB:PROGress:MCODer?
[:SOURce<hw>]:BB:PROGress:MCODer:ARBitrary:MCARrier?
[:SOURce<hw>]:BB:PROGress:MCODer:ARBitrary:WSEGment?

Queries the status of an initiated process, like for example the calculation of a signal in accordance to a digital standard, or the calculation of a multi-carrier or multi-segment waveform file.

Return values:

<WSegment> integer

Indicates the task progress in percent

Range: 0 to 100
*RST: 100

Example: See Example "Querying the status of the Create Waveform file process" on page 1024

Usage: Query only

[:SOURce<hw>]:BB:PROGress:MCODer:DM:FILTer?

Queries the status of an applied offline filtering, like for example during the calculation of a waveform and a multi carrier waveform file.

Return values:

<Filter> integer

Indicates the task progress in percent

Range: 0 to 100
*RST: 100

Example: see Example "Querying the status of the Create Waveform file process" on page 1024

Usage: Query only

13.18.5 SOURce:CORRection Subsystem

The `SOURce:CORRection` subsystem contains the commands for defining correction values for external test assemblies.

You can acquire the correction values any time, regardless of the modulation settings of the generator. The correction is performed by adding the correction values to the output level of the respective RF frequency.

Determine the correction values in one of the following ways:

- Measure the RF output level at several frequency points and enter the value pairs manually in a table
- Connect an R&S NRP to the generator output signal and send the command `[:SOURce<hw>]:CORRection:CSET:DATA[:SENSor<ch>][:POWer]:SONCe` to fill the table automatically.

Correction values can be stored in files with the predefined file extension `*.uco`.

Refer to Chapter 13.7.2, "Accessing Files in the Default or in a Specified Directory", on page 777 for general information on file handling in the default and in a specific directory.

Programming example

The examples in this section assume that:

- The files are stored in the default directory.
- The same tables are accessed for both paths. File operations such as creating, deleting and querying files are therefore path-independent.
  In this case, the suffix in the keyword `SOURce` **must** be omitted. An error message is displayed if the suffix is specified.
- `*RST` does not affect data lists.

Example: Create a table with user-defined correction values for the RF level

The following example shows a command sequence to create and activate a list for assigning level correction values to arbitrary RF frequencies. Further hardware settings are not considered.

```
// Reset the instrument to start from an initial state
// Query the available user correction list files in the default directory
// Select a file or create a new one
// ******************************************************************
*RST; *CLS
SOURce1:CORRection:CSET:CATalog?
// Response: shows the name of available user correction files (if applicable)
// Select a file
SOURce1:CORRection:CSET:SELect "/var/user/ucor1"
// Create a new file (if not existing)
SOURce1:CORRection:CSET:SELect "/var/user/ucor2"

// Enter the frequency/level value pairs in the table;
```

```
// existing data is overwritten
// Query the number of frequency/power entries in the selected list
SOURce1:CORRection:CSET:DATA:FREQuency 100MHz,110MHz,120MHz,130MHz,140MHz,150MHz
SOURce1:CORRection:CSET:DATA:POWer -10,-7.5,-5.0,-2.5,0,2.5
SOURce1:CORRection:CSET:DATA:FREQuency:POINts?
// 6
SOURce1:CORRection:CSET:DATA:POWer:POINTs?
// 6

// Enable user correction mode and RF output
SOURce1:CORRection:STATe 1
OUTPut1:STATe ON

// Query the currently used correction value
SOURce1:CORRection:VALue?
// -2.5

// Delete a user correction file
SOURce1:CORRection:CSET:DELEte "/var/user/ucor1.uco"
```

Example: Fill user correction data with sensor

The following example shows a command sequence to fill a user correction list automatically supported by a connected R&S NRP.

```
// Fill a user correction list with the level values
// measured by an R&S NRP,
// store the data in a file and enable multi level user correction.

*RST; *CLS

SOURce1:CORRection:CSET:SELect "/var/user/Ucor1_AutoFill.uco"
SOURce1:CORRection:CSET:DATA:FREQuency 100MHz,110MHz,120MHz,130MHz,140MHz,150MHz
SOURce1:CORRection:ZERoing:STATe 1
SOURce1:CORRection:CSET:DATA:SENSor1:POWer:SONCe
// Query the number of automatically filled correction level values
SOURce1:CORRection:CSET:DATA:POWer:POINTs?
// 6
SOURce1:CORRection:STATe 1

// Query the correction value at a certain frequency
FREQ 120000000
SOURce1:CORRection:VALue?
// -52.13
```

Example: User correction data exchange

The following example shows a command sequence to export a user correction list (here the list created with the example before) into an ASCII file. Further hardware settings are not considered.

```
// Select a user correction file for exporting to file in ASCII format
// Set ASCII data parameters
// Set the ASCII file extension, the decimal separator
// and the column separator for the ASCII data
SOURce1:CORRection:DEXChange:AFILe:CATalog?
// my_ucor
SOURce1:CORRection:CSET:CATalog?
// ucor1,Ucor1_AutoFill
SOURce1:CORRection:CSET:SELect "/var/user/Ucor1_AutoFill.uco"
SOURce1:CORRection:DEXChange:AFILe:EXTension CSV
SOURce1:CORRection:DEXChange:AFILe:SEParator:DECimal DOT
SOURce1:CORRection:DEXChange:AFILe:SEParator:COLumn COMMa

// Select source and destination
SOURce1:CORRection:DEXChange:AFILe:SELect "/var/user/ucor2ASCII"

// Export the user correction data into the ASCII file
SOURce1:CORRection:DEXChange:MODE EXPort
SOURce1:CORRection:DEXChange:EXECute

// Query the available ASCII files with extension .csv
SOURce1:CORRection:DEXChange:AFILe:CATalog?
// ucor2ASCII,my_ucor

// Import a user correction ASCII file
SOURce1:CORRection:DEXChange:MODE IMPort
SOURce1:CORRection:DEXChange:AFILE:SELect "/var/user/my_ucor"
SOURce1:CORRection:DEXChange:EXECute
```


13.18.5.1 Correction Settings

[:SOURce<hw>]:CORRection:CSET:DATA:FREQuency <Frequency>

Enters the frequency value in the table selected with `[:SOURce<hw>]:CORRection:CSET[:SELect]`.

Parameters:

<Frequency> Frequency#1[, Frequency#2, ...]

String of values with default unit Hz.

Example: See Example "Create a table with user-defined correction values for the RF level" on page 1026 .

Manual operation: See "Edit List Mode Data" on page 487

[:SOURce<hw>]:CORRection:CSET:DATA:POWer <Power>

Enters the level values to the table selected with `[:SOURce<hw>]:CORRection:CSET[:SELect]`.

Parameters:

<Power>	Power#1[, Power#2, ...] String of values with default unit dB.
Example:	See Example "Create a table with user-defined correction values for the RF level" on page 1026 .
Manual operation:	See "Edit List Mode Data" on page 487

[:SOURce<hw>]:CORRection:CSET:DATA:FREQuency:POINts?
[:SOURce<hw>]:CORRection:CSET:DATA:POWer:POINts?

Queries the number of frequency/level values in the selected table.

Return values:

<Points>	integer Range: 0 to 10000 *RST: 0
Example:	See Example "Create a table with user-defined correction values for the RF level" on page 1026 .
Usage:	Query only

[:SOURce<hw>]:CORRection:CSET:DATA[:SENSor<ch>][:POWer]:SONCe

Fills the selected user correction table with the level values measured by the power sensor for the given frequencies.

To select the used power sensor set the suffix in key word `SENSe`.

Example:	See Example "Fill user correction data with sensor" on page 1027.
Usage:	Event
Manual operation:	See " Fill User Correction Data with Sensor" on page 537

[:SOURce<hw>]:CORRection:CSET[:SELect] <Filename>

Selects or creates a file for the user correction data.

If the file with the selected name does not exist, a new file is created.

Parameters:

<Filename>	string Filename or complete file path; file extension can be omitted.

Example: See Example "Create a table with user-defined correction values for the RF level" on page 1026 .

Manual operation: See "User Cor. Data" on page 532

[:SOURce<hw>]:CORRection:VALue?

Queries the current value for user correction.

Return values:

<Value> float

Range: -100 to 100
Increment: 0.01
*RST: 0

Example: See Example "Create a table with user-defined correction values for the RF level" on page 1026 .

Usage: Query only

Manual operation: See "User Correction" on page 531

[:SOURce<hw>]:CORRection:ZERoing:STATe <State>

Activates the zeroing procedure before filling the user correction data acquired by a sensor.

Parameters:

<State> 0 | 1 | OFF | ON

*RST: 1

Example: See Example "Fill user correction data with sensor" on page 1027.

Manual operation: See " Fill User Correction Data with Sensor" on page 537

[:SOURce<hw>]:CORRection:CSET:DATA[:SENSor<ch>][:POWer]:SONCe

Fills the selected user correction list with the level values measured by the power sensor for the given frequencies.

Suffix:

SENSor<ch> Defines the used power sensor, i.e. the sensor whose values are used.

Example: See Example "Fill user correction data with sensor" on page 1027.

Usage: Event

[:SOURce<hw>]:CORRection[:STATe] <State>

Activates user correction with the currently selected table.

Parameters:

<State> 0 | 1 | OFF | ON

*RST: 0

Example: See Example "Create a table with user-defined correction values for the RF level" on page 1026 .

Manual operation: See "State" on page 531

[:SOURce]:CORRection:CSET:CATalog?

Queries a list of available user correction tables.

Return values:

<Catalog> string

List of list filenames, separated by commas

Example: See Example "Create a table with user-defined correction values for the RF level" on page 1026 .

Usage: Query only

Manual operation: See "User Cor. Data" on page 532

[:SOURce]:CORRection:CSET:DELete <Filename>

Deletes the specified user correction list file.

Setting parameters:

<Filename> string

Filename or complete file path; file extension is optional.

Example: See Example "Create a table with user-defined correction values for the RF level" on page 1026 .

Usage: Setting only

Manual operation: See "User Cor. Data" on page 532

13.18.5.2 SGamma Correction Data

The signal generator can perform a measured-value correction taking the complex reflection coefficient (source gamma) of the signal source into account. The state must be activated.

[:SOURce<hw>]:CORRection:CSET:DATA:POWer:SGAMma:STATe <State>

Activates the auto fill process to create a list with SGamma correction data.

SGamma correction is possible when the table is filled with multi level user correction data.

Parameters:

<State> 0 | 1 | OFF | ON

*RST: 0

Example: `CORR:CSET:DATA:POW:SGAM:STAT ON`

Activates the acquisition of multi level gamma correction data with a connected power sensor.

[:SOURce<hw>]:CORRection:CSET:DATA:SGAMma:DELete

Deletes SGamma correction values in the selected table.

Example: `CORR:CSET:DATA:SGAM:DEL "/var/user/temp/multilevelucor1"`

Usage: Event

[:SOURce<hw>]:CORRection:CSET:DATA:SGAMma:MAGNitude <Magnitude1>[, <MagnitudeN>]

Sets the magnitude of the reflection coefficient of the DUT for SGamma correction (multi level user correction data).

Parameters:

<Magnitude1> float

Range: 0.0 to 1.0
Increment: 0.01

<MagnitudeN> float

Range: 0.0 to 1.0
Increment: 0.01

Example: `CORR:CSET:DATA:POW:SGAM:MAG 0.1,0.2,0.3,0.4`

Sets the magnitude values.

[:SOURce<hw>]:CORRection:CSET:DATA:SGAMma:MAGNitude:POINts?

Queries the number of magnitude entries in the selected multi level user correction table.

Return values:

<Points> integer

Example:

```
CORR:CSET "/var/user/temp/multilevelucor"
CORR:CSET:DATA:SGAM:MAGN:POIN?
// 40
// the table contains 40 magnitude values
```

Usage: Query only

[:SOURce<hw>]:CORRection:CSET:DATA:SGAMma:PHASe <Phase1>[, <PhaseN>]

Sets the phase angle of the reflection coefficient of the DUT for SGamma correction (multi level user correction data).

Parameters:

<Phase1> float

Range: -360 to 360
Increment: 0.1

<PhaseN> float

Range: -360 to 360
Increment: 0.1

Example:
```
CORR:CSET:DATA:POW:SGAM:PHAS
-360.0,-180.0,180.0,360.0
```
Sets the phase values.

[:SOURce<hw>]:CORRection:CSET:DATA:SGAMma:PHASe:POINts?

Queries the number of phase entries in the selected multi level user correction table.

Return values:

<Points> integer

Example:
```
CORR:CSET "/var/user/temp/multilevelucor"
CORR:CSET:DATA:SGAM:PHAS:POIN?
// 40
// the table contains 40 phase values
```

Usage: Query only

13.18.5.3 Correction Data Exchange

With the following commands, you can configure user correction lists and export or import them accordingly.

[:SOURce<hw>]:CORRection:DEXChange:AFILe:CATalog?

Queries the available ASCII files for export or import of user correction data in the current or specified directory.

Return values:

<Catalog> string

List of ASCII files `*.txt` or `*.csv`, separated by commas.

Example: See Example "Create a table with user-defined correction values for the RF level" on page 1026 .

Usage: Query only

Manual operation: See "Select Source/Select Destination" on page 491

[:SOURce<hw>]:CORRection:DEXChange:AFILe:EXTension <Extension>

Determines the extension of the ASCII files for file import or export, or to query existing files.

Parameters:

<Extension> TXT | CSV

*RST: TXT

Example: See Example "User correction data exchange" on page 1027 .

Manual operation: See "ASCII File Settings" on page 490

[:SOURce<hw>]:CORRection:DEXChange:AFILe:SELect <Filename>

Selects the ASCII file to be imported or exported.

Parameters:

<Filename> string

Filename or complete file path; file extension can be omitted.

Example: See Example "User correction data exchange" on page 1027 .

Manual operation: See "Select Source/Select Destination" on page 491

[:SOURce<hw>]:CORRection:DEXChange:AFILe:SEParator:COLumn <Column>

Selects the separator between the frequency and level column of the ASCII table.

Parameters:

<Column> TABulator | SEMicolon | COMMa | SPACe

*RST: COMMa

Example: See Example "User correction data exchange" on page 1027 .

Manual operation: See "ASCII File Settings" on page 490

[:SOURce<hw>]:CORRection:DEXChange:AFILe:SEParator:DECimal <Decimal>

Sets the decimal separator used in the ASCII data between '.' (decimal point) and ',' (comma) with floating-point numerals.

Parameters:

<Decimal> DOT | COMMa

*RST: DOT

Example: See Example "User correction data exchange" on page 1027 .

Manual operation: See "ASCII File Settings" on page 490

[:SOURce<hw>]:CORRection:DEXChange:EXECute

Executes the import or export of the selected correction list, according to the previously set transfer direction with command `[:SOURce<hw>]:CORRection:DEXChange:MODE`.

Example: See Example "User correction data exchange" on page 1027 .

Usage: Event

Manual operation: See "Import / Export" on page 491

[:SOURce<hw>]:CORRection:DEXChange:MODE <Mode>

Determines import or export of a user correction list.

Specify the source or destination file with the command `[:SOURce<hw>]:CORRection:DEXChange:SELect`.

Parameters:

<Mode> IMPort | EXPort

*RST: IMPort

Example: See Example "User correction data exchange" on page 1027 .

Manual operation: See " Mode " on page 490

[:SOURce<hw>]:CORRection:DEXChange:SELect <Filename>

Selects the ASCII file for import or export, containing a user correction list.

Parameters:

<Filename> string

Filename or complete file path; file extension can be omitted.

Example: See Example "User correction data exchange" on page 1027 .

Manual operation: See "Select Source / Select ASCII Destination" on page 491

13.18.5.4 Frequency Response Commands

This section lists the frequency response commands related to the I/Q modulator.

Using user-defined frequency response correction requires option R&S SMW-K544.

See user manual R&S®SMW-K544 User-Defined Frequency Response Correction.

Example: Enabling high-quality optimization

```
SOURce1:CORRection:FRESponse:RF:OPTimization:MODE QHIG
SOURce1:CORRection:FRESponse:RF:OPTimization:BANDwidth:MODE AUTO
SOURce1:CORRection:FRESponse:RF:OPTimization:BANDwidth:VALue?
// 160000000
```


[:SOURce<hw>]:CORRection:FRESponse:RF:OPTimization:MODE <FreqRespOptMode>

Sets the optimization mode.

The value selected here is used also as optimization mode in the I/Q modulator, and vice versa.

For details, see R&S SMW User Manual.

Parameters:

<FreqRespOptMode> FAST | QHIGh | QHTable

FAST
Optimization by compensation for I/Q skew.

QHTable
Enabled if R&S SZU is connected to the R&S SMW and the connection is active.
Improved optimization by maintained speed.

QHIGh
Optimization by compensation for I/Q skew and frequency response correction.
This mode interrupts the RF signal. Do not use it in combination with the uninterrupted level settings and strictly monotone modes RF level modes

*RST: QHIGh / QHTable (if R&S SZU connected to R&S SMW and the connection is active)

Example: See Example "Enabling high-quality optimization" on page 1036.

Manual operation: See "Optimization Mode" on page 421

[:SOURce<hw>]:CORRection:FRESponse:RF:OPTimization:BANDwidth:MODE <FreqRespOptBwMo>

For `[:SOURce<hw>]:CORRection:FRESponse:RF:OPTimization:MODE QHIG|QHT`, sets how the signal bandwidth is estimated: automatically or manually with the command `[:SOURce<hw>]:CORRection:FRESponse:RF:OPTimization:BANDwidth[:VALue]`.

Parameters:
<FreqRespOptBwMo> AUTO | MANual

*RST: AUTO

Example: See Example "Enabling high-quality optimization" on page 1036.

Manual operation: See "Optimization Bandwidth" on page 423

[:SOURce<hw>]:CORRection:FRESponse:RF:OPTimization:BANDwidth[:VALue] <FreqRespOptBwVa>

Sets the signal compensation bandwidth for `[:SOURce<hw>]:CORRection:FRESponse:RF:OPTimization:BANDwidth:MODE` `MAN`.

Parameters:
<FreqRespOptBwVa> float

Example: See Example "Enabling high-quality optimization" on page 1036.

Manual operation: See "Bandwidth" on page 423

13.18.6 SOURce:FREQuency Subsystem

The `SOURce:FREQuency` subsystem contains the commands used to define the frequency settings for the RF sources and sweeps.

Example: Frequency configuration

```
SOURce1:FREQuency:MODE CW
SOURce1:FREQuency:CW 6000000000
SOURce1:FREQuency:OFFSet 2000000000
SOURce1:FREQuency:MULTiplier 1.5
SOURce1:FREQuency:CW?
// 11000000000

// SOURce1:FREQuency:STEP:MODE USER
// SOURce1:FREQuency:STEP:INCRement 1000000
// SOURce1:FREQuency:CW UP

SOURce1:PHASe 2
SOURce1:PHASe:REFerence
```

Example: Set up of LO coupling and LO level adjustment with three Rohde & Schwarz signal generators

This example shows the command sequences for coupling the oscillator signal of three signal generators connected according to the setup on Figure 13-14.

Connect the instruments in a daisy-chain regarding the LO signal:

- Connect the LO OUT of the first generator with LO IN of the second
- Connect the LO OUT of the second generator with LO IN of the third.

The frequency is assumed to be 5 GHz.

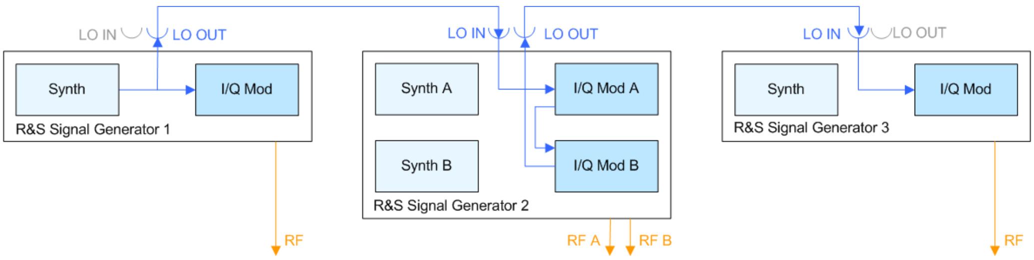


Figure 13-14: Set up of LO coupling with three Rohde & Schwarz signal generators

Signal generator#1 = One-path instrument, uses its internal oscillator signal.
Signal generator#1 = Two-paths instrument; receives the LO signal from the first instrument and assigns it to both paths
Signal generator#3 = One-path instrument; receives the LO signal from the second instrument

```
// ******************************************************************
// Address and reset the first instrument
// ******************************************************************
*RST; *CLS

// ******************************************************************
// Configure the oscillator signal of the first instrument
// ******************************************************************
SOURce:FREQuency 5 GHz
SOURce:FREQuency:LOSCillator:MODe INTernal
SOURce:FREQuency:LOSCillator:OUTPut:STATe ON
SOURce:FREQuency:LOSCillator:OUTPut:FREQuency?
// Response: 1000000000

// ******************************************************************
// Address and reset the second instrument
// ******************************************************************
*RST; *CLS

// ******************************************************************
// Configure the oscillator signal of the second instrument
// ******************************************************************
SOURce1:FREQuency 5 GHz
SOURce1:FREQuency:LOSCillator:MODe ECOupled
SOURce1:FREQuency:LOSCillator:OUTPut:STATe ON
```

```
// ******************************************************************
// If the attenuation between the interconnection LO OUT and LO IN is
// greater than 1 dB at the oscillator frequency,
// we recommend that you perform the LO level adjustment
// ******************************************************************
:CALibration:LOSCillator:Coupling:LOCAL?

// ******************************************************************
// Address and reset the third instrument
// ******************************************************************
*RST; *CLS

// ******************************************************************
// Configure the oscillator signal of the third instrument
// ******************************************************************
SOURce:FREQuency 5 GHz
SOURce:FREQuency:LOSCillator:MODe EXTernal
SOURce:FREQuency:LOSCillator:OUTPut:STATe OFF

// ******************************************************************
// If the attenuation between the interconnection LO OUT and LO IN is
// greater than 1 dB at the oscillator frequency,
// we recommend that you perform the LO level adjustment
// ******************************************************************
:CALibration:LOSCillator:Coupling:LOCAL?
```

See also:

- "To configure LO coupling with cascaded instruments to achieve phase coherence" on page 454
- "Level adjustment" on page 449

[:SOURce<hw>]:FREQuency:LOSCillator:INPut:FREQuency?

Queries the required external reference frequency.

Return values:

<Frequency> float

Range: 100E3 to 20E9
Increment: 0.01
*RST: 0

Example: See Example "Set up of LO coupling and LO level adjustment with three Rohde & Schwarz signal generators" on page 1038.

Usage: Query only

Manual operation: See "Mandatory LO IN Frequency" on page 452

[:SOURce<hw>]:FREQuency:LOSCillator:MODE <Mode>

Selects the mode of the local oscillator coupling.

Table 13-11: Cross-reference between <Mode> and the manual operation

`<Mode>`	**Parameter in manual operation**	**Description**
INTernal	A&B Internal / Internal (one path instrument)	Uses the internal oscillator signal in both paths.
EXTernal	A External & B Internal (one path instrument)	Uses an external signal in path A. B uses its internal signal.
COUPled	A Internal & A->B Coupled	Assigns the internal oscillator signal of path A also to path B.
ECOupled	A External & A->B Coupled	Assigns an externally supplied signal to both paths.
BOFF	A Internal & B RF Off	Uses the internal local oscillator signal of path A, if the selected frequency exceeds the maximum frequency of path B.
EBOFf	A External & B RF Off	Uses the LO IN signal for path A, if the selected RF frequency exceeds the maximum frequency of path B.
AOFF	A RF Off & B External	Uses the LO IN signal for path B, if the selected RF frequency exceeds the maximum frequency of path A.

Parameters:

<Mode> INTernal | EXTernal | COUPled | ECOupled | BOFF | AOFF

See Table 13-11

*RST: INTernal

Example: See Example "Set up of LO coupling and LO level adjustment with three Rohde & Schwarz signal generators" on page 1038.

Manual operation: See "Mode" on page 451

[:SOURce<hw>]:FREQuency:LOSCillator:OUTPut:FREQuency?

Queries the current frequency of the local oscillator at the LO OUT connector.

Return values:

<Frequency> float

Range: 100E3 to 20E9
Increment: 0.01
*RST: 0

Example: See Example "Set up of LO coupling and LO level adjustment with three Rohde & Schwarz signal generators" on page 1038.

Usage: Query only

Manual operation: See "LO OUT Frequency" on page 452

[:SOURce<hw>]:FREQuency:LOSCillator:OUTPut:STATe <State>

Activates the LO output in path B.

Parameters:

<State> 0 | 1 | OFF | ON

*RST: 0

Example: See Example "Set up of LO coupling and LO level adjustment with three Rohde & Schwarz signal generators" on page 1038.

Manual operation: See "Out State " on page 452

[:SOURce<hw>]:FREQuency:MODE <Mode>

Sets the frequency mode for generating the RF output signal. The selected mode determines the parameters to be used for further frequency settings.

Parameters:

<Mode> CW | FIXed | SWEep | LIST

CW|FIXed
Sets the fixed frequency mode. `CW` and `FIXed` are synonyms.
The instrument operates at a defined frequency, set with command `[:SOURce<hw>]:FREQuency[:CW|FIXed]`.

SWEep
Sets sweep mode.
The instrument processes frequency (and level) settings in defined sweep steps.
Set the range and current frequency with the commands:
`[:SOURce<hw>]:FREQuency:STARt` on page 1045 and `[:SOURce<hw>]:FREQuency:STOP` on page 1046,
`[:SOURce<hw>]:FREQuency:CENTer` on page 1045,
`[:SOURce<hw>]:FREQuency:SPAN` on page 1045,
`[:SOURce<hw>]:FREQuency:MANual` on page 1043

LIST
Sets list mode.
The instrument processes frequency and level settings by means of values loaded from a list.
To configure list mode settings, use the commands of the Chapter 13.18.11, "SOURce:LIST Subsystem", on page 1073.

*RST: CW

Example: See Example "Frequency configuration" on page 1037

Example: See Example "Setup an RF frequency or power sweep" on page 1107

Manual operation: See "State (RF frequency sweep)" on page 476

[:SOURce<hw>]:FREQuency[:CW|FIXed] <Fixed>

Sets the frequency of the RF output signal in the selected path.

The effect depends on the selected mode:

- In CW mode (FREQ:MODE CW | FIXed), the instrument operates at a fixed frequency.
- In sweep mode (FREQ:MODE SWE), the value applies to the sweep frequency. The instrument processes the frequency settings in defined sweep steps.
- In user mode (FREQ:STEP:MODE USER), you can vary the current frequency step by step.

Parameters:

<Fixed> float

The following settings influence the value range:
An offset set with the command `[:SOURce<hw>]:FREQuency:OFFSet`

Numerical value
Sets the frequency in CW and sweep mode

UP|DOWN
Varies the frequency step by step in user mode.
The frequency is increased or decreased by the value set with the command `[:SOURce<hw>]:FREQuency:STEP[:INCRement]`.

Range: (RFmin + OFFSet) to (RFmax + OFFSet)
*RST: 100 MHz

Example: See Example "Frequency configuration" on page 1037

Example: See Example "Setup an RF frequency or power sweep" on page 1107

Manual operation: See "Frequency" on page 431

[:SOURce<hw>]:FREQuency[:CW|FIXed]:RCL <Rcl>

Set whether the RF frequency value is retained or taken from a loaded instrument configuration, when you recall instrument settings with command `*RCL`.

The selected mode applies to both RF paths, i.e. a specified suffix is ignored.

Parameters:

<Rcl> INCLude | EXCLude

INCLude
Takes the frequency value of the loaded settings.

EXCLude
Retains the current frequency when an instrument configuration is loaded.

*RST: INCLude

Example: `SOURce1:FREQuency:CW:RCL INCLude`

Manual operation: See "Exclude Frequency" on page 632

[:SOURce<hw>]:FREQuency:MANual <Manual>

Sets the frequency and triggers a sweep step manually if SWEep:MODE MAN.

Parameters:

<Manual> float

You can select any frequency within the setting range, where:
`STARt` is set with `[:SOURce<hw>]:FREQuency:STARt`
`STOP` is set with `[:SOURce<hw>]:FREQuency:STOP`
`OFFSet` is set with `[:SOURce<hw>]:FREQuency:OFFSet`

Range: (STARt + OFFSet) to (STOP + OFFSet)
Increment: 0.01Hz
*RST: 100 MHz
Default unit: Hz

Example: See Example "Setup an RF frequency or power sweep" on page 1107

Manual operation: See " Current Frequency" on page 478

[:SOURce<hw>]:FREQuency:MULTiplier <Multiplier>

Sets the multiplication factor $N_{FREQ:MULT}$ of a subsequent downstream instrument.

The parameters offset $f_{FREQ:OFFSer}$ and multiplier $N_{FREQ:MULT}$ affect the frequency value set with the command FREQ.

The query FREQ? returns the value corresponding to the formula:

$f_{FREQ} = f_{RFout} * N_{FREQ:MULT} + f_{FREQ:OFFSer}$

See Chapter 7.3.4.1, "Displayed RF Frequency and Level Values with Downstream Instruments", on page 440.

Parameters:

<Multiplier> float

Range: 1 to dynamic
Increment: 0.001
*RST: 1

Example: See Example "Frequency configuration" on page 1037

Manual operation: See "Multiplier" on page 432

[:SOURce<hw>]:FREQuency:OFFSet <Offset>

Sets the frequency offset $f_{FREQ:OFFSet}$ of a downstream instrument.

The parameters offset $f_{FREQ:OFFSer}$ and multiplier $N_{FREQ:MULT}$ affect the frequency value set with the command FREQ.

The query FREQ? returns the value corresponding to the formula:

$f_{FREQ} = f_{RFout} * N_{FREQ:MULT} + f_{FREQ:OFFSer}$

See Chapter 7.3.4.1, "Displayed RF Frequency and Level Values with Downstream Instruments", on page 440.

Note: The offset also affects RF frequency sweep.

Parameters:

<Offset> float

Increment: 0.01
*RST: 0

Example: See Example "Frequency configuration" on page 1037

Manual operation: See "Offset" on page 432

[:SOURce<hw>]:FREQuency:CENTer <Center>

Sets the center frequency of the sweep.

See Chapter 7.6.2.1, "Correlating Parameters in Sweep Mode", on page 467.

Parameters:

<Center> float

Range: 300 kHz to RFmax
Increment: 0.01 Hz
*RST: 300E6
Default unit: Hz

Example: See Example "Setup an RF frequency or power sweep" on page 1107

Manual operation: See " Center Frequency " on page 478

[:SOURce<hw>]:FREQuency:SPAN <Span>

Sets the sapn of the frequency sweep range.

See Chapter 7.6.2.1, "Correlating Parameters in Sweep Mode", on page 467.

Parameters:

<Span> float

Full freqeuncy range

Increment: 0.01
*RST: 400E6

Example: See Example "Setup an RF frequency or power sweep" on page 1107

Manual operation: See "Span" on page 478

[:SOURce<hw>]:FREQuency:STARt <Start>

Sets the start frequency for the RF sweep.

See Chapter 7.6.2.1, "Correlating Parameters in Sweep Mode", on page 467.

Parameters:

<Start> float

Range: 300kHz to RFmax
Increment: 0.01Hz
*RST: 100 MHz

Example: See Example "Setup an RF frequency or power sweep" on page 1107

Manual operation: See " Start Frequency/Stop Frequency " on page 478

[:SOURce<hw>]:FREQuency:STOP <Stop>

Sets the stop frequency range for the RF sweep.

See Chapter 7.6.2.1, "Correlating Parameters in Sweep Mode", on page 467.

Parameters:

<Stop> float

Range: 300kHz to RFmax
Increment: 0.01Hz
*RST: 500 MHz
Default unit: Hz

Example: See Example "Setup an RF frequency or power sweep" on page 1107

Manual operation: See " Start Frequency/Stop Frequency " on page 478

[:SOURce<hw>]:FREQuency:STEP:MODE <Mode>

Defines the type of step size to vary the RF frequency at discrete steps with the commands FREQ UP or FREQ DOWN.

You can define the step mode for each path separately.

Parameters:

<Mode> DECimal | USER

DECimal
Increases or decreases the level in steps of ten.

USER
Increases or decreases the level in increments, set with the command FREQ:STEP[:INCR].

*RST: DECimal

Example:

```
// increasing the RF frequency with a step size of 50 KHz
SOURce1:FREQuency:STEP 50E3
SOURce1:FREQuency:STEP:MODE USER
SOURce1:FREQuency:CW UP
```

Manual operation: See "Variation Active" on page 433

[:SOURce<hw>]:FREQuency:STEP[:INCRement] <Increment>

Sets the step width.

You can use this value to vary the RF frequency with command FREQ UP or FREQ DOWN, if you have activated FREQ:STEP:MODE USER.

Note: This value also applies to the step width of the rotary knob on the instrument and, in user-defined step mode, increases or decreases the frequency.

Parameters:

<Increment> float

Range: 0 Hz to RFmax - 100 kHz
Increment: 0.01 Hz
*RST: 1E6

Example: See Example "Frequency configuration" on page 1037

Manual operation: See "Variation Step" on page 433

13.18.7 SOURce:INPut Subsystem

The `SOURce:INPut` subsystem contains the commands for configuring the inputs for external modulation signals. The instrument trigger setting influences all sweeps and is effective in the List mode (Instrument Trigger).

[:SOURce<hw>]:INPut:MODext:COUPling<ch> <Coupling>

Selects the coupling mode for an externally applied modulation signal.

Parameters:

<Coupling> AC | DC

AC
Passes the AC signal component of the modulation signal.

DC
Passes the modulation signal with both components, AC and DC.

*RST: AC

Example: `INP:MOD:COUP AC`
Selects the coupling mode AC for an externally applied modulation signal.

Manual operation: See "Coupling (AC/DC)" on page 505

[:SOURce<hw>]:INPut:MODext:IMPedance<ch> <Impedance>

Sets the impedance (50 kOhm or High = 100 kOhm to ground) for the externally supplied modulation signal.

Parameters:

<Impedance> G50 | HIGH

*RST: HIGH

Example: `INP:MOD:IMP 50`
sets 50 kOhm to ground.

Manual operation: See "Impedance" on page 505

[:SOURce<hw>]:INPut:TRIGger:SLOPe <Slope>

Sets the polarity of the active slope of an applied instrument trigger.

Parameters:

<Slope> NEGative | POSitive

*RST: POSitive

Manual operation: See "Trigger Slope" on page 480

13.18.8 SOURce:IQ Subsystem

The `SOURce:IQ` subsystem contains the commands for configuring the I/Q modulation.

Required options

See Chapter 6.1, "Required Options", on page 415.

[:SOURce<hw>]:IQ:SOURce <Source>

Selects the input signal source for the I/Q modulator.

Parameters:

<Source> BASeband | ANALog | DIFFerential

`DIFFerential` requires option R&S SMW-K739
You can feed `BASeband` as well as `ANALog` into both paths, while `DIFFerential` is provided in path A only.
Both, the external analog and external differential signals, disable the amplitude modulation, an enabled custom digital modulation, any configured digital standard or an applied digital baseband input signal.

*RST: BASeband

Example: `SOURce1:IQ:SOURce ANALog`
selects an external analog signal as the input signal. The signal must be applied at the inputs I and Q.

Manual operation: See "Source" on page 418

[:SOURce<hw>]:IQ:STATe <State>

Enables/disables the I/Q modulation.

Parameters:

<State> 0 | 1 | OFF | ON

*RST: 0

Example: `IQ:STAT ON`
activates I/Q modulation.

Manual operation: See "State" on page 418

[:SOURce<hw>]:IQ:GAIN <Gain>
[:SOURce<hw>]:BB:IQGain <IqGain>

Optimizes the modulation of the I/Q modulator for a subset of measurement requirement. Provided is a dynamic range of 16 dB divided into 2 dB steps.

Parameters:

<IqGain> DBM4 | DBM2 | DB0 | DB2 | DB4 | DB8 | DB6

DB0|DB2|DB4|DB6|DB8
Activates the specified gain of 0 dB, +2 dB, +4 dB, +6 dB, +8 dB

DBM2|DBM4
Activates the specified gain of -2 dB, -4 dB

DBM3|DB3
(setting only)
Provided only for backward compatibility with other Rohde & Schwarz signal generators.
The R&S SMW accepts these values and maps them automatically as follows:
DBM3 = DBM2, DB3 = DB2

AUTO
The gain value is retrieved form the connected R&S SZU. The I/Q modulator is configured automatically.

*RST: DB4

Example:
```
SOURce1:BB:IQGain DB2
SOURce1:IQGain DB2
```

Manual operation: See "(Baseband) Gain" on page 420

[:SOURce<hw>]:IQ:CRESTfactor <CrestFactor>

Specifies the crest factor for the external analog signal.

Parameters:

<CrestFactor> float

Range: 0 to 35
Increment: 0.01
*RST: 0
Default unit: dB

Example:
```
IQ:CRES 10
```
specifies 10 dB crest factor for the external analog signal.

Manual operation: See "Crest Factor" on page 420

[:SOURce<hw>]:IQ:SWAP[:STATe] <State>

Swaps the I and Q channel.

Parameters:

<State> 0 | 1 | OFF | ON

*RST: 0

Example:
```
IQ:SWAP ON
```
swaps the I and Q channel.

Manual operation: See "I/Q Swap" on page 419

[:SOURce<hw>]:IQ:WBSTate <WbState>

Activates wideband mode.

This setting automatically optimizes the settings for wideband modulation signals (>5 MHz, State ON).

Parameters:

<WbState> 0 | 1 | OFF | ON

*RST: 0

Example: `IQ:WBST ON`

switches to wideband mode.

Manual operation: See "I/Q Wideband" on page 420

13.18.9 SOURce:IQ:OUTPut Subsystem

The `SOURce:IQ:OUTPut` subsystem contains the commands for configuring the analog and digital I/Q output signals.

For information about the required options, see:

- Chapter 3.7, "I/Q Digital Output Settings", on page 178
- Chapter 3.8, "I/Q Analog Output Settings", on page 185

Refer to Table 3-4 for information on the connector and interfaces the analog and digital I/Q signals are output at.

13.18.9.1 SOURce:IQ:OUTPut:ANALog Commands

This section describes the commands of the output of an analog I/Q signal.

Example: Enabling and configuring the analog output

```
SCONfiguration:OUTPut:MAPPing:IQOutput1:STReam1:STATe 1

SOURce1:IQ:OUTPut:ANALog:PRESet
SOURce1:IQ:OUTPut:ANALog:TYPE DIFF
SOURce1:IQ:OUTPut:ANALog:MODE VAR
SOURce1:IQ:OUTPut:LEVel 2
SOURce1:IQ:OUTPut:ANALog:BIAS:COUPling:STATe 1
SOURce1:IQ:OUTPut:ANALog:BIAS:I 1
SOURce1:IQ:OUTPut:ANALog:BIAS:Q?
// 1
SOURce1:IQ:OUTPut:ANALog:OFFSet:I 2
SOURce1:IQ:OUTPut:ANALog:OFFSet:Q 2.5
SOURce1:IQ:OUTPut:ANALog:STATe 1
```

[:SOURce<hw>]:IQ:OUTPut:ANALog:STATe <State>

Activates the specified analog I/Q output.

Note: By default, the output connectors I/Q OUT X are deactivated.

Suffix:

:SOURce<hw> 1|2
Selects the I/Q OUT connectors

Parameters:

<State> 0 | 1 | OFF | ON
*RST: 0

Example: `SOURce:IQ:OUTPut:ANALog:STATe ON`
Activates the output of the analog I/Q signal on the I/Q OUT 1 connectors.

Manual operation: See "State" on page 187

[:SOURce<hw>]:IQ:OUTPut[:ANALog]:PRESet

Sets the default settings (`*RST` values specified for the commands).

Not affected are:

- The state set with the command `[:SOURce<hw>]:IQ:OUTPut:ANALog:STATe`.
- If `SCONfiguration:EXTernal:PBEHaviour 1`, the I/Q ouptput type set with the command `[:SOURce<hw>]:IQ:OUTPut[:ANALog]:TYPE`.

Example: See Example "Enabling and configuring the analog output" on page 1051.

Usage: Event

Manual operation: See "Set to Default" on page 187

[:SOURce<hw>]:IQ:OUTPut[:ANALog]:SETTing:CATalog?

Queries the files with I/Q output settings in the default directory. Listed are files with the file extension `*.iqout`.

Return values:

<Catalog> "<filename1>,<filename2>,..."

Returns a string of file names separated by commas.

Usage: Query only

Manual operation: See "Save/Recall" on page 188

[:SOURce<hw>]:IQ:OUTPut[:ANALog]:SETTing:STORe <Filename>

Stores the current settings into the selected file; the file extension (`*.iqout`) is assigned automatically.

Setting parameters:

<Filename> "<filename>"

Filename or complete file path

Usage: Setting only

Manual operation: See "Save/Recall" on page 188

[:SOURce<hw>]:IQ:OUTPut[:ANALog]:SETTing:LOAD <Filename>

Loads the selected file from the default or the specified directory. Loaded are files with extension `*.iqout`.

Setting parameters:

<Filename> "<filename>"

Filename or complete file path

Usage: Setting only

Manual operation: See "Save/Recall" on page 188

[:SOURce<hw>]:IQ:OUTPut[:ANALog]:SETTing:DELete <Filename>

Deletes the selected file from the default or specified directory. Deleted are files with the file extension `*.iqout`.

Setting parameters:

<Filename> "<filename>"

Filename or complete file path

Usage: Setting only

Manual operation: See "Save/Recall" on page 188

[:SOURce<hw>]:IQ:OUTPut[:ANALog]:TYPE <Type>

Sets the type of the analog signal.

Parameters:

<Type>	SINGle | DIFFerential	
	*RST:	SING (if SCONfiguration:EXTernal:PBEHaviour 0) / - (if SCONfiguration:EXTernal:PBEHaviour 1)

Example: `SOURce1:IQ:OUTPup:ANALog:TYPE DIFFerential`

Manual operation: See "I/Q Output Type" on page 158

[:SOURce<hw>]:IQ:OUTPut[:ANALog]:MODE <Mode>

Determines the mode for setting the output parameters.

Parameters:

<Mode> FIXed | VARiable

FIXed
Locks the I/Q output settings

VARiable
Unlocks the settings

*RST: FIXed

Example: See Example "Enabling and configuring the analog output" on page 1051.

Manual operation: See "Mode" on page 189

[SOURce<hw>]:IQ:OUTPut:LEVel <Level>

Sets the off-load voltage Vp of the analog I/Q signal output.

The value range is adjusted so that the maximum overall output voltage does not exceed 4V, see "Maximum overall output voltage" on page 186.

Parameters:

<Level> float

Range:	0.04V to 4V (R&S SMW-B10) / 0.04V to 2V (R&S SMW-B9)
Increment:	1E-4
*RST:	1
Default unit:	V

Example: See Example "Enabling and configuring the analog output" on page 1051.

Manual operation: See "I/Q Level Vp EMF" on page 189

[:SOURce<hw>]:IQ:OUTPut[:ANALog]:BIAS:COUPling[:STATe] <State>

Couples the bias setting of the I and Q signal components.

Parameters:

<State> 0 | 1 | OFF | ON

*RST: 0

Example: See Example "Enabling and configuring the analog output" on page 1051.

Manual operation: See "Couple I/Q Bias" on page 189

[:SOURce<hw>]:IQ:OUTPut[:ANALog]:BIAS:I <I>
[:SOURce<hw>]:IQ:OUTPut[:ANALog]:BIAS:Q <Q>

Specifies the amplifier bias V_{bias} of the respective I or Q component.

The value range is adjusted so that the maximum overall output voltage does not exceed 4V, see "Maximum overall output voltage" on page 186.

Parameters:

<Q> float

Range: $(-4+V_p/2+V_{offset}/2)$,V to $(4-V_p/2-V_{offset}/2)$,V (R&S SMW-B10) / -0.2V to 2.5V (R&S SMW-B9)
Increment: 1E-4
*RST: 0
Default unit: V

Example: See Example "Enabling and configuring the analog output" on page 1051.

Manual operation: See "Bias (EMF)" on page 189

[:SOURce<hw>]:IQ:OUTPut[:ANALog]:OFFSet:I <I>
[:SOURce<hw>]:IQ:OUTPut[:ANALog]:OFFSet:Q <Q>

Sets an offset V_{offset} between the inverting and non-inverting input of the differential analog I/Q output signal.

The value range is adjusted so that the maximum overall output voltage does not exceed 4V, see "Maximum overall output voltage" on page 186.

Parameters:

<Q> float

Range: $(-4+V_p/2+V_{bias}/2)$,V to $(4-V_p/2-V_{bias}/2)$,V (R&S SMW-B10) / $(-2+V_p)$,V to $(2-V_p)$,V (R&S SMW-B9)
Increment: 1E-4
*RST: 0
Default unit: V

Example: See Example "Enabling and configuring the analog output" on page 1051.

Manual operation: See "Offset (EMF)" on page 190

13.18.9.2 SOURce:IQ:OUTPut:DIGital Commands

This section describes the commands of the output of a digital I/Q signal.

Suffixes

The following suffixes are used:

Suffix	Value range	Description
`BBMM<ch>`	1 to 2	HS DIG I/Q connector
`FADer<ch>`	1 to 4*)	HS DIG I/Q connector

*) The number of the available DIG I/Q connectors on the FADER boards depends on the selected configuration (see also `:SCONfiguration:FADing`).

[:SOURce]:IQ:OUTPut:DIGital:BBMM<ch>:STATe <State>
[:SOURce]:IQ:OUTPut:DIGital:FADer<ch>:STATe <State>

Activates the digital I/Q signal output.

Parameters:

<State>	0 | 1 | OFF | ON *RST: 0

Example: `SOURce:IQ:OUTPut:DIGital:BBMM2:STATe ON`
Activates the BBMM 2 output connector.

Manual operation: See "State" on page 181

[:SOURce]:IQ:OUTPut:DIGital:BBMM<ch>:PON <Pon>
[:SOURce]:IQ:OUTPut:DIGital:FADer<ch>:PON <Pon>

Sets the power-on state of the selected digital I/Q output.

Parameters:

<Pon>	OFF | UNCHanged *RST: OFF

Example: `:SOURce:IQ:OUTPut:DIGital:BBMM2:PON OFF`
Deactivates the BBMM 2 output connector when the instrument is switched on.

Manual operation: See "Power-On State" on page 184

[:SOURce]:IQ:OUTPut:DIGital:BBMM<ch>:SRATe:SOURce <Source>
[:SOURce]:IQ:OUTPut:DIGital:FADer<ch>:SRATe:SOURce <Source>

Selects whether the sample rate is estimated based on the digital signal or is a user-defined value.

Parameters:

<Source>	USER | DOUT **DOUT** Enabled for BBMM1|BBMM2 connectors *RST: USER

Example: `:SOURce:IQ:OUTPut:DIGital:BBMM2:SRATe:SOURce DOUT`
Sample rate estimation based on the applied I/Q data clock
`:SOURce:IQ:OUTPut:DIGital:BBMM2:SRATe?`

Manual operation: See "Sample Rate Source" on page 181

[:SOURce]:IQ:OUTPut:DIGital:BBMM<ch>:SRATe <SRate>
[:SOURce]:IQ:OUTPut:DIGital:FADer<ch>:SRATe <SRate>

Sets/queries the sample rate of the digital I/Q output signal.

Parameters:

<SRate>	float Range for FADER connectors: `100E6|200E6` Range: 400 to depends on connected receiving device *RST: 1E8
Example:	See `[:SOURce]:IQ:OUTPut:DIGital:FADer<ch>:SRATe:SOURce` on page 1057
Manual operation:	See "Sample Rate Value" on page 181

[:SOURce]:IQ:OUTPut:DIGital:BBMM<ch>:CDEVice?
[:SOURce]:IQ:OUTPut:DIGital:FADer<ch>:CDEVice?

Queries information on the connected device.

Return values:

<CDevice>	string
Example:	`:SOURce:IQ:OUTPut:DIGital:BBMM2:CDEVice?`
Usage:	Query only
Manual operation:	See "Connected Device" on page 181

[:SOURce]:IQ:OUTPut:DIGital:BBMM<ch>:POWer:VIA <Via>
[:SOURce]:IQ:OUTPut:DIGital:FADer<ch>:POWer:VIA <Via>

Parameters:

<Via>	PEP | LEVel *RST: PEP
Manual operation:	See "Signal Output" on page 183

[:SOURce]:IQ:OUTPut:DIGital:BBMM<ch>:POWer:PEP <Pep>
[:SOURce]:IQ:OUTPut:DIGital:FADer<ch>:POWer:PEP <Pep>

Enters the peak level of the output signal relative to full scale of 0.5 V (in terms of dB full scale).

Parameters:

<Pep>	float Range: -60 to 0 Increment: 0.01 *RST: 0
Example:	`:SOURce:IQ:OUTPut:DIGital:BBMM2:POWer:PEP -10`
Manual operation:	See "Signal Output" on page 183

[:SOURce]:IQ:OUTPut:DIGital:BBMM<ch>:POWer:LEVel <Level>
[:SOURce]:IQ:OUTPut:DIGital:FADer<ch>:POWer:LEVel <Level>

Enters the RMS level of the output signal.

Parameters:

<Level> float

Range: -60 to 0
Increment: 0.01
*RST: 0

Example: `:SOURce:IQ:OUTPut:DIGital:BBMM2:POWer:LEVel -10`

Manual operation: See "Signal Output" on page 183

[:SOURce]:IQ:OUTPut:DIGital:BBMM<ch>:POWer:STEP:MODE <Mode>
[:SOURce]:IQ:OUTPut:DIGital:FADer<ch>:POWer:STEP:MODE <Mode>

Defines the type of step size to vary the digital output power step-by-step.

Parameters:

<Mode> DECimal | USER

DECimal
increases or decreases the level in steps of ten.

USER
increases or decreases the level in increments, determined with the command `[:SOURce]:IQ:OUTPut:DIGital:FADer<ch>:POWer:STEP[:INCRement]`.

*RST: DECimal

Example: `:SOURce:IQ:OUTPut:DIGital:BBMM2:POWer:STEP:MODE USER`
`:SOURce:IQ:OUTPut:DIGital:BBMM2:POWer:STEP 5 dB`
Activates the step width for level variation in 5 dB steps

Manual operation: See "User Variation" on page 183

[:SOURce]:IQ:OUTPut:DIGital:BBMM<ch>:POWer:STEP[:INCRement] <Increment>
[:SOURce]:IQ:OUTPut:DIGital:FADer<ch>:POWer:STEP[:INCRement] <Increment>

Sets the step width. Use this value to vary the digital I/Q output level step-by-step.

Parameters:

<Increment> float

Range: 0 to 100
Increment: 0.01
*RST: 1

Example: See `[:SOURce]:IQ:OUTPut:DIGital:FADer<ch>:POWer:STEP:MODE` on page 1059

Manual operation: See "User Variation" on page 183

[:SOURce]:IQ:OUTPut:DIGital:BBMM<ch>:GDELay:CState <CompState>

Enables/disables group delay compensation.

Parameters:
<CompState> 0 | 1 | OFF | ON
*RST: 0

Example: `SOURce:IQ:OUTPut:DIGital:BBMM1:GDELay:CSTate`

Manual operation: See "Group Delay Compensation" on page 184

[:SOURce]:IQ:OUTPut:DIGital:BBMM<ch>:OFLow:STATe?
[:SOURce]:IQ:OUTPut:DIGital:FADer<ch>:OFLow:STATe?

Queries whether the I/Q output signal is clipped or not.

Return values:
<State> 0 | 1 | OFF | ON
*RST: 0

Example: `:SOURce:IQ:OUTPut:DIGital:BBMM2:OFLow:STATe?`
Response: "0" (no overflow).

Usage: Query only

Manual operation: See "Signal Monitoring" on page 183

[:SOURce]:IQ:OUTPut:DIGital:BBMM<ch>:OFLow:HOLD:RESet
[:SOURce]:IQ:OUTPut:DIGital:FADer<ch>:OFLow:HOLD:RESet

Resets the overflow hold state and LED.

Example: `:SOURce:IQ:OUTPut:DIGital:BBMM2:OFLow:HOLD:RESet`

Usage: Event

Manual operation: See "Signal Monitoring" on page 183

[:SOURce]:IQ:OUTPut:DIGital:BBMM<ch>:OFLow:HOLD:STATe?
[:SOURce]:IQ:OUTPut:DIGital:FADer<ch>:OFLow:HOLD:STATe?

Queries an overload since last reset for evaluating the measurement.

Return values:
<State> 0 | 1 | OFF | ON
*RST: 0

Example: `:SOURce:IQ:OUTPut:DIGital:BBMM2:OFLow:HOLD:STATe?`

Usage: Query only

Manual operation: See "Signal Monitoring" on page 183

13.18.10 SOURce:LFOutput Subsystem

The `SOURce:LFOutput` subsystem contains the commands for setting the LF signal source in CW and Sweep mode and for analog modulation.

Example: Setup an LF sweep

The following example shows a command sequence to set up an LF sweep.

```
// Reset the instrument to start from an initial state
*RST; *CLS

// Set the trigger mode, the sweep mode and the sweep range
TRIGger1:LFFSweep:SOURce SINGle
SOURce1:LFOutput1:SWEep:FREQuency:MODE AUTO
SOURce1:LFOutput1:FREQuency:STARt 1 kHz
SOURce1:LFOutput1:FREQuency:STOP 7 kHz

// Select linear spacing
// Select the waveform shape for the frequency sweep cycle
// Set the step width and the dwell time.
SOURce1:LFOutput1:SWEep:FREQuency:SPACing LINear
SOURce1:LFOutput1:SWEep:FREQuency:SHAPe SAWtooth
SOURce1:LFOutput1:SWEep:FREQuency:STEP:LINear 100 Hz
SOURce1:LFOutput1:SWEep:FREQuency:DWELl 150 ms
// Alternatively to the step width set the number of steps
SOURce1:LFOutput1:SWEep:FREQuency:POINts 61

// Activate change to start frequency while waiting for next trigger
// Prerequisites: sweep mode single and sweep waveform sawtooth
SOURce1:LFOutput:SWEep:FREQuency:RETRace 1

// Activate the LF frequency sweep
SOURce1:LFOutput:FREQuency:MODE SWE

// Trigger the sweep(depending on the set mode) and query the status
// Perform a one-off LF sweep
SOURce1:LFOutput1:SWEep:FREQuency:EXECute
SOURce1:LFOutput1:SWEep:FREQuency:RUNNing?
// 1
// the sweep is running

// ***************************************************
// For manual step LF sweep use the following commands
*RST; *CLS
SOURce1:LFOutput:SWEep:FREQuency:MODE MANual
```

```
// Activate the LF frequency sweep
SOURce1:LFOutput:FREQuency:MODE SWEep
// Activate LF Output1.
SOURce1:LFOutput1:STATe 1
// Input the frequency manually for each step
SOURce1:LFOutput1:FREQuency:MANual 2 kHz
SOURce1:LFOutput1:FREQuency:MANual 2.1 kHz
// Alternatively use UP or DOWN parameter with set step width.
SOURce1:LFOutput1:SWEep:FREQuency:STEP:LINear 500 Hz
SOURce1:LFOutput1:FREQuency:MANual UP
```

Example: Configuring the LF generator

The following is a simple example on how to configure the LF generator and output the generated signal.

```
// configure the signal of the LF1 generator
SOURce1:LFOutput1:SHAPe SQU
SOURce1:LFOutput1:SHAPe:PULSe:PERiod 0.001
SOURce1:LFOutput1:SHAPe:PULSe:WIDTh 0.0005
SOURce1:LFOutput1:SHAPe:PULSe:DCYCle 0.5
// configure the signal of the LF1 generator
SOURce1:LFOutput2:SHAPe SINE
SOURce1:LFOutput2:FREQuency 1000000
SOURce1:LFOutput2:PERiod?
// 0.000001

// changing the LF signal shape
// SOURce1:LFOutput2:SHAPe TRAP
// SOURce1:LFOutput2:SHAPe:PULSe:PERiod 0.2
// SOURce1:LFOutput2:SHAPe:TRAPeze:RISE 0.0001
// SOURce1:LFOutput2:SHAPe:TRAPeze:FALL 0.001
// SOURce1:LFOutput2:SHAPe:TRAPeze:FALL 0.0001
// SOURce1:LFOutput2:SHAPe:TRAPeze:HIGH 0.0005
// SOURce1:LFOutput2:SHAPe TRI
// SOURce1:LFOutput2:SHAPe:PULSe:PERiod 0.1
// SOURce1:LFOutput2:SHAPe:TRIangle:RISE 0.0001

// activate the LF output and select the LF1 as signal source
// configure the LF output signal
SOURce1:LFOutput1:FREQuency:MODE CW
SOURce1:LFOutput1:STATe 1
SOURce1:LFOutput1:SOURce LF1
SOURce1:LFOutput1:INTernal:VOLTage 1
SOURce1:LFOutput1:OFFSet 0.001
```

13.18.10.1 LF Generator Settings

With the commands described in this section, you can configure the LF signal source.

[:SOURce]:LFOutput<ch>:BANDwidth?

Queries the bandwidth of the external LF signal.

Return values:

<Bandwidth> BW0M2 | BW10m

*RST: BW10m

Example:

```
LFO:BAND?
// BW10m
// the bandwidth of the externally supplied LF signal is 10 MHz
```

Usage: Query only

Manual operation: See "Bandwidth" on page 505

[:SOURce<hw>]:LFOutput<ch>:FREQuency <Frequency>

Sets the frequency of the LF signal in `[:SOURce<hw>]:LFOutput:FREQuency:MODE` `CW|FIXed` mode.

Note:

- If the LF generator is used as a signal source, the instrument performs the analog modulations (AM/FM/ΦM/PM) with this frequency.

- In sweep mode ([:SOURce<hw>]:LFOutput:FREQuency:MODE SWE), the frequency is coupled with the sweep frequency.

Parameters:

<Frequency> float

Range: 0.1 to depends on the installed options (R&S SMW-K24)
Increment: 0.1
*RST: 1000
Default unit: Hz

Example: See Example "Configuring the LF generator" on page 1062.

Manual operation: See "Frequency" on page 504

[:SOURce<hw>]:LFOutput<ch>:PERiod?

Queries the repetition frequency of the sine signal.

Return values:

<LfSinePeriod> float

Range: 1E-6 to 100
Increment: 10E-9
*RST: 0.001
Default unit: s

Example: See Example "Configuring the LF generator" on page 1062.

Usage: Query only

[:SOURce<hw>]:LFOutput:FREQuency:MANual <Manual>

Sets the frequency of the subsequent sweep step if LFO:SWE:MODE MAN.

Use a separate command for each sweep step.

Parameters:

<Manual> float

You can select any value within the setting range, where:
STARt is set with [:SOURce<hw>]:LFOutput:FREQuency:STARt
STOP is set with [:SOURce<hw>]:LFOutput:FREQuency:STOP

Range: STARt to STOP
Increment: 0.1
*RST: 1000

Example: See Example "Setup an LF sweep" on page 1061.

Manual operation: See " Current Frequency" on page 478

[:SOURce<hw>]:LFOutput:FREQuency:MODE <Mode>

Sets the mode for the output of the LF generator frequency, and determines the commands to be used for frequency settings.

Parameters:

<Mode> CW | FIXed | SWEep

CW|FIXed
Sets the fixed-frequency mode. `CW` and `FIXed` are synonyms.
To set the output frequency, use command `[:SOURce<hw>]:LFOutput<ch>:FREQuency`

SWEep
Sets sweep mode.
To set the frequency, use the commands:
`[:SOURce<hw>]:LFOutput:FREQuency:STARt` and `[:SOURce<hw>]:LFOutput:FREQuency:STOP`
Or `[:SOURce<hw>]:LFOutput:FREQuency:MANual`

*RST: CW

Example: See Example "Setup an LF sweep" on page 1061.

Manual operation: See "State (LF frequency sweep)" on page 476

[:SOURce<hw>]:LFOutput:FREQuency:STOP <Stop>
[:SOURce<hw>]:LFOutput:FREQuency:STARt <Start>

Sets the start/stop frequency for `[:SOURce<hw>]:LFOutput:FREQuency:MODE` `SWEep`.

Parameters:

<Start> float

Range: 0.1 Hz to 1 MHz
Increment: 0.1
*RST: 1 KHz

<Stop> float

Range: 0.1 Hz to 1 MHz
Increment: 0.1 Hz
*RST: 100 KHz

Example: See Example "Setup an LF sweep" on page 1061.

Manual operation: See " Start Frequency/Stop Frequency " on page 478

[:SOURce]:LFOutput<ch>:OFFSet <Offset>

Sets a DC offset at the selected LF Output.

Parameters:

<Offset> float

Range: -3.6 to 3.6
Increment: 2E-3
*RST: 0
Default unit: V

Example: See Example "Configuring the LF generator" on page 1062.

Manual operation: See "DC-Offset" on page 511

[:SOURce]:LFOutput<ch>:SOURce <Source>

Determines the LF signal to be synchronized, when monitoring is enabled.

Parameters:

<Source> LF1 | LF2 | NOISe | EXT1 | EXT2 | AM | FMPM | LF1B | LF2B | AMB | NOISB | FMPMB | LF1A | LF2A | NOISA | FMPMA | AMA

LF1|LF2|LF1A|LF2A|LF1B|LF2B
Selects an internally generated LF signal.

NOISe|NOISA|NOISB
Selects an internally generated noise signal.

EXT1|EXT2
Selects an externally supplied LF signal

AM|AMA|AMB
Selects the AM signal.

FMPM|FMPMA|FMPMB
Selects the signal also used by the frequency or phase modulations.

*RST: LF1

Example: See Example "Configuring the LF generator" on page 1062.

Manual operation: See "Source" on page 511

[:SOURce]:LFOutput<ch>:SOURce:PATH <SourPath>

Determines the path of the LF output source.

Parameters:

<SourPath> A | B

*RST: A

Example: `LFO:SOUR:PATH?`
Queries the currently set path for the LF output signal source.

[:SOURce]:LFOutput<ch>[:STATe] <State>

Activates LF signal output

Parameters:

<State> 0 | 1 | OFF | ON

*RST: 0

Example: See Example "Configuring the LF generator" on page 1062.

Manual operation: See "State" on page 511

[:SOURce]:LFOutput<ch>:VOLTage <Voltage>

Sets the output voltage of the selected LF output.

Parameters:

<Voltage> float

Range: dynamic (see data sheet)
Increment: 0.001
*RST: 1
Default unit: V

Example: `SOURce:LFOutput1:VOLTage 1.5`

Manual operation: See "Output Voltage" on page 511

[:SOURce]:LFOutput<ch>:INTernal:VOLTage <Voltage>

Sets the output voltage for the LF generators.

The sum of both values must not exceed the overall output voltage, set with command `[:SOURce]:LFOutput<ch>:VOLTage`.

Suffix:

<ch> [1]|2
LF1 and LF2

Parameters:

<Voltage> float

Range: 0 to 4
*RST: 0.5

Example: See Example "Configuring the LF generator" on page 1062.

[:SOURce<hw>]:LFOutput<ch>:SHAPe <Shape>

Selects the waveform shape of the LF signal.

Parameters:

<Shape> SINE | SQUare | TRIangle | TRAPeze

*RST: SINE

Example: See Example "Configuring the LF generator" on page 1062.

Options: TRIangle|TRAPeze require R&S SMW-K24

Manual operation: See "Shape" on page 504

[:SOURce<hw>]:LFOutput<ch>:SHAPe:PULSe:DCYCle <DCycle>

Sets the duty cycle for the shape pulse.

Parameters:

<DCycle> float

Range: 1E-6 to 100
Increment: 1E-6
*RST: 50
Default unit: PCT

Example: See Example "Configuring the LF generator" on page 1062.

Manual operation: See "Pulse Duty Cycle" on page 504

[:SOURce<hw>]:LFOutput<ch>:SHAPe:PULSe:PERiod <Period>

Sets the period of the generated pulse. The period determines the repetition frequency of the internal signal.

Parameters:

<Period> float

Range: 1E-6 to 100
Increment: 1E-8
*RST: 1E-3

Example: See Example "Configuring the LF generator" on page 1062.

Manual operation: See "Period" on page 504

[:SOURce<hw>]:LFOutput<ch>:SHAPe:PULSe:WIDTh <Width>

Sets the pulse width of the generated pulse.

Parameters:

<Width> float

Range: 1E-6 to 100
Increment: 1E-8
*RST: 5E-4

Example: See Example "Configuring the LF generator" on page 1062.

Manual operation: See "Pulse Width" on page 504

[:SOURce<hw>]:LFOutput<ch>:SHAPe:TRAPeze:FALL <Fall>

Selects the fall time for the trapezoid shape of the LF generator.

Parameters:

<Fall> float

Range: 1E-6 to 100
Increment: 10E-9
*RST: 250E-6

Example: See Example "Configuring the LF generator" on page 1062.

Manual operation: See "Trapezoid Rise / Fall" on page 505

[:SOURce<hw>]:LFOutput<ch>:SHAPe:TRAPeze:HIGH <High>

Sets the high time for the trapezoid signal of the LF generator.

Parameters:

<High> float

Range: 1E-6 to 100
Increment: 10E-9
*RST: 250E-6

Example: See Example "Configuring the LF generator" on page 1062.

Manual operation: See "Trapezoid High" on page 505

[:SOURce<hw>]:LFOutput<ch>:SHAPe:TRAPeze:PERiod <Period>

Sets the period of the generated trapezoid shape. The period determines the repetition frequency of the internal signal.

Parameters:

<Period> float

Range: 1E-6 to 100
Increment: 1E-8
*RST: 1E-3

Example: See Example "Configuring the LF generator" on page 1062.

Manual operation: See "Period" on page 504

[:SOURce<hw>]:LFOutput<ch>:SHAPe:TRAPeze:RISE <Rise>

Selects the rise time for the trapezoid shape of the LF generator.

Parameters:

<Rise> float

Range: 1E-6 to 100
Increment: 10E-9
*RST: 250E-6

Example: See Example "Configuring the LF generator" on page 1062.

Manual operation: See "Trapezoid Rise / Fall" on page 505

[:SOURce<hw>]:LFOutput<ch>:SHAPe:TRIangle:PERiod <Period>

Sets the period of the generated pulse. The period determines the repetition frequency of the internal signal.

Parameters:

<Period> float

Range: 1E-6 to 100
Increment: 10E-9
*RST: 0.001

Example: See Example "Configuring the LF generator" on page 1062.

Manual operation: See "Period" on page 504

[:SOURce<hw>]:LFOutput<ch>:SHAPe:TRIangle:RISE <Rise>

Selects the rise time for the triangle single of the LF generator.

Parameters:

<Rise> float

Range: 1E-6 to 100
Increment: 10E-9
*RST: 0.5E-3

Example: See Example "Configuring the LF generator" on page 1062.

Manual operation: See "Triangle Rise" on page 504

13.18.10.2 LF Sweep Settings

With the commands described in this section, you can configure the sweep of the LF signal.

[:SOURce<hw>]:LFOutput:SWEep[:FREQuency]:DWELl <Dwell>

Sets the dwell time for each frequency step of the sweep.

Parameters:

<Dwell> float

Range: 1E-3 to 100
Increment: 100E-6
*RST: 15E-3
Default unit: s

Example: See Example "Setup an LF sweep" on page 1061.

Manual operation: See " Dwell Time " on page 480

[:SOURce<hw>]:LFOutput:SWEep[:FREQuency]:EXECute

Immediately starts an LF sweep.

`[:SOURce<hw>]:LFOutput:SWEep[:FREQuency]:MODE` determines which sweep is executed, e.g. `SOURce:LFOutput:SWEep:FREQuency:MODE STEP`.

Example: See Example "Setup an LF sweep" on page 1061.

Usage: Event

Manual operation: See " Execute Single Sweep " on page 477

[:SOURce<hw>]:LFOutput:SWEep[:FREQuency]:MODE <Mode>

Sets the cycle mode of the LF sweep.

Parameters:

<Mode> AUTO | MANual | STEP

AUTO
Performs a complete sweep cycle from the start to the end value when a trigger event occurs.
The dwell time determines the time period until the signal switches to the next step.

MANual
Performs a single sweep step when a manual trigger event occurs.
The trigger system is not active. To trigger each frequency step of the sweep individually, use the command `[:SOURce<hw>]:LFOutput:FREQuency:MANual` on page 1064.

STEP
Each trigger command triggers one sweep step only.
The frequency increases by the value set with the coammnds:
`[:SOURce<hw>]:LFOutput:SWEep[:FREQuency]:STEP[:LINear]` (linear spacing)
`[:SOURce<hw>]:LFOutput:SWEep[:FREQuency]:STEP:LOGarithmic`(logarithmic spacing)

*RST: AUTO

Example: See Example "Setup an LF sweep" on page 1061.

Manual operation: See "Mode" on page 476

[:SOURce<hw>]:LFOutput:SWEep[:FREQuency]:POINts <Points>

Sets the number of steps in an LF sweep.

For information on how the value is calculated and the interdependency with other parameters, see Chapter 7.6.2.1, "Correlating Parameters in Sweep Mode", on page 467

Parameters:

<Points>	integer	
	Range:	2 to POINts
	Increment:	1
	*RST:	100

Example: See Example "Setup an LF sweep" on page 1061.

[:SOURce<hw>]:LFOutput:SWEep[:FREQuency]:RETRace <State>

Activates that the signal changes to the start frequency value while it is waiting for the next trigger event.

You can enable this feature, when you are working with sawtooth shapes in sweep mode "Single" or "External Single".

Parameters:

<State>	0 | 1 | OFF | ON	
	*RST:	0

Manual operation: See "Retrace" on page 479

[:SOURce<hw>]:LFOutput:SWEep[:FREQuency]:RUNNing?

Queries the current status of the LF frequency sweep mode.

Return values:

<State> 0 | 1 | OFF | ON

Example: See Example "Setup an LF sweep" on page 1061.

Usage: Query only

[:SOURce<hw>]:LFOutput:SWEep[:FREQuency]:SHAPe <Shape>

Sets the cycle mode for a sweep sequence (shape).

Parameters:

<Shape>	SAWTooth | TRIangle	
	*RST:	SAWTooth

Example: See Example "Setup an LF sweep" on page 1061.

Manual operation: See " Shape " on page 479

[:SOURce<hw>]:LFOutput:SWEep[:FREQuency]:SPACing <Spacing>

Selects linear or logarithmic sweep spacing.

Parameters:

<Spacing> LINear | LOGarithmic

*RST: LINear

Example: See Example "Setup an LF sweep" on page 1061.

Manual operation: See " Spacing" on page 479

[:SOURce<hw>]:LFOutput:SWEep[:FREQuency]:STEP:LOGarithmic <Logarithmic>

Sets the step width factor for logarithmic sweeps to calculate the frequencies of the steps.

For information on how the value is calculated and the interdependency with other parameters, see Chapter 7.6.2.1, "Correlating Parameters in Sweep Mode", on page 467

Parameters:

<Logarithmic> float

The unit is mandatory

Range: 0.01 to 100
Increment: 0.01
*RST: 1
Default unit: PCT

Example: See Example "Setup an LF sweep" on page 1061.

Manual operation: See " Step Linear/Step Logarithmic " on page 480

[:SOURce<hw>]:LFOutput:SWEep[:FREQuency]:STEP[:LINear] <Linear>

Sets the step width for the linear sweep.

For information on how the value is calculated and the interdependency with other parameters, see Chapter 7.6.2.1, "Correlating Parameters in Sweep Mode", on page 467

Parameters:

<Linear> float

Range: 0 to STOP-STARt
Increment: 0.1
*RST: 1000

Example: See Example "Setup an LF sweep" on page 1061.

Manual operation: See " Step Linear/Step Logarithmic " on page 480

13.18.11 SOURce:LIST Subsystem

The `SOURce:LIST` subsystem contains all commands for defining lists and for handling of list files.

List files have the predefined file extension `*.lsw`.

Refer to Chapter 13.7.2, "Accessing Files in the Default or in a Specified Directory", on page 777 for general information on file handling in the default and in a specific directory.

- You can use the same lists for both paths.
  File operations such as creating, deleting and querying lists are therefore path-independent. In this case, the suffix `<HW>` under `SOURce` **must** be omitted. An error message is displayed if the suffix is specified.
- `*RST` does not affect data lists.
- SCPI refers to the individual lists as segments.

Example: Create an RF list and activate the list mode

The following example shows a command sequence to create an RF list and to activate the list mode. Further hardware settings are not considered.

```
// Reset the instrument to start from an initial state
// Query the available list files in the default
// directory /var/user
// Select the list file or create it (if not existing)
*RST; *CLS
SOUR1:LIST:CAT?
// Response:- shows the name of available list files (if applicable)
SOUR1:LIST:SEL "/var/user/list1.lsw"

// Write the frequency/level/dwell time values in the selected list file
// existing data is overwritten
// Query the number of frequency/power/dwell time entries in the selected list
// Query the amount of free memory (in bytes) for list mode lists
SOUR1:LIST:FREQ 58 MHz, 61 MHz, 73 MHz, 86 MHz, 91 MHz, 92 MHz, 98 MHz
SOUR1:LIST:POW 13 dBm, 12 dBm, 5 dBm, 3 dBm, 0 dBm, 4 dBm, 6 dBm
SOUR1:LIST:DWEL:LIST 10000, 100000, 200000, 19000, 10000, 150000, 220000
SOUR1:LIST:FREQ:POIN?
// 7
SOUR1:LIST:POW:POINT?
// 7
SOUR1:LIST:DWEL:LIST:POIN?
// 7
SOUR1:LIST:FREE?
// 2147483647 (bytes of free memory)

// Use dwell times from list
// Configure the list mode parameters
// Enable RF output
SOUR1:LIST:MODE AUTO
SOUR1:LIST:TRIG:SOUR AUTO
SOUR1:LIST:DWEL:MODE "LIST"
```

```
// Learn list mode data
SOUR1:LIST:LEAR

OUTP1:STAT ON

// Use global dwell time
// Set only a part of the list (value pairs 3 to 5) to be processed
// Configure the list mode parameters using global dwell time
// Learn list mode data
// Enable RF output
SOUR1:LIST:IND:START 2
SOUR1:LIST:IND:STOP 4
SOUR1:LIST:MODE AUTO
SOUR1:LIST:TRIG:SOUR AUTO
SOUR1:LIST:DWEL:LIST 500 ms
SOUR1:LIST:LEAR
OUTP1:STAT ON
SOUR1:LIST:RMOD LEAR

// Enable the list mode
// Trigger the list (depending on the mode, not needed with trigger
// mode AUTO); query the current index
// Reset the list to the starting point
SOUR1:FREQ:MODE LIST
SOUR1:LIST:TRIG:EXEC
SOUR1:LIST:RUNN?
SOUR1:LIST:IND?
// 3
// value changes when the value is queried again
SOUR1:LIST:RES

// For list mode STEP use the following commands
*RST; *CLS
// Change list mode to STEP
SOUR1:LIST:MODE STEP
// Activate RF Output1
OUTP1:STAT 1
// Activate the list mode
SOUR1:FREQ:MODE LIST
// For each step: select frequency/powerlevel pair as index from the list
SOUR1:LIST:IND 2
SOUR1:LIST:IND 3
SOUR1:LIST:IND 4

// Use the selected list for path B (with List Mode B default settings)
SOUR2:LIST:SEL "/var/user/list1.lsw"
OUTP2:STAT ON
SOUR2:FREQ:MODE LIST
SOUR2:LIST:IND?
// 2
// value canges when the value is queried again
```

```
// Deactivate the list mode
SOUR1:FREQ:MODE CW

// Use global dwell time
// Set only a part of the list (value pairs 3 to 5) to be processed
// Configure the list mode parameters using global dwell time
// Enable RF output
SOUR1:LIST:IND:START 2
SOUR1:LIST:IND:STOP 4
SOUR1:LIST:MODE AUTO
SOUR1:LIST:TRIG:SOUR AUTO
SOUR1:LIST:DWEL:LIST 500 ms
OUTP1:STAT ON

// Enable the list mode
// Trigger the list (depending on the mode, not needed with trigger
// mode AUTO); query the current index
// Reset the list to the starting point
SOUR1:FREQ:MODE LIST
SOUR1:LIST:TRIG:EXEC
SOUR1:LIST:RUNN?
SOUR1:LIST:IND?
// 3
// value changes when the value is queried again
SOUR1:LIST:RES

// For list mode STEP use the following commands
*RST; *CLS
// Change list mode to STEP
SOUR1:LIST:MODE STEP
// Activate RF Output1
OUTP1:STAT 1
// Activate the list mode
SOUR1:FREQ:MODE LIST
// For each step: select frequency/powerlevel pair as index from the list
SOUR1:LIST:IND 2
SOUR1:LIST:IND 3
SOUR1:LIST:IND 4

// Use the selected list for path B (with List Mode B default settings)
SOUR2:LIST:SEL "/var/user/list1.lsw"
OUTP2:STAT ON
SOUR2:FREQ:MODE LIST
SOUR2:LIST:IND?
// 2
// value canges when the value is queried again

// Deactivate the list mode
SOUR1:FREQ:MODE CW
```

Example: List mode data exchange

The following example shows a command sequence to export a list (here the RF list created with the example before) into an ASCII file. Further hardware settings are not considered.

```
*RST; *CLS
LIST:DEXC:MODE EXP

// Set ASCII data parameters
// Set the ASCII file extension, the decimal separator
// and the column separator for the ASCII data
SOUR1:LIST:DEXC:AFIL:EXT CSV
SOUR1:LIST:DEXC:AFIL:SEP:DEC DOT
SOUR1:LIST:DEXC:AFIL:SEP:COL COMM

// Select source and destination path/directory
// Query available listfiles in default directory "/var/user"
SOUR1:LIST:CAT?
// list1
SOUR1:LIST:DEXC:AFIL:SEL "/var/user/list1ASCII"
SOUR1:LIST:DEXC:SEL "/var/user/list1"

// Export the list file data into the ASCII file
SOUR1:LIST:DEXC:EXEC

// Query the available ASCII files with extension .csv
SOUR1:LIST:DEXC:AFIL:CAT?
// Response: "list1ASCII"

// Deactivate the list mode
SOUR1:FREQ:MODE CW
```


13.18.11.1 List Mode Settings

With the following commands, you can create list mode data, select the trigger mode and determine the dwell time.

[:SOURce<hw>]:LIST:DWELl <Dwell>

Sets the global dwell time. The instrument generates the signal with the frequency / power value pairs of each list entry for that particular period.

See also Significant Parameters and Functions.

Parameters:

<Dwell> float

Range: 7E-4 to 100
Increment: 1E-4
*RST: 15E-3
Default unit: s

Example: See Example "Create an RF list and activate the list mode" on page 1074.

Manual operation: See "Global Dwell Time" on page 483

[:SOURce<hw>]:LIST:DWELl:MODE <DwelMode>

Selects the dwell time mode.

Parameters:

<DwelMode> LIST | GLOBal

LIST
uses the dwell time, specified in the data table for each value pair individually.

GLOBal
uses a constant dwell time, set with command `[:SOURce<hw>]:LIST:DWELl`.

*RST: GLOBal

Example: See Example "Create an RF list and activate the list mode" on page 1074.

Manual operation: See "Dwell Time Mode" on page 483

[:SOURce<hw>]:LIST:DWELl:LIST <Dwell>

Enters the dwell time values in the selected list in µs.

Parameters:

<Dwell> <Dwell#1>{, <Dwell#2>, ...} | block data

You can either enter the data as a list of numbers, or as binary block data. The list of numbers can be of any length, with the list entries separated by commas.
In binary block format, 8 (4) bytes are always interpreted as a floating-point number with double accuracy. See also `:FORMat[:DATA]` on page 802 for more details.

Example: See Example "Create an RF list and activate the list mode" on page 1074.

Manual operation: See "Edit List Mode Data" on page 487

[:SOURce<hw>]:LIST:DWELl:LIST:POINts?

Queries the number (points) of dwell time entries in the selected list.

Return values:

<Points> integer

Range: 0 to INT_MAX
*RST: 0

Example: See Example "Create an RF list and activate the list mode" on page 1074.

Usage: Query only

[:SOURce<hw>]:LIST:FREQuency <Frequency>

Enters the frequency values in the selected list.

Parameters:

<Frequency> <Frequency#1>{, <Frequency#2>, ...} | block data

You can either enter the data as a list of numbers, or as binary block data.
The list of numbers can be of any length, with the list entries separated by commas.
In binary block format, 8 (4) bytes are always interpreted as a floating-point number with double accuracy.
See also `:FORMat[:DATA]`.

Range: 300 kHz to RFmax (depends on the installed options)

Example: See Example "Create an RF list and activate the list mode" on page 1074.

Manual operation: See "Edit List Mode Data" on page 487

[:SOURce<hw>]:LIST:FREQuency:POINts?

Queries the number (points) of frequency entries in the seleced list.

Return values:

<Points> integer

Range: 0 to INT_MAX
*RST: 0

Example: See Example "Create an RF list and activate the list mode" on page 1074.

Usage: Query only

[:SOURce<hw>]:LIST:INDex <Index>

Sets the list index in LIST:MODE STEP.

After the trigger signal, the instrument processes the frequency and level settings of the selected index.

Parameters:

<Index> integer

*RST: 0

Example: See Example "Create an RF list and activate the list mode" on page 1074.

Manual operation: See " Current Index " on page 483

[:SOURce<hw>]:LIST:INDex:STARt <Start>
[:SOURce<hw>]:LIST:INDex:STOP <Stop>

Sets the start and stop index of the index range which defines a subgroup of frequency/level value pairs in the current list.

Parameters:

<Start>/<Stop> integer

Index range
Only values inside this range are processed in list mode

Range: 0 to list length
*RST: 0

Example: See Example "Create an RF list and activate the list mode" on page 1074.

Manual operation: See " List Range from/to " on page 485

[:SOURce<hw>]:LIST:LEARn

Learns the selected list to determine the hardware setting for all list entries. The results are saved with the list.

See also "Learn List Mode Data list processing mode" on page 473.

Example: See Example "Create an RF list and activate the list mode" on page 1074.

Usage: Event

Manual operation: See " Learn List Mode Data " on page 485

[:SOURce<hw>]:LIST:MODE <Mode>

Sets the list mode.

The instrument processes the list according to the selected mode and trigger source, see LIST:TRIG:SOUR AUTO, SING or EXT.

Parameters:

<Mode> AUTO | STEP

AUTO
Each trigger event triggers a complete list cycle.

STEP
Each trigger event triggers only one step in the list processing cycle. The list is processed in ascending order.
In this mode, you can select between LIST:TRIG:SOUR SING or EXT.

*RST: AUTO

Example: See Example "Create an RF list and activate the list mode" on page 1074.

Manual operation: See "Mode" on page 482

[:SOURce<hw>]:LIST:POWer <Power>

Enters the level values in the selected list. The number of level values must correspond to the number of frequency values. Existing data is overwritten.

Parameters:

<Power> <Power#1>{, <Power#2>, ...} | block data

You can either enter the data as a list of numbers, or as binary block data.
The list of numbers can be of any length, with the list entries separated by commas.
In binary block format, 8 (4) bytes are always interpreted as a floating-point number with double accuracy.
See also `:FORMat[:DATA]`.

Range: depends on the installed options
Default unit: dBm

Example: See Example "Create an RF list and activate the list mode" on page 1074.

Manual operation: See "Edit List Mode Data" on page 487

[:SOURce<hw>]:LIST:POWer:POINts?

Queries the number (points) of level entries in the selected list.

Return values:

<Points> integer

Range: 0 to INT_MAX
*RST: 0

Example: See Example "Create an RF list and activate the list mode" on page 1074.

Usage: Query only

[:SOURce<hw>]:LIST:RMODe <RMode>

Selects the run mode for processing the list.

Parameters:

<RMode> LEARned | LIVE

LEARned
Generates the signal by replaying the previously learned and saved data from the temporary memory.

LIVE
Generates the signal by processing the list directly.

*RST: LIVE

Example: See Example "Create an RF list and activate the list mode" on page 1074.

Manual operation: See "Run Mode" on page 485

[:SOURce<hw>]:LIST:TRIGger:EXECute

Starts the processing of a list in list mode.

Example: See Example "Create an RF list and activate the list mode" on page 1074.

Usage: Event

Manual operation: See " Execute Single" on page 482

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

Selects the trigger source for processing lists.

The names of the parameters correspond to those in sweep mode. SCPI standard uses other names for the parameters; these names are also accepted by the instrument. The SCPI names should be used if compatibility is an important consideration. For an overview, see the following table:

Rohde & Schwarz name	SCPI name	Command in manual control
AUTO	IMMediate	`MODE AUTO`
SINGle	BUS	`MODE SINGLE` or `STEP`
EXTernal	EXTernal	`MODE EXT TRIG SINGLE` or `EXT TRIG STEP`

Parameters:

<Source> SINGle | BUS | AUTO | EXTernal | IMMediate

AUTO|IMMediate
The trigger is free-running, i.e. the trigger condition is fulfilled continuously. The selected list is restarted as soon as it is finished.

SINGle|BUS
The list is triggered by the command `[:SOURce<hw>]:LIST:TRIGger:EXECute`. The list is executed once.

EXTernal
The list is triggered externally and executed once.

*RST: AUTO

Example: See Example "Create an RF list and activate the list mode" on page 1074.

Manual operation: See "Mode" on page 482

[:SOURce<hw>]:LIST:RUNNing?

Queries the current state of the list mode.

Return values:

<State> 0 | 1 | OFF | ON

1
Signal generation based on the list mode is active.

Example: See Example "Create an RF list and activate the list mode" on page 1074.

Usage: Query only

13.18.11.2 List Mode File Operation

The following section covers basic commands to file handling in list mode.

[:SOURce<hw>]:LIST:CATalog?

Queries the available list files in the specified directory.

Return values:

<Catalog> string

List of list filenames, separated by commas

Example: See Example "Create an RF list and activate the list mode" on page 1074.

Usage: Query only

Manual operation: See " List Mode Data " on page 484

[:SOURce<hw>]:LIST:DELete <Filename>

Deletes the specified list.

Setting parameters:

<Filename> string

Filename or complete file path; file extension is optional.

Example: See `[:SOURce<hw>]:LIST:DELete:ALL` on page 1084.

Usage: Setting only

Manual operation: See " List Mode Data " on page 484

[:SOURce<hw>]:LIST:DELete:ALL

Deletes all lists in the set directory.

This command can only be executed, if:

- No list file is selected.
- List mode is disabled.

Example:

```
SOUR1:LIST:CAT?
// list,my_list
SOUR1:LIST:DEL "/var/user/list1"
SOUR1:LIST:CAT?
// my_list
SOUR1:FREQ:MODE?
// LIST
SOUR1:LIST:SEL?
// /var/user/my_list.lsw
//deactivate list mode
SOUR1:FREQ:MODE CW
SOUR1:LIST:DELete:ALL
SOUR1:LIST:CAT?
// -
// all list files are deleted
```

Usage: Event

Manual operation: See " List Mode Data " on page 484

[:SOURce<hw>]:LIST:FREE?

Queries the amount of free memory (in bytes) for list mode lists.

Return values:

<Free> integer

Range: 0 to INT_MAX
*RST: 0

Example: See Example "Create an RF list and activate the list mode" on page 1074.

Usage: Query only

[:SOURce<hw>]:LIST:RESet

Jumps to the beginning of the list.

Example: See Example "Create an RF list and activate the list mode" on page 1074.

Usage: Event

Manual operation: See " Reset " on page 482

[:SOURce<hw>]:LIST:SELect <Filename>

Selects or creates a data list in list mode.

If the list with the selected name does not exist, a new list is created.

Parameters:

<Filename> string

Filename or complete file path; file extension can be omitted.

Example: See Example "Create an RF list and activate the list mode" on page 1074.

Manual operation: See " List Mode Data " on page 484

13.18.11.3 List Mode Data Exchange

With the following commands, you can configure lists in ASCII format and export or import them accordingly.

[:SOURce<hw>]:LIST:DEXChange:AFILe:CATalog?

Queries the available ASCII files for export or import of list mode data in the current or specified directory.

Return values:

<Catalog>	string List of ASCII files `*.txt` or `*.csv`, separated by commas.
Example:	See Example "List mode data exchange" on page 1077.
Usage:	Query only
Manual operation:	See "Select Source/Select Destination" on page 491

[:SOURce<hw>]:LIST:DEXChange:EXECute

Executes the import or export of the selected list file, according to the previously set transfer direction with command `[:SOURce<hw>]:LIST:DEXChange:MODE`

Example:	See Example "List mode data exchange" on page 1077.
Usage:	Event
Manual operation:	See "Import / Export" on page 491

[:SOURce<hw>]:LIST:DEXChange:AFILe:EXTension <Extension>

Determines the extension of the ASCII file for import or export, or to query existing files.

Parameters:

<Extension>	TXT | CSV *RST: TXT
Example:	See Example "List mode data exchange" on page 1077.
Manual operation:	See "ASCII File Settings" on page 490

[:SOURce<hw>]:LIST:DEXChange:AFILe:SELect <Filename>

Selects the ASCII file to be imported or exported.

Parameters:

<Filename>	string Filename or complete file path; file extension can be omitted.
Example:	See Example "List mode data exchange" on page 1077.

Manual operation: See "Select Source/Select Destination" on page 491

[:SOURce<hw>]:LIST:DEXChange:AFILe:SEParator:COLumn <Column>

Selects the separator between the frequency and level column of the ASCII table.

Parameters:

<Column> TABulator | SEMicolon | COMMa | SPACe

*RST: COMMa

Example: See Example "List mode data exchange" on page 1077.

Manual operation: See "ASCII File Settings" on page 490

[:SOURce<hw>]:LIST:DEXChange:AFILe:SEParator:DECimal <Decimal>

Sets "." (decimal point) or "," (comma) as the decimal separator used in the ASCII data with floating-point numerals.

Parameters:

<Decimal> DOT | COMMa

*RST: DOT

Example: See Example "List mode data exchange" on page 1077.

Manual operation: See "ASCII File Settings" on page 490

[:SOURce<hw>]:LIST:DEXChange:MODE <Mode>

Determines the import or export of a list.

Specify the source or destination file with the command `[:SOURce<hw>]:LIST:DEXChange:SELect`.

Parameters:

<Mode> IMPort | EXPort

*RST: IMPort

Example: See Example "List mode data exchange" on page 1077.

Manual operation: See " Mode " on page 490

[:SOURce<hw>]:LIST:DEXChange:SELect <Filename>

Selects the ASCII file for import or export, containing a list.

Parameters:

<Filename> string

Filename or complete file path; file extension can be omitted.

Example: See Example "List mode data exchange" on page 1077.

Manual operation: See "Select Source / Select ASCII Destination" on page 491

13.18.12 SOURce:NOISe Subsystem

The `SOURce:NOISe` subsystem contains the commands for setting the noise modulation signal. The noise generator is optional.

Example: Configuring the noise generator

```
SOURce1:NOISe:DISTribution GAUSs
SOURce1:NOISe:BANDwidth 10000000
SOURce1:NOISe:BANDwidth:STATe 1

SOURce1:LFOutput1:SOURce NOIS
SOURce1:LFOutput1:STATe 1

SOURce1:NOISe:LEVel1:RELative?
SOURce1:NOISe:LEVel1:ABSolute?
```

[:SOURce<hw>]:NOISe:BANDwidth|BWIDth <BWidth>

Sets the noise level in the system bandwidth when bandwidth limitation is enabled.

You can set distinct bandwidths between 10 kHz and 10 MHz:

- 100 kHz steps (range 100 .. 1 MHz)
- 1 MHz (range 1 MHz .. 5 MHz)
- 5 MHz (5 MHz … 10 MHz)

Parameters:

<BWidth> float

Range: 100E3 to 10E6
Increment: 100E3
*RST: 100E3

Example: See Example "Configuring the noise generator" on page 1088.

Manual operation: See "Bandwidth" on page 506

[:SOURce<hw>]:NOISe:BWIDth:STATe <State>

Activates noise bandwidth limitation.

Parameters:

<State> 0 | 1 | OFF | ON

*RST: 0

Example: See Example "Configuring the noise generator" on page 1088.

Manual operation: See "Bandwidth" on page 506

[:SOURce<hw>]:NOISe:DISTribution <Distribution>

Sets the distribution of the noise power density.

Parameters:

<Distribution>	GAUSs | EQUal	
	*RST:	GAUSs

Example: See Example "Configuring the noise generator" on page 1088.

Manual operation: See "Distribution" on page 505

[:SOURce<hw>]:NOISe:LEVel<ch>:RELative?

Queries the level of the noise signal per Hz in the total bandwidth.

Return values:

<Relative>	float	
	Range:	-149.18 to -52.67
	Increment:	0.1
	*RST:	-69.84

Example: See Example "Configuring the noise generator" on page 1088.

Usage: Query only

Manual operation: See "Noise Density" on page 506

[:SOURce<hw>]:NOISe:LEVel<ch>[:ABSolute]?

Queries the level of the noise signal in the system bandwidth within the enabled bandwidth limitation.

Return values:

<Absolute>	float	
	*RST:	3.84 MHz

Example: See Example "Configuring the noise generator" on page 1088.

Usage: Query only

Manual operation: See "Noise Level" on page 506

13.18.13 SOURce:PGEN Subsystem

The `PGEN` subsystem contains the commands for setting output of the pulse modulation signal.

Example: Using pulse generator as source for pulse modulation

```
// select pulse generator as source for pulse modulation
// enable pulse modulation
SOURce1:PULM:SOURce INT
SOURce1:PULM:STATe 1
// pulse generator and signal output are also activated
SOURce1:PGENerator:STATe?
// 1
SOURce1:PGENerator:OUTPut:STATe?
// 1
PGENerator:OUTPut:POLarity NORMal
// to disable pulse generator
SOURce1:PGENerator:STATe 0
// activate the pulse modulation of the RF carrier
SOURce1:PULM:STATe 1
```

[:SOURce<hw>]:PGENerator:OUTPut:POLarity <Polarity>

Sets the polarity of the pulse output signal.

Parameters:

<Polarity> NORMal | INVerted

NORMal
Outputs the pulse signal during the pulse width, that means during the high state.

INVerted
Inverts the pulse output signal polarity. The pulse output signal is suppressed during the pulse width, but provided during the low state.

*RST: NORMal

Example: See Example "Using pulse generator as source for pulse modulation" on page 1090.

Manual operation: See "Pulse Output Polarity" on page 516

[:SOURce<hw>]:PGENerator:OUTPut[:STATe] <State>

Activates the output of the pulse modulation signal.

Parameters:

<State> 0 | 1 | OFF | ON

*RST: 0

Example: See Example "Using pulse generator as source for pulse modulation" on page 1090.

Manual operation: See "Pulse Output State" on page 516

[:SOURce<hw>]:PGENerator:STATe <State>

Enables the output of the video/sync signal.

If the pulse generator is the current modulation source, activating the pulse modulation automatically activates the signal output and the pulse generator.

Parameters:

<State> 0 | 1 | OFF | ON

*RST: 0

Example: See Example "Using pulse generator as source for pulse modulation" on page 1090.

13.18.14 SOURce:PHASe Subsystem

This subsystem contains the commands for adjusting the phase of the RF output signal relative to a reference signal of the same frequency.

Example: Programming Example

```
// change the phase relative to the current phase
SOURce1:PHASe 2 DEG
// adopt the setting as the current phase
SOURce1:PHASe:REFerence
```

The following commands are available:

[:SOURce<hw>]:PHASe <Phase>

Specifies the phase variation relative to the current phase.

Parameters:

<Phase> float

Range: -720 to 720
Increment: 0.1
*RST: 0
Default unit: DEG

Example: See Example "Programming Example" on page 1091.

Manual operation: See "Delta Phase" on page 439

[:SOURce<hw>]:PHASe:REFerence

Assigns the value set with command `[:SOURce<hw>]:PHASe` as the reference phase.

Example: See Example "Programming Example" on page 1091.

Usage: Event

Manual operation: See "Reset Delta Phase Display" on page 440

13.18.15 SOURce:POWer Subsystem

The `SOURce:POWer` subsystem contains the commands for setting the output level, level control and level correction of the RF signal.

The default units are dBm. To change the units, perform on of the following:

- Enter the unit after the numerical value
  Example: `:POW 0.5V`
- Set the unit with the command `:UNIT:POWer`.

[:SOURce<hw>]:POWer:ALC:DAMPlifier <Amplifier>

Selects the driver amplifier switching state.

Parameters:

<Amplifier> OFF | ON | AUTO | FIX | ONMG

AUTO
Switches the attenuator automatically.

ON|OFF
Switches on or off the driver amplifier.

FIXed
Fixes the last setting.

ONMG
Supplies maximum level at the output.

*RST: AUTO

Example: `POW:ALC:DAMP AUTO`
Provides automatic switching of the attenuator.

Manual operation: See "Driver Amplifier" on page 528

[:SOURce<hw>]:POWer:ALC:DSENsitivity <Sensitivity>

Sets the sensitivity of the ALC detector.

Parameters:

<Sensitivity> AUTO | FIXed | LOW | MEDium | HIGH

AUTO
Selects the optimum sensitivity automatically.

LOW|MEDium|HIGH
Sets either low, medium or high sensitivity.

FIXed
Fixes the internal level detector.

*RST: AUTO

Example: `POW:ALC:DSEN FIX`

Manual operation: See "Detector Sensitivity" on page 527

[:SOURce<hw>]:POWer:ALC:OMODe <OffMode>

Sets the level control mode. It is activated when automatic level control is switched off (ALC Off).

Parameters:

<OffMode> SHOLd

Activates level control temporarily, when the level or frequency changes ("ALC Off Sample & Hold").

*RST: SHOLd

Example: POW:ALC OFF
Deactivates automatic level control at the RF output.
POW:ALC:OMOD SHOL
Activates the level control briefly when changing the frequency or level.

[:SOURce<hw>]:POWer:ALC:SEARch <Search>

Recalculates the instrument internal settings optimized for the current level. Not required for automatic modes.

Parameters:
<Search> ON
*RST: ON

Example: POW:ALC:SEAR ON
Executes readjustment.

[:SOURce<hw>]:POWer:ALC:SLEVel <SampLevel>

Sets the sample level of automatic level control (ALC).

NOTICE: Risk of DUT damage due to high input power

If you generate signals with large headroom using POW:ALC:SLEV FULL in combination with POW:ALC:STAT OFF or POW:ALC:STAT ONS, the R&S SMW generates a short sample pulse with full RF output power.

To protect the DUT from damage due to high input power, use the setting POW:ALC:SLEV ATT or POW:ALC:SLEV MIN.

Parameters:
<SampLevel> FULL | MINimum | ATTenuated
*RST: FULL

Example: POW:ALC:SLEV MIN
Sets the automatic level control to minimum.

Manual operation: See "Sample Level" on page 528

[:SOURce<hw>]:POWer:ALC:SONCe

Activates level control for correction purposes temporarily.

Example: POW:ALC OFF
Deactivates automatic level control at the RF output.
POW:ALC:SONC
Executes level control (once).

Usage: Event

Manual operation: See "Readjust" on page 437

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

Activates automatic level control in the selected mode.

Note: The ALC states `POW:ALC:STAT OFF` and `POW:ALC:STAT ONS` use a short sample pulse to set the level control. The level of this sample pulse corresponds to the sample level setting of the full PEP of the R&S SMW. It does not consider any signal headroom, e.g. applied to a user-defined ARB waveform.

NOTICE: Risk of DUT damage due to high input power

If you generate signals with large headroom using POW:ALC:SLEV FULL in combination with `POW:ALC:STAT OFF` or `POW:ALC:STAT ONS`, the R&S SMW generates a short sample pulse with full RF output power.

To protect the DUT from damage due to high input power, use the setting POW:ALC:SLEV ATT or POW:ALC:SLEV MIN.

Parameters:

<State> AUTO | OFFTable | ON | 1 | OFF | 0 | ONSample | ONTable

AUTO
Adjusts the output level to the operating conditions automatically.

OFFTable
Controls the level with the attenuation values of the internal ALC table.

ON
Activates internal level control permanently.

OFF
Deactivates internal level control, "Sample & Hold" mode is active.

ONSample
Starts the internal level control with the first change.

ONTable
Starts with the attenuation setting from the table and continues with automatic level control.

*RST: AUTO

Example: `POW:ALC ON`
Activates internal level control.

Manual operation: See "State" on page 527

[:SOURce<hw>]:POWer:ATTenuation <Attenuation>

Sets the attenuation value of the RF signal in manual mode, set with command `:OUTPut<hw>:AMODe`.

Parameters:

<Attenuation> integer

Range: depends on the installed options
Increment: 4|5|6
*RST: 0

Example: `SOURce1:POWer:ATTenuation 20dB`

Manual operation: See " Attenuation " on page 523

[:SOURce<hw>]:POWer:ATTenuation:DIGital <AttDigital>

Sets a relative attenuation value for the baseband signal.

Parameters:

<AttDigital> float

Range: -3.522 to 80
Increment: 1E-3
*RST: 0

Manual operation: See "Digital Attenuation" on page 435

[:SOURce<hw>]:POWer:ATTenuation:RFOFf:MODE <Mode>

Selects the state the attenuator is to assume if the RF signal is switched off.

Parameters:

<Mode> UNCHanged | FATTenuation

FATTenuation
The electronic step attenuator switches to maximum attenuation when RF is off.
Note: This setting overrides the RF level modes : `OUTPut<hw>:AMODe FIXed|MANual`.

UNCHanged
Freezes the current setting to keep the output impedance constant during RF off.

*RST: n.a. (factory preset: FATTenuation)

Example: `SOUR:POW:ATT:RFOF:MODE FATT`
Sets the attenuation to maximum when RF is switched off.

Manual operation: See " RF OFF Mode " on page 523

[:SOURce<hw>]:POWer:EMF:STATe <State>

Displays the signal level as voltage of the EMF. The displayed value represents the voltage over a 50 Ohm load.

Parameters:

<State> 0 | 1 | OFF | ON

*RST: n.a. (factory preset: 0)

Example: `POW:EMF:STAT 1`
Activates voltage level display.

Manual operation: See " Display Level as Voltage of EMF " on page 660

[:SOURce<hw>]:POWer:LBEHaviour <Behaviour>

Selects the level behavior at the RF output over time.

Parameters:

<Behaviour> AUTO | UNINterrupted | MONotone | CVSWr | CPHase

UNINterrupted|MONotone
Do not use the uninterrupted level settings and strictly monotone modes in combination with the high-quality optimization mode (see `[:SOURce<hw>]:BB:IMPairment:OPTimization:MODE` on page 1015)

CWSWr
Constant VSWR

CPHase
Continuous phase

*RST: AUTO

Example: `SOURce1:POWer:LBEHaviour AUTO`

Manual operation: See " Setting Characteristics " on page 436

[:SOURce<hw>]:POWer:LIMit[:AMPLitude] <Amplitude>

Limits the maximum RF output level in CW and sweep mode.

It does not influence the "Level" display or the response to the query `[:SOURce<hw>]:POWer[:LEVel][:IMMediate][:AMPLitude]`.

Parameters:

<Amplitude> float

Range: depends on the installed options
Increment: 0.01
*RST: n.a. (factory preset: 30)

Example: `SOURce1:POWer:LIMit:AMPLitude 10`

Manual operation: See " Limit " on page 438

[:SOURce<hw>]:POWer:LMODe <LevMode>

Sets the RF level mode.

Parameters:

<LevMode> NORMal | LOWNoise | LOWDistortion

NORMal
Supplies the RF signal with the standard power level of the instrument.

LOWNoise
Supplies a very low noise sinewave signal.

LOWDistortion
Supplies a very pure sinewave signal.

*RST: NORMal

Example: `SOURce1:POWer:LMODe LOWD`
Sets low distortion mode. The instrument reduces distortions of the RF signal to a minimum.

Manual operation: See " Mode " on page 436

[:SOURce<hw>]:POWer:MANual <Manual>

Sets the level for the subsequent sweep step if SWE:POW:MODE MAN.

Use a separate command for each sweep step.

Parameters:

<Manual> float

You can select any level within the setting range, where:
`STARt` is set with `[:SOURce<hw>]:POWer:STARt`
`STOP` is set with `[:SOURce<hw>]:POWer:STOP`
`OFFSet` is set with `[:SOURce<hw>]:POWer[:LEVel][:IMMediate]:OFFSet`

Range: (STARt + OFFSet) to (STOP + OFFSet)
Increment: 0.01
Default unit: dBm

Example: See Example "Setup an RF frequency or power sweep" on page 1107

Manual operation: See " Current Level" on page 479

[:SOURce<hw>]:POWer:MODE <Mode>

Selects the operating mode of the instrument to set the output level.

Parameters:

<Mode>	CW | FIXed | SWEep **CW|FIXed** Operates at a constant level. CW and FIXed are synonyms. To set the output level value, use the command [:SOURce<hw>]:POWer[:LEVel][:IMMediate][:AMPLitude]. **SWEep** Sets sweep mode. Set the range and current level with the commands: [:SOURce<hw>]:POWer:STARt and [:SOURce<hw>]:POWer:STOP, [:SOURce<hw>]:POWer:MANual. *RST: CW

Example: See Example "Setup an RF frequency or power sweep" on page 1107

Manual operation: See "State (RF level sweep)" on page 476

[:SOURce<hw>]:POWer:PEP?

Queries the PEP "**P**eak **E**nvelope **P**ower) of digital modulation or digital standards at the RF output. This value corresponds to the level specification, displayed in the status bar (header).

Return values:

<Pep> float

Example: POW:PEP?
Response: "4"
The PEP value of digital modulation is 4 dBm at the RF output.

Usage: Query only

[:SOURce<hw>]:POWer:POWer <Power>

Sets the level **at the RF output** connector.

This value does not consider a specified offset.

The command [:SOURce<hw>]:POWer[:LEVel][:IMMediate][:AMPLitude] sets the level of the "Level" display, that means the level containing offset.

See Chapter 7.3.4.1, "Displayed RF Frequency and Level Values with Downstream Instruments", on page 440.

Parameters:

<Power> float

Level at the RF output, without level offset

Range: See data sheet
Increment: 0.01
Default unit: dBm

Example: `SOURce1:POWer:POWer 15`
Sets the level at RF output

Manual operation: See "Amplitude" on page 435

[:SOURce<hw>]:POWer:STARt <Start>
[:SOURce<hw>]:POWer:STOP <Stop>

Sets the RF start/stop level in sweep mode.

Parameters:

<Stop> float

Sets the setting range calculated as follows:
`(Level_min + OFFSet) to (Level_max + OFFSet)`
Where the values are set with the commands:
`[:SOURce<hw>]:POWer[:LEVel][:IMMediate]:OFFSet`
`[:SOURce<hw>]:POWer:STARt`
`[:SOURce<hw>]:POWer:STOP`

Range: Minimum level to maximum level
*RST: -30 (Start)/ -10 (Stop)
Default unit: dBm

Example: See Example "Setup an RF frequency or power sweep" on page 1107

Manual operation: See " Start Level / Stop Level " on page 478

[:SOURce<hw>]:POWer:STEP:MODE <Mode>

Defines the type of step width to vary the RF output power step-by-step with the commands POW UP or POW DOWN.

You can define the step mode for each path separately.

Parameters:

<Mode> DECimal | USER

DECimal
Increases or decreases the level in steps of ten.

USER
Increases or decreases the level in increments, determined with the command `[:SOURce<hw>]:POWer:STEP[:INCRement]`.

*RST: DECimal

Example:
```
// increasing the RF level with a step size of 2 dB
SOURce1:POWer:STEP:INCRement 2
SOURce1:POWer:STEP:MODE USER
SOURce1:POWer:LEVel:IMMediate:AMPLitude UP
```

Manual operation: See "Variation Active" on page 433

[:SOURce<hw>]:POWer:STEP[:INCRement] <Increment>

Specifies the step width in the appropriate path for POW:STEP:MODE USER.

To adjust the level step-by-step with this increment value, use the command POW UP, or POW DOWN.

Note: The command also sets "Variation Step" in the manual control, that means the user-defined step width for setting the level with the rotary knob or the UP/DOWN arrow keys.

Parameters:

<Increment> float

Range: 0 to 100
Increment: 0.01
*RST: 1
Default unit: dB

Example: See `[:SOURce<hw>]:POWer:STEP:MODE` on page 1100.

Manual operation: See "Variation Step" on page 433

[:SOURce<hw>]:POWer[:LEVel][:IMMediate]:OFFSet <Offset>

Sets the level offset of a downstream instrument.

The level at the RF output is not changed.

To query the resulting level, as it is at the output of the downstream instrument, use the command `[:SOURce<hw>]:POWer[:LEVel][:IMMediate][:AMPLitude]`.

See Chapter 7.3.4.1, "Displayed RF Frequency and Level Values with Downstream Instruments", on page 440.

Note: The level offset also affects the RF level sweep.

Parameters:

<Offset> float

Range: -100 to 100
Increment: 0.01
*RST: 0
Default unit: dB
Level offset is always expreced in dB; linear units (V, W, etc.) are not supported

Example: `POWer:OFFSet 10`
Sets the RF level offset to 10 dB

Manual operation: See " Offset " on page 438

[:SOURce<hw>]:POWer[:LEVel][:IMMediate]:RCL <Rcl>

Determines whether the current level is retained or if the stored level setting is adopted when an instrument configuration is loaded.

The setting is valid for both paths. If a suffix is specified, it is ignored.

Parameters:

<Rcl> INCLude | EXCLude

INCLude
Takes the current level when an instrument configuration is loaded.

EXCLude
Retains the current level when an instrument configuration is loaded.

*RST: INCLude

Example: `POW:RCL INCL`
Takes the level value from an instrument configuration loaded with command `*RCL`.

Manual operation: See "Exclude Level" on page 632

[:SOURce<hw>]:POWer[:LEVel][:IMMediate][:AMPLitude] <Amplitude>

Sets the RF level applied to the DUT.

To activate the RF output use command `:OUTPut<hw>[:STATe]` ("RF On"/"RF Off").

The following applies `POWer = RF output level + OFFSet`, where:

- `POWer` is the values set with `[:SOURce<hw>]:POWer[:LEVel][:IMMediate][:AMPLitude]`
- RF output level is set with `[:SOURce<hw>]:POWer:POWer`
- `OFFSet` is set with `[:SOURce<hw>]:POWer[:LEVel][:IMMediate]:OFFSet`

Parameters:

<Amplitude> float

The following settings influence the value range:
OFFSet set with the command [:SOURce<hw>]:POWer[:LEVel][:IMMediate]:OFFSet

Numerical value
Sets the level

UP|DOWN
Varies the level step by step.
The level is increased or decreased by the value set with the command [:SOURce<hw>]:POWer:STEP[:INCRement].

Range: (Level_min + OFFSet) to (Level_max + OFFStet)
*RST: -30
Default unit: dBm

Example: POWer -30
Sets the RF level

Example: See also [:SOURce<hw>]:POWer:STEP:MODE on page 1100.

Manual operation: See "Amplitude" on page 435

13.18.16 SOURce:ROSCillator Subsystem

The SOURce:ROSCillator subsystem contains the commands for setting the external and internal reference frequency.

The commands of this subsystem are not affected by an instrument reset (*RST on page 772).

Example: Configuring the reference oscillator

```
// Using 10 MHz external reference source
SOURce:ROSCillator:SOURce EXT
SOURce:ROSCillator:EXTernal:RFOFf:STATe 1
SOURce:ROSCillator:EXTernal:FREQuency 10MHZ
SOURce:ROSCillator:EXTernal:SBANdwidth WIDE
SOURce:ROSCillator:INTernal:ADJust:STATe 0
```

[:SOURce]:ROSCillator:SOURce <Source>

Selects between internal or external reference frequency.

Parameters:

<Source> INTernal | EXTernal

*RST: n.a. (factory preset: INTernal)

Example: See Example "Configuring the reference oscillator" on page 1103.

Manual operation: See "Source" on page 446

[:SOURce]:ROSCillator:EXTernal:RFOFf[:STATe] <State>

Determines that the RF output is turned off when the external reference signal is selected, but missing.

Parameters:

<State> 0 | 1 | OFF | ON

*RST: n.a. (factory preset: 0)

Example: See Example "Configuring the reference oscillator" on page 1103.

Manual operation: See "Deactivate RF Output (if external reference is missing)" on page 446

[:SOURce]:ROSCillator:EXTernal:FREQuency <Frequency>

Sets the frequency of the external reference.

Parameters:

<Frequency> 10MHZ | VARiable | 5MHZ | 13MHZ

*RST: n.a. (factory preset: 10MHZ)

Example: See Example "Configuring the reference oscillator" on page 1103.

Manual operation: See "External Reference Frequency" on page 446

[:SOURce]:ROSCillator:EXTernal:FREQuency:VARiable <Frequency>

Specifies the user-defined external reference frequency.

Parameters:

<Frequency> float

Range: 1E6 to 100E6
Increment: 0.1
*RST: n.a. (factory preset: 1E7)
Default unit: Hz

Example: See Example "Configuring the reference oscillator" on page 1103.

Manual operation: See "Variable Reference Frequency" on page 447

[:SOURce]:ROSCillator:EXTernal:SBANdwidth <SBandwidth>

Selects the synchronization bandwidth for the external reference signal.

Parameters:

<SBandwidth> WIDE | NARRow

NARRow
The synchronization bandwidth is a few Hz (with option R&S SMW-B22) or approximately 20 Hz otherwise.

WIDE
The synchronization bandwidth is a 100 Hz (with option R&S SMW-B22) and approximately 750 Hz otherwise.

*RST: n.a. (factory preset)

Example: See Example "Configuring the reference oscillator" on page 1103.

Manual operation: See "Synchronization Bandwidth" on page 447

[:SOURce]:ROSCillator:OUTPut:FREQuency:MODE <OutpFreqMode>

Selects the mode for the determination and output of the reference frequency.

Parameters:

<OutpFreqMode> DER10M | SAME

*RST: n.a. (factory preset: DER10M)

Example: See Example "Configuring the reference oscillator" on page 1103.

Manual operation: See "Reference Frequency Output" on page 447

[:SOURce]:ROSCillator[:INTernal]:ADJust:VALue <Value>

Specifies the frequency correction value (adjustment value).

Parameters:

<Value> integer

*RST: 0

Example: See `[:SOURce]:ROSCillator[:INTernal]:ADJust[:STATe]` on page 1106

Manual operation: See "Adjustment DAC Value" on page 448

[:SOURce]:ROSCillator[:INTernal]:ADJust[:STATe] <State>

Determines whether the calibrated (off) or a user-defined (on) **adjustment value** is used for fine adjustment of the frequency.

Parameters:

<State> 0 | 1 | OFF | ON

0
Fine adjustment with the calibrated frequency value

1
User-defined adjustment value
The instrument is no longer in the calibrated state.
The calibration value is, however, not changed. The instrument resumes the calibrated state if you send
`SOURce:ROSCillator:INTernal:ADJust:STATe 0`.

*RST: n.a. (factory preset: 0)

Example:

```
// query calibration value
CALibration:ROSCillator?
// 32767
// Set an internal source
// Activate user-defined adjustment value of 1000
SOURce:ROSCillator:SOURce INT
SOURce:ROSCillator:INTernal:ADJust:STATe 1
SOURce:ROSCillator:INTernal:ADJust:VALue 1000

// to resume calibrated state
SOURce:ROSCillator:INTernal:ADJust:VALue 0
SOURce:ROSCillator:INTernal:ADJust:STATe 0
// or
// SYSTem:FPRest
```

Manual operation: See "Adjustment Active" on page 448

13.18.17 SOURce:SWEep Subsystem

The `SOURce:SWEep` subsystem contains the commands for configuring RF sweep signals.

- The keyword `[:FREQuency]` can be omitted, then the commands are SCPI-compliant.
- To activate an RF sweep mode, use the following commands:
  - RF frequency sweep: `SOURce:FREQuency:MODE SWEep` (`SOURce:FREQuency:MODE CW` (off))
  - RF level sweep: `SOURce:POWer:MODE SWEep` (`SOURce:POWer:MODE CW` (off))
- All sweeps, including the LF sweep, can be set independently of each other.

For detailed information on the sweep modes and the triggering, see Chapter 7.6.1, "Signal Generation and Triggering in the Sweep and List Modes", on page 459.

Example: Setup an RF frequency or power sweep

The following example shows a command sequence to set up an RF frequency sweep, triggered by the execute command. For an RF power sweep, replace `FREQuency` in the `SWEep` commands with `POWer`.

Exceptions are the power spacing (defined with `LINear` only) and the power step width (defined with `LOGarithmic` only).

```
// Reset the instrument to start from an initial state
// Switch off display update to improve performance
// (especially with short dwell times)
// Set the sweep mode (first two commands) and the sweep range
// Select linear spacing
// Select the waveform shape for the frequency sweep
*RST; *CLS
SYSTem:DISPlay:UPDate OFF
TRIGger1:FSWeep:SOURce SINGle
SOURce1:SWEep:FREQuency:MODE AUTO
SOURce1:FREQuency:SPAN 300 MHz
SOURce1:FREQuency:CENTer 200 MHz
// Alternatively use
// SOURce1:FREQuency:STARt 50 MHz
// SOURce1:FREQuency:STOP 350 MHz
SOURce1:SWEep:FREQuency:SPACing LINear
SOURce1:SWEep:FREQuency:SHAPe SAWTooth

// Activate change to start frequency while waiting for next trigger
// Prerequisites: sweep mode single and sweep waveform sawtooth
SOURce1:SWEep:FREQuency:RETRace 1
// Alternatively reset all sweeps to their initial value
SOURce1:SWEep:RESet:ALL

// Set the step width and dwell time
SOURce1:SWEep:FREQuency:STEP:LINear 1 MHz
// Alternatively set the number of steps, then the sweep step width is
// set automatically
SOURce1:SWEep:FREQuency:POINts 301
```

```
SOURce1:SWEep:FREQuency:DWELl 500 ms
// With logarithmic spacing select the step width as follows
// (steps of 10 percent of the previous frequency in each instance)
SOURce1:SWEep:FREQuency:SPACing LOG
SOURce1:SWEep:FREQuency:STEP:LOGarithmic 10PCT

// Activate the sweep
// Trigger the sweep (depending on the set mode) and query the status
SOURce1:FREQuency:MODE SWEep
// Perform a one-off RF frequency sweep
SOURce1:SWEep:FREQuency:EXECute
SOURce1:SWEep:FREQuency:RUNNing?
// 1
// the frequency sweep is running

// For manual step RF sweep use the following commands
*RST; *CLS
// Activate manual step RF sweep
SOURce1:SWEep:FREQuency:MODE MANual
// Activate the RF frequency sweep.
SOURce1:FREQuency:MODE SWEep
// Activate RF Output1.
Output1:STATe 1
// Input the frequency manually for each step
SOURce1:FREQuency:MANual 200 MHz
SOURce1:FREQuency:MANual 201 MHz
// Alternatively use the UP or DOWN commands with the set step width.
SOURce1:SWEep:FREQuency:STEP:LINear 1 MHz
SOURce1:FREQuency:MANual UP
```

[:SOURce<hw>]:SWEep:POWer:AMODe <AMode>

Selects the power attenuator mode for the level sweep.

Parameters:

<AMode> NORMal | HPOWer

NORMal
Performs the level settings in the range of the built-in attenuator.

HPOWer
Performs the level settings in the high level range.

*RST: NORMal(HighPower)|AUTO

Example: `SWE:POW:AMOD NORM`
Selects the high level ranges for level sweep.

[:SOURce<hw>]:SWEep:POWer:DWELl <Dwell>

Sets the dwell time for a level sweep step.

Parameters:

<Dwell> float

Range: 1E-3 to 100
Increment: 100E-6
*RST: 15E-3
Default unit: s

Example: See Example "Setup an RF frequency or power sweep" on page 1107.

Manual operation: See " Dwell Time " on page 480

[:SOURce<hw>]:SWEep:POWer:MODE <Mode>

Sets the cycle mode for the level sweep.

Parameters:

<Mode> AUTO | MANual | STEP

AUTO
Each trigger triggers exactly one complete sweep.

MANual
The trigger system is not active. You can trigger every step individually with the command `[:SOURce<hw>]:POWer:MANual`.
The level value increases at each step by the value that you define with `[:SOURce<hw>]:POWer:STEP[:INCRement]`.
Values directly entered with the command `[:SOURce<hw>]:POWer:MANual` are not taken into account.

STEP
Each trigger triggers one sweep step only. The level increases by the value entered with `[:SOURce<hw>]:POWer:STEP[:INCRement]`.

*RST: AUTO

Example: See Example "Setup an RF frequency or power sweep" on page 1107.

Manual operation: See "Mode" on page 476

[:SOURce<hw>]:SWEep:POWer:POINts <Points>

Sets the number of steps within the RF level sweep range.

See Chapter 7.6.2.1, "Correlating Parameters in Sweep Mode", on page 467.

Parameters:

<Points> integer

Range: 2 to Max

Example: See Example "Setup an RF frequency or power sweep" on page 1107.

[:SOURce<hw>]:SWEep:POWer:SPACing:MODE?

Queries the level sweep spacing. The sweep spacing for level sweeps is always linear.

Return values:

<Mode> LINear

*RST: LINear

Example: `SWE:POW:SPAC:MODE?`
queries the sweep spacing for a level sweep at RF output.
Result: `"LIN"`
linear spacing

Usage: Query only

[:SOURce<hw>]:SWEep:POWer:STEP[:LOGarithmic] <Logarithmic>

Sets a logarithmically determined step width for the RF level sweep. The level is increased by a logarithmically calculated fraction of the current level.

See Chapter 7.6.2.1, "Correlating Parameters in Sweep Mode", on page 467.

Parameters:

<Logarithmic> float

The unit `dB` is manatory.

Range: 0.01 to 139 dB
Increment: 0.01
*RST: 1
Default unit: dB

Example: See Example "Setup an RF frequency or power sweep" on page 1107.

Manual operation: See " Step " on page 480

[:SOURce<hw>]:SWEep[:FREQuency]:DWELl <Dwell>

Sets the dwell time for a frequency sweep step.

Parameters:

<Dwell> float

Range: 2E-3 to 100
Increment: 100E-6
*RST: 15E-3
Default unit: s

Example: See Example "Setup an RF frequency or power sweep" on page 1107.

Manual operation: See " Dwell Time " on page 480

[:SOURce<hw>]:SWEep[:FREQuency]:MODE <Mode>

Sets the cycle mode for the frequency sweep.

Parameters:

<Mode> AUTO | MANual | STEP

AUTO
Each trigger event triggers exactly one complete sweep.

MANual
The trigger system is not active. You can trigger every step individually by input of the frequencies with the command `[:SOURce<hw>]:FREQuency:MANual`.

STEP
Each trigger event triggers one sweep step. The frequency increases by the value entered with `[:SOURce<hw>]:SWEep[:FREQuency]:STEP[:LINear]` (linear spacing) or `[:SOURce<hw>]:SWEep[:FREQuency]:STEP:LOGarithmic` (logarithmic spacing).

*RST: AUTO

Example: See Example "Setup an RF frequency or power sweep" on page 1107.

Manual operation: See "Mode" on page 476

[:SOURce<hw>]:SWEep[:FREQuency]:POINts <Points>

Sets the number of steps within the RF frequency sweep range.

See Chapter 7.6.2.1, "Correlating Parameters in Sweep Mode", on page 467.

Two separate `POINts` values are used for linear or logarithmic sweep spacing (`[:SOURce<hw>]:SWEep[:FREQuency]:SPACing` LIN | LOG). The command always affects the currently set sweep spacing.

Parameters:

<Points> integer

Range: 2 to Max

Example: See Example "Setup an RF frequency or power sweep" on page 1107.

[:SOURce<hw>]:SWEep[:FREQuency]:SPACing <Spacing>

Selects the mode for the calculation of the frequency intervals, with which the current frequency at each step is increased or decreased.

The keyword `[:FREQuency]` can be omitted; then the command is SCPI-compliant.

Parameters:

<Spacing>	LINear | LOGarithmic **LINear** Sets a fixed frequency value as step width and adds it to the current frequency. The linear step width is entered in Hz, see `[:SOURce<hw>]:SWEep[:FREQuency]:STEP[:LINear]`. **LOGarithmic** Sets a constant fraction of the current frequency as step width and adds it to the current frequency. The logarithmic step width is entered in %, see `[:SOURce<hw>]:SWEep[:FREQuency]:STEP:LOGarithmic`. *RST: LINear

Example: See Example "Setup an RF frequency or power sweep" on page 1107.

Manual operation: See " Spacing" on page 479

[:SOURce<hw>]:SWEep:POWer:SHAPe <Shape>
[:SOURce<hw>]:SWEep[:FREQuency]:SHAPe <Shape>

Determines the waveform shape for a frequency sweep sequence.

Parameters:

<Shape>	SAWTooth | TRIangle *RST: SAWTooth

Example: See Example "Setup an RF frequency or power sweep" on page 1107.

Manual operation: See " Shape " on page 479

[:SOURce<hw>]:SWEep:POWer:EXECute
[:SOURce<hw>]:SWEep[:FREQuency]:EXECute

Executes an RF frequency sweep.

The command performs a single sweep and is therefore only effective in manual sweep mode.

Example: See Example "Setup an RF frequency or power sweep" on page 1107.

Usage: Event

Manual operation: See " Execute Single Sweep " on page 477

[:SOURce<hw>]:SWEep:POWer:RETRace <State>
[:SOURce<hw>]:SWEep[:FREQuency]:RETRace <State>

Activates that the signal changes to the start frequency value while it is waiting for the next trigger event.

You can enable this feature, when you are working with sawtooth shapes in sweep mode "Single" or "External Single".

Parameters:

<State> 0 | 1 | OFF | ON

*RST: 0

Example: See Example "Setup an RF frequency or power sweep" on page 1107.

Manual operation: See "Retrace" on page 479

[:SOURce<hw>]:SWEep:POWer:RUNNing?
[:SOURce<hw>]:SWEep[:FREQuency]:RUNNing?

Queries the current sweep state.

Return values:

<State> 0 | 1 | OFF | ON

Example: See Example "Setup an RF frequency or power sweep" on page 1107.

Usage: Query only

[:SOURce<hw>]:SWEep[:FREQuency]:STEP:LOGarithmic <Logarithmic>

Sets a logarithmically determined step width for the RF frequency sweep. The value is added at each sweep step to the current frequency.

See Chapter 7.6.2.1, "Correlating Parameters in Sweep Mode", on page 467.

Parameters:

<Logarithmic> float

The unit is mandatory.

Range: 0.01 to 100
Increment: 1E-3
*RST: 1
Default unit: PCT

Example: See Example "Setup an RF frequency or power sweep" on page 1107.

Manual operation: See " Step Linear/Step Logarithmic " on page 480

[:SOURce<hw>]:SWEep[:FREQuency]:STEP[:LINear] <Linear>

Sets the step width for linear sweeps.

See Chapter 7.6.2.1, "Correlating Parameters in Sweep Mode", on page 467.

Omit the optional keywords so that the command is SCPI-compliant.

Parameters:

<Linear> float

Range: 0.001 Hz to (STOP - STARt)
Increment: 0.01

Example: See Example "Setup an RF frequency or power sweep" on page 1107.

Manual operation: See " Step Linear/Step Logarithmic " on page 480

[:SOURce<hw>]:SWEep:RESet[:ALL]

Resets all active sweeps to the starting point.

Usage: Event

Manual operation: See " Reset Sweep " on page 478

13.19 SYSTem Subsystem

The `SYSTem` subsystem contains a series of commands for general functions which do not directly affect signal generation.

Example: Retrieving information on network-related settings

```
SYSTem:COMMunicate:NETWork:STATus?
// 1
SYSTem:PROTection1:STATe 0,123456

SYSTem:COMMunicate:NETWork:IPADdress:MODE STAT
SYSTem:COMMunicate:NETWork:IPADdress "10.113.0.104"
SYSTem:COMMunicate:NETWork:IPADdress:DNS "10.0.2.166"
SYSTem:COMMunicate:NETWork:COMMon:HOSTname?
// "SMW200A-102030"
SYSTem:COMMunicate:NETWork:COMMon:WORKgroup "instrument"
SYSTem:COMMunicate:NETWork:COMMon:DOMain "rsint.net"
SYSTem:COMMunicate:NETWork:IPADdress:GATeway "10.113.0.1"
SYSTem:COMMunicate:NETWork:IPADdress:SUBNet:MASK "255.255.252.0"
SYSTem:COMMunicate:NETWork:MACaddress "08 00 27 a3 a1 70"
SYSTem:PROTection1:STATe 1
```

Example: Finding out the used VISA ressource strings

```
SYSTem:COMMunicate:NETWork:RESource?
// "TCPIP::10.113.0.104::inst0::INSTR"

SYSTem:COMMunicate:HISLip:RESource?
// "TCPIP::10.113.0.104::hislip0::INSTR"

SYSTem:COMMunicate:GPIB:RESource?
// "GPIB::28::INSTR"
SYSTem:COMMunicate:GPIB:SELF:ADDRess?
// 28
SYSTem:COMMunicate:GPIB:LTERminator?
// STAN

SYSTem:COMMunicate:SERial:RESource?
// "ASRL1::INSTR"
SYSTem:COMMunicate:SERial:SBITs?
// 1
SYSTem:COMMunicate:SERial:BAUD?
// 115200
SYSTem:COMMunicate:SERial:PARity?
// NONE

SYSTem:COMMunicate:SOCKet:RESource?
// "TCPIP::10.113.0.104::5025::SOCKET"
SYSTem:COMMunicate:USB:RESource?
// "USB::0x0AAD::0x0092::100001::INSTR"
```

Example: Querying the error queue

```
SYSTem:ERRor:STATic?
// -221,"Settings conflict", 153,"Input voltage out of range", ...
// returns all static errors that are collectred in the error queue

SYSTem:ERRor:HISTory?
// 90,"Info;(*)Instrument startup... (Mar-13-2017/ 10:25:16-601 ms)",
90,"Info;(*)Information generated while processing license keys.,
Repaired Error!
COND: ( hr == false )
FILE: /home/sa_okbuildserver/jenkins/workspace/OK-Legacy-Distribution-30/
ok_services_oklib/Src/CServiceExtension.cpp
LINE: 3554
ADDITIONAL INFO: Init ServiceExtension failed, 2877, -2147218613
HRESULT = 80001007
", 90,"Info;(A)Baseband info, [RF A] No frequency calibration data found.
Please run Adjust All!", 90,"Info;(A)Baseband info, [RF B] No frequency
calibration data found. Please run Adjust All!", 90,"Info;
(A)Baseband info, [BB A],...
// returns all entries of the error queue

SYSTem:ERRor:HISTory:ClEar
// Deletes the history entries
```

:SYSTem:ERRor:ALL?

Queries the error/event queue for all unread items and removes them from the queue. The response is a comma separated list of error number and a short description of the error in FIFO order.

Positive error numbers are instrument-dependent. Negative error numbers are reserved by the SCPI standard.

Return values:

<All> string

List of: Error/event_number,"Error/event_description>[;Device-dependent info]"
If the queue is empty, the response is `0,"No error"`

Example: `SYST:ERR:ALL?`
queries all entries in the error queue.
`Response: 0, 'no error'`
No errors have occurred since the error queue was last read out.

Usage: Query only

Manual operation: See "Clear History" on page 1176

:SYSTem:ERRor:CODE:ALL?

Queries all entries in the error queue and then deletes them. Only the error numbers are returned and not the entire error text.

Return values:

<All> string

0
"No error", i.e. the error queue is empty

positive value
Positive error numbers denote device-specific errors

negative value
Negative error numbers denote error messages defined by SCPI.

Example: `SYST:ERR:CODE:ALL`
queries all entries in the error queue.
`Response: 0`
no errors have occurred since the error queue was last read out.

Usage: Query only

:SYSTem:ERRor:CODE[:NEXT]?

Queries the oldest entry in the error queue and then deletes it. Only the error number is returned and not the entire error text.

Return values:

<Next> string

0
"No error", i.e. the error queue is empty

positive value
Positive error numbers denote device-specific errors

negative value
Negative error numbers denote error messages defined by SCPI.

Example: `SYST:ERR:CODE`
queries the oldest entry in the error queue.
`Response: 0`
No errors have occurred since the error queue was last read out.

Usage: Query only

:SYSTem:ERRor:COUNt?

Queries the number of entries in the error queue. If the error queue is empty, '0' is returned.

Return values:

<Count> integer

Example: `SYST:ERR:COUN`
queries the number of entries in the error queue.
`Response: 1`
One error has occurred since the error queue was last read out.

Usage: Query only

:SYSTem:ERRor[:NEXT]?

Queries the error/event queue for the oldest item and removes it from the queue. The response consists of an error number and a short description of the error.

Positive error numbers are instrument-dependent. Negative error numbers are reserved by the SCPI standard.

Return values:

<Next> string

Error/event_number,"Error/event_description>[;Device-dependent info]"
If the queue is empty, the response is `0,"No error"`

Example: `SYST:ERR?`
queries the oldest entry in the error queue.
`Response: 0, 'no error'`
No errors have occurred since the error queue was last read out.

Usage: Query only

Manual operation: See "Show History / Show Static" on page 1176

:SYSTem:ERRor:HISTory?

Queries the error history.

Note that the result can amount several kilobytes.

Return values:

<ErrorHistory> string

Example: See Example "Querying the error queue" on page 1117

Usage: Query only

:SYSTem:ERRor:HISTory:CLEar

Clears the error history.

Example: See Example "Querying the error queue" on page 1117

Usage: Event

Manual operation: See "Clear History" on page 1176

:SYSTem:ERRor:STATic?

Returns a list of all errors existing at the time when the query is started. This list corresponds to the display on the info page under manual control.

Return values:

<StaticErrors> string

Example: See Example "Querying the error queue" on page 1117

Usage: Query only

:SYSTem:ULOCk <Mode>

Locks or unlocks the user interface of the instrument.

Parameters:

<Mode> ENABled | DONLy | DISabled | TOFF | VNConly

ENABled
Unlocks the display, the touchscreen and all controls for the manual operation.

DONLy
Locks the touchscreen and controls for the rmanual operation of the instrument. The display shows the current settings.

VNConly
Locks the touchscreen and controls for the rmanual operation, and enables remote operation over VNC. The display shows the current settings.

TOFF
Locks the touchscreen for the rmanual operation of the instrument. The display shows the current settings.

DISabled
Locks the display, the touchscreen and all controls for the manual operation.

*RST: n.a. (factory preset: ENABled)

Example: `SYST:ULOC ON`
activates the user interface lock.

Manual operation: See "User Interface" on page 689

:SYSTem:DLOCk <DispLockStat>

Disables the manual operation via the display, including the front panel keyboard of the instrument and the LOCAL key.

Parameters:

<DispLockStat> 0 | 1 | OFF | ON

*RST: n.a. (factory preset: 0)

Example: `SYST:DLOC ON`
Activates the display lock. The instrument cannot be operated via the display until it has been enabled with `SYST:DLOC OFF`.

Manual operation: See "User Interface" on page 689

:SYSTem:KLOCk <State>

Disables the front panel keyboard of the instrument including the LOCAL key.

Parameters:

<State> 0 | 1 | OFF | ON

*RST: n.a. (factory preset: 0)

Example: `SYST:KLOC ON`
Locks the front panel and external controls.
To enable the controls, send `SYST:KLOC OFF`.

Manual operation: See "User Interface" on page 689

:SYSTem:PROTect<ch>[:STATe] <State>[, <Key>]

Activates and deactivates the specified protection level.

Suffix:

<ch> Indicates the protection level.
See also "Protection" on page 686

Parameters:

<State> 0 | 1 | OFF | ON

*RST: n.a. (factory preset: 1)

Setting parameters:

<Key> integer

The respective functions are disabled when the protection level is activated. No password is required for activation of a level.
A password must be entered to deactivate the protection level.
The default password for the first level is 123456.
This protection level is required to unlock internal adjustments for example.

Example:

```
// to activate protection level
SYSTem:PROTect1:STATe 1
// internal adjustments or hostname cannot be changed
// to unlock protection level 1
SYSTem:PROTect1:STATe 0,123456
// internal adjustments are accessible
```

Manual operation: See "Protection Level/Password" on page 688

:SYSTem:SECurity:VOLMode[:STATe] <SecPassWord>, <MmemProtState>

Activates volatile mode, so that no user data can be written to the internal memory permanently.

To enable volatile mode, reboot the instrument. Otherwise the change has no effect.

Parameters:

<MmemProtState> 0 | 1 | OFF | ON

*RST: 0

Setting parameters:

<SecPassWord> string

Current security password
The default password is 123456.

Example:

```
SYSTem:SECurity:VOLMode:STATe "123456", 1
SYSTem:REBoot
```

Manual operation: See "Volatile Mode" on page 689

:SYSTem:COMMunicate:GPIB:LTERminator <LTerminator>

Sets the terminator recognition for remote control via GPIB interface.

Parameters:

<LTerminator> STANdard | EOI

EOI
Recognizes an LF (Line Feed) as the terminator only when it is sent with the line message EOI (End of Line). This setting is recommended particularly for binary block transmissions, as binary blocks may coincidentally contain a characater with value LF (Line Feed), although it is not determined as a terminator.

STANdard
Recognizes an LF (Line Feed) as the terminator regardless of whether it is sent with or without EOI.

*RST: n.a. (factory preset: STANdard)

Example: See Example "Finding out the used VISA ressource strings" on page 1116.

:SYSTem:COMMunicate:GPIB:RESource?

Queries the visa resource string for remote control via the GPIB interface.

To change the GPIB address, use the command `:SYSTem:COMMunicate:GPIB[:SELF]:ADDRess`.

Return values:

<Resource> string

Example: See Example "Finding out the used VISA ressource strings" on page 1116.

Usage: Query only

:SYSTem:COMMunicate:GPIB[:SELF]:ADDRess <Address>

Sets the GPIB address.

Parameters:

<Address> integer

Range: 0 to 30
*RST: 28

Example: See Example "Finding out the used VISA ressource strings" on page 1116.

:SYSTem:COMMunicate:HISLip:RESource?

Queries the VISA resource string . This string is used for remote control of the instrument with HiSLIP protocol.

Return values:

<Resource> string

Example: See Example "Finding out the used VISA ressource strings" on page 1116.

Usage: Query only

:SYSTem:COMMunicate:NETWork:IPADdress <IpAddress>

Sets the IP address.

Parameters:

<IpAddress> string

Range: 0.0.0.0. to ff.ff.ff.ff

Example: See Example "Retrieving information on network-related settings" on page 1115.

Manual operation: See "IP Address" on page 716

:SYSTem:COMMunicate:NETWork:IPADdress:MODE <Mode>

Selects manual or automatic setting of the IP address.

Parameters:

<Mode> AUTO | STATic

*RST: n.a. (factory preset: AUTO)

Example: See Example "Retrieving information on network-related settings" on page 1115.

Example:

```
SYSTem:COMMunicate:NETWork:IPADdress:MODE STATic
SYSTem:COMMunicate:NETWork:IPADdress "10.113.0.105"
```

Manual operation: See "Address Mode" on page 716

:SYSTem:COMMunicate:NETWork:MACaddress <MacAddress>

Queries the MAC address of the network adapter.

This is a password-protected function. Unlock the protection level 1 to access it, see `:SYSTem:PROTect<ch>[:STATe]`.

Parameters:

<MacAddress> string

Example: See Example "Retrieving information on network-related settings" on page 1115.

Manual operation: See "MAC Address" on page 716

:SYSTem:COMMunicate:NETWork:RESource?

Queries the visa resource string for Ethernet instruments.

Return values:

<Resource> string

Example: See Example "Finding out the used VISA ressource strings" on page 1116.

Usage: Query only

:SYSTem:COMMunicate:NETWork:RESTart

Restarts the network.

Example:
```
SYSTem:COMMunicate:NETWork:RESTart
// Terminates the network connection and sets it up again
```

Usage: Event

Manual operation: See "Restart Network" on page 718

:SYSTem:COMMunicate:NETWork:STATus?

Queries the network configuration state.

Return values:

<State> 0 | 1 | OFF | ON

Example: See Example "Retrieving information on network-related settings" on page 1115.

Usage: Query only

Manual operation: See "Network Status" on page 715

:SYSTem:COMMunicate:NETWork[:COMMon]:DOMain <Domain>

Determines the primary suffix of the network domain.

Parameters:

<Domain> string

Example: See Example "Retrieving information on network-related settings" on page 1115.

Manual operation: See "DNS Suffix" on page 717

:SYSTem:COMMunicate:NETWork[:COMMon]:HOSTname <Hostname>

Sets an individual hostname for the R&S SMW.

Note: We recommend that you do not change the hostname to avoid problems with the network connection. If you change the hostname, be sure to use a unique name.

This is a password-protected function. Unlock the protection level 1 to access it, see `:SYSTem:PROTect<ch>[:STATe]`.

Parameters:

<Hostname>	string
Example:	See Example "Retrieving information on network-related settings" on page 1115.
Manual operation:	See "Hostname" on page 715

:SYSTem:COMMunicate:NETWork[:COMMon]:WORKgroup <Workgroup>

Sets an individual workgroup name for the instrument.

Parameters:

<Workgroup>	string
Example:	See Example "Retrieving information on network-related settings" on page 1115.
Manual operation:	See "Workgroup" on page 716

:SYSTem:COMMunicate:NETWork[:IPADdress]:DNS <DNS>

Determines or queries the network DNS server to resolve the name.

Parameters:

<DNS>	string
Example:	See Example "Retrieving information on network-related settings" on page 1115.
Manual operation:	See "DNS Server" on page 717

:SYSTem:COMMunicate:NETWork[:IPADdress]:GATeway <Gateway>

Sets the IP address of the default gateway.

Parameters:

<Gateway>	string Range: 0.0.0.0 to ff.ff.ff.ff
Example:	See Example "Retrieving information on network-related settings" on page 1115.
Manual operation:	See "Default Gateway" on page 717

:SYSTem:COMMunicate:NETWork[:IPADdress]:SUBNet:MASK <Mask>

Sets the subnet mask.

Parameters:

<Mask>	string
Example:	See Example "Retrieving information on network-related settings" on page 1115.

Manual operation: See "Subnet Mask" on page 717

:SYSTem:COMMunicate:SERial:BAUD <Baud>

Defines the baudrate for the serial remote control interface.

Parameters:

<Baud> 2400 | 4800 | 9600 | 19200 | 38400 | 57600 | 115200

*RST: n.a. (factory preset: 115200)

Example: See Example "Finding out the used VISA ressource strings" on page 1116.

Manual operation: See "Baud Rate" on page 720

:SYSTem:COMMunicate:SERial:PARity <Parity>

Enters the parity for the serial remote control interface.

Parameters:

<Parity> NONE | ODD | EVEN

*RST: n.a. (factory preset: NONE)

Example: See Example "Finding out the used VISA ressource strings" on page 1116.

Manual operation: See "Parity" on page 720

:SYSTem:COMMunicate:SERial:RESource?

Queries the visa resource string for the serial remote control interface. This string is used for remote control of the instrument.

Return values:

<Resource> string

Example: See Example "Finding out the used VISA ressource strings" on page 1116.

Usage: Query only

:SYSTem:COMMunicate:SERial:SBITs <SBits>

Defines the number of stop bits for the serial remote control interface.

Parameters:

<SBits> 1 | 2

*RST: n.a. (factory preset: 1)

Example: See Example "Finding out the used VISA ressource strings" on page 1116.

Manual operation: See "Stop Bits" on page 720

:SYSTem:COMMunicate:SOCKet:RESource?

Queries the visa resource string for remote control via LAN interface, using TCP/IP socket protocol.

Return values:

<Resource>	string
Example:	See Example "Finding out the used VISA ressource strings" on page 1116.
Usage:	Query only

:SYSTem:COMMunicate:USB:RESource?

Queries the visa resource string for remote control via the USB interface.

Return values:

<Resource>	string
Example:	See Example "Finding out the used VISA ressource strings" on page 1116.
Usage:	Query only

:SYSTem:HELP:EXPort

Saves the online help as zip archive in the user directory.

Example:

```
:SYSTem:HELP:EXPort
MMEM:CDIR?
// "/var/user"
MMEM:CAT?
// ..,"Log,DIR,4096","help.tgz,BIN,69836600"
// confirms that help zip archive is saved.
```

Usage: Event

Manual operation:: "Setup > Help > Export Help to User Path"

:SYSTem:IDENtification <Identification>

Selects the mode to determine the "IDN String" and the "OPT String" for the instrument, selected with command `:SYSTem:LANGuage`.

Note: While working in a emulation mode, the R&S SMW specific command set is disabled, that is, the SCPI command `SYST:IDEN` will be discarded.

Parameters:

<Identification> AUTO | USER

AUTO
Automatically determines the "IDN String" and the "OPT String".

USER
Enables the selection of user definable "IDN String" and "OPT String".

*RST: n.a. (factory preset: AUTO)

Example: `SYST:IDEN AUTO`
automatically asigns the OPT and IDN strings accoording to the selected instrument language.

Manual operation: See "Mode" on page 721

:SYSTem:IDENtification:PRESet

Sets the `*IDN` and `*OPT` strings in user defined mode to default values.

Example:
```
SYST:IDEN USER
SYST:IDEN:PRES
```

Usage: Event

Manual operation: See "Set to Default" on page 721

:SYSTem:LANGuage <Language>

Sets the remote control command set.

Parameters:

<Language> string

Example:
```
SYSTem:LANGuage "SCPI"
// selects SCPI command set
```

Manual operation: See "Language" on page 721

:SYSTem:INFormation:SCPI <InfoString>

Inserts system information in recorded SCPI command lists, for example information on a missing command.

Parameters:

<InfoString> string

Example: `SYST:INF:SCPI "missing command"`
enters the information into a recorded SCPI command list.

:SYSTem:SPECification? <Id>

Retrieves data sheet information for a specific parameter.

Setting parameters:

<Id> string

Identifies the name of the entry in the data sheet, as quieried with the command `:SYSTem:SPECification:IDENtification:CATalog?` on page 1132

Return values:

<ValList> float

Comma-separated list with the specified and, if available, the typical value of the parameter, as specified in the data sheet.

See also "Data Sheet" on page 730.

Example: Retrieving instruments specification

Note: The following values are merely an example.

```
// query the data sheet versions stored in the instrument
:SYSTem:SPECification:VERSion:CATalog?
// "04.03,04.02,04.01,04.00,03.04,03.03,03.02,03.01,03.00,
// 02.96,02.95,02.94,02.02,02.01,02.00,01.03,01.02,01.01,01.00"

// query the data sheet version with that the instrument was delivered
:SYSTem:SPECification:VERSion:FACTory?
// "04.00"

// select a data sheet version
:SYSTem:SPECification:VERSion?
// 04.00
:SYSTem:SPECification:VERSion "04.01"
// selects one particular data sheet version
// queries regarding data sheet parameters (IDs) and their values
// refer to this particular data sheet

// query the IDs of all parameters
// listed in the selected data sheet version
:SYSTem:SPECification:IDENtification:CATalog?
// "ID_RF_FREQ_SETTING_TIME_ALC_ON_MS,ID_RF_FREQ_SETTING_TIME_MS,..."

// query the data sheet information on a specific parameter,
// defined by its ID
:SYSTem:SPECification? "ID_RF_FREQ_SETTING_TIME_ALC_ON_MS"
// returned is the specified and, if available,
// the typical value of the parameter
```

Usage: Query only

:SYSTem:SPECification:VERSion <Version>

Selects a data sheet version from the data sheets saved on the instrument.

Further queries regarding the data sheet parameters (`<Id>`) and their values refer to the selected data sheet.

To query the list of data sheet versions, use the command :SYSTem:SPECification:VERSion:CATalog? on page 1132.

Parameters:

<Version> string

Example: See :SYSTem:SPECification? on page 1130.

:SYSTem:SPECification:IDENtification:CATalog?

Queries the parameter identifiers (<Id>) available in the data sheet.

Return values:

<IdList> string

Comma-separated string of the parameter identifiers (<Id>)

Example: See :SYSTem:SPECification? on page 1130.

Usage: Query only

:SYSTem:SPECification:PARameter? <Id>[, <Parameter>]

Retrieves data sheet information for a specific parameter.

Setting parameters:

<Id> string

Identifies the name of the entry in the data sheet.
Query the data sheet parameters with the command :SYSTem:SPECification:IDENtification:CATalog?.

<Parameter> float

An additional value the result (ValList) depends on.

Return values:

<ValList> float

Comma-separated list with the specified and, if available, the typical value of the parameter, as specified in the data sheet.

Example: **Note:** The following values are merely an example. Your instrument may not support the same parameters.

```
SYST:SPEC:PAR? "ID_RF_FREQ_SETTING_TIME_MS",0.1
SYST:SPEC:PAR? "ID_RF_LEVEL_MAX_GENERAL_DBM",
0.1
```

Usage: Query only

:SYSTem:SPECification:VERSion:CATalog?

Queries all data sheet versions stored in the instrument.

Return values:

<VersCatalog> string

Example: See `:SYSTem:SPECification?` on page 1130.

Usage: Query only

:SYSTem:SPECification:VERSion:FACTory?

Queries the data sheet version of the factory setting.

Return values:

<Version> string

Example: See `:SYSTem:SPECification?` on page 1130.

Usage: Query only

Manual operation: See "Versions" on page 1171

:SYSTem:SRData?

Queris the SCPI recording data from the internal file.

This feature enables you to transfer an instrument configuration to other test environments, as e.g. laboratory virtual instruments.

Return values:

<FileData> block data

Example:

```
SYSTem:SRData?
// #3118:SOURce1:ROSCillator:SOURce EXT
:SOURce1:FREQuency:CW 4000000000
:SOURce1:FREQuency:OFFSet 1000000
:SOURce1:AM1:STATe 1
:OUTPut1:STATe 1
```

Usage: Query only

:SYSTem:STARtup:COMPlete?

Queries if the startup of the instrument is completed.

Return values:

<Complete> 0 | 1 | OFF | ON

*RST: 0

Example: `SYST:STAR:COMP?`

Response: `1`

the startup of the instrument is completed.

Usage: Query only

:SYSTem:DATE <Year>, <Month>, <Day>

Queries or sets the date for the instrument-internal calendar.

This is a password-protected function. Unlock the protection level 1 to access it, see :SYSTem:PROTect<ch>[:STATe].

Parameters:

<Year>	integer	
<Month>	integer	
	Range:	1 to 12
<Day>	integer	
	Range:	1 to 31

Example:
```
:SYSTem:DATE?
// 2016,05,01
```

Manual operation: See "Date" on page 1154

:SYSTem:NTP:HOSTname <NTPName>

Sets the address of the NTP server. You can enter the IP address, or the hostname.

Parameters:

<NTPName> string

Example:
```
SYSTem:NTP:HOSTname "pool.ntp.org"
```

Manual operation: See "NTP Address" on page 1155

:SYSTem:NTP:STATe <UseNtpState>

Activates clock synchronization via NTP.

Parameters:

<UseNtpState>	0 | 1 | OFF | ON	
	*RST:	0

Example:
```
SYSTem:NTP:STATe 1
```

Manual operation: See "Use Time from NTP Server" on page 1155

:SYSTem:TIME <Hour>, <Minute>, <Second>

Queries or sets the time for the instrument-internal clock.

This is a password-protected function. Unlock the protection level 1 to access it, see :SYSTem:PROTect<ch>[:STATe].

Parameters:

<Hour>	integer	
	Range:	0 to 23
<Minute>	integer	
	Range:	0 to 59

<Second> integer
Range: 0 to 59

Example: `SYSTem:TIME?`
`// 10,27,14`

Manual operation: See "Time" on page 1155

:SYSTem:TIME:ZONE <TimeZone>

Sets the timezone. You can query the list of the available timezones with `:SYSTem:TIME:ZONE:CATalog?`.

Parameters:
<TimeZone> string

Manual operation: See "Timezone" on page 1155

:SYSTem:TIME:ZONE:CATalog?

Querys the list of available timezones.

Return values:
<Catalog>

Usage: Query only

Manual operation: See "Timezone" on page 1155

:SYSTem:UPTime?

Queries the up time of the operating system.

Return values:
<UpTime> "<ddd.hh:mm:ss>"

Example: `SYSTem:UPTime?`
Response: "`0.08:11:00`"

Usage: Query only

:SYSTem:BIOS:VERSion?

Queries the BIOS version of the instrument.

Return values:
<Version> string

Example: `SYST:BIOS:VERS?`
queries the BIOS version.
`Response: 123456`

Usage: Query only

:SYSTem:VERSion?

Queries the SCPI version the instrument's command set complies with.

Return values:

<Version> string

Example: `SYST:VERS`

queries the SCPI version.

Response: "`1996`"

The instrument complies with the SCPI version from 1996.

Usage: Query only

:SYSTem:OSYStem?

Queries the operating system of the instrument.

Return values:

<OperSystem> string

Example: `SYSTem:OSYStem?`

Response: "`Linux`"

Usage: Query only

:SYSTem:MMEMory:PATH:USER?

Queries the user directory, that means the directory the R&S SMW stores user files on.

Return values:

<PathUser> string

Example: `SYSTem:MMEMory:PATH:USER?`

Response: "'/var/user/"

Usage: Query only

:SYSTem:DFPR?

Queries the device footprint of the instrument. The retrieved information is in machine-readable form suitable for automatic further processing.

If you are obtaining technical support as described in Chapter 15.6, "Collecting Information for Technical Support", on page 1177, this information is automatically retrieved and is part of the created `*.tar.gz` support file.

Return values:

<DeviceFootprint> string

Information on the instrument type and details on the installed FW version, hardware and software options.

Example: `:SYSTem:DFPR?`

Usage: Query only

:SYSTem:REBoot

Reboots the instrument including the operating system.

Usage: Event

:SYSTem:RESTart

Restarts the instrument without restarting the operating system.

Usage: Event

:SYSTem:SHUTdown

Shuts down the instrument.

Usage: Event

:SYSTem:WAIT <TimeMs>

Delays the execution of the subsequent remote command by the specified time.

This function is useful, for example to execute an SCPI sequence automatically but with a defined time delay between some commands.

See Chapter 11.3, "Assigning Actions to the User Key", on page 678.

Setting parameters:

<TimeMs> integer

Wait time in ms

Range: 0 to 10000
*RST: 0

Example:

```
:SYSTem:WAIT 10000
// waits 10s before resetting the instrument
*RST
```

Usage: Setting only

Manual operation: See "Wizard" on page 680

13.20 STATus Subsystem

This system contains the commands for the status reporting system. See also Chapter B.1.5, "Status Reporting System", on page 1203 for detailed information.

`*RST` on page 772 has no effect on the status registers.

Value ranges

- Queries return the current value of the respective register, which permits a check of the device status.
  Return values: A decimal value in the range 0 to 32767 (=2^{15}-1)
- The configuration commands set the respective register thus determining which status changes of the R&S SMW cause the status registers to be changed.
  Setting values: A decimal value in the range 0 to 32767 (=2^{15}-1)

:STATus:OPERation:CONDition <Condition>

Sets the content of the CONDition part of the STATus:OPERation register. This part contains information on the action currently being performed in the instrument. The content is not deleted after being read out because it indicates the current hardware status.

Parameters:

<Condition> string

Example: `:STATus:OPERation:CONDition?`
queries the Status:Operation:Condition register.

:STATus:OPERation:ENABle <Enable>

Sets the bits of the ENABle part of the STATus:OPERation register. This setting determines which events of the Status-Event part are forwarded to the sum bit in the status byte. These events can be used for a service request.

Parameters:

<Enable> string

Example: `:STAT:OPER:ENAB 32767`
all events are forwarded to the sum bit of the status byte.

:STATus:OPERation[:EVENt] <Event>

Queries the content of the EVENt part of the STATus:OPERation register. This part contains information on the actions performed in the instrument since the last readout. The content of the EVENt part is deleted after being read out.

Parameters:

<Event> string

Example: `:STAT:OPER:EVEN?`
queries the STATus:OPERation:EVENt register.

:STATus:OPERation:NTRansition <Ntransition>

Sets the bits of the NTRansition part of the STATus:OPERation register. If a bit is set, a transition from 1 to 0 in the condition part causes an entry to be made in the EVENt part of the register. The disappearance of an event in the hardware is thus registered, for example the end of an adjustment.

Parameters:

<Ntransition> string

Example: `:STAT:OPER:NTR 0`
a transition from 1 to 0 in the condition part of the Status:Operation register does not cause an entry to be made in the EVENt part.

:STATus:OPERation:PTRansition <Ptransition>

Sets the bits of the PTRansition part of the STATus:OPERation register. If a bit is set, a transition from 0 to 1 in the condition part causes an entry to be made in the EVENt part of the register. A new event in the hardware is thus registered, for example the start of an adjustment.

Parameters:

<Ptransition> string

Example: `:STAT:OPER:PTR 32767`
all transitions from 0 to 1 in the condition part of the Status:Operation register cause an entry to be made in the EVENt part.

:STATus:PRESet <Preset>

Resets the status registers. All PTRansition parts are set to FFFFh (32767), 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 parts of STATus:OPERation and STATus:QUEStionable are set to 0, i.e. all events in these registers are not passed on.

Parameters:

<Preset> string

Example: `STAT:PRES`
resets the status registers.

:STATus:QUEStionable:CONDition <Condition>

Queries the content of the CONDition part of the STATus:QUEStionable register. This part contains information on the action currently being performed in the instrument. The content is not deleted after being read out since it indicates the current hardware status.

Parameters:

<Condition> string

Example: `:STATus:QUEStionable:CONDition?`
queries the Status:Questionable:Condition register.

:STATus:QUEStionable:ENABle <Enable>

Sets the bits of the ENABle part of the STATus:QUEStionable register. The enable part determines which events of the STATus:EVENt part are enabled for the summary bit in the status byte. These events can be used for a service request.

If a bit in the ENABle part is 1, and the correesponding EVENt bit is true, a positive transition occurs in the summary bit. This transition is reportet to the next higher level.

Parameters:

<Enable> string

Example: `STAT:QUES:ENAB 1`
Problems when performing an adjustment cause an entry to be made in the sum bit.

:STATus:QUEStionable[:EVENt] <Event>

Queries the content of the EVENt part of the `STATus:QUEStionable` register. This part contains information on the actions performed in the instrument since the last readout. The content of the EVENt part is deleted after being read out.

Parameters:

<Event> string

Example: `STAT:QUES:EVEN?`
queries the Status:Questionable:Event register.

:STATus:QUEStionable:NTRansition <Ntransition>

Sets the bits of the NTRansition part of the STATus:QUEStionable register. If a bit is set, a transition from 1 to 0 in the condition part causes an entry to be made in the EVENt part of the register.

Parameters:

<Ntransition> string

Example: `STAT:QUES:NTR 0`
a transition from 1 to 0 in the condition part of the STATus:QUEStionable register does not cause an entry to be made in the EVENt part

---

:STATus:QUEStionable:PTRansition <PTransition>

Sets the bits of the NTRansition part of the STATus:QUEStionable register. If a bit is set, a transition from 1 to 0 in the condition part causes an entry to be made in the EVENt part of the register.

Parameters:
<PTransition> string

Example: `STAT:QUES:PTR 32767`
all transitions from 0 to 1 in the condition part of the STATus:QUEStionable register cause an entry to be made in the EVENt part

---

:STATus:QUEue[:NEXT]?

Queries the oldest entry in the error queue and then deletes it. Positive error numbers denote device-specific errors, and negative error numbers denote error messages defined by SCPI. If the error queue is empty, 0 ("No error") is returned.

The command is identical to `:SYSTem:ERRor[:NEXT]?` on page 1120.

Return values:
<Next> string

Example: `:STATus:QUEue?`
queries the oldest entry in the error queue.
`Response: 0, 'no error'`
no errors have occurred since the error queue was last read out

Usage: Query only

13.21 TEST Subsystem

The `TEST` subsystem contains the commands for performing test routines directly at the hardware assemblies.

The selftetst responses with a `0` if the test is performed successfully, otherwise a value other than `0` is returned. None of the commands of this system has a `*RST` value.

:TEST:BASeband?

Queries the result of the baseband selftest.

Return values:

<TestBbError> 0 | 1 | RUNning | STOPped

Example:

```
:TEST:BASeband?
// 0
// the test succeeded
```

Usage: Query only

Manual operation: See "Execute Baseband Selftest" on page 1167

:TEST:BB:BNC:CONNection?

Queries the BNC connection test result.

This is a password-protected function. Unlock the protection level 1 to access it, see :SYSTem:PROTect<ch>[:STATe].

Return values:

<TestStatus> 0 | 1 | RUNning | STOPped

Example:

```
SYSTem:PROTect1 0,123456
TEST:BB:BNC:CONNection?
// RUN
// test is running
```

Usage: Query only

Manual operation: See "Execute BNC Connection Test" on page 1164

:TEST:BB:BNC:DESTination <BncDestination>

Selects the BNC connection test destination.

This is a password-protected function. Unlock the protection level 1 to access it, see :SYSTem:PROTect<ch>[:STATe].

Parameters:

<BncDestination> AUTO | USER1 | USER2 | USER3 | USER4 | USER5 | USER6 | TRGA | TRGB | C1TMC1 | C1TM2 | C1TM3 | C2TMC4 | C2TM5 | C2TM6 | F1TMC1 | F1TM2 | F1TM3 | F2TMC4 | F2TM5 | F2TM6 | F3TMC1 | F3TM2 | F3TM3 | F4TMC4 | F4TM5 | F4TM6

*RST: USER2

Example:

```
SYSTem:PROTect1 0,123456
TEST:BB:BNC:DESTination TRGA
// tests the instrument trigger connector of path A
```

Manual operation: See "Destination" on page 1164

:TEST:BB:BNC:LOG?

Queries the log message reported during the BNC connector test.

This is a password-protected function. Unlock the protection level 1 to access it, see :SYSTem:PROTect<ch>[:STATe].

Return values:

<Log> string

Example:

```
SYSTem:PROTect1 0,123456
TEST:BB:BNC:LOG?
// queries the reported message
```

Usage: Query only

Manual operation: See "Log Message" on page 1164

:TEST:BB:BNC:SOURce <BncSource>

Selects the BNC connection test source.

This is a password-protected function. Unlock the protection level 1 to access it, see :SYSTem:PROTect<ch>[:STATe].

Parameters:

<BncSource> AUTO | USER1 | USER2 | USER3 | USER4 | USER5 | USER6 | TRGA | TRGB | C1TMC1 | C1TM2 | C1TM3 | C2TMC4 | C2TM5 | C2TM6 | F1TMC1 | F1TM2 | F1TM3 | F2TMC4 | F2TM5 | F2TM6 | F3TMC1 | F3TM2 | F3TM3 | F4TMC4 | F4TM5 | F4TM6

*RST: USER1

Example:

```
SYSTem:PROTect1 0,123456
TEST:BB:BNC:SOURce AUTO
// detects the connectors that have assigned a signal and performs the co
```

Manual operation: See "Source" on page 1164

:TEST:BB:GENerator:ARBitrary <Filename>

Selects the ARB waveform to be tested.

Parameters:

<Filename> string

Example:
```
TEST:BB:GENerator:ARBitrary "3gpp.wv"
// Tests the loaded waveform, generating a 3GPP signal
```

Manual operation: See "Select ARB File" on page 1166

:TEST:BB:GENerator:FREQuency <Frequency>

Sets the frequency of the test sine or constant I/Q test signal.

Parameters:

<Frequency> float

Range: -80E6 to 80E6
Increment: 1E-3
*RST: 1E6

Example:
```
TEST:BB:GENerator:FREQuency 50 MHz
```

Manual operation: See "Frequency" on page 1166

:TEST:BB:GENerator:GAIN <Gain>

Sets the gain for a sine or constant I/Q test signal.

Parameters:

<Gain> float

Range: -1 to 1
Increment: 1E-6
*RST: 1

Example:
```
TEST:BB:GENerator:GAIN 0.5
// amplifies the test signal by a factor of 0.5
```

Manual operation: See "Gain" on page 1165

:TEST:BB:GENerator:SOURce <IqSource>

Selects the test signal source.

Parameters:

<IqSource> SINE | CONStant | ARB

SINE
Generates a sine waveform. To determine the frequency, use command :TEST:BB:GENerator:FREQuency on page 1144.

CONStant
Uses a constant I/Q test signal.

ARB
Selects a signal generated generated by the ARB.

*RST: SINE

Example: `:TEST:BB:GENerator:SOURce SINE`

Manual operation: See "IQ Source" on page 1165

:TEST:BB:GENerator:STATe <State>

Starts the test generator.

Parameters:

<State> 0 | 1 | OFF | ON

*RST: 0

Example:
```
TEST:BB:GENerator:STATe ON
// activates the test generator
```

Manual operation: See "State" on page 1165

:TEST<hw>:DIRect:BLOCk:READ? <Board>, <SubAddr>, <Addr>[, <Len>]

Direct mode for querying the blocks of the hardware assemblies directly.

The respective hardware assembly responds directly to the command; any safety mechanisms are bypassed. This function is only available via remote control.

Query parameters:

<Board> string

<SubAddr> integer

<Addr> integer

<Len> integer

Return values:

<DataBlock> string

Example: `TEST:DIR:BLOC:READ? 'TEST_SCPI',0,0,8`

Usage: Query only

:TEST<hw>:DIRect:BLOCk:WRITe <BoardName>, <SubAddress>, <StartAddress>, <DataBlock>

Direct mode for programming the blocks of the hardware assemblies directly.

The respective hardware assembly responds directly to the command; any safety mechanisms are bypassed. This function is only available via remote control.

Setting parameters:

<BoardName> string
<Board>, <SubAddr>, <Addr>, <Data>

<SubAddress> integer

<StartAddress> integer

<DataBlock> string

Example: `TEST:DIR:BLOC:WRIT 'TEST_SCPI',0,0,#1812345678`

Usage: Setting only

:TEST<hw>:ALL:STARt

Usage: Event

Starts the selftest. Use the command `:TEST<hw>:ALL:RESult?` to query the result.

:TEST<hw>:ALL:RESult?

Queries the result of the performed selftest. Start the selftest with `:TEST<hw>:ALL:STARt`.

Return values:

<Result> 0 | 1 | RUNning | STOPped
*RST: STOPped

Usage: Query only

13.22 TRIGger Subsystem

The `TRIGger` system contains the commands for selecting the trigger source for the RF and LF sweep.

You can work with an internal or with an externally applied trigger signal. In this case, use the commands in the `SOURce:INPut` subsystem to configure the signal.

The trigger system of the R&S SMW is a simplified implementation of the SCPI trigger system. The `TRIGger` system differs from the SCPI system as follows:

- No `INITiate` command; the instrument behaves as if `INITiate:CONTinuous ON` were set.

- Under TRIGger several sweep subsystems exist.
- The trigger source names correspond directly to the various settings of manual control. SCPI uses different names which are also accepted by the instrument. The SCPI names should be used if compatibility is an important consideration.

In addition to these commands, see more trigger-related commands in the modulation and RF signal subsystems.

Table 13-12: Cross-reference between the manual and remote control

R&S proprietary value name	SCPI conform value name	Parameter in manual control
AUTO	IMMediate	"Auto" mode
SINGle	BUS	"Single" mode.
EXTernal	EXTernal	"Ext Single" and "Ext Step" mode. Use command LFO:SWEep:MODE to select between the two sweep modes.
EAUTo	-	"Ext Start/Stop" mode.

:TRIGger<hw>:FSWeep:SOURce <Source>
:TRIGger<hw>:LFFSweep:SOURce <Source>
:TRIGger<hw>:PSWeep:SOURce <Source>
:TRIGger<hw>[:SWEep]:SOURce <Source>

Selects the trigger source for the corresponding sweeps:

- FSWeep - RF frequency
- LFFSweep - LF frequency
- PSWeep - RF level
- SWEep - all sweeps

The source names of the parameters correspond to the values provided in manual control of the instrument. They differ from the SCPI-compliant names, but the instrument accepts both variants.

Use the SCPI name, if compatibility is an important issue. Find the corresponding SCPI-compliant commands in Cross-reference between the manual and remote control .

Setting parameters:

<Source> AUTO | IMMediate | SINGle | BUS | EXTernal | EAUTo

AUTO [IMMediate]
Executes a sweep automatically.
In this free-running mode, the trigger condition is met continuously. I.e. when a sweep is completed, the next one starts immediately.

SINGle [BUS]
Executes one complete sweep cycle.
The following commands initiate a trigger event:
`*TRG` on page 773
`[:SOURce<hw>]:SWEep:POWer:EXECute`
`[:SOURce<hw>]:SWEep[:FREQuency]:EXECute`
`:TRIGger<hw>[:SWEep][:IMMediate]`, `:TRIGger<hw>:PSWeep[:IMMediate]` and `:TRIGger<hw>:FSWeep[:IMMediate]`.
Set the sweep mode with the commands:
`[:SOURce<hw>]:SWEep:POWer:MODE AUTO|STEP`
`[:SOURce<hw>]:SWEep[:FREQuency]:MODE AUTO|STEP`
`[:SOURce<hw>]:LFOutput:SWEep[:FREQuency]:MODE AUTO|STEP`
In step mode (`STEP`), the instrument executes only one step.

EXTernal
An external signal triggers the sweep.

EAUTo
An external signal triggers the sweep. When one sweep is finished, the next sweep starts.
A second trigger event stops the sweep at the current frequency, a third trigger event starts the trigger at the start frequency, and so on.

*RST: AUTO

Example: See Example "Setup an LF sweep" on page 1061

Usage: Setting only

Manual operation: See "Mode" on page 476

:TRIGger<hw>:FSWeep[:IMMediate]
:TRIGger<hw>:PSWeep[:IMMediate]
:TRIGger<hw>:LFFSweep:IMMediate
:TRIGger<hw>[:SWEep][:IMMediate]

Performs a single sweep and immediately starts the activated, corresponding sweep:

- `FSWeep` - RF frequency
- `PSWeep` - RF level
- `LFFSweep` - LF frequency
- `SWEep` - all sweeps

Effective in the following configuration:

- `TRIG:FSW|LFFS|PSW|[:SWE]:SOUR` **SING**
- `SOUR:SWE:FREQ|POW:MODE` **AUTO** or `SOUR:LFO:SWE:[FREQ:]MODE` **AUTO**

Alternativelly, you can use the `IMMediate` command instead of the respective `SWEep:[FREQ:]|POW:EXECute` command.

Example: `TRIG`
Starts all active sweeps.

Usage: Event

Manual operation: See " Execute Single Sweep " on page 477

:TRIGger<hw>:LFFSweep

Executes an LF frequency sweep in the following configuration:

- `TRIG:LFFS:SOUR` **SING**
- `LFO:SWE:MODE` **AUTO**

Example:
```
LFO:SWE:MODE AUTO
TRIG:LFFS:SOUR SING
TRIG:LFFSweep
```

Usage: Event

13.23 UNIT Subsystem

The `UNIT` subsystem is used to set default units for parameters if no unit is indicated in a command. These settings are valid for the entire instrument.

Example: Setting default units for remote control

```
UNIT:POW V
UNIT:ANGL DEG

UNIT:VEL KMH
```

Sets V (volts) as unit of all power parameters, DEG (degrees) for the phase modulation angle and KMH for the speed.

:UNIT:ANGLe <Angle>

Sets the default unit for phase modulation angle. The command affects no other parameters, such as RF phase, or the manual control or display.

Parameters:

<Angle>	DEGree | DEGRee | RADian	
	*RST:	RADian

:UNIT:POWer <Power>

Sets the default unit for all power parameters. This setting affects the GUI, as well as all remote control commands that determine power values.

Parameters:

<Power>	V | DBUV | DBM	
	*RST:	DBM

:UNIT:VELocity <Velocity>

Sets the default unit for the velocity of the wave.

Parameters:

<Velocity>	MPS | KMH | MPH | NMPH	
	*RST:	MPS

14 Maintenance

The instrument does not need periodic maintenance. Only the cleaning of the instrument is essential.

Follow the instructions in the service manual and the safety instructions when exchanging modules or ordering spares. The order no. for spare parts is included in the service manual. The service manual includes further information particularly on troubleshooting, repair, exchange of modules and alignment.

The address of our support center and a list of all Rohde & Schwarz service centers can be found at the beginning of this manual.

NOTICE

Risk of damage during shipment

Insufficient protection against mechanical and electrostatic effects during shipment can damage the instrument.

- When shipping an instrument, use the original packaging. If you do not have the original packaging, use sufficient padding to prevent the instrument from moving around inside the box.
- Pack the instrument in antistatic wrap to protect it from electrostatic charging.
- Secure the instrument to prevent any movement and other mechanical effects during transportation.

14.1 Cleaning

⚠ WARNING

Risk of electric shock

If moisture enters the casing, for example if you clean the instrument using a moist cloth, contact with the instrument can lead to electric shock. Before cleaning the instrument other than with a dry cloth, make sure that the instrument is switched off and disconnected from all power supplies.