



VNA Control 5

Programmer's Guide

SCPI Edition

LA Techniques Ltd

Chancerygate Business Centre, Unit 5
Red Lion Road, Surrey KT6 7RA, UK
VAT no GB 689 4720 79
Registered in England No 3356289 Registered Office as above

Tel: 0208 3973150
Fax: 0208 3975372
E-mail: info@latechniques.com
Web site: www.latechniques.com

Contents

1. Interface	5
2. SCPI control in headless mode	5
3. Example programs for SCPI control	6
3.1. Linux: telnet	6
3.2. Python	7
3.3. MATLAB	7
4. Overview of the interface	7
4.1. Command syntax	7
4.2. Parameters	8
4.3. Return values	9
4.4. Data representation	10
4.5. Specifying channels	10
4.6. Commands that use the filesystem	10
4.7. Compatibility mode	11
4.8. Differences from other SCPI implementations	11
5. Overview of key SCPI commands	12
5.1. Getting help	12
5.2. Setting up measurements	12
5.3. Starting a measurement sweep	12
5.4. Retrieving data	12
5.5. Exporting data to a Touchstone or CSV file	13
5.6. Using markers	13
6. SCPI Command Reference: top-level commands	14
6.0.1. HELP	14
6.0.2. *IDN?	15
6.0.3. *OPC?	15
6.0.4. *OPT?	15
6.0.5. *RST	15
6.0.6. *TRG	16
6.0.7. ABORt	16

7. SCPI Command Reference: CALCulate subsystem	16
7.0.1. CALCulate:AVERaging	16
7.0.2. CALCulate:AVERaging:COUNT	17
7.0.3. CALCulate:AVERaging:COUNT?	17
7.0.4. CALCulate:AVERaging?	17
7.0.5. CALCulate:DATA	17
7.0.6. CALCulate:DATA:LIVE	18
7.0.7. CALCulate:DATA:MEMory	19
7.0.8. CALCulate:DATA:STIMulus?	19
7.0.9. CALCulate:SMOothing	19
7.0.10. CALCulate:SMOothing?	19
7.0.11. CALCulate:SMOothing:FACTOR	20
7.0.12. CALCulate:SMOothing:FACTOR?	20
7.0.13. CALCulate:TIME	20
7.0.14. CALCulate:TIME?	20
7.0.15. CALCulate:TIME:DCterm	20
7.0.16. CALCulate:TIME:DCterm?	20
7.0.17. CALCulate:TIME:EFFD	21
7.0.18. CALCulate:TIME:EFFD?	21
7.0.19. CALCulate:TIME:RESPonse	21
7.0.20. CALC:TIME:RESP?	21
7.0.21. CALCulate:TIME:RTERM	21
8. SCPI Command Reference: CHANnel subsystem	22
8.0.1. CHANnel<Ch>:COPY	22
8.0.2. CHANnel:MEMory:CATalog:COUNT?	22
8.0.3. CHANnel:MEMory:CATalog?:	22
8.0.4. CHANnel:MEMory:DElete	22
8.0.5. CHANnel:MEMory:DElete:ALL:	22
8.0.6. HELP	23
8.0.7. INITiate:	23
8.0.8. INSTrument:PORT:COUNT?	23
8.0.9. MARKer<M>	23
8.0.10. MARKer<M>:DELETE	24
8.0.11. MARKer<M>:Query	24
8.0.12. MARKer<M>:READOUT	25
8.0.13. MARKer<M>:READOUT:AOFF	25

8.0.14. MARKer<M>:TRACKing	25
8.0.15. MARKer<M>:TYPe	27
8.0.16. MARKer<M>:TYPe?	27
8.0.17. MARKer<M>:VISible	27
8.0.18. MARKer<M>:VISible?	27
8.0.19. MARKer<M>:X	27
8.0.20. MARKer<M>:X?	27
8.0.21. MARKERT<M>	28
8.0.22. MARKERT<M>:DELETE	28
8.0.23. MARKERT<M>:Query	28
8.0.24. MARKer<M>:TRACKing	28
8.0.25. MARKERT<M>:TYPe	29
8.0.26. MARKERT<M>:TYPe?	30
8.0.27. MARKERT<M>:VISible	30
8.0.28. MARKERT<M>:VISible?	30
8.0.29. MARKERT<M>:X	30
8.0.30. MARKERT<M>:X?	30
8.0.31. MMEMory:APPLY:CALibration	30
8.0.32. MMEMory:CATalog?	31
8.0.33. MMEMory:CDirectory	31
8.0.34. MMEMory:COPIY	31
8.0.35. MMEMory:CP	31
8.0.36. MMEMory:DELete	31
8.0.37. MMEMory:LOAD:SESSion	32
8.0.38. MMEMory:LOAD:STATe	32
8.0.39. MMEMory:LS	32
8.0.40. MMEMory:MKDIRectory	32
8.0.41. MMEMory:MOVE	32
8.0.42. MMEMory:MV	33
8.0.43. MMEMory:RM	33
8.0.44. MMEMory:RMDIRectory	33
8.0.45. MMEMory:STORE:SESSion	33
8.0.46. MMEMory:STORE:STATe	33
8.0.47. MMEMory:STORE:TRACe	33
8.0.48. MMEMory:STORE:TRACe:OPTion:CSVDATAFORMAT	34
8.0.49. MMEMory:STORE:TRACe:OPTion:NUMPORTS	35

8.0.50. MMEMory:STORe:TRACe:OPTion:ONEPORTPARAMETER	36
8.0.51. MMEMory:STORe:TRACe:OPTion:SEPARATOR	36
8.0.52. MMEMory:STORe:TRACe:OPTion:TABS	36
8.0.53. MMEMory:STORe:TRACe:OPTion:TOUCHSTONEDATAFORMAT	36
8.0.54. SENSE:Bandwidth	37
8.0.55. SENSE:Bandwidth?	37
8.0.56. SENSE:FREQuency:STARt	37
8.0.57. SENSE:FREQuency:STARt?	37
8.0.58. SENSE:FREQuency:STOP	37
8.0.59. SENSE:FREQuency:STOP?	37
8.0.60. SENSE:LEVel	38
8.0.61. SENSE:LEVel?	38
8.0.62. SENSE:SWEEp:POINts	38
8.0.63. SENSE:SWEEp:POINts?	38
8.0.64. SENSE:SWEEp:STEP	38
8.0.65. SENSE:SWEEp:STEP?	39
8.0.66. SENSE:TEMPerature?	39

1 Interface

By default, the VNA Control 5 software listens for SCPI commands on TCP port 5025. The SCPI interface can be disabled, or the TCP port changed, via the User Preferences (Network), shown in Figure 1.

A VNA instrument can be controlled concurrently from both the User Interface and SCPI commands. For example, a measurement could be set up via the User Interface and then initiated via SCPI, or vice-versa.

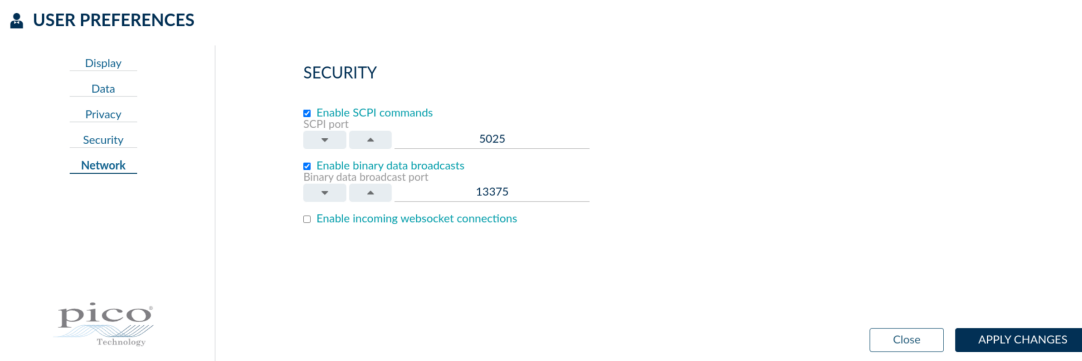


Figure 1: User preferences for configuring SCPI via the User Interface.

2 SCPI control in headless mode

If the User Interface is not required, the VNA Control 5 software may be started in headless mode. All SCPI functionality is available in headless mode. The serial number of the instrument to use must be provided (via command line arguments) when starting the software in headless mode.

Headless mode (Windows)

Open a Command Prompt and change the working directory to where VNA Control 5 is installed (by default C:\Program Files\LA Techniques Ltd\VNA Control 5). Then start the server application using:

```
bin\vnaserver -P <scpi_port> --instrument <instrument_serial_number>
```

This process is shown in Figure 2.

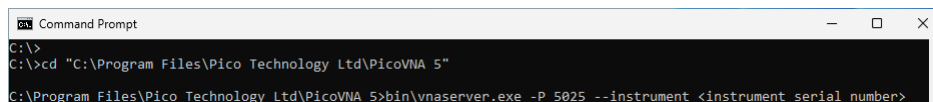


Figure 2: Starting the VNA Control 5 software in headless mode on Windows.

Headless mode (Linux)

Open a terminal and change the working directory to where VNA Control 5 is installed (by default /opt/vnacontrol). Then start the server application using:

```
bin/vnaserver -P <scpi_port> --instrument <instrument_serial_number>
```

This process is shown in Figure 3.

```

~$ cd /opt/picovna
~/opt/picovna$ ./bin/vnserver -P 5025 --instrument <instrument_serial_number>

```

Figure 3: Starting the VNA Control 5 software in headless mode on Linux.

Headless mode (macOS)

Open a terminal and change the working directory to where VNA Control 5 is installed (by default /Applications/vnacontrol5.app/Contents/MacOS). Then start the server application using:

```
bin/vnserver -P <scpi_port> --instrument <instrument_serial_number>
```

This process is shown in Figure 4.



```

MacOS -- zsh -- 108x24
~ % cd /Applications/picovna5.app/Contents/MacOS
MacOS % ./bin/vnserver -P 5025 --instrument <instrument_serial_number>

```

Figure 4: Starting the VNA Control 5 software in headless mode on macOS.

Command line arguments

In headless mode, command line arguments can be specified after the executable name. The following command line arguments are supported:

<i>Argument</i>	<i>Required?</i>	<i>Description</i>
<code>--instrument <serial></code>	Yes	The serial number of the instrument to use
<code>-h</code>	No	Show help
<code>-P <scpi_port></code>	No (Default 5025)	Listen port for SCPI
<code>-B <bb_port></code>	No (Default 13375)	Server port for binary data broadcasts

Controlling multiple connected VNA instruments in headless mode

It is possible to control multiple VNA instruments connected to the same PC. One instance of `vnserver` must be started for each instrument to be controlled. A different SCPI server port must be specified for each instance of `vnserver` that is started.

3 Example programs for SCPI control

A set of short example programs that connect to PicoVNA 5 and initiate a sweep are presented below. A more comprehensive set of example programs are available at https://github.com/LA-Techniques/VNAControl5_SDK_Examples.

3.1 Linux: telnet

SCPI commands can be typed directly into a telnet prompt. For example, to start a single sweep from a terminal prompt running on the same host as the PicoVNA 5 software:

```

telnet localhost 5025
*IDN?
INIT

```

3.2 Python

We recommend using the `pyvisa` package for controlling the instrument via SCPI from Python. Ensure that the `pyvisa`, `pyvisa-py` and `zeroconf` packages are installed.

The following script will store the instrument ID in the variable `id` and then start a single sweep, when running on the same host as the PicoVNA 5 software:

```
import pyvisa

rm = pyvisa.ResourceManager()
vna = rm.open_resource("TCPIP::127.0.0.1::5025::SOCKET")
vna.read_termination = '\n'
vna.write_termination = '\n'

id = vna.query('*IDN?')
vna.query('INIT')
```

3.3 MATLAB

The following script will store the instrument ID in the variable `id` and then start a single sweep from MATLAB running on the same host as the PicoVNA 5 software:

```
vnaInterface = tcpip('localhost', 5025);
fopen(vnaInterface);
id = query(vnaInterface, '*IDN?');
query(vnaInterface, 'INIT')
```

4 Overview of the interface

4.1 Command syntax

SCPI commands can be written in full or abbreviated form. The documentation will specify many commands in mixed lower and upper case. The command can either be issued in full by copying the full form in the documentation, or in abbreviated form by only issuing the capitalised parts of the command. SCPI commands are not case sensitive (except for units of numeric quantities) so can be issued in upper-case, lower-case, or any combination of upper- and lower-case.

Multiple commands can be issued together by concatenating them on a single line, separated by a semi-colon.

Some SCPI commands are used to set or query a parameter on the instrument, for example the number of points in the measurement. In these cases, the command to query the parameter is formed by appending a question-mark (?) to the end of the command used to set the parameter.

Commands with integral suffixes

Some commands require an *integral suffix* (usually an integer), where the suffix is integrated into the command itself. These commands are not valid unless the suffix is included with the command.

An example is the `MARKER<M>:X?` command. `M` denotes the integral suffix. `MARKER1:X?` would be used to query the x -coordinate of Marker 1, but `MARKER:X?` would not be a valid command as it lacks the integral suffix.

Integral suffixes should not be confused with parameters. Commands have at most one integral suffix and no space is left between the integral suffix and the command.

4.2 Parameters

Many SCPI commands require one or more parameters. The list of parameters should be separated from the command with a single space, and then parameters should be passed as a comma-separated list without spaces between parameters. For example, a command (COMMAND) with three parameters (PARAM1, PARAM2 and PARAM3) would be issued as:

```
COMMAND PARAM1 ,PARAM2 ,PARAM3
```

Required and optional parameters

In the documentation:

- `< . . . >` indicates a required parameter.
- `[. . .]` indicates an optional parameter.

The square and angle brackets used for illustration in the documentation should not be issued with the command.

Required parameters should not be confused with integral suffixes, which are indicated in the documentation using the same angle bracket notation but without a space between the integral suffix and the command.

Parameter types

In the documentation, parameters are written as:

X: Y

Where X indicates the name of the parameter and Y indicates its type. Types are notated in this way for the purposes of documentation only, and should not be included when issuing a command.

Parameter types may be:

- **BOOL.** A Boolean value. True can be written as TRUE, 1, ON or YES. False can be written as FALSE, 0, OFF or NO.
- **DOUBLE.** An IEEE-754 double-precision floating point number.
- **INT.** An integer value.
- **STRING.** A string value, used for example to specify a file path.
- **KEYWORD.** Some parameters accept a finite set of values, specified in the documentation. In these cases, the type of the parameter is specified as VAL1 | VAL2 | VAL3, where VAL1, VAL2 and VAL3 indicate the values that can be passed for that parameter.

Regardless of the parameter type, parameters must be specified in ASCII text. There is no binary input mode for parameters.

Units

Where parameters require a unit (for example when specifying a frequency, bandwidth or power), enter it after the value. A space between the value and unit is optional. Units are case-sensitive.

Units that can be used are:

- m – metre
- Hz – Hertz
- s – seconds
- dB – decibels
- W – Watts
- dBm – decibels relative to 1 mW
- v_{rms} – RMS voltage
- v_{pp} – peak-to-peak voltage
- deg – degrees
- rad – radians
- Ω – ohms
- π rad – multiples of π radians
- U – unitless

4.3 Return values

Commands that have data to return (for example, a getter command, or a command retrieving sweep data) will return that data in a single line terminated with a newline character (unless otherwise specified).

Commands that do not produce a specific output (for example, setter commands) will return the string OK (followed by a newline character) if the operation succeeded. If the command failed, it will return a single-line string (terminated with a newline character) that indicates the nature of the error that occurred.

The strings returned by commands that do not produce a specific output include, but are not limited, to:

- OK – the operation succeeded.
- Unknown SCPI command: [command] – if the command issued was not a valid command.
- Too few arguments provided to SCPI command. Need [...] got [...]. – if the number of arguments supplied was not correct.
- No device connected – the operation requires an instrument to be connected, but no instrument is connected.
- This command has no effect – if the command is included for compatibility reasons with other software and instruments, but it has no effect in PicoVNA 5.
- Command disabled by user for security – returned by commands that operate on the filesystem, if filesystem commands have not been explicitly enabled in User Preferences in the PicoVNA 5 software.

4.4 Data representation

Measurement data is retrieved from the instrument using the `CALCulate:DATA` series of commands. Data can be retrieved in binary or text-based (ASCII) format. The default is binary. Use the `FORMat` command to select the desired data output format.

In binary mode, data will be returned as a concatenation of IEEE-754 floating-point values. The bit-length of each number (32 or 64 bits) can be set using the `FORMat` command. The default is 64 bits (double precision). The endianness of the output data can be set using the `FORMat:BOReDer` command. The default is big-endian.

In ASCII mode, data is returned as a comma-separated list of values.

If a data point is not available (for example, if the sweep was aborted early, or group delay was requested at the first measurement point), it will be replaced with a *NaN*. *NaN* is represented by the string `nan` in ASCII mode. In binary mode, a non-signalling NaN is returned. The exact byte representation of this value depends on your hardware/OS, but it will certainly have all bits of the exponent set and at least one bit of the fraction set (that is, it is an IEEE754-compliant NaN, but whether or not it triggers your hardware's floating-point exception mechanism will be platform-dependent).

4.5 Specifying channels

Many commands require the user to specify what channel the command should be applied to, i.e. the live data channel, or a specific memory channel. The live data channel is specified as 0. A memory channel is specified by the index of the memory channel, i.e. 1, 2, 3, ...

4.6 Commands that use the filesystem

Some commands produce their output by writing to files (for example, when exporting data to a Touchstone or CSV file). Other commands require a file as input (for example, when loading a calibration). These commands require a path to be specified as one of their arguments.

Some commands also operate directly on the filesystem, for example to copy, create or remove files or directories.

Paths

Paths can be specified as either an absolute path or a relative path.

PicoVNA 5 maintains a virtual “working directory”, where relative paths are specified from. To change the working directory, use the `MMEMemory:CDIRECTORY` command (which in turn can take an absolute path, or a relative path from the current working directory).

Safety

By default, the following filesystem commands that operate directly on the filesystem are disabled for security purposes:

- `MMEMemory:COpy`
- `MMEMemory:DELeTe`
- `MMEMemory:RMDIRECTORY`
- `MMEMemory:MOVe`
- `MMEMemory:MKDIRECTORY`

- `MMEemory:CATalog?`

This is important, otherwise a remote attacker could open a connection to PicoVNA 5 and gain access to your filesystem.

These commands can be enabled using the User Preferences modal in PicoVNA 5.

4.7 Compatibility mode

Compatibility mode may be helpful if you wish to run a SCPI program that was written for another instrument. Some adaptation may still be required, but compatibility mode introduces many new command aliases to try to look more like some other SCPI implementations.

Compatibility mode is not helpful unless facilitating interoperability with other instruments, since the introduced aliases clutter the output of `HELP` and typically represent more convoluted ways of doing things.

Compatibility mode is disabled by default. Use the `SYSTem:COMPATibility` command to enable it.

4.8 Differences from other SCPI implementations

This SCPI implementation differs from some other implementations in a few notable ways. These differences are expected to generally make the interface easier and more convenient to use, but awareness of them is useful.

- *Units are case-sensitive.* Unlike other strings, units *are* case-sensitive. Some SCPI implementations accept case-insensitive units and resolve any ambiguity (e.g. between mHz and MHz) from context. This implementation requires units to be cased appropriately, and does not perform such inference.
- *Help commands.* There is a non-standard command `HELP`. `HELP` can be used to retrieve a list of valid SCPI commands, or ask for help on a particular command (`COMMAND`) by issuing `HELP COMMAND`.
- *All commands have a reply.* All commands that do not produce a specific output will return a single-line status message. The status message will either be `OK` or a single-line human-readable error message, as described in Section 4.3. This mechanism feeds back any errors instantly, but requires the user to perform a read after every write. Use your programming environment's `readLine()` mechanism to do this.

Status replies (`OK`/`not-OK`) will always be one line long, so it is sufficient to perform a single `readLine` operation after every such command. Cases where commands produce multiline output are noted separately.

Users are discouraged from writing code that sends a command and then reads exactly 3 bytes for the response, as this will fail if any errors are encountered.

- *Command execution is synchronous or commands implicitly synchronise.* Some SCPI implementations have a distinction between synchronous and asynchronous commands, and require the user to call `*WAI` when you want to wait for asynchronous operations (such as a measurement sweep) to complete before doing something else. This becomes especially problematic when it comes to receiving errors from the device, because these end up in an asynchronously-populated buffer you must explicitly poll.

In this implementation, commands are either synchronous, or implicitly synchronise. Commands that do things like change settings are synchronous, instantaneous, and return immediately with `"OK"` or an error message. Commands that take a nontrivial amount of time (such

as “do a sweep”) start asynchronously, and the next command you send that depends on the outcome of the asynchronous thing causes it to wait for it to finish.

Consequently, you can send a sequence of commands such as “do some configuration”, “do a sweep”, “read a sweep”, “change more settings”, “do another sweep”, all without any explicit synchronisation messages, and it will be interpreted in the obvious and reasonable way. As well as removing the possibility of mistakes from forgetting to send an explicit synchronisation command, this arrangement means that all errors appear at the time they happen.

Since the SCPI commands are handled by software rather than the VNA itself, there are no performance implications of doing all error/success replies immediately. It's just a round trip on a TCP socket. The cost of having to do a read/write pair for every command is more than outweighed by the benefit of not having to deal with asynchronous error/state handling.

5 Overview of key SCPI commands

5.1 Getting help

Help can be obtained via SCPI. To list all available SCPI commands together with summary information, use `HELP` (with no parameters). If you pass the name of a SCPI command to `HELP`, detailed information about that command is given. If you give the name of a SCPI subtree, summary information about all commands in that SCPI subtree is given.

5.2 Setting up measurements

A measurement can be set up fully configured via SCPI, including loading a calibration, setting the sweep parameters and setting up any de-embedding or enhancement required. This user manual describes relevant commands in the `SENSE`, and `CALCulate` subsystems for doing so.

However, it is often convenient to do the majority of the setup live via the PicoVNA 5 graphical user interface. Alternatively, SCPI can be used to load a session file that was previously configured using the PicoVNA 5 software. To load a session file for a previously-configured session, use the `MMEMory:LOAD:SESSion` command.

5.3 Starting a measurement sweep

A single measurement sweep can be initiated using the `INITiate` command. If a measurement sweep is already running, this command will cause it to be restarted.

5.4 Retrieving data

A note on the utility of markers

The simplest way to retrieve a measurement at one frequency is to place a marker at that point. If the requested frequency does not align with a sweep point, the measurement will be interpolated using the current interpolation preferences set via the PicoVNA 5 user interface. Markers are described in Section 5.6 and the remainder of this section will describe how to retrieve the data from an entire sweep.

Retrieving all the data from a frequency sweep

To retrieve sweep data, use:

- the `CALCulate:DATA <MParam> <Type>` command to retrieve data from the `LIVE` channel,

- the `CALCulate:DATA:MEMory<M> <MParam> <Type>` command to retrieve data from a memory channel, specifying which memory channel using the integral suffix `M`.

These commands will return the data for one `S`-parameter in the requested format. If multiple `S`-parameters are required, simply run the commands repeatedly.

Synchronisation and semantics for retrieving measurement data from the LIVE channel

If a measurement is not in progress, the `CALCulate:DATA` command will return immediately with the data from the most recent sweep. If the most recent sweep was aborted early, the command will return the data that was collected during the partial sweep followed by as many NaNs as are needed to pad the result to the needed number of measurement points.

If continuous-mode sweeping was started from the PicoVNA 5 User Interface, and then stopped, then the `CALCulate:DATA` command will return the data corresponding to what the UI is showing (a mixture of data from two different sweeps).

If continuous-mode sweeping is currently in progress, the `CALCulate:DATA` command will return immediately with an error. If data retrieval is required during continuous mode sweeping, either save the sweep data to a new memory channel and retrieve it from that, or alternatively use the separate binary data broadcast data retrieval interface.

If a sweep (other than continuous-mode sweeping via the PicoVNA 5 User Interface) is in progress, the `CALCulate:DATA` command will block until it has finished so the complete data from the current sweep can be returned.

5.5 Exporting data to a Touchstone or CSV file

The `MMEMemory:STORe:TRACe` command is used to export data to a Touchstone or CSV file. Any type of data (including raw `a/b` data) can be exported to a CSV file, regardless of whether or not it is currently displayed on a graph. The type of file (Touchstone, CSV or raw `a/b`), channel to export (live data or a specified memory channel) and the file path of the output are specified using this command's parameters.

The touchstone data format (`dB/Arg`, `Mag/Arg` or `Re/Im`) is set using the `MMEMemory:STORe:TRACe:OPTion:TOUCHSTONEDATAFORMAT` command. The default is `real/imaginary`. This option is persistent with a session when configured.

The data types to export to a CSV file (`LogMag`, `Time Domain`, `Group Delay`, ...) are set using the `MMEMemory:STORe:TRACe:OPTion:CSVDATAFORMAT` command. The default is to export real and imaginary values. This option is also persistent with a session when configured.

5.6 Using markers

Markers in the frequency-domain

The `MARKer<M>` command can be used to create a marker, or move an existing marker between channels/plots/`S`-parameters. For example, if Marker 1 did not currently exist, the following would create Marker 1 and place it on the LIVE `S11` `LogMag` trace:

```
MARKer1 0,LOGMAG,S11
```

The frequency of the marker can be set using `MARKer<M>:X`. For example, to set Marker 1 to a frequency of 2 GHz, the following command would be used:

```
MARKer1:X 2 GHz
```

The readout from a marker can be obtained using the `MARKer:Query` command. It takes one parameter: the type of data to readout (LogMag, Phase, ...). If the requested frequency (or time) does not align with a measurement point, the measurement will be interpolated using the current interpolation preferences set via the PicoVNA 5 user interface.

Markers in the time-domain

Markers in the time-domain are similar to markers in the frequency-domain, but the marker ID is prefixed with T. To create a time-domain marker 1 on the LIVE S11 Time Domain trace, use:

```
MARKERT1 0,S11
```

The time (or distance) of the marker can be set using `MARKERT<M>:X`. For example, to set Marker T1 to a time of 1 ns, the following command would be used:

```
MARKERT1:X 1 ns
```

Markers are indexed separately in the time-domain from those in the frequency-domain. That is, `MARKER1` (in the frequency-domain) and `MARKERT1` (in the time-domain) have no relation.

Using tracking markers

Tracking markers can be used to find the position of a peak or other features within a trace. To configure tracking for frequency-domain markers use the `MARKer<M>:TRACKing` command, and to configure tracking for time-domain markers use the `MARKERT<M>:TRACKing` command.

For example, the position and value of the maximum within a LogMag trace can be determined as follows:

```
MARKER1:TRACK LOGMAG,GLOBALMAX
MARKER1:X?
MARKER1:Q LOGMAG
```

6 SCPI Command Reference: top-level commands

HELP

Usage:

```
HELP [Command: STRING]
```

Description:

Get help about SCPI commands.

If you pass the name of a SCPI command to this command, detailed information about that command is given. If you give the name of a SCPI subtree, summary information about all commands in that SCPI subtree is given. If no argument is given, summary information is given for all commands.

Parameters:

`Command:` `STRING` Optional. The name of the SCPI command, or subtree, to get help for.

Example:

To get help on the `SENSE:BANDWIDTH` command, use:

```
HELP SENSE:BANDWIDTH
```

Example command returns:

```
SENSe:BAWdth      <Bandwidth: DOUBLE>
```

Change the bandwidth of the sweep.

```
##### PARAMETERS #####
```

```
<Bandwidth: DOUBLE>      The new bandwidth.
```

*IDN?

Usage:

```
*IDN?
```

Description:

Print a string identifying the device and associated software stack.

Returns a comma-separated string identifying the device connected and software version. The fields are: <vendor>,<instrument>,<instrument serial>,<picovna5 version> — <instrument firmware version>

To facilitate interoperability with other software, the value returned by this command may be modified using the SYSTem:IDENtify command.

Parameters:

This command takes no parameters.

Example:

To get information about the connected device and software version, use:

```
*IDN?
```

Example command returns:

```
Pico Technology Ltd,PicoVNA-108,A0000.00000,5.2.7-0-0
```

*OPC?

Usage:

```
*OPC?
```

Description:

Returns 1.

The semantics of the server already guarantee command ordering works correctly, so this command is not required. It exists for compatibility with other instruments.

Parameters:

This command takes no parameters.

*OPT?

Usage:

```
*OPT?
```

Description:

Print a string describing the optional hardware features present on the device.

At present, no such optional features exist, so this command simply returns 1.

To facilitate interoperability with other instruments, the value returned by this command may be modified using the SYSTem:OPTions command.

Parameters:

This command takes no parameters.

RST*Usage:**

*RST

Description:

Reset the device and workspaces.

Any ongoing measurement is stopped. Then:

- If a user reset file has been set using `SYSTem:PRESet:USER:NAME`, and user reset has been enabled using `SYSTem:PRESet:USER`, the current workspace is overwritten with what is stored in that file.
- If `SYSTem:PRESet:SCOPE` has been used to set the reset scope to apply to all workspaces, the same procedure is then performed on all other workspaces.

Parameters:

This command takes no parameters.

TRG*Usage:**

*TRG

Description:

Send a manual trigger event to the device.

Parameters:

This command takes no parameters.

ABORt**Usage:**

ABORt

Description:

Stop all ongoing measurements.

Parameters:

This command takes no parameters.

7 SCPI Command Reference: CALCulate subsystem

CALCulate:AVERaging**Usage:**

`CALCulate:AVERaging <Average: BOOL>`

Description:

Change whether averaging is enabled.

Parameters:

`Average: BOOL` Required. Whether or not averaging should be enabled on the live channel.

Example 1:

To turn averaging on for the live channel, use:

```
CALCULATE: AVERAGING ON
```

Example command returns:

OK

Example 2:

To turn averaging off for the live channel, use:

```
CALCulate:AVERaging OFF
```

Example command returns:

OK

CALCulate:AVERaging:COUNT

Usage:

```
CALCulate:AVERaging:COUNT <NumAverages: INT>
```

Description:

Change the number of averages.

Parameters:

NumAverages: INT Required. The new number of averages.

CALCulate:AVERaging:COUNT?

Usage:

```
CALCulate:AVERaging:COUNT?
```

Description:

Query the number of averages.

Parameters:

This command takes no parameters.

CALCulate:AVERaging?

Usage:

```
CALCulate:AVERaging?
```

Description:

Query if averaging is enabled.

Parameters:

This command takes no parameters.

CALCulate:DATA

Usage:

```
CALCulate:DATA
```

```
<MParam: (S11 | S21 | S12 | S22)>
```

```
<Type: (LOGMAG | MAG | PHASe | TD | REAL | IMAGinary | GD | VSWR | POLARlinear)>
```

Description:

Query the response values for a measurement parameter, for the live channel.

This command will get you the data for one sparam in the requested format. The FORMat:* command can be used to select whether this command outputs ASCII data or raw floating point bytes, and which endianness/precision is used.

If you require multiple measurement parameters, simply run this command repeatedly.

If a measurement is not in progress, this command will return immediately with the data from the most recent sweep. If the most recent sweep was aborted early, this command will return the data that

was collected during the partial sweep followed by as many NaNs as are needed to pad the result to the needed number of measurement points.

If continuous-mode sweeping was started from the PicoVNA 5 User Interface, and then stopped, then this command will return the data corresponding to what the UI is showing (a mixture of data from two different sweeps).

If continuous-mode sweeping is currently in progress, this command will return immediately with an error. If data retrieval is required during continuous mode sweeping, either save the sweep data to a new memory channel and retrieve it from that, or alternatively use the separate binary data broadcast data retrieval interface (see the PicoVNA 5 User Manual for further details).

If a sweep (other than continuous-mode sweeping via the PicoVNA 5 User Interface) is in progress, this command will block until it has finished so the complete data can be returned.

This command does not currently support averaging of data across multiple sweeps (any desired averaging must be implemented by the caller). This command will return the same data regardless of whether or not averaging is enabled.

A natural consequence of this behaviour is that you can perform measurements extremely straightforwardly, without having to think about concurrency unless you're intentionally doing exotic things with multiple SCPI connections. For example, the following code will start a sweep and then retrieve the data from it:

```
# Start a sweep
INIT
```

```
# Retrieve the resulting measurement data. There will be a pause after running the first of these
# while we wait for measurement to finish. The latter 3 will return instantly.
CALC:DATA S11,LOGMAG
CALC:DATA S12,LOGMAG
CALC:DATA S21,LOGMAG
CALC:DATA S22,LOGMAG
```

NaN is represented by the string nan in ASCII mode. In binary mode, a non-signalling NaN is returned. The exact byte representation of this value depends on your hardware/OS, but it will certainly have all bits of the exponent set and at least one bit of the fraction set. That is: it's an IEEE754-compliant NaN, but whether or not it triggers your hardware's floating-point exception mechanism will depend on your platform (and also whether it supports quiet NaNs at all).

You may open a second SCPI connection and send an ABORT command to unblock a SCPI connection that is waiting for this command to complete. This will result in the same partial data semantics as described above. This may be useful if you discover a defect in the measurement configuration or DUT and wish to start again without having to wait. Alternatively, you may drop the SCPI connection, reconnect, and then send ABORT.

Parameters:

MParam Required. Measurement parameter you want data for. Values: S11|S21|S12|S22.
Type Required. The data you want.
 Values: LOGMAG|MAG|PHASe|TD|REAL|IMAGinary|GD|VSWR|POLARlinear.

POLARlinear data outputs the real and imaginary parts of each measurement point, in order.

CALCulate:DATA:LIVE

Usage:

```
CALCulate:DATA:LIVE
```

```
<MParam: (S11 | S21 | S12 | S22)>  
<Type: (LOGMAG | MAG | PHASe | TD | REAL | IMAGinary | GD | VSWR | POLARlinear)>
```

Description:

An alias of CALCulate:DATA.

CALCulate:DATA:MEMory**Usage:**

```
CALCulate:DATA:MEMory
```

```
<MParam: (S11 | S21 | S12 | S22)>
```

```
<Type: (LOGMAG | MAG | PHASe | TD | REAL | IMAGinary | GD | VSWR | POLARlinear)>
```

Description:

Identical to CALC:DATA, but used to retrieve data for memory channels.

Specify the desired memory channel in the integral suffix of this command name, such as:

```
CALC:DATA:MEM0
```

```
CALC:DATA:MEM4
```

etc.

It is always valid to retrieve data from a memory trace, regardless of whether or not continuous-mode sweeping is enabled via the PicoVNA 5 User Interface.

Parameters:

MParam Required. Measurement parameter you want data for. Values: S11|S21|S12|S22.

Type Required. The data you want.

Values: LOGMAG|MAG|PHASe|TD|REAL|IMAGinary|GD|VSWR|POLARlinear.

CALCulate:DATA:STIMulus?**Usage:**

```
CALCulate:DATA:STIMulus?
```

Description:

Output the stimulus values for the given sweep segment.

Parameters:

This command takes no parameters.

CALCulate:SMOothing**Usage:**

```
CALCulate:SMOothing <Smooth: BOOL>
```

Description:

Change whether smoothing is enabled.

Parameters:

Smooth: BOOL Required. Whether or not smoothing should be enabled on this channel.

CALCulate:SMOothing?**Usage:**

```
CALCulate:SMOothing?
```

Description:

Query if smoothing is enabled.

Parameters:

This command takes no parameters.

CALCulate:SMOothing:FACTOR**Usage:**

CALCulate:SMOothing:FACTOR <Smoothing: DOUBLE>

Description:

Change the smoothing factor.

Parameters:

Smoothing: DOUBLE Required. The new smoothing factor.

CALCulate:SMOothing:FACTOR?**Usage:**

CALCulate:SMOothing:FACTOR?

Description:

Determine the smoothing factor.

Parameters:

This command takes no parameters.

CALCulate:TIME**Usage:**

CALCulate:TIME <Mode: (BPASs | LPASs)>

Description:

Set whether to use a low-pass or band-pass time-domain transform.

The default is low-pass.

Parameters:

Mode Required. The new time-domain mode. Values: BPASs|LPASs.

CALCulate:TIME?**Usage:**

CALCulate:TIME?

Description:

Query whether we are currently using a low-pass or band-pass time-domain transform.

Parameters:

This command takes no parameters.

CALCulate:TIME:DCterm**Usage:**

CALCulate:TIME:DCterm <Termination: (AUTO | OPEN | SHORT | RESISTive)>

Description:

Set the type of DC termination to use in the time domain transform.

Parameters:

Termination Required. The type of DC termination to use.
Values: AUTO|OPEN|SHORT|RESISTive.

CALCulate:TIME:DCterm?

Usage:

CALCulate:TIME:DCterm?

Description:

Query the type of DC termination used in the time domain transform.

Returns: AUTO, OPEN, SHORT, RESISTIVE or NONE.

Parameters:

This command takes no parameters.

CALCulate:TIME:EFFD**Usage:**

CALCulate:TIME:EFFD <D: DOUBLE>

Description:

Set the effective dielectric constant used in conversions between time and distance for time-domain measurements.

Parameters:

EffD: DOUBLE Required. The new effective dielectric constant used in for time-domain measurements.

CALCulate:TIME:EFFD?**Usage:**

CALCulate:TIME:EFFD?

Description:

Query the effective dielectric constant used in conversions between time and distance for time-domain measurements.

Parameters:

This command takes no parameters.

CALCulate:TIME:RESPonse**Usage:**

CALCulate:TIME:RESPonse <Mode: (IMPulse | STEP)>

Description:

Set whether to return the impulse or step response of the DUT for time-domain measurements.

Parameters:

Mode Required. The new stimulus mode.
Values: =IMPulse|STEP=

CALC:TIME:RESP?**Usage:**

CALCulate:TIME:RESPonse?

Description:

Query whether to return the impulse or step response of the DUT for time-domain measurements.

Returns IMPULSE or STEP.

Parameters:

This command takes no parameters.

CALCulate:TIME:RTERM

Usage:

CALCulate:TIME:RTERM <RT: DOUBLE>

Description:

8 SCPI Command Reference: CHANnel subsystem

CHANnel<Ch>:COPY

Usage:

CHANnel<Ch>:COPY [MemChanID: INT]

Description:

Save a channel to a memory channel.

If no parameters are given, the channel is saved to a new memory channel and its ID is printed.

If you give an ID of a memory channel, it will be overwritten or created.

Afterwards, the new channel can be assigned a friendly name using CONFIGure:CHANnel:NAME.

Parameters:

MemChanID: INT Optional. The ID of the memory channel to write.

CHANnel:MEMory:CATalog:COUNT?

Usage:

CHANnel:MEMory:CATalog:COUNT?

Description:

Print the number of existing memory channels.

Parameters:

This command takes no parameters.

CHANnel:MEMory:CATalog?:

Usage:

CHANnel:MEMory:CATalog?

Description:

Print the names and ID of all memory channels that exist.

Parameters:

This command takes no parameters.

CHANnel:MEMory:DELeTe

Usage:

CHANnel:MEMory:DELeTe <MemChanID: INT>

Description:

Delete the memory channel with the given ID.

Parameters:

<MemChanID: INT> Required. The ID of the memory channel to delete.

CHANnel:MEMory:DELeTe:ALL:

Usage:

CHANnel:MEMory:DELeTe:ALL

Description:

Delete all memory channels.

Parameters:

This command takes no parameters.

HELP**Usage:**

HELP [Command: STRING]

Description:

Get help about SCPI commands.

If you pass the name of a SCPI command to this command, detailed information about that command is given. If you give the name of a SCPI subtree, summary information about all commands in that SCPI subtree is given. If no argument is given, summary information is given for all commands.

Parameters:

Command: STRING Optional. The name of the SCPI command, or subtree, to get help for.

INITiate:**Usage:**

INITiate

Description:

Initiate a measurement sweep. If a measurement is already running, this command will cause it to be restarted.

For compatibility with other instruments, this command includes an optional integral suffix. However, if the integral suffix is specified, the command will return an error unless it is 0.

Parameters:

This command takes no parameters.

INSTrument:PORT:COUNT?**Usage:**

INSTrument:PORT:COUNT?

Determine how many ports the attached VNA has.

Parameters:

This command takes no parameters.

MARKer<M>**Usage:**

```
MARKer<M> <Channel: STRING>
           <SeriesType>
           <SParam: (S11 | S12 | S21 | S22)>
```

Description:

Create a frequency-domain marker, or move it between channels/plots/S-parameters.

Set marker M to be on the given channel/trace/measurement-parameter. If the marker does not exist, it is created. If a marker with that ID already exists, it is changed accordingly.

Parameters:

<code>Channel</code> : <code>STRING</code>	Required. The name or ID of the channel the marker should be placed on.
<code>SeriesType</code>	Required. The name or ID of the channel the marker should be placed on.
<code>SParam</code> : (<code>S11</code> <code>S12</code> <code>S21</code> <code>S22</code>)	Required. The measurement parameter the marker is to be associated with.

The following values can be passed to the `SeriesType` parameter:

- `LOGMAG`. Marker should appear on logmag traces.
- `LINMAG`. Marker should appear on linear magnitude traces.
- `PHASe`. Marker should appear on phase traces.
- `REa1`. Marker should appear on real traces.
- `IMag`. Marker should appear on imaginary traces.
- `GD`. Marker should appear on group-delay charts.
- `VSWR`. Marker should appear on VSWR charts.
- `POLARlinear`. Marker should appear on polar-linear charts.
- `SMITH`. Marker should appear on Smith charts.

MARKer<M>:DELETE**Usage:**

MARKer<M>:DELETE

Description:

Delete the frequency-domain marker.

MARKer<M>:Query**Usage:**

MARKer<M>:Query <Type: (`LOGMAG` | `LINMAG` | `PHASe` | `REa1` | `IMag` | `GD` | `VSWR`)>

Description:

Print a measurement made at the frequency of this frequency-domain marker.

Parameters:

`Type` Required. The measurement to print.

The following values can be passed to the `Type` parameter:

- `LOGMAG`. Print log-magnitude for the associated sparam.
- `LINMAG`. Print linear magnitude for the associated sparam.
- `PHASe`. Print phase for the associated sparam.
- `REa1`. Print real for the associated sparam.
- `IMag`. Print imaginary for the associated sparam.
- `GD`. Print Group-delay for the associated sparam.

- VSWR. Print VSWR for the associated sparam.
- ZRE. Print the real part of the impedance in ohms (reflection measurements only).
- ZIM. Print the imaginary part of the impedance (reflection measurements only).
- INDCAP. Print the imaginary part of the impedance as an inductance or capacitance measurement (reflection measurements only).

MARKer<M>:READOUT

Usage:

MARKer<M>:READOUT

<Type: (LOGMAG | LINMAG | PHASe | TD | REa1 | IMag | GD | VSWR)>
<State: BOOL>

Description:

Configure the readouts shown in the UI for this frequency-domain marker.

Parameters:

Type Required. The value to extract.
State: BOOL Required. Whether or not to show this readout value in the UI.

The following values can be passed to the Type parameter:

- LOGMAG. Log-magnitude.
- LINMAG. Linear magnitude.
- PHASe. Phase.
- REa1. Real.
- IMag. Imaginary.
- GD. Group Delay.
- VSWR. VSWR.

MARKer<M>:READOUT:AOFF

Usage:

MARKer<M>:READOUT:AOFF

Description:

Disable all readouts from this frequency-domain marker in the PicoVNA 5 User Interface.

Parameters:

This command takes no parameters.

MARKer<M>:TRACKing

Usage:

MARKer<M>:TRACKing

<TrackingFunction: (LOGMAG | LINMAG | PHASe | REa1 | IMag | GD | VSWR | OFF)>
[TrackingFeature: (GLOBALMAX | GLOBALMIN | NTHMAX | NTHMIN | NTHEXTREME
| NTHMINUSXDBPOINT | NTHPLUSXDBPOINT)]
[OptionalParam1: STRING]

[OptionalParam2: STRING]

Description:

Turn tracking on or off for a frequency-domain marker, and specify the series type to track. To turn tracking on, specify a tracking function, tracking feature and any other parameters required for the tracking feature. To turn tracking off, specify OFF for the tracking function and no further arguments.

This command may require a number of additional parameters, depending on the tracking feature specified.

Supported tracking features are:

- Global maximum (GLOBALMAX). Required parameters: none.
- Global minimum (GLOBALMIN). Required parameters: none.
- n th local maximum (NTHMAX). Required parameters:
 - n (natural number)
 - hysteresis (real number)
- n th local minimum (NTHMIN). Required parameters:
 - n (natural number)
 - hysteresis (real number)
- n th local extremum (NTHEXTREME). Required parameters:
 - n (natural number)
 - hysteresis (real number)
- n th x dB point (NTHXDBPOINT). Only available for LogMag traces. Required parameters:
 - n (natural number) x (real number)

Optional parameters are specified by appending them to the required parameters (see examples below).

Parameters:

TrackingFunction	Required. The series type to track, or OFF to disable tracking.
TrackingFeature	Required if enabling tracking. The feature to track (see list above).
OptionalParam1	Required only for some tracking features (see above).
OptionalParam2	Required only for some tracking features (see above).

Example:

To set marker 1 to track the global minimum of SWR:

```
MARKER1:TRACK VSWR,GLOBALMIN
```

Example:

To set marker 1 to track the 3rd local extremum of phase:

```
MARKER1:TRACK PHASE,NTHEXTREME,3,0.5
```

Example:

To set marker 1 to track the 1st (trace max)-6 dB point.

```
MARKER1:TRACK LOGMAG,NTHMINUXDBPOINT,1,-6
```

MARKer<M>:TYPE**Usage:**

MARKer<M>:TYPE <MarkerType: (REFerence | NORmal)>

Description:

Change the type of a frequency-domain marker (reference or normal).

Does nothing if the marker is already of the specified type.

Parameters:

MarkerType Required. The type of marker that this marker should become.

The following values can be passed to the MarkerType parameter:

- REFerence. The marker will become a reference marker.
- NORmal. The marker will become a normal (non-reference) marker.

MARKer<M>:TYPE?**Usage:**

MARKer<M>:TYPE?

Description:

Query if a frequency-domain marker is a reference marker or not.

Parameters:

This command takes no parameters.

MARKer<M>:VISible**Usage:**

MARKer<M>:VISible <Visible: BOOL>

Description:

Temporarily hide or show a frequency-domain marker.

Parameters:

Visible: BOOL ON (show marker), OFF (hide marker)

MARKer<M>:VISible?**Usage:**

MARKer<M>:VISible?

Description:

Determine if a frequency-domain marker has been hidden.

Parameters:

This command takes no parameters.

MARKer<M>:X**Usage:**

MARKer<M>:X <X: DOUBLE>

Description:

Set the x-position of the frequency-domain marker. Specify the frequency at which to position the marker.

Parameters:

X: DOUBLE The x-value at which to position the marker.

MARKer<M>:X?**Usage:**

MARKer<M>:X?

Description:

Get the x -position of the frequency-domain marker.

Parameters:

This command takes no parameters.

MARKERT<M>**Usage:**

```
MARKERT<M>    <Channel: STRING>
               <SParam: (S11 | S12 | S21 | S22)>
```

Description:

Create a time-domain marker, or move it between channels/S-parameters.

Set marker M to be on the given channel/trace/measurement-parameter. If the marker does not exist, it is created. If a marker with that ID already exists, it is changed accordingly.

Parameters:

Channel: STRING	Required. The name or ID of the channel the marker should be placed on.
SParam: (S11 S12 S21 S22)	Required. The measurement parameter the marker is to be associated with.

MARKER<M>:DELETE**Usage:**

MARKER<M>:DELETE

Description:

Delete the time-domain marker.

MARKERT<M>:Query**Usage:**

MARKERT<M>:Query

Description:

Extract some interesting value at the point this frequency-domain marker is associated with.

Parameters:

This command takes no parameters.

MARKer<M>:TRACKing**Usage:**

```
MARKERT<M>:TRACKing
    <TrackingFunction: (TD | OFF)>
    [TrackingFeature: (GLOBALMAX | GLOBALMIN | NTHMAX | NTHMIN | NTHEXTREME)]
    [OptionalParam1: STRING]
    [OptionalParam2: STRING]
```

Description:

Turn tracking on or off for a time-domain marker, and specify the series type to track. To turn tracking on, specify a tracking function, tracking feature and any other parameters required for the tracking feature. To turn tracking off, specify OFF for the tracking function and no further arguments.

This command may require a number of additional parameters, depending on the tracking feature specified.

Supported tracking features are:

- Global maximum (GLOBALMAX). Required parameters: none.
- Global minimum (GLOBALMIN). Required parameters: none.
- n th local maximum (NTHMAX). Required parameters:
 - n (natural number)
 - hysteresis (real number)
- n th local minimum (NTHMIN). Required parameters:
 - n (natural number)
 - hysteresis (real number)
- n th local extremum (NTHEXTREME). Required parameters:
 - n (natural number)
 - hysteresis (real number)

Optional parameters are specified by appending them to the required parameters (see examples below).

Parameters:

TrackingFunction	Required. The series type to track, or OFF to disable tracking.
TrackingFeature	Required if enabling tracking. The feature to track (see list above).
OptionalParam1	Required only for some tracking features (see above).
OptionalParam2	Required only for some tracking features (see above).

Example:

To set marker T1 to track the global minimum of the step/impulse response (as configured separately):

```
MARKERT1:TRACK TD,GLOBALMIN
```

Example:

To set marker T1 to track the 3rd local extremum of the step/impulse response (as configured separately):

```
MARKERT1:TRACK TD,NTHEXTREME,3,0.5
```

MARKERT<M>:TYPE

Usage:

```
MARKERT<M>:TYPE <MarkerType: (REFERence | NORmal)>
```

Description:

Change the type of a time-domain marker (reference or normal).

Does nothing if the marker is already of the specified type.

Parameters:

MarkerType Required. The type of marker that this marker should become.

The following values can be passed to the MarkerType parameter:

- REFERENCE. The marker will become a reference marker.
- NORmal. The marker will become a normal (non-reference) marker.

MARKERT<M>:TYPE?**Usage:**

MARKERT<M>:TYPE?

Description:

Query if a time-domain marker is a reference marker or not.

Parameters:

This command takes no parameters.

MARKERT<M>:VISible**Usage:**

MARKERT<M>:VISible <Visible: BOOL>

Description:

Temporarily hide or show a time-domain marker.

Parameters:

Visible: BOOL ON (show marker), OFF (hide marker)

MARKERT<M>:VISible?**Usage:**

MARKERT<M>:VISible?

Description:

Determine if a time-domain marker has been hidden.

Parameters:

This command takes no parameters.

MARKERT<M>:X**Usage:**

MARKERT<M>:X <X: DOUBLE>

Description:

Set the x -position of the time-domain marker. Specify a time (or distance) at which to position the marker.

Parameters:

X: DOUBLE The x -value at which to position the marker.

MARKERT<M>:X?**Usage:**

MARKERT<M>:X?

Description:

Get the x -position of the time-domain marker.

Parameters:

This command takes no parameters.

MMEMory:APPLY:CALibration**Usage:**

MMEMory:APPLY:CALibration

Description:

Load a user calibration from a file and immediately apply it in the current workspace.

The user calibration to be loaded must have been performed from the PicoVNA 5 software (i.e. it must be a .calx file). It is not possible to load calibrations that were performed by the PicoVNA 3 software (from .cal files) via SCPI.

To export a calibration from the PicoVNA 5 software to a file, so this command can be used to load it, use the Export feature in the Calibration History modal, accessed via the Calibrations → My Calibrations menu.

This does not add the calibration to the list of calibrations known to the server (it does not appear in the PicoVNA 5's my calibrations list).

Parameters:

Path: `STRING` Required. Path to the calibration to load.

MMEMory:CATalog?

Usage:

MMEMory:CATalog? [Dir: `STRING`]

Description:

List the files in the specified directory, or the current directory if none is specified.

For security reasons, this command will have no effect unless filesystem commands are explicitly enabled via User Preferences in the PicoVNA 5 software.

Parameters:

Dir: `STRING` The directory to list.

MMEMory:CDirectory

Usage:

MMEMory:CDirectory <NewPath: `STRING`>

Description:

Change the current working directory (the directory that paths are interpreted in relation to).

Parameters:

NewPath Required. The new path to interpret relative paths in relation to.

MMEMory:COPY

Usage:

MMEMory:COPY <Source: `STRING`> <Dest: `STRING`>

Description:

Copies a file.

For security reasons, this command will have no effect unless filesystem commands are explicitly enabled via User Preferences in the PicoVNA 5 software.

Parameters:

Source: `STRING` Required. Copy from.

Dest: `STRING` Required. Copy to.

MMEMory:CP

Alias of MMEM: COPY.

MMEMory:DELeTe

Usage:

MMEMory:DELeTe <File: `STRING`> [Dir: `BOOL`]

Description:

Delete the selected file.

For security reasons, this command will have no effect unless filesystem commands are explicitly enabled via User Preferences in the PicoVNA 5 software.

Parameters:

File: STRING The file to delete.
Dir: BOOL Optional. Whether or not to delete read-only files.

MMEMemory:LOAD:SESSion**Usage:**

MMEMemory:LOAD:SESSion [File: STRING]

Description:

Import a previously-saved session.

The session may have been saved using MMEMemory:STORe:SESSion, or the the save session function in the PicoVNA 5 software.

Parameters:

File: STRING Optional. The file to load from.

MMEMemory:LOAD:STATe**Usage:**

MMEMemory:LOAD:STATe <Compatibility: STRING> [File: STRING]

Description:

Import the current configuration from a file.

Parameters:

Compatibility: STRING Required. May be 1 (for compatibility with SCPI standard), or may be the target file.
File: STRING Optional. The file to load from, if using the other argument for SCPI compatibility. Otherwise, omitted.

MMEMemory:LS

Alias of MMEMemory:CATalog?.

MMEMemory:MKDIRectory**Usage:**

MMEMemory:MKDIRectory <Dir: STRING>

Description:

Creates a directory.

Parameters:

Dir: STRING Required. The directory to create.

For security reasons, this command will have no effect unless filesystem commands are explicitly enabled via User Preferences in the PicoVNA 5 software.

MMEMemory:MOVE**Usage:**

MMEMemory:MOVE <Source: STRING> <Dest: STRING>

Usage:

Move a file to another location.

For security reasons, this command will have no effect unless filesystem commands are explicitly enabled via User Preferences in the PicoVNA 5 software.

Parameters:

Source: STRING Required. Move from.
Dest: STRING Required. Move to.

MMEMemory:MV

Alias of MMEMemory:MOVE.

MMEMemory:RM

Alias of MMEMemory:DELEte.

MMEMemory:RMDIRectory**Usage:**

MMEMemory:RMDIRectory <File: STRING>

Description:

Delete the selected directory and its contents.

For security reasons, this command will have no effect unless filesystem commands are explicitly enabled via User Preferences in the PicoVNA 5 software.

Parameters:

File: STRING Required. The directory to delete.

MMEMemory:STORe:SESSion**Usage:**

MMEMemory:STORe:SESSion [File: STRING]

Description:

Export the current session to a file.

Parameters:

File: STRING Optional. The file to export to.

MMEMemory:STORe:STATe**Usage:**

MMEMemory:STORe:STATe <Compatibility: STRING> [File: STRING]

Description:

Export the current configuration to a file.

Parameters:

Compatibility: STRING Required. May be 1 (for compatibility with SCPI standard), or may be the target file.

File: STRING Optional. The file to export to, if using the other argument for SCPI compatibility. Otherwise, omitted.

MMEMemory:STORe:TRACe**Usage:**

MMEMemory:STORe:TRACe <Channel: STRING> <Type: STRING> <OutputFilePath: STRING>

Description:

Output data to Touchstone or CSV format. Raw A/B data can also be exported where it is available.

The data (S-parameters) to export are set by the following commands:

- `MMEMory:STORe:TRACe:OPTion:NUMPORTS`
- `MMEMory:STORe:TRACe:OPTion:ONEPORTPARAMETER`

Touchstone options are set by the following commands:

- `MMEMory:STORe:TRACe:OPTion:TABS`
- `MMEMory:STORe:TRACe:OPTion:TOUCHSTONEDATAFORMAT`

CSV options are set by the following commands:

- `MMEMory:STORe:TRACe:OPTion:SEPARATOR`
- `MMEMory:STORe:TRACe:OPTion:CSVDATAFORMAT`

This command synchronises in the same way as the `CALCulate:DATA` command. If the channel to export data from is a memory channel, the command will export the data and return immediately. Otherwise:

- If a measurement is not in progress, this command will export the data from the most recent sweep and return immediately.
- If the most recent sweep was aborted early, this command will export the data that was collected during the partial sweep followed by as many NaNs as are needed to pad the result to the needed number of measurement points.
- If continuous-mode sweeping was started from the PicoVNA 5 User Interface, and then stopped, then this command will export the data corresponding to what the UI is showing (a mixture of data from two different sweeps).
- If continuous-mode sweeping is currently in progress, this command will return immediately with an error and not export any data. If data export is required during continuous mode sweeping, save the sweep data to a new memory channel and export it from that.
- If a sweep (other than continuous-mode sweeping via the PicoVNA 5 User Interface) is in progress, this command will block until it has finished so the complete data from the current sweep can be exported.

This command does not currently support averaging of data across multiple sweeps (any desired averaging must be implemented by the caller). This command will export the same data regardless of whether or not averaging is enabled.

Parameters:

<code>Channel:</code>	<code>STRING</code>	Required. Name of the channel to export. See documentation on specifying channels.
<code>Type:</code>	<code>STRING</code>	Required. The type of file to export. Valid options are <code>S2P CSV RAWAB</code> .
<code>OutputFilePath:</code>	<code>STRING</code>	Required. The relative path (including file name and extension) of the output file that will be created or overwritten. See documentation on specifying file paths.

`MMEMory:STORe:TRACe:OPTion:CSVDATAFORMAT`

Usage:

MMEMory:STORe:TRACe:OPTion:CSVDATAFORMAT <DataTypes: STRING>

Description:

Set the data types that will be exported to a CSV file by the MMEMory:STORe:TRACe command when the CSV format is specified. The data types to export are specified as a colon-separated list.

Note that for group delay, the first value will always be NaN, because group delay cannot be computed for the first measurement point.

It is not valid to request both frequency-domain data types and time-domain data types in the same CSV file (it is necessary to perform multiple CSV exports if both frequency-domain and time-domain data are required).

For example, to cause the MMEMory:STORe:TRACe to output (only) time domain data for each measurement parameter, use: MMEMory:STORe:TRACe:OPTion:CSVDATAFORMAT TD

To cause the MMEMory:STORe:TRACe to output linear magnitude and phase data for each measurement parameter, use: MMEMory:STORe:TRACe:OPTion:CSVDATAFORMAT MAG;PHASE

Parameters:

DataTypes: STRING Required. A colon-separated list of data types that should be exported whenever saving a CSV file.

The data types can be any of:

- LOGMAG – Log-magnitude data (dB)
- MAG – Linear magnitude (U)
- PHASE – Phase (degrees)
- TD – Time Domain (U)
- REAL – Real (U)
- IMAG – Imaginary (U)
- GD – Group Delay.
- VSWR – VSWR (U)
- POLAR – Outputs the real and imaginary parts of each measurement point, in order.

MMEMory:STORe:TRACe:OPTion:NUMPORTS**Usage:**

MMEMory:STORe:TRACe:OPTion:NUMPORTS <NumPorts: INT>

Description:

For Touchstone and CSV exports, set whether to export:

- 1-port S-parameters – export data for S11/S22 (1-port S-parameters), *OR*
- 2-port S-parameters – export data for S11,S21,S12,S22 (2-port S-parameters)

By default, the 2-port option is selected and all S-parameters are exported when using the `MMEMemory:STORe:TRACe` command for exporting CSV or Touchstone files.

If the 2-port option is selected, the `MMEMemory:STORe:TRACe:OPTion:ONEPORTPARAMETER` command can be used to select which of S11 OR S22 is exported by the `MMEMemory:STORe:TRACe` command.

Parameters:

`NumPorts`: INT Required. A single character: either 1 (for S11/S22 exports) or 2 (for S11,S21,S12,S22 exports).

`MMEMemory:STORe:TRACe:OPTion:ONEPORTPARAMETER`

Usage:

`MMEMemory:STORe:TRACe:OPTion:ONEPORTPARAMETER <OnePortParameter: STRING>`

Description:

If 1-port exports have been selected using `MMEMemory:STORe:TRACe:OPTion:NUMPORTS`, which of S11 or S22 will be exported in text file exports?

Parameters:

`OnePortParameter`: STRING Required. Either S11 or S22.

`MMEMemory:STORe:TRACe:OPTion:SEPARATOR`

Usage:

`MMEMemory:STORe:TRACe:OPTion:SEPARATOR <Separator: STRING>`

Description:

Set the separator for CSV file exports (including raw a/b data exports). The separator must be a single character.

Parameters:

`Separator`: STRING Required. The separator to use for CSV file exports. The separator must be a single character.

`MMEMemory:STORe:TRACe:OPTion:TABS`

Usage:

`MMEMemory:STORe:TRACe:OPTion:TABS <UseTabs: BOOL>`

Change whether tabs are used as column separators in Touchstone exports.

Parameters:

`UseTabs`: BOOL Required. Whether or not to use tabs instead of spaces as column separators in touchstone files.

`MMEMemory:STORe:TRACe:OPTion:TOUCHSTONEDATAFORMAT`

Usage:

`MMEMemory:STORe:TRACe:OPTion:TOUCHSTONEDATAFORMAT <Format: STRING>`

Description:

Set the data format used by the `MMEMemory:STORe:TRACe` command during Touchstone export.

Parameters:

`Format`: STRING Required. One of: REIM, MAGANG or DBANG.

The following values can be passed to the Format parameter:

- REIM – Real/Imaginary.

- MAGANG – Magnitude/Angle.
- DBANG – dB Magnitude/Angle.

The default is Real/Imaginary.

SENSe:BANDwidth

Usage:

SENSe:BANDwidth <Bandwidth: DOUBLE>

Description:

Change the bandwidth of the sweep.

Parameters:

Bandwidth: DOUBLE The new bandwidth.

SENSe:BANDwidth?

Usage:

SENSe:BANDwidth?

Description:

Query the bandwidth of the sweep.

Parameters:

This command takes no parameters.

SENSe:FREQuency:STARt

Usage:

SENSe:FREQuency:STARt <StartFreq: DOUBLE>

Description:

Change the start frequency of the sweep.

Parameters:

StartFreq: DOUBLE The new start frequency.

SENSe:FREQuency:STARt?

Usage:

SENSe:FREQuency:STARt?

Description:

Query the start frequency of the sweep.

Parameters:

This command takes no parameters.

SENSe:FREQuency:STOP

Usage:

SENSe:FREQuency:STOP <StopFreq: DOUBLE>

Description:

Change the end frequency of the sweep.

Parameters:

StopFreq: DOUBLE The new end frequency.

SENSe:FREQuency:STOP?

Usage:

SENSe:FREQuency:STOP?

Description:

Query the end frequency of the sweep.

Parameters:

This command takes no parameters.

SENSe:LEVel**Usage:**

SENSe:LEVel <PowerLevel: DOUBLE>

Description:

Change the power level of the sweep.

Parameters:

PowerLevel: DOUBLE The new power level.

SENSe:LEVel?**Usage:**

SENSe:LEVel?

Description:

Query the power level of the sweep.

Parameters:

This command takes no parameters.

SENSe:SWEp:POINts**Usage:**

SENSe:SWEp:STEP <FreqStep: DOUBLE>

Description:

Change the frequency step of the sweep.

This will update the number of sweep points as necessary to produce a sweep with this step over the configured frequency range.

Parameters:

FreqStep: DOUBLE The new frequency step.

SENSe:SWEp:POINts?**Usage:**

SENSe:SWEp:POINts?

Description:

Query the point count of the sweep.

Parameters:

This command takes no parameters.

SENSe:SWEp:STEP**Usage:**

SENSe:SWEp:STEP <FreqStep: DOUBLE>

Description:

Change the frequency step of the sweep.

This will update the number of sweep points as necessary to produce a sweep with this step over the configured frequency range.

Parameters:

FreqStep: DOUBLE The new frequency step.

SENSe:SWEEP:STEP?**Usage:**

SENSe:SWEEP:STEP?

Description:

Query the frequency step of the sweep.

Parameters:

This command takes no parameters.

SENSe:TEMPerature?**Usage:**

SENSe:TEMPerature?

Description:

Returns the instrument's internal temperature in degrees centigrade.

If no instrument is yet connected, or an instrument is connected that does not support temperature readings, an error message will be returned.

If the instrument is still initialising, NaN will be returned. In this case, run this command again later to obtain a temperature reading.

Parameters:

This command takes no parameters.

