RsCMPX_BluetoothMeas

Release 5.0.70.3

Rohde & Schwarz

Apr 19, 2024

CONTENTS:


5G
CMX/P

CHAPTER

ONE

REVISION HISTORY

1.1 RsCMPX_BluetoothMeas

Rohde & Schwarz CMX/CMP Bluetooth Measurement RsCMPX_BluetoothMeas instrument driver.

Basic Hello-World code:

```
from RsCMPX_BluetoothMeas import *

instr = RsCMPX_BluetoothMeas('TCPIP::192.168.2.101::hislip0')
idn = instr.query('*IDN?')
print('Hello, I am: ' + idn)
```

Supported instruments: CMX500, CMP180, PVT360

The package is hosted here: https://pypi.org/project/RsCMPX-BluetoothMeas/

Documentation: https://RsCMPX-BluetoothMeas.readthedocs.io/

Examples: https://github.com/Rohde-Schwarz/Examples/

1.1.1 Version history

Latest release notes summary: Update for FW 5.0.70

Version 5.0.70

- Update for FW 5.0.70

Version 4.0.185

- First release for FW 4.0.185

CHAPTER

TWO

GETTING STARTED

2.1 Introduction

RsCMPX_BluetoothMeas is a Python remote-control communication module for Rohde & Schwarz SCPI-based Test and Measurement Instruments. It represents SCPI commands as fixed APIs and hence provides SCPI autocompletion and helps you to avoid common string typing mistakes.

Basic example of the idea:
SCPI command:
`SYSTem:REFerence:FREQuency:SOURce`
Python module representation:
writing:
`driver.system.reference.frequency.source.set()`
reading:
`driver.system.reference.frequency.source.get()`

Check out this example for RsCmpx-Base and RsCmpx-Gprf:

```
"""
# GitHub examples repository path: CMXP/Python/RsCmxp_xxx_ScpiPackages

Example on how to use the python RsCmx auto-generated instrument drivers for
RsCmpx_Base and RsCmpx_Gprf (Base and GPRF) in one script with shared VISA session.
"""

from RsCMPX_Base.RsCMPX_Base import RsCMPX_Base  # install from pypi.org
from RsCMPX_Base import enums as base_enums
from RsCMPX_Base import repcap as base_repcap

from RsCMPX_Gprf.RsCMPX_Gprf import RsCMPX_Gprf  # install from pypi.org
from RsCMPX_Gprf.CustomFiles.reliability import ReliabilityEventArgs
from RsCMPX_Gprf import enums as gprf_enums
from RsCMPX_Gprf import repcap as gprf_repcaps
```

(continues on next page)

(continued from previous page)

```
# CMX Base init
cmx_base = RsCMPX_Base('TCPIP::10.112.1.116', False, True)
print(f'CMX Base IND: {cmx_base.utilities.idn_string}')
print(f'CMX Instrument options:\n{",".join(cmx_base.utilities.instrument_options)}')
cmx_base.utilities.visa_timeout = 5000  # default is 10000

# Sends OPC after each command
cmx_base.utilities.opc_query_after_write = False
# Checks for syst:err? after each command / query - default value after init is True
cmx_base.utilities.instrument_status_checking = True

# Self-test
self_test = cmx_base.utilities.self_test()
print(f'CMW self-test result: {self_test} - {"Passed" if self_test[0] == 0 else "Failed"}
↪"')
# Reference Frequency Source
cmx_base.system.reference.frequency.source_set(base_enums.SourceIntExt.INTernal)

# CMX RsCMPX_Gprf Init - reuse the session of the cmx_base, rather than creating another␣
↪one
cmx_gprf = RsCMPX_Gprf.from_existing_session(cmx_base)
cmx_gprf.utilities.visa_timeout = 5000

# Driver's Interface reliability offers a convenient way of reacting on the return value␣
↪Reliability Indicator
cmx_gprf.reliability.ExceptionOnError = True  # default is 10000


# Callback to use for the reliability indicator update events
def my_reliability_handler(event_args: ReliabilityEventArgs):
        print(f'GPRF Reliability updated.\nContext: {event_args.context}\nMessage:
↪{event_args.message}')


# We register a callback for each change in the reliability indicator
cmx_gprf.reliability.on_update_handler = my_reliability_handler

# You can obtain the last value of the returned reliability
print(f"\nReliability last value: {cmx_gprf.reliability.last_value}, context '{cmx_gprf.
↪reliability.last_context}', message: {cmx_gprf.reliability.last_message}")

# Close the sessions
cmx_gprf.close()
cmx_base.close()
```

Couple of reasons why to choose this module over plain SCPI approach:

- Type-safe API using typing module
- You can still use the plain SCPI communication
- You can select which VISA to use or even not use any VISA at all
- Initialization of a new session is straight-forward, no need to set any other properties

- Many useful features are already implemented - reset, self-test, opc-synchronization, error checking, option checking
- Binary data blocks transfer in both directions
- Transfer of arrays of numbers in binary or ASCII format
- File transfers in both directions
- Events generation in case of error, sent data, received data, chunk data (for big files transfer)
- Multithreading session locking - you can use multiple threads talking to one instrument at the same time
- Logging feature tailored for SCPI communication - different for binary and ascii data

2.2 Installation

RsCMPX_BluetoothMeas is hosted on pypi.org. You can install it with pip (for example, `pip.exe` for Windows), or if you are using Pycharm (and you should be :-) direct in the Pycharm `Packet Management GUI`.

Preconditions

- Installed VISA. You can skip this if you plan to use only socket LAN connection. Download the Rohde & Schwarz VISA for Windows, Linux, Mac OS from here

Option 1 - Installing with pip.exe under Windows

- Start the command console: `WinKey + R`, type `cmd` and hit ENTER
- Change the working directory to the Python installation of your choice (adjust the user name and python version in the path):

```
cd c:\Users\John\AppData\Local\Programs\Python\Python37\Scripts
```

- Install with the command: `pip install RsCMPX_BluetoothMeas`

Option 2 - Installing in Pycharm

- In Pycharm Menu `File->Settings->Project->Project Interpreter` click on the '+' button on the top left (the last PyCharm version)
- Type `RsCMPX_BluetoothMeas` in the search box
- If you are behind a Proxy server, configure it in the Menu: `File->Settings->Appearance->System Settings->HTTP Proxy`

For more information about Rohde & Schwarz instrument remote control, check out our Instrument_Remote_Control_Web_Series .

Option 3 - Offline Installation

If you are still reading the installation chapter, it is probably because the options above did not work for you - proxy problems, your boss saw the internet bill… Here are 6 step for installing the RsCMPX_BluetoothMeas offline:

- Download this python script (**Save target as**): rsinstrument_offline_install.py This installs all the preconditions that the RsCMPX_BluetoothMeas needs.
- Execute the script in your offline computer (supported is python 3.6 or newer)
- Download the RsCMPX_BluetoothMeas package to your computer from the pypi.org: https://pypi.org/project/RsCMPX_BluetoothMeas/#files to for example `c:\temp\`
- Start the command line `WinKey + R`, type `cmd` and hit ENTER
- Change the working directory to the Python installation of your choice (adjust the user name and python version in the path):

```
cd c:\Users\John\AppData\Local\Programs\Python\Python37\Scripts
```

- Install with the command: `pip install c:\temp\RsCMPX_BluetoothMeas-5.0.70.3.tar`

2.3 Finding Available Instruments

Like the pyvisa's ResourceManager, the RsCMPX_BluetoothMeas can search for available instruments:

```
"""
Find the instruments in your environment
"""

from RsCMPX_BluetoothMeas import *

# Use the instr_list string items as resource names in the RsCMPX_BluetoothMeas␣
↪constructor
instr_list = RsCMPX_BluetoothMeas.list_resources("?*")
print(instr_list)
```

If you have more VISAs installed, the one actually used by default is defined by a secret widget called Visa Conflict Manager. You can force your program to use a VISA of your choice:

```
"""
Find the instruments in your environment with the defined VISA implementation
"""

from RsCMPX_BluetoothMeas import *

# In the optional parameter visa_select you can use for example 'rs' or 'ni'
# Rs Visa also finds any NRP-Zxx USB sensors
instr_list = RsCMPX_BluetoothMeas.list_resources('?*', 'rs')
print(instr_list)
```

Tip: We believe our R&S VISA is the best choice for our customers. Here are the reasons why:

- Small footprint
- Superior VXI-11 and HiSLIP performance

- Integrated legacy sensors NRP-Zxx support
- Additional VXI-11 and LXI devices search
- Availability for Windows, Linux, Mac OS

2.4 Initiating Instrument Session

RsCMPX_BluetoothMeas offers four different types of starting your remote-control session. We begin with the most typical case, and progress with more special ones.

Standard Session Initialization

Initiating new instrument session happens, when you instantiate the RsCMPX_BluetoothMeas object. Below, is a simple Hello World example. Different resource names are examples for different physical interfaces.

```
"""
Simple example on how to use the RsCMPX_BluetoothMeas module for remote-controlling your␣
↪instrument
Preconditions:

- Installed RsCMPX_BluetoothMeas Python module Version 5.0.70 or newer from pypi.org
- Installed VISA, for example R&S Visa 5.12 or newer
"""

from RsCMPX_BluetoothMeas import *

# A good practice is to assure that you have a certain minimum version installed
RsCMPX_BluetoothMeas.assert_minimum_version('5.0.70')
resource_string_1 = 'TCPIP::192.168.2.101::INSTR'  # Standard LAN connection (also␣
↪called VXI-11)
resource_string_2 = 'TCPIP::192.168.2.101::hislip0'  # Hi-Speed LAN connection - see␣
↪1MA208
resource_string_3 = 'GPIB::20::INSTR'  # GPIB Connection
resource_string_4 = 'USB::0x0AAD::0x0119::022019943::INSTR'  # USB-TMC (Test and␣
↪Measurement Class)

# Initializing the session
driver = RsCMPX_BluetoothMeas(resource_string_1)

idn = driver.utilities.query_str('*IDN?')
print(f"\nHello, I am: '{idn}'")
print(f'RsCMPX_BluetoothMeas package version: {driver.utilities.driver_version}')
print(f'Visa manufacturer: {driver.utilities.visa_manufacturer}')
print(f'Instrument full name: {driver.utilities.full_instrument_model_name}')
print(f'Instrument installed options: {",".join(driver.utilities.instrument_options)}')

# Close the session
driver.close()
```

Note: If you are wondering about the missing `ASRL1::INSTR`, yes, it works too, but come on… it's 2023.

Do not care about specialty of each session kind; RsCMPX_BluetoothMeas handles all the necessary session settings for you. You immediately have access to many identification properties in the interface `driver.utilities` . Here are same of them:

- `idn_string`
- `driver_version`
- `visa_manufacturer`
- `full_instrument_model_name`
- `instrument_serial_number`
- `instrument_firmware_version`
- `instrument_options`

The constructor also contains optional boolean arguments `id_query` and `reset`:

```
driver = RsCMPX_BluetoothMeas('TCPIP::192.168.56.101::hislip0', id_query=True,␣
↪reset=True)
```

- Setting `id_query` to True (default is True) checks, whether your instrument can be used with the RsCMPX_BluetoothMeas module.
- Setting `reset` to True (default is False) resets your instrument. It is equivalent to calling the `reset()` method.

Selecting a Specific VISA

Just like in the function `list_resources()`, the RsCMPX_BluetoothMeas allows you to choose which VISA to use:

```
"""
Choosing VISA implementation
"""

from RsCMPX_BluetoothMeas import *

# Force use of the Rs Visa. For NI Visa, use the "SelectVisa='ni'"
driver = RsCMPX_BluetoothMeas('TCPIP::192.168.56.101::INSTR', True, True, "SelectVisa='rs
↪'")

idn = driver.utilities.query_str('*IDN?')
print(f"\nHello, I am: '{idn}'")
print(f"\nI am using the VISA from: {driver.utilities.visa_manufacturer}")

# Close the session
driver.close()
```

No VISA Session

We recommend using VISA when possible preferrably with HiSlip session because of its low latency. However, if you are a strict VISA denier, RsCMPX_BluetoothMeas has something for you too - **no Visa installation raw LAN socket**:

```
"""
Using RsCMPX_BluetoothMeas without VISA for LAN Raw socket communication
"""

from RsCMPX_BluetoothMeas import *

driver = RsCMPX_BluetoothMeas('TCPIP::192.168.56.101::5025::SOCKET', True, True,
↪"SelectVisa='socket'")
print(f'Visa manufacturer: {driver.utilities.visa_manufacturer}')
print(f"\nHello, I am: '{driver.utilities.idn_string}'")

# Close the session
driver.close()
```

Warning: Not using VISA can cause problems by debugging when you want to use the communication Trace Tool. The good news is, you can easily switch to use VISA and back just by changing the constructor arguments. The rest of your code stays unchanged.

Simulating Session

If a colleague is currently occupying your instrument, leave him in peace, and open a simulating session:

```
driver = RsCMPX_BluetoothMeas('TCPIP::192.168.56.101::hislip0', True, True,
↪"Simulate=True")
```

More `option_string` tokens are separated by comma:

```
driver = RsCMPX_BluetoothMeas('TCPIP::192.168.56.101::hislip0', True, True, "SelectVisa=
↪'rs', Simulate=True")
```

Shared Session

In some scenarios, you want to have two independent objects talking to the same instrument. Rather than opening a second VISA connection, share the same one between two or more RsCMPX_BluetoothMeas objects:

```
"""
Sharing the same physical VISA session by two different RsCMPX_BluetoothMeas objects
"""

from RsCMPX_BluetoothMeas import *

driver1 = RsCMPX_BluetoothMeas('TCPIP::192.168.56.101::INSTR', True, True)
driver2 = RsCMPX_BluetoothMeas.from_existing_session(driver1)

print(f'driver1: {driver1.utilities.idn_string}')
```

(continues on next page)

(continued from previous page)

```
print(f'driver2: {driver2.utilities.idn_string}')

# Closing the driver2 session does not close the driver1 session - driver1 is the
↪'session master'
driver2.close()
print(f'driver2: I am closed now')

print(f'driver1: I am  still opened and working: {driver1.utilities.idn_string}')
driver1.close()
print(f'driver1: Only now I am closed.')
```

Note: The `driver1` is the object holding the 'master' session. If you call the `driver1.close()`, the `driver2` loses its instrument session as well, and becomes pretty much useless.

2.5 Plain SCPI Communication

After you have opened the session, you can use the instrument-specific part described in the RsCMPX_BluetoothMeas API Structure. If for any reason you want to use the plain SCPI, use the `utilities` interface's two basic methods:

- `write_str()` - writing a command without an answer, for example ***RST**
- `query_str()` - querying your instrument, for example the ***IDN?** query

You may ask a question. Actually, two questions:

- **Q1**: Why there are not called `write()` and `query()` ?
- **Q2**: Where is the `read()` ?

Answer 1: Actually, there are - the `write_str()` / `write()` and `query_str()` / `query()` are aliases, and you can use any of them. We promote the `_str` names, to clearly show you want to work with strings. Strings in Python3 are Unicode, the *bytes* and *string* objects are not interchangeable, since one character might be represented by more than 1 byte. To avoid mixing string and binary communication, all the method names for binary transfer contain `_bin` in the name.

Answer 2: Short answer - you do not need it. Long answer - your instrument never sends unsolicited responses. If you send a set command, you use `write_str()`. For a query command, you use `query_str()`. So, you really do not need it...

Bottom line - if you are used to `write()` and `query()` methods, from pyvisa, the `write_str()` and `query_str()` are their equivalents.

Enough with the theory, let us look at an example. Simple write, and query:

```
"""
Basic string write_str / query_str
"""

from RsCMPX_BluetoothMeas import *

driver = RsCMPX_BluetoothMeas('TCPIP::192.168.56.101::INSTR')
driver.utilities.write_str('*RST')
response = driver.utilities.query_str('*IDN?')
```

(continues on next page)

(continued from previous page)

```
print(response)

# Close the session
driver.close()
```

This example is so-called "*University-Professor-Example*" - good to show a principle, but never used in praxis. The abovementioned commands are already a part of the driver's API. Here is another example, achieving the same goal:

```
"""
Basic string write_str / query_str
"""

from RsCMPX_BluetoothMeas import *

driver = RsCMPX_BluetoothMeas('TCPIP::192.168.56.101::INSTR')
driver.utilities.reset()
print(driver.utilities.idn_string)

# Close the session
driver.close()
```

One additional feature we need to mention here: **VISA timeout**. To simplify, VISA timeout plays a role in each `query_xxx()`, where the controller (your PC) has to prevent waiting forever for an answer from your instrument. VISA timeout defines that maximum waiting time. You can set/read it with the `visa_timeout` property:

```
# Timeout in milliseconds
driver.utilities.visa_timeout = 3000
```

After this time, the RsCMPX_BluetoothMeas raises an exception. Speaking of exceptions, an important feature of the RsCMPX_BluetoothMeas is **Instrument Status Checking**. Check out the next chapter that describes the error checking in details.

For completion, we mention other string-based `write_xxx()` and `query_xxx()` methods - all in one example. They are convenient extensions providing type-safe float/boolean/integer setting/querying features:

```
"""
Basic string write_xxx / query_xxx
"""

from RsCMPX_BluetoothMeas import *

driver = RsCMPX_BluetoothMeas('TCPIP::192.168.56.101::INSTR')
driver.utilities.visa_timeout = 5000
driver.utilities.instrument_status_checking = True
driver.utilities.write_int('SWEEP:COUNT ', 10)  # sending 'SWEEP:COUNT 10'
driver.utilities.write_bool('SOURCE:RF:OUTPUT:STATE ', True)  # sending
↪'SOURCE:RF:OUTPUT:STATE ON'
driver.utilities.write_float('SOURCE:RF:FREQUENCY ', 1E9)  # sending 'SOURCE:RF:FREQUENCY␣
↪1000000000'

sc = driver.utilities.query_int('SWEEP:COUNT?')  # returning integer number sc=10
out = driver.utilities.query_bool('SOURCE:RF:OUTPUT:STATE?')  # returning boolean␣
↪out=True
```

(continues on next page)

(continued from previous page)

```
freq = driver.utilities.query_float('SOURCE:RF:FREQUENCY?')  # returning float number␣
↪freq=1E9

# Close the session
driver.close()
```

Lastly, a method providing basic synchronization: `query_opc()`. It sends query ***OPC?** to your instrument. The instrument waits with the answer until all the tasks it currently has in a queue are finished. This way your program waits too, and this way it is synchronized with the actions in the instrument. Remember to have the VISA timeout set to an appropriate value to prevent the timeout exception. Here's the snippet:

```
driver.utilities.visa_timeout = 3000
driver.utilities.write_str("INIT")
driver.utilities.query_opc()

# The results are ready now to fetch
results = driver.utilities.query_str("FETCH:MEASUREMENT?")
```

Tip: Wait, there's more: you can send the ***OPC?** after each `write_xxx()` automatically:

```
# Default value after init is False
driver.utilities.opc_query_after_write = True
```

2.6 Error Checking

RsCMPX_BluetoothMeas pushes limits even further (internal R&S joke): It has a built-in mechanism that after each command/query checks the instrument's status subsystem, and raises an exception if it detects an error. For those who are already screaming: **Speed Performance Penalty!!!**, don't worry, you can disable it.

Instrument status checking is very useful since in case your command/query caused an error, you are immediately informed about it. Status checking has in most cases no practical effect on the speed performance of your program. However, if for example, you do many repetitions of short write/query sequences, it might make a difference to switch it off:

```
# Default value after init is True
driver.utilities.instrument_status_checking = False
```

To clear the instrument status subsystem of all errors, call this method:

```
driver.utilities.clear_status()
```

Instrument's status system error queue is clear-on-read. It means, if you query its content, you clear it at the same time. To query and clear list of all the current errors, use this snippet:

```
errors_list = driver.utilities.query_all_errors()
```

See the next chapter on how to react on errors.

2.7 Exception Handling

The base class for all the exceptions raised by the RsCMPX_BluetoothMeas is `RsInstrException`. Inherited exception classes:

- `ResourceError` raised in the constructor by problems with initiating the instrument, for example wrong or non-existing resource name
- `StatusException` raised if a command or a query generated error in the instrument's error queue
- `TimeoutException` raised if a visa timeout or an opc timeout is reached

In this example we show usage of all of them. Because it is difficult to generate an error using the instrument-specific SCPI API, we use plain SCPI commands:

```
"""
Showing how to deal with exceptions
"""

from RsCMPX_BluetoothMeas import *

driver = None
# Try-catch for initialization. If an error occures, the ResourceError is raised
try:
    driver = RsCMPX_BluetoothMeas('TCPIP::10.112.1.179::hislip0')
except ResourceError as e:
    print(e.args[0])
    print('Your instrument is probably OFF...')
    # Exit now, no point of continuing
    exit(1)

# Dealing with commands that potentially generate errors OPTION 1:
# Switching the status checking OFF termporarily
driver.utilities.instrument_status_checking = False
driver.utilities.write_str('MY:MISSpelled:COMMand')
# Clear the error queue
driver.utilities.clear_status()
# Status checking ON again
driver.utilities.instrument_status_checking = True

# Dealing with queries that potentially generate errors OPTION 2:
try:
    # You migh want to reduce the VISA timeout to avoid long waiting
    driver.utilities.visa_timeout = 1000
    driver.utilities.query_str('MY:WRONg:QUERy?')

except StatusException as e:
    # Instrument status error
    print(e.args[0])
    print('Nothing to see here, moving on...')

except TimeoutException as e:
    # Timeout error
    print(e.args[0])
    print('That took a long time...')
```

(continues on next page)

(continued from previous page)

```
except RsInstrException as e:
    # RsInstrException is a base class for all the RsCMPX_BluetoothMeas exceptions
    print(e.args[0])
    print('Some other RsCMPX_BluetoothMeas error...')

finally:
    driver.utilities.visa_timeout = 5000
    # Close the session in any case
    driver.close()
```

Tip: General rules for exception handling:

- If you are sending commands that might generate errors in the instrument, for example deleting a file which does not exist, use the **OPTION 1** - temporarily disable status checking, send the command, clear the error queue and enable the status checking again.
- If you are sending queries that might generate errors or timeouts, for example querying measurement that can not be performed at the moment, use the **OPTION 2** - try/except with optionally adjusting the timeouts.

2.8 Transferring Files

Instrument -> PC

You definitely experienced it: you just did a perfect measurement, saved the results as a screenshot to an instrument's storage drive. Now you want to transfer it to your PC. With RsCMPX_BluetoothMeas, no problem, just figure out where the screenshot was stored on the instrument. In our case, it is */var/user/instr_screenshot.png*:

```
driver.utilities.read_file_from_instrument_to_pc(
    r'/var/user/instr_screenshot.png',
    r'c:\temp\pc_screenshot.png')
```

PC -> Instrument

Another common scenario: Your cool test program contains a setup file you want to transfer to your instrument: Here is the RsCMPX_BluetoothMeas one-liner split into 3 lines:

```
driver.utilities.send_file_from_pc_to_instrument(
    r'c:\MyCoolTestProgram\instr_setup.sav',
    r'/var/appdata/instr_setup.sav')
```

2.9 Writing Binary Data

Writing from bytes

An example where you need to send binary data is a waveform file of a vector signal generator. First, you compose your wform_data as `bytes`, and then you send it with `write_bin_block()`:

```
# MyWaveform.wv is an instrument file name under which this data is stored
driver.utilities.write_bin_block(
    "SOUR:BB:ARB:WAV:DATA 'MyWaveform.wv',",
    wform_data)
```

Note: Notice the `write_bin_block()` has two parameters:

- `string` parameter `cmd` for the SCPI command
- `bytes` parameter `payload` for the actual binary data to send

Writing from PC files

Similar to querying binary data to a file, you can write binary data from a file. The second parameter is then the PC file path the content of which you want to send:

```
driver.utilities.write_bin_block_from_file(
    "SOUR:BB:ARB:WAV:DATA 'MyWaveform.wv',",
    r"c:\temp\wform_data.wv")
```

2.10 Transferring Big Data with Progress

We can agree that it can be annoying using an application that shows no progress for long-lasting operations. The same is true for remote-control programs. Luckily, the RsCMPX_BluetoothMeas has this covered. And, this feature is quite universal - not just for big files transfer, but for any data in both directions.

RsCMPX_BluetoothMeas allows you to register a function (programmers fancy name is `callback`), which is then periodicaly invoked after transfer of one data chunk. You can define that chunk size, which gives you control over the callback invoke frequency. You can even slow down the transfer speed, if you want to process the data as they arrive (direction instrument -> PC).

To show this in praxis, we are going to use another *University-Professor-Example*: querying the ***IDN?** with chunk size of 2 bytes and delay of 200ms between each chunk read:

```
"""
Event handlers by reading
"""

from RsCMPX_BluetoothMeas import *
import time


def my_transfer_handler(args):
```

(continues on next page)

(continued from previous page)

```
    """Function called each time a chunk of data is transferred"""
    # Total size is not always known at the beginning of the transfer
    total_size = args.total_size if args.total_size is not None else "unknown"

    print(f"Context: '{args.context}{'with opc' if args.opc_sync else ''}', "
        f"chunk {args.chunk_ix}, "
        f"transferred {args.transferred_size} bytes, "
        f"total size {total_size}, "
        f"direction {'reading' if args.reading else 'writing'}, "
        f"data '{args.data}'")

    if args.end_of_transfer:
        print('End of Transfer')
    time.sleep(0.2)


driver = RsCMPX_BluetoothMeas('TCPIP::192.168.56.101::INSTR')

driver.events.on_read_handler = my_transfer_handler
# Switch on the data to be included in the event arguments
# The event arguments args.data will be updated
driver.events.io_events_include_data = True
# Set data chunk size to 2 bytes
driver.utilities.data_chunk_size = 2
driver.utilities.query_str('*IDN?')
# Unregister the event handler
driver.utilities.on_read_handler = None

# Close the session
driver.close()
```

If you start it, you might wonder (or maybe not): why is the `args.total_size = None`? The reason is, in this particular case the RsCMPX_BluetoothMeas does not know the size of the complete response up-front. However, if you use the same mechanism for transfer of a known data size (for example, file transfer), you get the information about the total size too, and hence you can calculate the progress as:

*progress [pct] = 100 * args.transferred_size / args.total_size*

Snippet of transferring file from PC to instrument, the rest of the code is the same as in the previous example:

```
driver.events.on_write_handler = my_transfer_handler
driver.events.io_events_include_data = True
driver.data_chunk_size = 1000
driver.utilities.send_file_from_pc_to_instrument(
    r'c:\MyCoolTestProgram\my_big_file.bin',
    r'/var/user/my_big_file.bin')
# Unregister the event handler
driver.events.on_write_handler = None
```

2.11 Multithreading

You are at the party, many people talking over each other. Not every person can deal with such crosstalk, neither can measurement instruments. For this reason, RsCMPX_BluetoothMeas has a feature of scheduling the access to your instrument by using so-called **Locks**. Locks make sure that there can be just one client at a time *talking* to your instrument. Talking in this context means completing one communication step - one command write or write/read or write/read/error check.

To describe how it works, and where it matters, we take three typical mulithread scenarios:

One instrument session, accessed from multiple threads

You are all set - the lock is a part of your instrument session. Check out the following example - it will execute properly, although the instrument gets 10 queries at the same time:

```
"""
Multiple threads are accessing one RsCMPX_BluetoothMeas object
"""

import threading
from RsCMPX_BluetoothMeas import *


def execute(session):
    """Executed in a separate thread."""
    session.utilities.query_str('*IDN?')


driver = RsCMPX_BluetoothMeas('TCPIP::192.168.56.101::INSTR')
threads = []
for i in range(10):
    t = threading.Thread(target=execute, args=(driver, ))
    t.start()
    threads.append(t)
print('All threads started')

# Wait for all threads to join this main thread
for t in threads:
    t.join()
print('All threads ended')

driver.close()
```

Shared instrument session, accessed from multiple threads

Same as the previous case, you are all set. The session carries the lock with it. You have two objects, talking to the same instrument from multiple threads. Since the instrument session is shared, the same lock applies to both objects causing the exclusive access to the instrument.

Try the following example:

```
"""
Multiple threads are accessing two RsCMPX_BluetoothMeas objects with shared session
"""

import threading
from RsCMPX_BluetoothMeas import *


def execute(session: RsCMPX_BluetoothMeas, session_ix, index) -> None:
    """Executed in a separate thread."""
    print(f'{index} session {session_ix} query start...')
    session.utilities.query_str('*IDN?')
    print(f'{index} session {session_ix} query end')


driver1 = RsCMPX_BluetoothMeas('TCPIP::192.168.56.101::INSTR')
driver2 = RsCMPX_BluetoothMeas.from_existing_session(driver1)
driver1.utilities.visa_timeout = 200
driver2.utilities.visa_timeout = 200
# To see the effect of crosstalk, uncomment this line
# driver2.utilities.clear_lock()

threads = []
for i in range(10):
    t = threading.Thread(target=execute, args=(driver1, 1, i,))
    t.start()
    threads.append(t)
    t = threading.Thread(target=execute, args=(driver2, 2, i,))
    t.start()
    threads.append(t)
print('All threads started')

# Wait for all threads to join this main thread
for t in threads:
    t.join()
print('All threads ended')

driver2.close()
driver1.close()
```

As you see, everything works fine. If you want to simulate some party crosstalk, uncomment the line `driver2.utilities.clear_lock()`. Thich causes the driver2 session lock to break away from the driver1 session lock. Although the driver1 still tries to schedule its instrument access, the driver2 tries to do the same at the same time, which leads to all the fun stuff happening.

Multiple instrument sessions accessed from multiple threads

Here, there are two possible scenarios depending on the instrument's VISA interface:

- Your are lucky, because you instrument handles each remote session completely separately. An example of such instrument is SMW200A. In this case, you have no need for session locking.
- Your instrument handles all sessions with one set of in/out buffers. You need to lock the session for the duration of a talk. And you are lucky again, because the RsCMPX_BluetoothMeas takes care of it for you. The text below describes this scenario.

Run the following example:

```
"""
Multiple threads are accessing two RsCMPX_BluetoothMeas objects with two separate␣
↪sessions
"""

import threading
from RsCMPX_BluetoothMeas import *


def execute(session: RsCMPX_BluetoothMeas, session_ix, index) -> None:
    """Executed in a separate thread."""
    print(f'{index} session {session_ix} query start...')
    session.utilities.query_str('*IDN?')
    print(f'{index} session {session_ix} query end')


driver1 = RsCMPX_BluetoothMeas('TCPIP::192.168.56.101::INSTR')
driver2 = RsCMPX_BluetoothMeas('TCPIP::192.168.56.101::INSTR')
driver1.utilities.visa_timeout = 200
driver2.utilities.visa_timeout = 200

# Synchronise the sessions by sharing the same lock
driver2.utilities.assign_lock(driver1.utilities.get_lock())  # To see the effect of␣
↪crosstalk, comment this line

threads = []
for i in range(10):
    t = threading.Thread(target=execute, args=(driver1, 1, i,))
    t.start()
    threads.append(t)
    t = threading.Thread(target=execute, args=(driver2, 2, i,))
    t.start()
    threads.append(t)
print('All threads started')

# Wait for all threads to join this main thread
for t in threads:
    t.join()
print('All threads ended')

driver2.close()
driver1.close()
```

You have two completely independent sessions that want to talk to the same instrument at the same time. This will not go well, unless they share the same session lock. The key command to achieve this is `driver2.utilities.assign_lock(driver1.utilities.get_lock())` Try to comment it and see how it goes. If despite commenting the line the example runs without issues, you are lucky to have an instrument similar to the SMW200A.

2.12 Logging

Yes, the logging again. This one is tailored for instrument communication. You will appreciate such handy feature when you troubleshoot your program, or just want to protocol the SCPI communication for your test reports.

What can you actually do with the logger?

- Write SCPI communication to a stream-like object, for example console or file, or both simultaneously
- Log only errors and skip problem-free parts; this way you avoid going through thousands lines of texts
- Investigate duration of certain operations to optimize your program's performance
- Log custom messages from your program

Let us take this basic example:

```
"""
Basic logging example to the console
"""

from RsCMPX_BluetoothMeas import *

driver = RsCMPX_BluetoothMeas('TCPIP::192.168.1.101::INSTR')

# Switch ON logging to the console.
driver.utilities.logger.log_to_console = True
driver.utilities.logger.mode = LoggingMode.On
driver.utilities.reset()

# Close the session
driver.close()
```

Console output:

```
10:29:10.819     TCPIP::192.168.1.101::INSTR     0.976 ms  Write: *RST
10:29:10.819     TCPIP::192.168.1.101::INSTR  1884.985 ms  Status check: OK
10:29:12.704     TCPIP::192.168.1.101::INSTR     0.983 ms  Query OPC: 1
10:29:12.705     TCPIP::192.168.1.101::INSTR     2.892 ms  Clear status: OK
10:29:12.708     TCPIP::192.168.1.101::INSTR     3.905 ms  Status check: OK
10:29:12.712     TCPIP::192.168.1.101::INSTR     1.952 ms  Close: Closing session
```

The columns of the log are aligned for better reading. Columns meaning:

- (1) Start time of the operation
- (2) Device resource name (you can set an alias)
- (3) Duration of the operation
- (4) Log entry

Tip: You can customize the logging format with `set_format_string()`, and set the maximum log entry length with the properties:

- `abbreviated_max_len_ascii`
- `abbreviated_max_len_bin`
- `abbreviated_max_len_list`

See the full logger help *here*.

Notice the SCPI communication starts from the line `driver.utilities.reset()`. If you want to log the initialization of the session as well, you have to switch the logging ON already in the constructor:

```
driver = RsCMPX_BluetoothMeas('TCPIP::192.168.56.101::hislip0', options='LoggingMode=On')
```

Parallel to the console logging, you can log to a general stream. Do not fear the programmer's jargon'... under the term **stream** you can just imagine a file. To be a little more technical, a stream in Python is any object that has two methods: `write()` and `flush()`. This example opens a file and sets it as logging target:

```
"""
Example of logging to a file
"""

from RsCMPX_BluetoothMeas import *

driver = RsCMPX_BluetoothMeas('TCPIP::192.168.1.101::INSTR')

# We also want to log to the console.
driver.utilities.logger.log_to_console = True

# Logging target is our file
file = open(r'c:\temp\my_file.txt', 'w')
driver.utilities.logger.set_logging_target(file)
driver.utilities.logger.mode = LoggingMode.On

# Instead of the 'TCPIP::192.168.1.101::INSTR', show 'MyDevice'
driver.utilities.logger.device_name = 'MyDevice'

# Custom user entry
driver.utilities.logger.info_raw('----- This is my custom log entry. ---- ')

driver.utilities.reset()

# Close the session
driver.close()

# Close the log file
file.close()
```

Tip: To make the log more compact, you can skip all the lines with `Status check: OK`:

```
driver.utilities.logger.log_status_check_ok = False
```

Hint: You can share the logging file between multiple sessions. In such case, remember to close the file only after you have stopped logging in all your sessions, otherwise you get a log write error.

For logging to a UDP port in addition to other log targets, use one of the lines:

```
driver.utilities.logger.log_to_udp = True
driver.utilities.logger.log_to_console_and_udp = True
```

You can select the UDP port to log to, the default is 49200:

```
driver.utilities.logger.udp_port = 49200
```

Another cool feature is logging only errors. To make this mode usefull for troubleshooting, you also want to see the circumstances which lead to the errors. Each driver elementary operation, for example, `write_str()`, can generate a group of log entries - let us call them **Segment**. In the logging mode `Errors`, a whole segment is logged only if at least one entry of the segment is an error.

The script below demonstrates this feature. We use a direct SCPI communication to send a misspelled SCPI command *CLS, which leads to instrument status error:

```
"""
Logging example to the console with only errors logged
"""

from RsCMPX_BluetoothMeas import *

driver = RsCMPX_BluetoothMeas('TCPIP::192.168.1.101::INSTR', options='LoggingMode=Errors
↪')

# Switch ON logging to the console.
driver.utilities.logger.log_to_console = True

# Reset will not be logged, since no error occurred there
driver.utilities.reset()

# Now a misspelled command.
driver.utilities.write('*CLaS')

# A good command again, no logging here
idn = driver.utilities.query('*IDN?')

# Close the session
driver.close()
```

Console output:

```
12:11:02.879 TCPIP::192.168.1.101::INSTR     0.976 ms  Write string: *CLaS
12:11:02.879 TCPIP::192.168.1.101::INSTR     6.833 ms  Status check: StatusException:
                                                       Instrument error detected: Undefined header;
↪*CLaS
```

Notice the following:

- Although the operation **Write string: *CLaS** finished without an error, it is still logged, because it provides the context for the actual error which occurred during the status checking right after.
- No other log entries are present, including the session initialization and close, because they were all error-free.

CHAPTER

THREE

ENUMS

3.1 AddressType

```
# Example value:
value = enums.AddressType.PUBLic
# All values (2x):
PUBLic | RANDom
```

3.2 AutoManualMode

```
# Example value:
value = enums.AutoManualMode.AUTO
# All values (2x):
AUTO | MANual
```

3.3 BaudRate

```
# First value:
value = enums.BaudRate.B110
# Last value:
value = enums.BaudRate.B96K
# All values (24x):
B110 | B115k | B12K | B14K | B19K | B1M | B1M5 | B234k
B24K | B28K | B2M | B300 | B38K | B3M | B3M5 | B460k
B48K | B4M | B500k | B576k | B57K | B600 | B921k | B96K
```

3.4 BrEdrChannelsRange

```
# Example value:
value = enums.BrEdrChannelsRange.CH21
# All values (2x):
CH21 | CH79
```

3.5 BrPacketType

```
# Example value:
value = enums.BrPacketType.DH1
# All values (3x):
DH1 | DH3 | DH5
```

3.6 BurstType

```
# Example value:
value = enums.BurstType.BR
# All values (4x):
BR | EDR | LE | QHSL
```

3.7 CmwSingleConnector

```
# First value:
value = enums.CmwSingleConnector.R11
# Last value:
value = enums.CmwSingleConnector.RB8
# All values (48x):
R11 | R12 | R13 | R14 | R15 | R16 | R17 | R18
R21 | R22 | R23 | R24 | R25 | R26 | R27 | R28
R31 | R32 | R33 | R34 | R35 | R36 | R37 | R38
R41 | R42 | R43 | R44 | R45 | R46 | R47 | R48
RA1 | RA2 | RA3 | RA4 | RA5 | RA6 | RA7 | RA8
RB1 | RB2 | RB3 | RB4 | RB5 | RB6 | RB7 | RB8
```

3.8 CodingScheme

```
# Example value:
value = enums.CodingScheme.S2
# All values (2x):
S2 | S8
```

3.9 CommProtocol

```
# Example value:
value = enums.CommProtocol.HCI
# All values (2x):
HCI | TWO
```

3.10 CtePacketType

```
# Example value:
value = enums.CtePacketType.AOA1us
# All values (5x):
AOA1us | AOA2us | AOAus | AOD1us | AOD2us
```

3.11 CteType

```
# Example value:
value = enums.CteType.AOA
# All values (3x):
AOA | AOD1 | AOD2
```

3.12 DataBits

```
# Example value:
value = enums.DataBits.D7
# All values (2x):
D7 | D8
```

3.13 DetectedPatternType

```
# Example value:
value = enums.DetectedPatternType.ALTernating
# All values (4x):
ALTernating | OTHer | P11 | P44
```

3.14 DetectedPhyType

```
# Example value:
value = enums.DetectedPhyType.P2Q
# All values (5x):
P2Q | P3Q | P4Q | P5Q | P6Q
```

3.15 DisplayMeasurement

```
# Example value:
value = enums.DisplayMeasurement.MEV
# All values (1x):
MEV
```

3.16 DisplayView

```
# First value:
value = enums.DisplayView.DEVM
# Last value:
value = enums.DisplayView.SOBW
# All values (15x):
DEVM | FDEViation | FRANge | IQABs | IQDiff | IQERr | MODulation | OVERview
PDIFference | PENCoding | POWer | PVTime | SACP | SGACp | SOBW
```

3.17 EdrPacketType

```
# Example value:
value = enums.EdrPacketType.E21P
# All values (6x):
E21P | E23P | E25P | E31P | E33P | E35P
```

3.18 FilterWidth

```
# Example value:
value = enums.FilterWidth.NARRow
# All values (2x):
NARRow | WIDE
```

3.19 HwInterface

```
# Example value:
value = enums.HwInterface.NONE
# All values (3x):
NONE | RS232 | USB
```

3.20 LeChannelsRange

```
# Example value:
value = enums.LeChannelsRange.CH10
# All values (2x):
CH10 | CH40
```

3.21 LePacketType

```
# Example value:
value = enums.LePacketType.ADVertiser
# All values (3x):
ADVertiser | RFCTe | RFPHytest
```

3.22 LePatternType

```
# Example value:
value = enums.LePatternType.OTHer
# All values (3x):
OTHer | P11 | P44
```

3.23 LePhysicalType

```
# Example value:
value = enums.LePhysicalType.LE1M
# All values (3x):
LE1M | LE2M | LELR
```

3.24 LePhyType

```
# Example value:
value = enums.LePhyType.LE1M
# All values (8x):
LE1M | LE2M | LELR | P2Q | P3Q | P4Q | P5Q | P6Q
```

3.25 LeRangePaternType

```
# Example value:
value = enums.LeRangePaternType.ALL0
# All values (6x):
ALL0 | ALL1 | OTHer | P11 | P44 | PRBS9
```

3.26 LeSymolTimeError

```
# Example value:
value = enums.LeSymolTimeError.NEG50
# All values (3x):
NEG50 | OFF | POS50
```

3.27 LogCategory

```
# Example value:
value = enums.LogCategory.CONTinue
# All values (4x):
CONTinue | ERRor | INFO | WARNing
```

3.28 MeasureScope

```
# Example value:
value = enums.MeasureScope.ALL
# All values (2x):
ALL | SINGle
```

3.29 MevPatternType

```
# Example value:
value = enums.MevPatternType.ALL1
# All values (5x):
ALL1 | ALTernating | OTHer | P11 | P44
```

3.30 PacketTypeA

```
# First value:
value = enums.PacketTypeA.E21P
# Last value:
value = enums.PacketTypeA.E85P
# All values (9x):
E21P | E23P | E25P | E41P | E43P | E45P | E81P | E83P
E85P
```

3.31 PacketTypeB

```
# Example value:
value = enums.PacketTypeB.DATA
# All values (1x):
DATA
```

3.32 PacketTypeC

```
# Example value:
value = enums.PacketTypeC.H41P
# All values (6x):
H41P | H43P | H45P | H81P | H83P | H85P
```

3.33 ParameterSetMode

```
# Example value:
value = enums.ParameterSetMode.GLOBal
# All values (2x):
GLOBal | LIST
```

3.34 Parity

```
# Example value:
value = enums.Parity.EVEN
# All values (3x):
EVEN | NONE | ODD
```

3.35 PatternIndependent

```
# Example value:
value = enums.PatternIndependent.PINDependent
# All values (2x):
PINDependent | SPECconform
```

3.36 PatternType

```
# Example value:
value = enums.PatternType.ALL0
# All values (5x):
ALL0 | ALL1 | P11 | P44 | PRBS9
```

3.37 PatternTypeHdr

```
# Example value:
value = enums.PatternTypeHdr.OTHer
# All values (2x):
OTHer | PRBS9
```

3.38 PayloadCoding

```
# Example value:
value = enums.PayloadCoding.L12D
# All values (3x):
L12D | L34D | NONE
```

3.39 PayloadLength

```
# Example value:
value = enums.PayloadLength._255
# All values (2x):
_255 | _37
```

3.40 PduType

```
# Example value:
value = enums.PduType.ADVDirect
# All values (7x):
ADVDirect | ADVind | ADVNonconn | ADVScan | CONReq | SCReq | SCRSp
```

3.41 PhysicalLayer

```
# Example value:
value = enums.PhysicalLayer.P4HP
# All values (2x):
P4HP | P8HP
```

3.42 Protocol

```
# Example value:
value = enums.Protocol.CTSRts
# All values (3x):
CTSRts | NONE | XONXoff
```

3.43 Repeat

```
# Example value:
value = enums.Repeat.CONTinuous
# All values (2x):
CONTinuous | SINGleshot
```

3.44 ResourceState

```
# Example value:
value = enums.ResourceState.ACTive
# All values (8x):
ACTive | ADJusted | INValid | OFF | PENDing | QUEued | RDY | RUN
```

3.45 Result

```
# Example value:
value = enums.Result.FAIL
# All values (2x):
FAIL | PASS
```

3.46 ResultStatus2

```
# First value:
value = enums.ResultStatus2.DC
# Last value:
value = enums.ResultStatus2.ULEU
# All values (10x):
DC | INV | NAV | NCAP | OFF | OFL | OK | UFL
ULEL | ULEU
```

3.47 RxConnector

```
# First value:
value = enums.RxConnector.I11I
# Last value:
value = enums.RxConnector.RH8
# All values (163x):
I11I | I13I | I15I | I17I | I21I | I23I | I25I | I27I
I31I | I33I | I35I | I37I | I41I | I43I | I45I | I47I
IFI1 | IFI2 | IFI3 | IFI4 | IFI5 | IFI6 | IQ1I | IQ3I
IQ5I | IQ7I | R10D | R11 | R11C | R11D | R12 | R12C
R12D | R12I | R13 | R13C | R14 | R14C | R14I | R15
R16 | R17 | R18 | R21 | R21C | R22 | R22C | R22I
R23 | R23C | R24 | R24C | R24I | R25 | R26 | R27
R28 | R31 | R31C | R32 | R32C | R32I | R33 | R33C
R34 | R34C | R34I | R35 | R36 | R37 | R38 | R41
R41C | R42 | R42C | R42I | R43 | R43C | R44 | R44C
R44I | R45 | R46 | R47 | R48 | RA1 | RA2 | RA3
RA4 | RA5 | RA6 | RA7 | RA8 | RB1 | RB2 | RB3
RB4 | RB5 | RB6 | RB7 | RB8 | RC1 | RC2 | RC3
RC4 | RC5 | RC6 | RC7 | RC8 | RD1 | RD2 | RD3
```

(continues on next page)

(continued from previous page)

```
RD4 | RD5 | RD6 | RD7 | RD8 | RE1 | RE2 | RE3
RE4 | RE5 | RE6 | RE7 | RE8 | RF1 | RF1C | RF2
RF2C | RF2I | RF3 | RF3C | RF4 | RF4C | RF4I | RF5
RF5C | RF6 | RF6C | RF7 | RF7C | RF8 | RF8C | RF9C
RFAC | RFBC | RFBI | RG1 | RG2 | RG3 | RG4 | RG5
RG6 | RG7 | RG8 | RH1 | RH2 | RH3 | RH4 | RH5
RH6 | RH7 | RH8
```

3.48 RxConverter

```
# First value:
value = enums.RxConverter.IRX1
# Last value:
value = enums.RxConverter.RX44
# All values (40x):
IRX1 | IRX11 | IRX12 | IRX13 | IRX14 | IRX2 | IRX21 | IRX22
IRX23 | IRX24 | IRX3 | IRX31 | IRX32 | IRX33 | IRX34 | IRX4
IRX41 | IRX42 | IRX43 | IRX44 | RX1 | RX11 | RX12 | RX13
RX14 | RX2 | RX21 | RX22 | RX23 | RX24 | RX3 | RX31
RX32 | RX33 | RX34 | RX4 | RX41 | RX42 | RX43 | RX44
```

3.49 RxQualityMeasMode

```
# Example value:
value = enums.RxQualityMeasMode.PER
# All values (3x):
PER | SENS | SPOT
```

3.50 SegmentPacketType

```
# First value:
value = enums.SegmentPacketType.ADVertiser
# Last value:
value = enums.SegmentPacketType.RFPHytest
# All values (13x):
ADVertiser | DATA | DH1 | DH3 | DH5 | E21P | E23P | E25P
E31P | E33P | E35P | RFCTe | RFPHytest
```

3.51 StopBits

```
# Example value:
value = enums.StopBits.S1
# All values (2x):
S1 | S2
```

3.52 StopCondition

```
# Example value:
value = enums.StopCondition.NONE
# All values (2x):
NONE | SLFail
```

3.53 SyncState

```
# Example value:
value = enums.SyncState.ADJusted
# All values (2x):
ADJusted | PENDing
```

3.54 TargetMainState

```
# Example value:
value = enums.TargetMainState.OFF
# All values (3x):
OFF | RDY | RUN
```

3.55 TestScenario

```
# Example value:
value = enums.TestScenario.CSPath
# All values (3x):
CSPath | SALone | UNDefined
```

3.56 TransmitPatternType

```
# Example value:
value = enums.TransmitPatternType.ALL1
# All values (2x):
ALL1 | OTHer
```

3.57 TxConnector

```
# First value:
value = enums.TxConnector.I120
# Last value:
value = enums.TxConnector.RH18
# All values (86x):
I120 | I140 | I160 | I180 | I220 | I240 | I260 | I280
I320 | I340 | I360 | I380 | I420 | I440 | I460 | I480
IFO1 | IFO2 | IFO3 | IFO4 | IFO5 | IFO6 | IQ20 | IQ40
IQ60 | IQ80 | R10D | R118 | R1183 | R1184 | R11C | R11D
R110 | R1103 | R1104 | R12C | R12D | R13C | R130 | R14C
R214 | R218 | R21C | R210 | R22C | R23C | R230 | R24C
R258 | R318 | R31C | R310 | R32C | R33C | R330 | R34C
R418 | R41C | R410 | R42C | R43C | R430 | R44C | RA18
RB14 | RB18 | RC18 | RD18 | RE18 | RF18 | RF1C | RF10
RF2C | RF3C | RF30 | RF4C | RF5C | RF6C | RF7C | RF8C
RF9C | RFAC | RFA0 | RFBC | RG18 | RH18
```

3.58 TxConnectorBench

```
# First value:
value = enums.TxConnectorBench.R118
# Last value:
value = enums.TxConnectorBench.RH18
# All values (15x):
R118 | R214 | R218 | R258 | R318 | R418 | RA18 | RB14
RB18 | RC18 | RD18 | RE18 | RF18 | RG18 | RH18
```

3.59 TxConverter

```
# First value:
value = enums.TxConverter.ITX1
# Last value:
value = enums.TxConverter.TX44
# All values (40x):
ITX1 | ITX11 | ITX12 | ITX13 | ITX14 | ITX2 | ITX21 | ITX22
ITX23 | ITX24 | ITX3 | ITX31 | ITX32 | ITX33 | ITX34 | ITX4
```

(continues on next page)

(continued from previous page)

```
ITX41 | ITX42 | ITX43 | ITX44 | TX1 | TX11 | TX12 | TX13
TX14 | TX2 | TX21 | TX22 | TX23 | TX24 | TX3 | TX31
TX32 | TX33 | TX34 | TX4 | TX41 | TX42 | TX43 | TX44
```

CHAPTER

FOUR

REPCAPS

4.1 Instance (Global)

```
# Setting:
driver.repcap_instance_set(repcap.Instance.Inst1)
# Range:
Inst1 .. Inst16
# All values (16x):
Inst1 | Inst2 | Inst3 | Inst4 | Inst5 | Inst6 | Inst7 | Inst8
Inst9 | Inst10 | Inst11 | Inst12 | Inst13 | Inst14 | Inst15 | Inst16
```

4.2 Segment

```
# First value:
value = repcap.Segment.S1
# Range:
S1 .. S128
# All values (128x):
S1 | S2 | S3 | S4 | S5 | S6 | S7 | S8
S9 | S10 | S11 | S12 | S13 | S14 | S15 | S16
S17 | S18 | S19 | S20 | S21 | S22 | S23 | S24
S25 | S26 | S27 | S28 | S29 | S30 | S31 | S32
S33 | S34 | S35 | S36 | S37 | S38 | S39 | S40
S41 | S42 | S43 | S44 | S45 | S46 | S47 | S48
S49 | S50 | S51 | S52 | S53 | S54 | S55 | S56
S57 | S58 | S59 | S60 | S61 | S62 | S63 | S64
S65 | S66 | S67 | S68 | S69 | S70 | S71 | S72
S73 | S74 | S75 | S76 | S77 | S78 | S79 | S80
S81 | S82 | S83 | S84 | S85 | S86 | S87 | S88
S89 | S90 | S91 | S92 | S93 | S94 | S95 | S96
S97 | S98 | S99 | S100 | S101 | S102 | S103 | S104
S105 | S106 | S107 | S108 | S109 | S110 | S111 | S112
S113 | S114 | S115 | S116 | S117 | S118 | S119 | S120
S121 | S122 | S123 | S124 | S125 | S126 | S127 | S128
```

CHAPTER

FIVE

EXAMPLES

For more examples, visit our Rohde & Schwarz Github repository.

```
"""
# GitHub examples repository path: CMXP/Python/RsCmxp_xxx_ScpiPackages

Example on how to use the python RsCmx auto-generated instrument drivers for
RsCmpx_Base and RsCmpx_Gprf (Base and GPRF) in one script with shared VISA session.
"""

from RsCMPX_Base.RsCMPX_Base import RsCMPX_Base  # install from pypi.org
from RsCMPX_Base import enums as base_enums
from RsCMPX_Base import repcap as base_repcap

from RsCMPX_Gprf.RsCMPX_Gprf import RsCMPX_Gprf  # install from pypi.org
from RsCMPX_Gprf.CustomFiles.reliability import ReliabilityEventArgs
from RsCMPX_Gprf import enums as gprf_enums
from RsCMPX_Gprf import repcap as gprf_repcaps


# CMX Base init
cmx_base = RsCMPX_Base('TCPIP::10.112.1.116', False, True)
print(f'CMX Base IND: {cmx_base.utilities.idn_string}')
print(f'CMX Instrument options:\n{",".join(cmx_base.utilities.instrument_options)}')
cmx_base.utilities.visa_timeout = 5000  # default is 10000

# Sends OPC after each command
cmx_base.utilities.opc_query_after_write = False
# Checks for syst:err? after each command / query - default value after init is True
cmx_base.utilities.instrument_status_checking = True

# Self-test
self_test = cmx_base.utilities.self_test()
print(f'CMW self-test result: {self_test} - {"Passed" if self_test[0] == 0 else "Failed"}
↪"')
# Reference Frequency Source
cmx_base.system.reference.frequency.source_set(base_enums.SourceIntExt.INTernal)

# CMX RsCMPX_Gprf Init - reuse the session of the cmx_base, rather than creating another␣
↪one
cmx_gprf = RsCMPX_Gprf.from_existing_session(cmx_base)
```

(continues on next page)

(continued from previous page)

```
cmx_gprf.utilities.visa_timeout = 5000

# Driver's Interface reliability offers a convenient way of reacting on the return value␣
↪Reliability Indicator
cmx_gprf.reliability.ExceptionOnError = True  # default is 10000


# Callback to use for the reliability indicator update events
def my_reliability_handler(event_args: ReliabilityEventArgs):
        print(f'GPRF Reliability updated.\nContext: {event_args.context}\nMessage:
↪{event_args.message}')


# We register a callback for each change in the reliability indicator
cmx_gprf.reliability.on_update_handler = my_reliability_handler

# You can obtain the last value of the returned reliability
print(f"\nReliability last value: {cmx_gprf.reliability.last_value}, context '{cmx_gprf.
↪reliability.last_context}', message: {cmx_gprf.reliability.last_message}")

# Close the sessions
cmx_gprf.close()
cmx_base.close()
```

CHAPTER

SIX

RSCMPX_BLUETOOTHMEAS API STRUCTURE

Global RepCaps

```
driver = RsCMPX_BluetoothMeas('TCPIP::192.168.2.101::hislip0')
# Instance range: Inst1 .. Inst16
rc = driver.repcap_instance_get()
driver.repcap_instance_set(repcap.Instance.Inst1)
```

class RsCMPX_BluetoothMeas(*resource_name: str*, *id_query: bool = True*, *reset: bool = False*, *options: str = None*, *direct_session: object = None*)

1320 total commands, 8 Subgroups, 0 group commands

Initializes new RsCMPX_BluetoothMeas session.

Parameter options tokens examples:

- `Simulate=True` - starts the session in simulation mode. Default: `False`
- `SelectVisa=socket` - uses no VISA implementation for socket connections - you do not need any VISA-C installation
- `SelectVisa=rs` - forces usage of RohdeSchwarz Visa
- `SelectVisa=ivi` - forces usage of National Instruments Visa
- `QueryInstrumentStatus = False` - same as `driver.utilities.instrument_status_checking = False`. Default: `True`
- `WriteDelay = 20, ReadDelay = 5` - Introduces delay of 20ms before each write and 5ms before each read. Default: `0ms` for both
- `OpcWaitMode = OpcQuery` - mode for all the opc-synchronised write/reads. Other modes: StbPolling, StbPollingSlow, StbPollingSuperSlow. Default: `StbPolling`
- `AddTermCharToWriteBinBLock = True` - Adds one additional LF to the end of the binary data (some instruments require that). Default: `False`
- `AssureWriteWithTermChar = True` - Makes sure each command/query is terminated with termination character. Default: Interface dependent
- `TerminationCharacter = "\r"` - Sets the termination character for reading. Default: `\n` (LineFeed or LF)
- `DataChunkSize = 10E3` - Maximum size of one write/read segment. If transferred data is bigger, it is split to more segments. Default: `1E6` bytes
- `OpcTimeout = 10000` - same as driver.utilities.opc_timeout = 10000. Default: `30000ms`
- `VisaTimeout = 5000` - same as driver.utilities.visa_timeout = 5000. Default: `10000ms`

- `ViClearExeMode = Disabled` - viClear() execution mode. Default: `execute_on_all`
- `OpcQueryAfterWrite = True` - same as driver.utilities.opc_query_after_write = True. Default: `False`
- `StbInErrorCheck = False` - if true, the driver checks errors with *STB? If false, it uses SYST:ERR?. Default: `True`
- `ScpiQuotes = double'. - for SCPI commands, you can define how strings are quoted. With single or double quotes. Possible values: single | double | {char}. Default: ``single`
- `LoggingMode = On` - Sets the logging status right from the start. Default: `Off`
- `LoggingName = 'MyDevice'` - Sets the name to represent the session in the log entries. Default: `'resource_name'`
- `LogToGlobalTarget = True` - Sets the logging target to the class-property previously set with RsCMPX_BluetoothMeas.set_global_logging_target() Default: `False`
- `LoggingToConsole = True` - Immediately starts logging to the console. Default: False
- `LoggingToUdp = True` - Immediately starts logging to the UDP port. Default: False
- `LoggingUdpPort = 49200` - UDP port to log to. Default: 49200

Parameters

- **resource_name** – VISA resource name, e.g. 'TCPIP::192.168.2.1::INSTR'
- **id_query** – if True, the instrument's model name is verified against the models supported by the driver and eventually throws an exception.
- **reset** – Resets the instrument (sends *RST command) and clears its status sybsystem.
- **options** – string tokens alternating the driver settings.
- **direct_session** – Another driver object or pyVisa object to reuse the session instead of opening a new session.

static assert_minimum_version(*min_version: str*) → None

Asserts that the driver version fulfills the minimum required version you have entered. This way you make sure your installed driver is of the entered version or newer.

classmethod clear_global_logging_relative_timestamp() → None

Clears the global relative timestamp. After this, all the instances using the global relative timestamp continue logging with the absolute timestamps.

close() → None

Closes the active RsCMPX_BluetoothMeas session.

classmethod from_existing_session(*session: object*, *options: str = None*) → RsCMPX_BluetoothMeas

Creates a new RsCMPX_BluetoothMeas object with the entered 'session' reused.

Parameters

- **session** – can be another driver or a direct pyvisa session.
- **options** – string tokens alternating the driver settings.

classmethod get_global_logging_relative_timestamp() → datetime

Returns global common relative timestamp for log entries.

classmethod get_global_logging_target()

Returns global common target stream.

get_session_handle() → object

Returns the underlying session handle.

get_total_execution_time() → timedelta

Returns total time spent by the library on communicating with the instrument. This time is always shorter than get_total_time(), since it does not include gaps between the communication. You can reset this counter with reset_time_statistics().

get_total_time() → timedelta

Returns total time spent by the library on communicating with the instrument. This time is always shorter than get_total_time(), since it does not include gaps between the communication. You can reset this counter with reset_time_statistics().

static list_resources(*expression: str = '?*::INSTR'*, *visa_select: str = None*) → List[str]

Finds all the resources defined by the expression

- '?*' - matches all the available instruments
- 'USB::?*' - matches all the USB instruments
- 'TCPIP::192?*' - matches all the LAN instruments with the IP address starting with 192

Parameters

- **expression** – see the examples in the function
- **visa_select** – optional parameter selecting a specific VISA. Examples: '@ivi', '@rs'

reset_time_statistics() → None

Resets all execution and total time counters. Affects the results of get_total_time() and get_total_execution_time()

restore_all_repcaps_to_default() → None

Sets all the Group and Global repcaps to their initial values

classmethod set_global_logging_relative_timestamp(*timestamp: datetime*) → None

Sets global common relative timestamp for log entries. To use it, call the following: io.utilities.logger.set_relative_timestamp_global()

classmethod set_global_logging_relative_timestamp_now() → None

Sets global common relative timestamp for log entries to this moment. To use it, call the following: io.utilities.logger.set_relative_timestamp_global().

classmethod set_global_logging_target(*target*) → None

Sets global common target stream that each instance can use. To use it, call the following: io.utilities.logger.set_logging_target_global(). If an instance uses global logging target, it automatically uses the global relative timestamp (if set). You can set the target to None to invalidate it.

Subgroups

6.1 Bluetooth

class BluetoothCls

Bluetooth commands group definition. 903 total commands, 1 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.bluetooth.clone()
```

Subgroups

6.1.1 Measurement

class MeasurementCls

Measurement commands group definition. 903 total commands, 8 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.bluetooth.measurement.clone()
```

Subgroups

6.1.1.1 BhRate

SCPI Commands :

```
ABORt:BLUetooth:MEASurement<Instance>:BHRate
STOP:BLUetooth:MEASurement<Instance>:BHRate
INITiate:BLUetooth:MEASurement<Instance>:BHRate
```

class BhRateCls

BhRate commands group definition. 71 total commands, 7 Subgroups, 3 group commands

abort() → None

```
# SCPI: ABORt:BLUetooth:MEASurement<Instance>:BHRate
driver.bluetooth.measurement.bhRate.abort()
```

No command help available

abort_with_opc(*opc_timeout_ms: int = -1*) → None

```
# SCPI: ABORt:BLUetooth:MEASurement<Instance>:BHRate
driver.bluetooth.measurement.bhRate.abort_with_opc()
```

No command help available

Same as abort, but waits for the operation to complete before continuing further. Use the RsCMPX_BluetoothMeas.utilities.opc_timeout_set() to set the timeout value.

param opc_timeout_ms
Maximum time to wait in milliseconds, valid only for this call.

initiate() → None

```
# SCPI: INITiate:BLUetooth:MEASurement<Instance>:BHRate
driver.bluetooth.measurement.bhRate.initiate()
```

No command help available

initiate_with_opc(*opc_timeout_ms: int = -1*) → None

```
# SCPI: INITiate:BLUetooth:MEASurement<Instance>:BHRate
driver.bluetooth.measurement.bhRate.initiate_with_opc()
```

No command help available

Same as initiate, but waits for the operation to complete before continuing further. Use the RsCMPX_BluetoothMeas.utilities.opc_timeout_set() to set the timeout value.

param opc_timeout_ms
Maximum time to wait in milliseconds, valid only for this call.

stop() → None

```
# SCPI: STOP:BLUetooth:MEASurement<Instance>:BHRate
driver.bluetooth.measurement.bhRate.stop()
```

No command help available

stop_with_opc(*opc_timeout_ms: int = -1*) → None

```
# SCPI: STOP:BLUetooth:MEASurement<Instance>:BHRate
driver.bluetooth.measurement.bhRate.stop_with_opc()
```

No command help available

Same as stop, but waits for the operation to complete before continuing further. Use the RsCMPX_BluetoothMeas.utilities.opc_timeout_set() to set the timeout value.

param opc_timeout_ms
Maximum time to wait in milliseconds, valid only for this call.

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.bluetooth.measurement.bhRate.clone()
```

Subgroups

6.1.1.1.1 InputSignal

class InputSignalCls

InputSignal commands group definition. 2 total commands, 1 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.bluetooth.measurement.bhRate.inputSignal.clone()
```

Subgroups

6.1.1.1.1.1 Adetected

class AdetectedCls

Adetected commands group definition. 2 total commands, 2 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.bluetooth.measurement.bhRate.inputSignal.adetected.clone()
```

Subgroups

6.1.1.1.1.2 Plength

SCPI Command :

```
FETCh:BLUetooth:MEASurement<Instance>:BHRate:ISIGnal:ADETected:PLENgth
```

class PlengthCls

Plength commands group definition. 1 total commands, 0 Subgroups, 1 group commands

fetch() → int

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:BHRate:ISIGnal:ADETected:PLENgth
value: int = driver.bluetooth.measurement.bhRate.inputSignal.adetected.plength.
↪fetch()
```

No command help available

Suppressed linked return values: reliability

return
payload_length: No help available

6.1.1.1.1.3 Ptype

SCPI Command :

```
FETCh:BLUetooth:MEASurement<Instance>:BHRate:ISIGnal:ADETected:PTYPe
```

class PtypeCls

Ptype commands group definition. 1 total commands, 0 Subgroups, 1 group commands

fetch() → PacketTypeC

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:BHRate:ISIGnal:ADETected:PTYPe
value: enums.PacketTypeC = driver.bluetooth.measurement.bhRate.inputSignal.
↪adetected.ptype.fetch()
```

No command help available

Suppressed linked return values: reliability

return
packet_type: No help available

6.1.1.1.2 Modulation

class ModulationCls

Modulation commands group definition. 12 total commands, 4 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.bluetooth.measurement.bhRate.modulation.clone()
```

Subgroups

6.1.1.1.2.1 Average

SCPI Commands :

```
READ:BLUetooth:MEASurement<Instance>:BHRate:MODulation:AVERage
FETCh:BLUetooth:MEASurement<Instance>:BHRate:MODulation:AVERage
CALCulate:BLUetooth:MEASurement<Instance>:BHRate:MODulation:AVERage
```

class AverageCls

Average commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Bursts_Out_Of_Tol: float or bool: No parameter help available

- Wi: float or bool: No parameter help available
- W_0_Wi: float or bool: No parameter help available
- W_0_Max: float or bool: No parameter help available
- Rms_Devm: float or bool: No parameter help available
- Peak_Devm: float or bool: No parameter help available
- Nominal_Power: float or bool: No parameter help available

class FetchStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Bursts_Out_Of_Tol: float: No parameter help available
- Wi: float: No parameter help available
- W_0_Wi: float: No parameter help available
- W_0_Max: float: No parameter help available
- Rms_Devm: float: No parameter help available
- Peak_Devm: float: No parameter help available
- Nominal_Power: float: No parameter help available

class ReadStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Bursts_Out_Of_Tol: int: No parameter help available
- Wi: float: No parameter help available
- W_0_Wi: float: No parameter help available
- W_0_Max: float: No parameter help available
- Rms_Devm: float: No parameter help available
- Peak_Devm: float: No parameter help available
- Nominal_Power: float: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>:BHRate:MODulation:AVERage
value: CalculateStruct = driver.bluetooth.measurement.bhRate.modulation.average.
↪calculate()
```

No command help available

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → FetchStruct

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:BHRate:MODulation:AVERage
value: FetchStruct = driver.bluetooth.measurement.bhRate.modulation.average.
↪fetch()
```

No command help available

return
structure: for return value, see the help for FetchStruct structure arguments.

read() → ReadStruct

```
# SCPI: READ:BLUetooth:MEASurement<Instance>:BHRate:MODulation:AVERage
value: ReadStruct = driver.bluetooth.measurement.bhRate.modulation.average.
↪read()
```

No command help available

return
structure: for return value, see the help for ReadStruct structure arguments.

6.1.1.1.2.2 Current

SCPI Commands :

```
READ:BLUetooth:MEASurement<Instance>:BHRate:MODulation:CURRent
FETCh:BLUetooth:MEASurement<Instance>:BHRate:MODulation:CURRent
CALCulate:BLUetooth:MEASurement<Instance>:BHRate:MODulation:CURRent
```

class CurrentCls

Current commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Bursts_Out_Of_Tol: float or bool: No parameter help available
- Wi: float or bool: No parameter help available
- W_0_Wi: float or bool: No parameter help available
- W_0_Max: float or bool: No parameter help available
- Rms_Devm: float or bool: No parameter help available
- Peak_Devm: float or bool: No parameter help available
- P_99_Devm: float or bool: No parameter help available
- Nominal_Power: float or bool: No parameter help available

class ResultData

Response structure. Fields:

- Reliability: int: No parameter help available
- Bursts_Out_Of_Tol: float: No parameter help available
- Wi: float: No parameter help available
- W_0_Wi: float: No parameter help available
- W_0_Max: float: No parameter help available
- Rms_Devm: float: No parameter help available

- Peak_Devm: float: No parameter help available
- P_99_Devm: float: No parameter help available
- Nominal_Power: float: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>:BHRate:MODulation:CURRent
value: CalculateStruct = driver.bluetooth.measurement.bhRate.modulation.current.
↪calculate()
```

No command help available

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → ResultData

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:BHRate:MODulation:CURRent
value: ResultData = driver.bluetooth.measurement.bhRate.modulation.current.
↪fetch()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

read() → ResultData

```
# SCPI: READ:BLUetooth:MEASurement<Instance>:BHRate:MODulation:CURRent
value: ResultData = driver.bluetooth.measurement.bhRate.modulation.current.
↪read()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

6.1.1.1.2.3 Maximum

SCPI Commands :

```
READ:BLUetooth:MEASurement<Instance>:BHRate:MODulation:MAXimum
FETCh:BLUetooth:MEASurement<Instance>:BHRate:MODulation:MAXimum
CALCulate:BLUetooth:MEASurement<Instance>:BHRate:MODulation:MAXimum
```

class MaximumCls

Maximum commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Bursts_Out_Of_Tol: float or bool: No parameter help available
- Wi: float or bool: No parameter help available

- W_0_Wi: float or bool: No parameter help available
- W_0_Max: float or bool: No parameter help available
- Rms_Devm: float or bool: No parameter help available
- Peak_Devm: float or bool: No parameter help available
- Nominal_Power: float or bool: No parameter help available

class ResultData

Response structure. Fields:

- Reliability: int: No parameter help available
- Bursts_Out_Of_Tol: float: No parameter help available
- Wi: float: No parameter help available
- W_0_Wi: float: No parameter help available
- W_0_Max: float: No parameter help available
- Rms_Devm: float: No parameter help available
- Peak_Devm: float: No parameter help available
- Nominal_Power: float: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>:BHRate:MODulation:MAXimum
value: CalculateStruct = driver.bluetooth.measurement.bhRate.modulation.maximum.
↪calculate()
```

No command help available

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → ResultData

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:BHRate:MODulation:MAXimum
value: ResultData = driver.bluetooth.measurement.bhRate.modulation.maximum.
↪fetch()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

read() → ResultData

```
# SCPI: READ:BLUetooth:MEASurement<Instance>:BHRate:MODulation:MAXimum
value: ResultData = driver.bluetooth.measurement.bhRate.modulation.maximum.
↪read()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

6.1.1.1.2.4 StandardDev

SCPI Commands :

```
READ:BLUetooth:MEASurement<Instance>:BHRate:MODulation:SDEViation
FETCh:BLUetooth:MEASurement<Instance>:BHRate:MODulation:SDEViation
CALCulate:BLUetooth:MEASurement<Instance>:BHRate:MODulation:SDEViation
```

class StandardDevCls

StandardDev commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Bursts_Out_Of_Tol: float or bool: No parameter help available
- Nominal_Power: float: No parameter help available
- Wi: float or bool: No parameter help available
- W_0_Wi: float or bool: No parameter help available
- W_0_Max: float or bool: No parameter help available
- Rms_Devm: float or bool: No parameter help available
- Peak_Devm: float or bool: No parameter help available

class FetchStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Bursts_Out_Of_Tol: float: No parameter help available
- Nominal_Power: float: No parameter help available
- Wi: float: No parameter help available
- W_0_Wi: float: No parameter help available
- W_0_Max: float: No parameter help available
- Rms_Devm: float: No parameter help available
- Peak_Devm: float: No parameter help available

class ReadStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Bursts_Out_Of_Tol: float: No parameter help available
- Wi: float: No parameter help available
- W_0_Wi: float: No parameter help available
- W_0_Max: float: No parameter help available
- Rms_Devm: float: No parameter help available
- Peak_Devm: float: No parameter help available

- Nominal_Power: float: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>:BHRate:MODulation:SDEViation
value: CalculateStruct = driver.bluetooth.measurement.bhRate.modulation.
↪standardDev.calculate()
```

No command help available

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → FetchStruct

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:BHRate:MODulation:SDEViation
value: FetchStruct = driver.bluetooth.measurement.bhRate.modulation.standardDev.
↪fetch()
```

No command help available

return
structure: for return value, see the help for FetchStruct structure arguments.

read() → ReadStruct

```
# SCPI: READ:BLUetooth:MEASurement<Instance>:BHRate:MODulation:SDEViation
value: ReadStruct = driver.bluetooth.measurement.bhRate.modulation.standardDev.
↪read()
```

No command help available

return
structure: for return value, see the help for ReadStruct structure arguments.

6.1.1.1.3 Pencoding

class PencodingCls

Pencoding commands group definition. 3 total commands, 1 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.bluetooth.measurement.bhRate.pencoding.clone()
```

Subgroups

6.1.1.1.3.1 Current

SCPI Commands :

```
READ:BLUetooth:MEASurement<Instance>:BHRate:PENCoding:CURRent
FETCh:BLUetooth:MEASurement<Instance>:BHRate:PENCoding:CURRent
CALCulate:BLUetooth:MEASurement<Instance>:BHRate:PENCoding:CURRent
```

class CurrentCls

Current commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Nominal_Power: float or bool: No parameter help available
- Sync_Bit_Errors: float or bool: No parameter help available
- Trailer_Bit_Error: float or bool: No parameter help available

class ResultData

Response structure. Fields:

- Reliability: int: No parameter help available
- Nominal_Power: float: No parameter help available
- Sync_Bit_Errors: int: No parameter help available
- Trailer_Bit_Error: int: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>:BHRate:PENCoding:CURRent
value: CalculateStruct = driver.bluetooth.measurement.bhRate.pencoding.current.
↪calculate()
```

No command help available

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → ResultData

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:BHRate:PENCoding:CURRent
value: ResultData = driver.bluetooth.measurement.bhRate.pencoding.current.
↪fetch()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

read() → ResultData

```
# SCPI: READ:BLUetooth:MEASurement<Instance>:BHRate:PENCoding:CURRent
value: ResultData = driver.bluetooth.measurement.bhRate.pencoding.current.read()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

6.1.1.1.4 PowerVsTime

class PowerVsTimeCls

PowerVsTime commands group definition. 12 total commands, 4 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.bluetooth.measurement.bhRate.powerVsTime.clone()
```

Subgroups

6.1.1.1.4.1 Average

SCPI Commands :

```
READ:BLUetooth:MEASurement<Instance>:BHRate:PVTime:AVERage
FETCh:BLUetooth:MEASurement<Instance>:BHRate:PVTime:AVERage
CALCulate:BLUetooth:MEASurement<Instance>:BHRate:PVTime:AVERage
```

class AverageCls

Average commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Bursts_Out_Of_Tol: float or bool: No parameter help available
- Nominal_Power: float or bool: No parameter help available
- Gfsk_Power: float or bool: No parameter help available
- Psk_Power: float or bool: No parameter help available
- Psk_Gfsk_Power: float or bool: No parameter help available
- Guard_Period: float or bool: No parameter help available

class ResultData

Response structure. Fields:

- Reliability: int: No parameter help available

- Bursts_Out_Of_Tol: float: No parameter help available
- Nominal_Power: float: No parameter help available
- Gfsk_Power: float: No parameter help available
- Psk_Power: float: No parameter help available
- Psk_Gfsk_Power: float: No parameter help available
- Guard_Period: float: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>:BHRate:PVTime:AVERage
value: CalculateStruct = driver.bluetooth.measurement.bhRate.powerVsTime.
↪average.calculate()
```

No command help available

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → ResultData

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:BHRate:PVTime:AVERage
value: ResultData = driver.bluetooth.measurement.bhRate.powerVsTime.average.
↪fetch()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

read() → ResultData

```
# SCPI: READ:BLUetooth:MEASurement<Instance>:BHRate:PVTime:AVERage
value: ResultData = driver.bluetooth.measurement.bhRate.powerVsTime.average.
↪read()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

6.1.1.1.4.2 Current

SCPI Commands :

```
READ:BLUetooth:MEASurement<Instance>:BHRate:PVTime:CURRent
FETCh:BLUetooth:MEASurement<Instance>:BHRate:PVTime:CURRent
CALCulate:BLUetooth:MEASurement<Instance>:BHRate:PVTime:CURRent
```

class CurrentCls

Current commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Bursts_Out_Of_Tol: float or bool: No parameter help available
- Nominal_Power: float or bool: No parameter help available
- Gfsk_Power: float or bool: No parameter help available
- Psk_Power: float or bool: No parameter help available
- Psk_Gfsk_Power: float or bool: No parameter help available
- Guard_Period: float or bool: No parameter help available

class ResultData

Response structure. Fields:

- Reliability: int: No parameter help available
- Bursts_Out_Of_Tol: float: No parameter help available
- Nominal_Power: float: No parameter help available
- Gfsk_Power: float: No parameter help available
- Psk_Power: float: No parameter help available
- Psk_Gfsk_Power: float: No parameter help available
- Guard_Period: float: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>:BHRate:PVTime:CURRent
value: CalculateStruct = driver.bluetooth.measurement.bhRate.powerVsTime.
↪current.calculate()
```

No command help available

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → ResultData

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:BHRate:PVTime:CURRent
value: ResultData = driver.bluetooth.measurement.bhRate.powerVsTime.current.
↪fetch()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

read() → ResultData

```
# SCPI: READ:BLUetooth:MEASurement<Instance>:BHRate:PVTime:CURRent
value: ResultData = driver.bluetooth.measurement.bhRate.powerVsTime.current.
↪read()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

6.1.1.1.4.3 Maximum

SCPI Commands :

```
READ:BLUetooth:MEASurement<Instance>:BHRate:PVTime:MAXimum
FETCh:BLUetooth:MEASurement<Instance>:BHRate:PVTime:MAXimum
CALCulate:BLUetooth:MEASurement<Instance>:BHRate:PVTime:MAXimum
```

class MaximumCls

Maximum commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Bursts_Out_Of_Tol: float or bool: No parameter help available
- Nominal_Power: float or bool: No parameter help available
- Gfsk_Power: float or bool: No parameter help available
- Psk_Power: float or bool: No parameter help available
- Psk_Gfsk_Power: float or bool: No parameter help available
- Guard_Period: float or bool: No parameter help available

class ResultData

Response structure. Fields:

- Reliability: int: No parameter help available
- Bursts_Out_Of_Tol: float: No parameter help available
- Nominal_Power: float: No parameter help available
- Gfsk_Power: float: No parameter help available
- Psk_Power: float: No parameter help available
- Psk_Gfsk_Power: float: No parameter help available
- Guard_Period: float: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>:BHRate:PVTime:MAXimum
value: CalculateStruct = driver.bluetooth.measurement.bhRate.powerVsTime.
↪maximum.calculate()
```

No command help available

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → ResultData

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:BHRate:PVTime:MAXimum
value: ResultData = driver.bluetooth.measurement.bhRate.powerVsTime.maximum.
↪fetch()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

read() → ResultData

```
# SCPI: READ:BLUetooth:MEASurement<Instance>:BHRate:PVTime:MAXimum
value: ResultData = driver.bluetooth.measurement.bhRate.powerVsTime.maximum.
↪read()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

6.1.1.1.4.4 Minimum

SCPI Commands :

```
READ:BLUetooth:MEASurement<Instance>:BHRate:PVTime:MINimum
FETCh:BLUetooth:MEASurement<Instance>:BHRate:PVTime:MINimum
CALCulate:BLUetooth:MEASurement<Instance>:BHRate:PVTime:MINimum
```

class MinimumCls

Minimum commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Bursts_Out_Of_Tol: float or bool: No parameter help available
- Nominal_Power: float or bool: No parameter help available
- Gfsk_Power: float or bool: No parameter help available
- Psk_Power: float or bool: No parameter help available
- Psk_Gfsk_Power: float or bool: No parameter help available
- Guard_Period: float or bool: No parameter help available

class ResultData

Response structure. Fields:

- Reliability: int: No parameter help available
- Bursts_Out_Of_Tol: float: No parameter help available
- Nominal_Power: float: No parameter help available
- Gfsk_Power: float: No parameter help available

- Psk_Power: float: No parameter help available
- Psk_Gfsk_Power: float: No parameter help available
- Guard_Period: float: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>:BHRate:PVTime:MINimum
value: CalculateStruct = driver.bluetooth.measurement.bhRate.powerVsTime.
↪minimum.calculate()
```

No command help available

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → ResultData

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:BHRate:PVTime:MINimum
value: ResultData = driver.bluetooth.measurement.bhRate.powerVsTime.minimum.
↪fetch()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

read() → ResultData

```
# SCPI: READ:BLUetooth:MEASurement<Instance>:BHRate:PVTime:MINimum
value: ResultData = driver.bluetooth.measurement.bhRate.powerVsTime.minimum.
↪read()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

6.1.1.1.5 Sgacp

class SgacpCls

Sgacp commands group definition. 3 total commands, 1 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.bluetooth.measurement.bhRate.sgacp.clone()
```

Subgroups

6.1.1.1.5.1 Current

SCPI Commands :

```
READ:BLUetooth:MEASurement<Instance>:BHRate:SGACp[:CURRent]
FETCh:BLUetooth:MEASurement<Instance>:BHRate:SGACp[:CURRent]
CALCulate:BLUetooth:MEASurement<Instance>:BHRate:SGACp[:CURRent]
```

class CurrentCls

Current commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Bursts_Out_Of_Tol: float or bool: No parameter help available
- Nominal_Power: float or bool: No parameter help available
- Ptx_Ref_One_Mh_Z: float or bool: No parameter help available
- Ptx_Ref_Two_Mh_Z: float or bool: No parameter help available
- Ptx_Ref_Four_Mh_Z: float or bool: No parameter help available
- Ptx_Minus_26_N_1_Abs: float or bool: No parameter help available
- Ptx_Minus_26_N_1_Rel: float or bool: No parameter help available
- Ptx_Minus_25_N_2_Abs: float or bool: No parameter help available
- Ptx_Minus_25_N_2_Rel: float or bool: No parameter help available
- Ptx_Minus_7_N_3_Abs: float or bool: No parameter help available
- Ptx_Minus_7_N_3_Rel: float or bool: No parameter help available
- Ptx_Minus_26_P_1_Abs: float or bool: No parameter help available
- Ptx_Minus_26_P_1_Rel: float or bool: No parameter help available
- Ptx_Minus_25_P_2_Abs: float or bool: No parameter help available
- Ptx_Minus_25_P_2_Rel: float or bool: No parameter help available
- Ptx_Minus_7_P_3_Abs: float or bool: No parameter help available
- Ptx_Minus_7_P_3_Rel: float or bool: No parameter help available

class ResultData

Response structure. Fields:

- Reliability: int: No parameter help available
- Bursts_Out_Of_Tol: float: No parameter help available
- Nominal_Power: float: No parameter help available
- Ptx_Ref_One_Mh_Z: float: No parameter help available
- Ptx_Ref_Two_Mh_Z: float: No parameter help available

- Ptx_Ref_Four_Mh_Z: float: No parameter help available
- Ptx_Minus_26_N_1_Abs: float: No parameter help available
- Ptx_Minus_26_N_1_Rel: float: No parameter help available
- Ptx_Minus_25_N_2_Abs: float: No parameter help available
- Ptx_Minus_25_N_2_Rel: float: No parameter help available
- Ptx_Minus_7_N_3_Abs: float: No parameter help available
- Ptx_Minus_7_N_3_Rel: float: No parameter help available
- Ptx_Minus_26_P_1_Abs: float: No parameter help available
- Ptx_Minus_26_P_1_Rel: float: No parameter help available
- Ptx_Minus_25_P_2_Abs: float: No parameter help available
- Ptx_Minus_25_P_2_Rel: float: No parameter help available
- Ptx_Minus_7_P_3_Abs: float: No parameter help available
- Ptx_Minus_7_P_3_Rel: float: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>:BHRate:SGACp[:CURRent]
value: CalculateStruct = driver.bluetooth.measurement.bhRate.sgacp.current.
↪calculate()
```

No command help available

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → ResultData

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:BHRate:SGACp[:CURRent]
value: ResultData = driver.bluetooth.measurement.bhRate.sgacp.current.fetch()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

read() → ResultData

```
# SCPI: READ:BLUetooth:MEASurement<Instance>:BHRate:SGACp[:CURRent]
value: ResultData = driver.bluetooth.measurement.bhRate.sgacp.current.read()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

6.1.1.1.6 State

SCPI Command :

```
FETCh:BLUetooth:MEASurement<Instance>:BHRate:STATe
```

class StateCls

State commands group definition. 2 total commands, 1 Subgroups, 1 group commands

fetch(*timeout: float = None*, *target_main_state: ResourceState = None*, *target_sync_state: SyncState = None*) → ResourceState

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:BHRate:STATe
value: enums.ResourceState = driver.bluetooth.measurement.bhRate.state.
↪fetch(timeout = 1.0, target_main_state = enums.ResourceState.ACTive, target_
↪sync_state = enums.SyncState.ADJusted)
```

No command help available

param timeout
No help available

param target_main_state
No help available

param target_sync_state
No help available

return
meas_state: No help available

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.bluetooth.measurement.bhRate.state.clone()
```

Subgroups

6.1.1.1.6.1 All

SCPI Command :

```
FETCh:BLUetooth:MEASurement<Instance>:BHRate:STATe:ALL
```

class AllCls

All commands group definition. 1 total commands, 0 Subgroups, 1 group commands

fetch(*timeout: float = None*, *target_main_state: ResourceState = None*, *target_sync_state: SyncState = None*) → ResourceState

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:BHRate:STATe:ALL
value: enums.ResourceState = driver.bluetooth.measurement.bhRate.state.all.
↪fetch(timeout = 1.0, target_main_state = enums.ResourceState.ACTive, target_
↪sync_state = enums.SyncState.ADJusted)
```

No command help available

param timeout
No help available

param target_main_state
No help available

param target_sync_state
No help available

return
meas_states: No help available

6.1.1.1.7 Trace

class TraceCls

Trace commands group definition. 34 total commands, 7 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.bluetooth.measurement.bhRate.trace.clone()
```

Subgroups

6.1.1.1.7.1 DevMagnitude

class DevMagnitudeCls

DevMagnitude commands group definition. 6 total commands, 3 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.bluetooth.measurement.bhRate.trace.devMagnitude.clone()
```

Subgroups

6.1.1.1.7.2 Average

SCPI Commands :

```
READ:BLUetooth:MEASurement<Instance>:BHRate:TRACe:DEVMagnitude:AVERage
FETCh:BLUetooth:MEASurement<Instance>:BHRate:TRACe:DEVMagnitude:AVERage
```

class AverageCls

Average commands group definition. 2 total commands, 0 Subgroups, 2 group commands

fetch() → List[float]

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:BHRate:TRACe:DEVMagnitude:AVERage
value: List[float] = driver.bluetooth.measurement.bhRate.trace.devMagnitude.
↪average.fetch()
```

No command help available

Suppressed linked return values: reliability

return
devm: No help available

read() → List[float]

```
# SCPI: READ:BLUetooth:MEASurement<Instance>:BHRate:TRACe:DEVMagnitude:AVERage
value: List[float] = driver.bluetooth.measurement.bhRate.trace.devMagnitude.
↪average.read()
```

No command help available

Suppressed linked return values: reliability

return
devm: No help available

6.1.1.1.7.3 Current

SCPI Commands :

```
READ:BLUetooth:MEASurement<Instance>:BHRate:TRACe:DEVMagnitude:CURRent
FETCh:BLUetooth:MEASurement<Instance>:BHRate:TRACe:DEVMagnitude:CURRent
```

class CurrentCls

Current commands group definition. 2 total commands, 0 Subgroups, 2 group commands

fetch() → List[float]

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:BHRate:TRACe:DEVMagnitude:CURRent
value: List[float] = driver.bluetooth.measurement.bhRate.trace.devMagnitude.
↪current.fetch()
```

No command help available

Suppressed linked return values: reliability

return
devm: No help available

read() → List[float]

```
# SCPI: READ:BLUetooth:MEASurement<Instance>:BHRate:TRACe:DEVMagnitude:CURRent
value: List[float] = driver.bluetooth.measurement.bhRate.trace.devMagnitude.
↪current.read()
```

No command help available

Suppressed linked return values: reliability

return
devm: No help available

6.1.1.1.7.4 Maximum

SCPI Commands :

```
READ:BLUetooth:MEASurement<Instance>:BHRate:TRACe:DEVMagnitude:MAXimum
FETCh:BLUetooth:MEASurement<Instance>:BHRate:TRACe:DEVMagnitude:MAXimum
```

class MaximumCls

Maximum commands group definition. 2 total commands, 0 Subgroups, 2 group commands

fetch() → List[float]

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:BHRate:TRACe:DEVMagnitude:MAXimum
value: List[float] = driver.bluetooth.measurement.bhRate.trace.devMagnitude.
↪maximum.fetch()
```

No command help available

Suppressed linked return values: reliability

return
devm: No help available

read() → List[float]

```
# SCPI: READ:BLUetooth:MEASurement<Instance>:BHRate:TRACe:DEVMagnitude:MAXimum
value: List[float] = driver.bluetooth.measurement.bhRate.trace.devMagnitude.
↪maximum.read()
```

No command help available

Suppressed linked return values: reliability

return
devm: No help available

6.1.1.1.7.5 IqAbs

SCPI Commands :

```
FETCh:BLUetooth:MEASurement<Instance>:BHRate:TRACe:IQABs
READ:BLUetooth:MEASurement<Instance>:BHRate:TRACe:IQABs
```

class IqAbsCls

IqAbs commands group definition. 2 total commands, 0 Subgroups, 2 group commands

class ResultData

Response structure. Fields:

- Reliability: int: No parameter help available
- Iphase: List[float]: No parameter help available
- Qphase: List[float]: No parameter help available

fetch() → ResultData

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:BHRate:TRACe:IQABs
value: ResultData = driver.bluetooth.measurement.bhRate.trace.iqAbs.fetch()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

read() → ResultData

```
# SCPI: READ:BLUetooth:MEASurement<Instance>:BHRate:TRACe:IQABs
value: ResultData = driver.bluetooth.measurement.bhRate.trace.iqAbs.read()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

6.1.1.1.7.6 IqDifference

SCPI Commands :

```
FETCh:BLUetooth:MEASurement<Instance>:BHRate:TRACe:IQDiff
READ:BLUetooth:MEASurement<Instance>:BHRate:TRACe:IQDiff
```

class IqDifferenceCls

IqDifference commands group definition. 2 total commands, 0 Subgroups, 2 group commands

class ResultData

Response structure. Fields:

- Reliability: int: No parameter help available
- Iphase: List[float]: No parameter help available
- Qphase: List[float]: No parameter help available

fetch() → ResultData

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:BHRate:TRACe:IQDiff
value: ResultData = driver.bluetooth.measurement.bhRate.trace.iqDifference.
↪fetch()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

read() → ResultData

```
# SCPI: READ:BLUetooth:MEASurement<Instance>:BHRate:TRACe:IQDiff
value: ResultData = driver.bluetooth.measurement.bhRate.trace.iqDifference.
↪read()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

6.1.1.1.7.7 IqError

SCPI Commands :

```
READ:BLUetooth:MEASurement<Instance>:BHRate:TRACe:IQERr
FETCh:BLUetooth:MEASurement<Instance>:BHRate:TRACe:IQERr
```

class IqErrorCls

IqError commands group definition. 2 total commands, 0 Subgroups, 2 group commands

class ResultData

Response structure. Fields:

- Reliability: int: No parameter help available
- Iphase: List[float]: No parameter help available
- Qphase: List[float]: No parameter help available

fetch() → ResultData

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:BHRate:TRACe:IQERr
value: ResultData = driver.bluetooth.measurement.bhRate.trace.iqError.fetch()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

read() → ResultData

```
# SCPI: READ:BLUetooth:MEASurement<Instance>:BHRate:TRACe:IQERr
value: ResultData = driver.bluetooth.measurement.bhRate.trace.iqError.read()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

6.1.1.1.7.8 Pdifference

class PdifferenceCls

Pdifference commands group definition. 6 total commands, 3 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.bluetooth.measurement.bhRate.trace.pdifference.clone()
```

Subgroups

6.1.1.1.7.9 Average

SCPI Commands :

```
READ:BLUetooth:MEASurement<Instance>:BHRate:TRACe:PDIFference:AVERage
FETCh:BLUetooth:MEASurement<Instance>:BHRate:TRACe:PDIFference:AVERage
```

class AverageCls

Average commands group definition. 2 total commands, 0 Subgroups, 2 group commands

fetch() → List[float]

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:BHRate:TRACe:PDIFference:AVERage
value: List[float] = driver.bluetooth.measurement.bhRate.trace.pdifference.
↪average.fetch()
```

No command help available

Suppressed linked return values: reliability

return
phase_difference: No help available

read() → List[float]

```
# SCPI: READ:BLUetooth:MEASurement<Instance>:BHRate:TRACe:PDIFference:AVERage
value: List[float] = driver.bluetooth.measurement.bhRate.trace.pdifference.
↪average.read()
```

No command help available

Suppressed linked return values: reliability

return
phase_difference: No help available

6.1.1.1.7.10 Current

SCPI Commands :

```
READ:BLUetooth:MEASurement<Instance>:BHRate:TRACe:PDIFference:CURRent
FETCh:BLUetooth:MEASurement<Instance>:BHRate:TRACe:PDIFference:CURRent
```

class CurrentCls

Current commands group definition. 2 total commands, 0 Subgroups, 2 group commands

fetch() → List[float]

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:BHRate:TRACe:PDIFference:CURRent
value: List[float] = driver.bluetooth.measurement.bhRate.trace.pdifference.
↪current.fetch()
```

No command help available

Suppressed linked return values: reliability

return
phase_difference: No help available

read() → List[float]

```
# SCPI: READ:BLUetooth:MEASurement<Instance>:BHRate:TRACe:PDIFference:CURRent
value: List[float] = driver.bluetooth.measurement.bhRate.trace.pdifference.
↪current.read()
```

No command help available

Suppressed linked return values: reliability

return
phase_difference: No help available

6.1.1.1.7.11 Maximum

SCPI Commands :

```
READ:BLUetooth:MEASurement<Instance>:BHRate:TRACe:PDIFference:MAXimum
FETCh:BLUetooth:MEASurement<Instance>:BHRate:TRACe:PDIFference:MAXimum
```

class MaximumCls

Maximum commands group definition. 2 total commands, 0 Subgroups, 2 group commands

fetch() → List[float]

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:BHRate:TRACe:PDIFference:MAXimum
value: List[float] = driver.bluetooth.measurement.bhRate.trace.pdifference.
↪maximum.fetch()
```

No command help available

Suppressed linked return values: reliability

return
phase_difference: No help available

read() → List[float]

```
# SCPI: READ:BLUetooth:MEASurement<Instance>:BHRate:TRACe:PDIFference:MAXimum
value: List[float] = driver.bluetooth.measurement.bhRate.trace.pdifference.
↪maximum.read()
```

No command help available

Suppressed linked return values: reliability

return
phase_difference: No help available

6.1.1.1.7.12 PowerVsTime

class PowerVsTimeCls

PowerVsTime commands group definition. 8 total commands, 4 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.bluetooth.measurement.bhRate.trace.powerVsTime.clone()
```

Subgroups

6.1.1.1.7.13 Average

SCPI Commands :

```
READ:BLUetooth:MEASurement<Instance>:BHRate:TRACe:PVTime:AVERage
FETCh:BLUetooth:MEASurement<Instance>:BHRate:TRACe:PVTime:AVERage
```

class AverageCls

Average commands group definition. 2 total commands, 0 Subgroups, 2 group commands

fetch() → List[float]

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:BHRate:TRACe:PVTime:AVERage
value: List[float] = driver.bluetooth.measurement.bhRate.trace.powerVsTime.
↪average.fetch()
```

No command help available

Suppressed linked return values: reliability

return
pvt_trace: No help available

read() → List[float]

```
# SCPI: READ:BLUetooth:MEASurement<Instance>:BHRate:TRACe:PVTime:AVERage
value: List[float] = driver.bluetooth.measurement.bhRate.trace.powerVsTime.
↪average.read()
```

No command help available

Suppressed linked return values: reliability

return
pvt_trace: No help available

6.1.1.1.7.14 Current

SCPI Commands :

```
READ:BLUetooth:MEASurement<Instance>:BHRate:TRACe:PVTime:CURRent
FETCh:BLUetooth:MEASurement<Instance>:BHRate:TRACe:PVTime:CURRent
```

class CurrentCls

Current commands group definition. 2 total commands, 0 Subgroups, 2 group commands

fetch() → List[float]

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:BHRate:TRACe:PVTime:CURRent
value: List[float] = driver.bluetooth.measurement.bhRate.trace.powerVsTime.
↪current.fetch()
```

No command help available

Suppressed linked return values: reliability

return
pvt_trace: No help available

read() → List[float]

```
# SCPI: READ:BLUetooth:MEASurement<Instance>:BHRate:TRACe:PVTime:CURRent
value: List[float] = driver.bluetooth.measurement.bhRate.trace.powerVsTime.
↪current.read()
```

No command help available

Suppressed linked return values: reliability

return
pvt_trace: No help available

6.1.1.1.7.15 Maximum

SCPI Commands :

```
READ:BLUetooth:MEASurement<Instance>:BHRate:TRACe:PVTime:MAXimum
FETCh:BLUetooth:MEASurement<Instance>:BHRate:TRACe:PVTime:MAXimum
```

class MaximumCls

Maximum commands group definition. 2 total commands, 0 Subgroups, 2 group commands

fetch() → List[float]

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:BHRate:TRACe:PVTime:MAXimum
value: List[float] = driver.bluetooth.measurement.bhRate.trace.powerVsTime.
↪maximum.fetch()
```

No command help available

Suppressed linked return values: reliability

return
pvt_trace: No help available

read() → List[float]

```
# SCPI: READ:BLUetooth:MEASurement<Instance>:BHRate:TRACe:PVTime:MAXimum
value: List[float] = driver.bluetooth.measurement.bhRate.trace.powerVsTime.
↪maximum.read()
```

No command help available

Suppressed linked return values: reliability

return
pvt_trace: No help available

6.1.1.1.7.16 Minimum

SCPI Commands :

```
READ:BLUetooth:MEASurement<Instance>:BHRate:TRACe:PVTime:MINimum
FETCh:BLUetooth:MEASurement<Instance>:BHRate:TRACe:PVTime:MINimum
```

class MinimumCls

Minimum commands group definition. 2 total commands, 0 Subgroups, 2 group commands

fetch() → List[float]

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:BHRate:TRACe:PVTime:MINimum
value: List[float] = driver.bluetooth.measurement.bhRate.trace.powerVsTime.
↪minimum.fetch()
```

No command help available

Suppressed linked return values: reliability

return
pvt_trace: No help available

read() → List[float]

```
# SCPI: READ:BLUetooth:MEASurement<Instance>:BHRate:TRACe:PVTime:MINimum
value: List[float] = driver.bluetooth.measurement.bhRate.trace.powerVsTime.
↪minimum.read()
```

No command help available

Suppressed linked return values: reliability

return
pvt_trace: No help available

6.1.1.1.7.17 Sgacp

class SgacpCls

Sgacp commands group definition. 8 total commands, 4 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.bluetooth.measurement.bhRate.trace.sgacp.clone()
```

Subgroups

6.1.1.1.7.18 Average

SCPI Commands :

```
READ:BLUetooth:MEASurement<Instance>:BHRate:TRACe:SGACp:AVERage
FETCh:BLUetooth:MEASurement<Instance>:BHRate:TRACe:SGACp:AVERage
```

class AverageCls

Average commands group definition. 2 total commands, 0 Subgroups, 2 group commands

fetch() → List[float]

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:BHRate:TRACe:SGACp:AVERage
value: List[float] = driver.bluetooth.measurement.bhRate.trace.sgacp.average.
↪fetch()
```

No command help available

Suppressed linked return values: reliability

return
sgacp_trace: No help available

read() → List[float]

```
# SCPI: READ:BLUetooth:MEASurement<Instance>:BHRate:TRACe:SGACp:AVERage
value: List[float] = driver.bluetooth.measurement.bhRate.trace.sgacp.average.
↪read()
```

No command help available

Suppressed linked return values: reliability

return
sgacp_trace: No help available

6.1.1.1.7.19 Current

SCPI Commands :

```
READ:BLUetooth:MEASurement<Instance>:BHRate:TRACe:SGACp:CURRent
FETCh:BLUetooth:MEASurement<Instance>:BHRate:TRACe:SGACp:CURRent
```

class CurrentCls

Current commands group definition. 2 total commands, 0 Subgroups, 2 group commands

fetch() → List[float]

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:BHRate:TRACe:SGACp:CURRent
value: List[float] = driver.bluetooth.measurement.bhRate.trace.sgacp.current.
↪fetch()
```

No command help available

Suppressed linked return values: reliability

return
sgacp_trace: No help available

read() → List[float]

```
# SCPI: READ:BLUetooth:MEASurement<Instance>:BHRate:TRACe:SGACp:CURRent
value: List[float] = driver.bluetooth.measurement.bhRate.trace.sgacp.current.
↪read()
```

No command help available

Suppressed linked return values: reliability

return
sgacp_trace: No help available

6.1.1.1.7.20 Maximum

SCPI Commands :

```
READ:BLUetooth:MEASurement<Instance>:BHRate:TRACe:SGACp:MAXimum
FETCh:BLUetooth:MEASurement<Instance>:BHRate:TRACe:SGACp:MAXimum
```

class MaximumCls

Maximum commands group definition. 2 total commands, 0 Subgroups, 2 group commands

fetch() → List[float]

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:BHRate:TRACe:SGACp:MAXimum
value: List[float] = driver.bluetooth.measurement.bhRate.trace.sgacp.maximum.
↪fetch()
```

No command help available

Suppressed linked return values: reliability

return
sgacp_trace: No help available

read() → List[float]

```
# SCPI: READ:BLUetooth:MEASurement<Instance>:BHRate:TRACe:SGACp:MAXimum
value: List[float] = driver.bluetooth.measurement.bhRate.trace.sgacp.maximum.
↪read()
```

No command help available

Suppressed linked return values: reliability

return
sgacp_trace: No help available

6.1.1.1.7.21 Ptx

SCPI Commands :

```
READ:BLUetooth:MEASurement<Instance>:BHRate:TRACe:SGACp[:PTX]
FETCh:BLUetooth:MEASurement<Instance>:BHRate:TRACe:SGACp[:PTX]
```

class PtxCls

Ptx commands group definition. 2 total commands, 0 Subgroups, 2 group commands

fetch() → List[float]

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:BHRate:TRACe:SGACp[:PTX]
value: List[float] = driver.bluetooth.measurement.bhRate.trace.sgacp.ptx.fetch()
```

No command help available

Suppressed linked return values: reliability

return
sgacp_trace: No help available

read() → List[float]

```
# SCPI: READ:BLUetooth:MEASurement<Instance>:BHRate:TRACe:SGACp[:PTX]
value: List[float] = driver.bluetooth.measurement.bhRate.trace.sgacp.ptx.read()
```

No command help available

Suppressed linked return values: reliability

return
sgacp_trace: No help available

6.1.1.2 DtMode

class DtModeCls

DtMode commands group definition. 26 total commands, 1 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.bluetooth.measurement.dtMode.clone()
```

Subgroups

6.1.1.2.1 RxQuality

class RxQualityCls

RxQuality commands group definition. 26 total commands, 2 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.bluetooth.measurement.dtMode.rxQuality.clone()
```

Subgroups

6.1.1.2.1.1 Per

SCPI Commands :

```
INITiate:BLUetooth:MEASurement<Instance>:DTMode:RXQuality:PER
STOP:BLUetooth:MEASurement<Instance>:DTMode:RXQuality:PER
ABORt:BLUetooth:MEASurement<Instance>:DTMode:RXQuality:PER
```

class PerCls

Per commands group definition. 13 total commands, 2 Subgroups, 3 group commands

abort(*opc_timeout_ms: int = -1*) → None

```
# SCPI: ABORt:BLUetooth:MEASurement<Instance>:DTMode:RXQuality:PER
driver.bluetooth.measurement.dtMode.rxQuality.per.abort()
```

No command help available

param opc_timeout_ms
Maximum time to wait in milliseconds, valid only for this call.

initiate(*opc_timeout_ms: int = -1*) → None

```
# SCPI: INITiate:BLUetooth:MEASurement<Instance>:DTMode:RXQuality:PER
driver.bluetooth.measurement.dtMode.rxQuality.per.initiate()
```

No command help available

param opc_timeout_ms
Maximum time to wait in milliseconds, valid only for this call.

stop(*opc_timeout_ms: int = -1*) → None

```
# SCPI: STOP:BLUetooth:MEASurement<Instance>:DTMode:RXQuality:PER
driver.bluetooth.measurement.dtMode.rxQuality.per.stop()
```

No command help available

param opc_timeout_ms
Maximum time to wait in milliseconds, valid only for this call.

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.bluetooth.measurement.dtMode.rxQuality.per.clone()
```

Subgroups

6.1.1.2.1.2 LowEnergy

class LowEnergyCls

LowEnergy commands group definition. 9 total commands, 3 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.bluetooth.measurement.dtMode.rxQuality.per.lowEnergy.clone()
```

Subgroups

6.1.1.2.1.3 Le1M

SCPI Commands :

```
READ:BLUetooth:MEASurement<Instance>:DTMode:RXQuality:PER:LENergy:LE1M
FETCh:BLUetooth:MEASurement<Instance>:DTMode:RXQuality:PER:LENergy:LE1M
CALCulate:BLUetooth:MEASurement<Instance>:DTMode:RXQuality:PER:LENergy:LE1M
```

class Le1MCls

Le1M commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Per: float or bool: No parameter help available
- Packets_Received: float or bool: No parameter help available

class ResultData

Response structure. Fields:

- Reliability: int: No parameter help available
- Per: float: No parameter help available
- Packets_Received: int: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>
↪:DTMode:RXQuality:PER:LENergy:LE1M
value: CalculateStruct = driver.bluetooth.measurement.dtMode.rxQuality.per.
↪lowEnergy.le1M.calculate()
```

No command help available

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → ResultData

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:DTMode:RXQuality:PER:LENergy:LE1M
value: ResultData = driver.bluetooth.measurement.dtMode.rxQuality.per.lowEnergy.
↪le1M.fetch()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

read() → ResultData

```
# SCPI: READ:BLUetooth:MEASurement<Instance>:DTMode:RXQuality:PER:LENergy:LE1M
value: ResultData = driver.bluetooth.measurement.dtMode.rxQuality.per.lowEnergy.
↪le1M.read()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

6.1.1.2.1.4 Le2M

SCPI Commands :

```
READ:BLUetooth:MEASurement<Instance>:DTMode:RXQuality:PER:LENergy:LE2M
FETCh:BLUetooth:MEASurement<Instance>:DTMode:RXQuality:PER:LENergy:LE2M
CALCulate:BLUetooth:MEASurement<Instance>:DTMode:RXQuality:PER:LENergy:LE2M
```

class Le2MCls

Le2M commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Per: float or bool: No parameter help available
- Packets_Received: float or bool: No parameter help available

class ResultData

Response structure. Fields:

- Reliability: int: No parameter help available
- Per: float: No parameter help available
- Packets_Received: int: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>
↪:DTMode:RXQuality:PER:LENergy:LE2M
value: CalculateStruct = driver.bluetooth.measurement.dtMode.rxQuality.per.
↪lowEnergy.le2M.calculate()
```

No command help available

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → ResultData

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:DTMode:RXQuality:PER:LENergy:LE2M
value: ResultData = driver.bluetooth.measurement.dtMode.rxQuality.per.lowEnergy.
↪le2M.fetch()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

read() → ResultData

```
# SCPI: READ:BLUetooth:MEASurement<Instance>:DTMode:RXQuality:PER:LENergy:LE2M
value: ResultData = driver.bluetooth.measurement.dtMode.rxQuality.per.lowEnergy.
↪le2M.read()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

6.1.1.2.1.5 Lrange

SCPI Commands :

```
READ:BLUetooth:MEASurement<Instance>:DTMode:RXQuality:PER:LENergy:LRANge
FETCh:BLUetooth:MEASurement<Instance>:DTMode:RXQuality:PER:LENergy:LRANge
CALCulate:BLUetooth:MEASurement<Instance>:DTMode:RXQuality:PER:LENergy:LRANge
```

class LrangeCls

Lrange commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Per: float or bool: No parameter help available
- Packets_Received: float or bool: No parameter help available

class ResultData

Response structure. Fields:

- Reliability: int: No parameter help available
- Per: float: No parameter help available
- Packets_Received: int: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>
↪:DTMode:RXQuality:PER:LENergy:LRANge
value: CalculateStruct = driver.bluetooth.measurement.dtMode.rxQuality.per.
↪lowEnergy.lrange.calculate()
```

No command help available

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → ResultData

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:DTMode:RXQuality:PER:LENergy:LRANge
value: ResultData = driver.bluetooth.measurement.dtMode.rxQuality.per.lowEnergy.
↪lrange.fetch()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

read() → ResultData

```
# SCPI: READ:BLUetooth:MEASurement<Instance>:DTMode:RXQuality:PER:LENergy:LRANge
value: ResultData = driver.bluetooth.measurement.dtMode.rxQuality.per.lowEnergy.
↪lrange.read()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

6.1.1.2.1.6 State

SCPI Command :

```
FETCh:BLUetooth:MEASurement<Instance>:DTMode:RXQuality:PER:STATe
```

class StateCls

State commands group definition. 1 total commands, 0 Subgroups, 1 group commands

fetch(*timeout: float = None*, *target_main_state: ResourceState = None*, *target_sync_state: SyncState = None*) → ResourceState

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:DTMode:RXQuality:PER:STATe
value: enums.ResourceState = driver.bluetooth.measurement.dtMode.rxQuality.per.
↪state.fetch(timeout = 1.0, target_main_state = enums.ResourceState.ACTive,␣
↪target_sync_state = enums.SyncState.ADJusted)
```

No command help available

param timeout
No help available

param target_main_state
No help available

param target_sync_state
No help available

return
meas_status: No help available

6.1.1.2.1.7 Search

class SearchCls

Search commands group definition. 13 total commands, 1 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.bluetooth.measurement.dtMode.rxQuality.search.clone()
```

Subgroups

6.1.1.2.1.8 Per

SCPI Commands :

```
INITiate:BLUetooth:MEASurement<Instance>:DTMode:RXQuality:SEARch:PER
STOP:BLUetooth:MEASurement<Instance>:DTMode:RXQuality:SEARch:PER
ABORt:BLUetooth:MEASurement<Instance>:DTMode:RXQuality:SEARch:PER
```

class PerCls

Per commands group definition. 13 total commands, 2 Subgroups, 3 group commands

abort(*opc_timeout_ms: int = -1*) → None

```
# SCPI: ABORt:BLUetooth:MEASurement<Instance>:DTMode:RXQuality:SEARch:PER
driver.bluetooth.measurement.dtMode.rxQuality.search.per.abort()
```

No command help available

param opc_timeout_ms
Maximum time to wait in milliseconds, valid only for this call.

initiate(*opc_timeout_ms: int = -1*) → None

```
# SCPI: INITiate:BLUetooth:MEASurement<Instance>:DTMode:RXQuality:SEARch:PER
driver.bluetooth.measurement.dtMode.rxQuality.search.per.initiate()
```

No command help available

param opc_timeout_ms
Maximum time to wait in milliseconds, valid only for this call.

stop(*opc_timeout_ms: int = -1*) → None

```
# SCPI: STOP:BLUetooth:MEASurement<Instance>:DTMode:RXQuality:SEARch:PER
driver.bluetooth.measurement.dtMode.rxQuality.search.per.stop()
```

No command help available

param opc_timeout_ms
Maximum time to wait in milliseconds, valid only for this call.

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.bluetooth.measurement.dtMode.rxQuality.search.per.clone()
```

Subgroups

6.1.1.2.1.9 LowEnergy

class LowEnergyCls

LowEnergy commands group definition. 9 total commands, 3 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.bluetooth.measurement.dtMode.rxQuality.search.per.lowEnergy.clone()
```

Subgroups

6.1.1.2.1.10 Le1M

SCPI Commands :

```
READ:BLUetooth:MEASurement<Instance>:DTMode:RXQuality:SEARch:PER:LENergy:LE1M
FETCh:BLUetooth:MEASurement<Instance>:DTMode:RXQuality:SEARch:PER:LENergy:LE1M
CALCulate:BLUetooth:MEASurement<Instance>:DTMode:RXQuality:SEARch:PER:LENergy:LE1M
```

class Le1MCls

Le1M commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Per: float or bool: No parameter help available
- Packets_Received: float or bool: No parameter help available
- Search_Result: float or bool: No parameter help available

class ResultData

Response structure. Fields:

- Reliability: int: No parameter help available
- Per: float: No parameter help available
- Packets_Received: int: No parameter help available
- Search_Result: float: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>
↪:DTMode:RXQuality:SEARch:PER:LENergy:LE1M
value: CalculateStruct = driver.bluetooth.measurement.dtMode.rxQuality.search.
↪per.lowEnergy.le1M.calculate()
```

No command help available

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → ResultData

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:DTMode:RXQuality:SEARch:PER:LENergy:LE1M
value: ResultData = driver.bluetooth.measurement.dtMode.rxQuality.search.per.
↪lowEnergy.le1M.fetch()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

read() → ResultData

```
# SCPI: READ:BLUetooth:MEASurement<Instance>
↪:DTMode:RXQuality:SEARch:PER:LENergy:LE1M
value: ResultData = driver.bluetooth.measurement.dtMode.rxQuality.search.per.
↪lowEnergy.le1M.read()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

6.1.1.2.1.11 Le2M

SCPI Commands :

```
READ:BLUetooth:MEASurement<Instance>:DTMode:RXQuality:SEARch:PER:LENergy:LE2M
FETCh:BLUetooth:MEASurement<Instance>:DTMode:RXQuality:SEARch:PER:LENergy:LE2M
CALCulate:BLUetooth:MEASurement<Instance>:DTMode:RXQuality:SEARch:PER:LENergy:LE2M
```

class Le2MCls

Le2M commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Per: float or bool: No parameter help available
- Packets_Received: float or bool: No parameter help available
- Search_Result: float or bool: No parameter help available

class ResultData

Response structure. Fields:

- Reliability: int: No parameter help available
- Per: float: No parameter help available
- Packets_Received: int: No parameter help available
- Search_Result: float: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>
↪:DTMode:RXQuality:SEARch:PER:LENergy:LE2M
value: CalculateStruct = driver.bluetooth.measurement.dtMode.rxQuality.search.
↪per.lowEnergy.le2M.calculate()
```

No command help available

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → ResultData

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:DTMode:RXQuality:SEARch:PER:LENergy:LE2M
value: ResultData = driver.bluetooth.measurement.dtMode.rxQuality.search.per.
↪lowEnergy.le2M.fetch()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

read() → ResultData

```
# SCPI: READ:BLUetooth:MEASurement<Instance>
↪:DTMode:RXQuality:SEARch:PER:LENergy:LE2M
value: ResultData = driver.bluetooth.measurement.dtMode.rxQuality.search.per.
↪lowEnergy.le2M.read()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

6.1.1.2.1.12 Lrange

SCPI Commands :

```
READ:BLUetooth:MEASurement<Instance>:DTMode:RXQuality:SEARch:PER:LENergy:LRANge
FETCh:BLUetooth:MEASurement<Instance>:DTMode:RXQuality:SEARch:PER:LENergy:LRANge
CALCulate:BLUetooth:MEASurement<Instance>:DTMode:RXQuality:SEARch:PER:LENergy:LRANge
```

class LrangeCls

Lrange commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Per: float or bool: No parameter help available
- Packets_Received: float or bool: No parameter help available
- Search_Result: float or bool: No parameter help available

class ResultData

Response structure. Fields:

- Reliability: int: No parameter help available
- Per: float: No parameter help available
- Packets_Received: int: No parameter help available
- Search_Result: float: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>
↪:DTMode:RXQuality:SEARch:PER:LENergy:LRANge
value: CalculateStruct = driver.bluetooth.measurement.dtMode.rxQuality.search.
↪per.lowEnergy.lrange.calculate()
```

No command help available

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → ResultData

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:DTMode:RXQuality:SEARch:PER:LENergy:LRANge
value: ResultData = driver.bluetooth.measurement.dtMode.rxQuality.search.per.
↪lowEnergy.lrange.fetch()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

read() → ResultData

```
# SCPI: READ:BLUetooth:MEASurement<Instance>
↪:DTMode:RXQuality:SEARch:PER:LENergy:LRANge
value: ResultData = driver.bluetooth.measurement.dtMode.rxQuality.search.per.
↪lowEnergy.lrange.read()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

6.1.1.2.1.13 State

SCPI Command :

```
FETCh:BLUetooth:MEASurement<Instance>:DTMode:RXQuality:SEARch:PER:STATe
```

class StateCls

State commands group definition. 1 total commands, 0 Subgroups, 1 group commands

fetch(*timeout: float = None*, *target_main_state: ResourceState = None*, *target_sync_state: SyncState = None*) → ResourceState

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:DTMode:RXQuality:SEARch:PER:STATe
value: enums.ResourceState = driver.bluetooth.measurement.dtMode.rxQuality.
↪search.per.state.fetch(timeout = 1.0, target_main_state = enums.ResourceState.
↪ACTive, target_sync_state = enums.SyncState.ADJusted)
```

No command help available

param timeout
No help available

param target_main_state
No help available

param target_sync_state
No help available

return
meas_status: No help available

6.1.1.3 Hdr

SCPI Commands :

```
ABORt:BLUetooth:MEASurement<Instance>:HDR
STOP:BLUetooth:MEASurement<Instance>:HDR
INITiate:BLUetooth:MEASurement<Instance>:HDR
```

class HdrCls

Hdr commands group definition. 74 total commands, 7 Subgroups, 3 group commands

abort() → None

```
# SCPI: ABORt:BLUetooth:MEASurement<Instance>:HDR
driver.bluetooth.measurement.hdr.abort()
```

No command help available

abort_with_opc(*opc_timeout_ms: int = -1*) → None

```
# SCPI: ABORt:BLUetooth:MEASurement<Instance>:HDR
driver.bluetooth.measurement.hdr.abort_with_opc()
```

No command help available

Same as abort, but waits for the operation to complete before continuing further. Use the RsCMPX_BluetoothMeas.utilities.opc_timeout_set() to set the timeout value.

param opc_timeout_ms
Maximum time to wait in milliseconds, valid only for this call.

initiate() → None

```
# SCPI: INITiate:BLUetooth:MEASurement<Instance>:HDR
driver.bluetooth.measurement.hdr.initiate()
```

No command help available

initiate_with_opc(*opc_timeout_ms: int = -1*) → None

```
# SCPI: INITiate:BLUetooth:MEASurement<Instance>:HDR
driver.bluetooth.measurement.hdr.initiate_with_opc()
```

No command help available

Same as initiate, but waits for the operation to complete before continuing further. Use the RsCMPX_BluetoothMeas.utilities.opc_timeout_set() to set the timeout value.

param opc_timeout_ms
Maximum time to wait in milliseconds, valid only for this call.

stop() → None

```
# SCPI: STOP:BLUetooth:MEASurement<Instance>:HDR
driver.bluetooth.measurement.hdr.stop()
```

No command help available

stop_with_opc(*opc_timeout_ms: int = -1*) → None

```
# SCPI: STOP:BLUetooth:MEASurement<Instance>:HDR
driver.bluetooth.measurement.hdr.stop_with_opc()
```

No command help available

Same as stop, but waits for the operation to complete before continuing further. Use the RsCMPX_BluetoothMeas.utilities.opc_timeout_set() to set the timeout value.

param opc_timeout_ms
Maximum time to wait in milliseconds, valid only for this call.

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.bluetooth.measurement.hdr.clone()
```

Subgroups

6.1.1.3.1 InputSignal

class InputSignalCls

InputSignal commands group definition. 2 total commands, 1 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.bluetooth.measurement.hdr.inputSignal.clone()
```

Subgroups

6.1.1.3.1.1 Adetected

class AdetectedCls

Adetected commands group definition. 2 total commands, 2 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.bluetooth.measurement.hdr.inputSignal.adetected.clone()
```

Subgroups

6.1.1.3.1.2 Plength

SCPI Command :

```
FETCh:BLUetooth:MEASurement<Instance>:HDR:ISIGnal:ADETected:PLENgth
```

class PlengthCls

Plength commands group definition. 1 total commands, 0 Subgroups, 1 group commands

fetch() → int

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:HDR:ISIGnal:ADETected:PLENgth
value: int = driver.bluetooth.measurement.hdr.inputSignal.adetected.plength.
↪fetch()
```

No command help available

Suppressed linked return values: reliability

return
payload_length: No help available

6.1.1.3.1.3 Ptype

SCPI Command :

```
FETCh:BLUetooth:MEASurement<Instance>:HDR:ISIGnal:ADETected:PTYPe
```

class PtypeCls

Ptype commands group definition. 1 total commands, 0 Subgroups, 1 group commands

fetch() → PacketTypeC

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:HDR:ISIGnal:ADETected:PTYPe
value: enums.PacketTypeC = driver.bluetooth.measurement.hdr.inputSignal.
↪adetected.ptype.fetch()
```

No command help available

Suppressed linked return values: reliability

return
packet_type: No help available

6.1.1.3.2 Modulation

class ModulationCls

Modulation commands group definition. 12 total commands, 4 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.bluetooth.measurement.hdr.modulation.clone()
```

Subgroups

6.1.1.3.2.1 Average

SCPI Commands :

```
READ:BLUetooth:MEASurement<Instance>:HDR:MODulation:AVERage
FETCh:BLUetooth:MEASurement<Instance>:HDR:MODulation:AVERage
CALCulate:BLUetooth:MEASurement<Instance>:HDR:MODulation:AVERage
```

class AverageCls

Average commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Bursts_Out_Of_Tol: float or bool: No parameter help available

- Wi: float or bool: No parameter help available
- W_0_Wi: float or bool: No parameter help available
- W_0_Max: float or bool: No parameter help available
- Rms_Devm: float or bool: No parameter help available
- Peak_Devm: float or bool: No parameter help available
- Nominal_Power: float or bool: No parameter help available

class FetchStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Bursts_Out_Of_Tol: float: No parameter help available
- Wi: float: No parameter help available
- W_0_Wi: float: No parameter help available
- W_0_Max: float: No parameter help available
- Rms_Devm: float: No parameter help available
- Peak_Devm: float: No parameter help available
- Nominal_Power: float: No parameter help available

class ReadStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Bursts_Out_Of_Tol: int: No parameter help available
- Wi: float: No parameter help available
- W_0_Wi: float: No parameter help available
- W_0_Max: float: No parameter help available
- Rms_Devm: float: No parameter help available
- Peak_Devm: float: No parameter help available
- Nominal_Power: float: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>:HDR:MODulation:AVERage
value: CalculateStruct = driver.bluetooth.measurement.hdr.modulation.average.
↪calculate()
```

No command help available

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → FetchStruct

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:HDR:MODulation:AVERage
value: FetchStruct = driver.bluetooth.measurement.hdr.modulation.average.fetch()
```

No command help available

return
structure: for return value, see the help for FetchStruct structure arguments.

read() → ReadStruct

```
# SCPI: READ:BLUetooth:MEASurement<Instance>:HDR:MODulation:AVERage
value: ReadStruct = driver.bluetooth.measurement.hdr.modulation.average.read()
```

No command help available

return
structure: for return value, see the help for ReadStruct structure arguments.

6.1.1.3.2.2 Current

SCPI Commands :

```
READ:BLUetooth:MEASurement<Instance>:HDR:MODulation:CURRent
FETCh:BLUetooth:MEASurement<Instance>:HDR:MODulation:CURRent
CALCulate:BLUetooth:MEASurement<Instance>:HDR:MODulation:CURRent
```

class CurrentCls

Current commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Bursts_Out_Of_Tol: float or bool: No parameter help available
- Wi: float or bool: No parameter help available
- W_0_Wi: float or bool: No parameter help available
- W_0_Max: float or bool: No parameter help available
- Rms_Devm: float or bool: No parameter help available
- Peak_Devm: float or bool: No parameter help available
- P_99_Devm: float or bool: No parameter help available
- Nominal_Power: float or bool: No parameter help available

class ResultData

Response structure. Fields:

- Reliability: int: No parameter help available
- Bursts_Out_Of_Tol: float: No parameter help available
- Wi: float: No parameter help available
- W_0_Wi: float: No parameter help available
- W_0_Max: float: No parameter help available
- Rms_Devm: float: No parameter help available

- Peak_Devm: float: No parameter help available
- P_99_Devm: float: No parameter help available
- Nominal_Power: float: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>:HDR:MODulation:CURRent
value: CalculateStruct = driver.bluetooth.measurement.hdr.modulation.current.
↪calculate()
```

No command help available

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → ResultData

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:HDR:MODulation:CURRent
value: ResultData = driver.bluetooth.measurement.hdr.modulation.current.fetch()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

read() → ResultData

```
# SCPI: READ:BLUetooth:MEASurement<Instance>:HDR:MODulation:CURRent
value: ResultData = driver.bluetooth.measurement.hdr.modulation.current.read()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

6.1.1.3.2.3 Maximum

SCPI Commands :

```
READ:BLUetooth:MEASurement<Instance>:HDR:MODulation:MAXimum
FETCh:BLUetooth:MEASurement<Instance>:HDR:MODulation:MAXimum
CALCulate:BLUetooth:MEASurement<Instance>:HDR:MODulation:MAXimum
```

class MaximumCls

Maximum commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Bursts_Out_Of_Tol: float or bool: No parameter help available
- Wi: float or bool: No parameter help available
- W_0_Wi: float or bool: No parameter help available

- W_0_Max: float or bool: No parameter help available
- Rms_Devm: float or bool: No parameter help available
- Peak_Devm: float or bool: No parameter help available
- Nominal_Power: float or bool: No parameter help available

class ResultData

Response structure. Fields:

- Reliability: int: No parameter help available
- Bursts_Out_Of_Tol: float: No parameter help available
- Wi: float: No parameter help available
- W_0_Wi: float: No parameter help available
- W_0_Max: float: No parameter help available
- Rms_Devm: float: No parameter help available
- Peak_Devm: float: No parameter help available
- Nominal_Power: float: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>:HDR:MODulation:MAXimum
value: CalculateStruct = driver.bluetooth.measurement.hdr.modulation.maximum.
↪calculate()
```

No command help available

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → ResultData

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:HDR:MODulation:MAXimum
value: ResultData = driver.bluetooth.measurement.hdr.modulation.maximum.fetch()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

read() → ResultData

```
# SCPI: READ:BLUetooth:MEASurement<Instance>:HDR:MODulation:MAXimum
value: ResultData = driver.bluetooth.measurement.hdr.modulation.maximum.read()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

6.1.1.3.2.4 StandardDev

SCPI Commands :

```
READ:BLUetooth:MEASurement<Instance>:HDR:MODulation:SDEViation
FETCh:BLUetooth:MEASurement<Instance>:HDR:MODulation:SDEViation
CALCulate:BLUetooth:MEASurement<Instance>:HDR:MODulation:SDEViation
```

class StandardDevCls

StandardDev commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Bursts_Out_Of_Tol: float or bool: No parameter help available
- Nominal_Power: float: No parameter help available
- Wi: float or bool: No parameter help available
- W_0_Wi: float or bool: No parameter help available
- W_0_Max: float or bool: No parameter help available
- Rms_Devm: float or bool: No parameter help available
- Peak_Devm: float or bool: No parameter help available

class FetchStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Bursts_Out_Of_Tol: float: No parameter help available
- Nominal_Power: float: No parameter help available
- Wi: float: No parameter help available
- W_0_Wi: float: No parameter help available
- W_0_Max: float: No parameter help available
- Rms_Devm: float: No parameter help available
- Peak_Devm: float: No parameter help available

class ReadStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Bursts_Out_Of_Tol: float: No parameter help available
- Wi: float: No parameter help available
- W_0_Wi: float: No parameter help available
- W_0_Max: float: No parameter help available
- Rms_Devm: float: No parameter help available
- Peak_Devm: float: No parameter help available

- Nominal_Power: float: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>:HDR:MODulation:SDEViation
value: CalculateStruct = driver.bluetooth.measurement.hdr.modulation.
↪standardDev.calculate()
```

No command help available

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → FetchStruct

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:HDR:MODulation:SDEViation
value: FetchStruct = driver.bluetooth.measurement.hdr.modulation.standardDev.
↪fetch()
```

No command help available

return
structure: for return value, see the help for FetchStruct structure arguments.

read() → ReadStruct

```
# SCPI: READ:BLUetooth:MEASurement<Instance>:HDR:MODulation:SDEViation
value: ReadStruct = driver.bluetooth.measurement.hdr.modulation.standardDev.
↪read()
```

No command help available

return
structure: for return value, see the help for ReadStruct structure arguments.

6.1.1.3.3 Pencoding

class PencodingCls

Pencoding commands group definition. 6 total commands, 2 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.bluetooth.measurement.hdr.pencoding.clone()
```

Subgroups

6.1.1.3.3.1 Current

SCPI Commands :

```
READ:BLUetooth:MEASurement<Instance>:HDR:PENCoding:CURRent
FETCh:BLUetooth:MEASurement<Instance>:HDR:PENCoding:CURRent
CALCulate:BLUetooth:MEASurement<Instance>:HDR:PENCoding:CURRent
```

class CurrentCls

Current commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Nominal_Power: float or bool: No parameter help available
- Bit_Error_Rate: float or bool: No parameter help available
- Packets_0_Errors: float or bool: No parameter help available

class ResultData

Response structure. Fields:

- Reliability: int: No parameter help available
- Nominal_Power: float: No parameter help available
- Bit_Error_Rate: float: No parameter help available
- Packets_0_Errors: float: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>:HDR:PENCoding:CURRent
value: CalculateStruct = driver.bluetooth.measurement.hdr.pencoding.current.
↪calculate()
```

No command help available

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → ResultData

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:HDR:PENCoding:CURRent
value: ResultData = driver.bluetooth.measurement.hdr.pencoding.current.fetch()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

read() → ResultData

```
# SCPI: READ:BLUetooth:MEASurement<Instance>:HDR:PENCoding:CURRent
value: ResultData = driver.bluetooth.measurement.hdr.pencoding.current.read()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

6.1.1.3.3.2 Ssequence

class SsequenceCls

Ssequence commands group definition. 3 total commands, 1 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.bluetooth.measurement.hdr.pencoding.ssequence.clone()
```

Subgroups

6.1.1.3.3.3 Current

SCPI Commands :

```
READ:BLUetooth:MEASurement<Instance>:HDR:PENCoding:SSEQuence:CURRent
FETCh:BLUetooth:MEASurement<Instance>:HDR:PENCoding:SSEQuence:CURRent
CALCulate:BLUetooth:MEASurement<Instance>:HDR:PENCoding:SSEQuence:CURRent
```

class CurrentCls

Current commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Nominal_Power: float or bool: No parameter help available
- Sync_Bit_Errors: float or bool: No parameter help available
- Trailer_Bit_Error: float or bool: No parameter help available

class ResultData

Response structure. Fields:

- Reliability: int: No parameter help available
- Nominal_Power: float: No parameter help available
- Sync_Bit_Errors: int: No parameter help available
- Trailer_Bit_Error: int: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>
↪:HDR:PENCoding:SSEQuence:CURRent
value: CalculateStruct = driver.bluetooth.measurement.hdr.pencoding.ssequence.
↪current.calculate()
```

No command help available

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → ResultData

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:HDR:PENCoding:SSEQuence:CURRent
value: ResultData = driver.bluetooth.measurement.hdr.pencoding.ssequence.
↪current.fetch()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

read() → ResultData

```
# SCPI: READ:BLUetooth:MEASurement<Instance>:HDR:PENCoding:SSEQuence:CURRent
value: ResultData = driver.bluetooth.measurement.hdr.pencoding.ssequence.
↪current.read()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

6.1.1.3.4 PowerVsTime

class PowerVsTimeCls

PowerVsTime commands group definition. 12 total commands, 4 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.bluetooth.measurement.hdr.powerVsTime.clone()
```

Subgroups

6.1.1.3.4.1 Average

SCPI Commands :

```
READ:BLUetooth:MEASurement<Instance>:HDR:PVTime:AVERage
FETCh:BLUetooth:MEASurement<Instance>:HDR:PVTime:AVERage
CALCulate:BLUetooth:MEASurement<Instance>:HDR:PVTime:AVERage
```

class AverageCls

Average commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Bursts_Out_Of_Tol: float or bool: No parameter help available
- Nominal_Power: float or bool: No parameter help available
- Gfsk_Power: float or bool: No parameter help available
- Dpsk_Power: float or bool: No parameter help available
- Dpsk_Gfsk_Power: float or bool: No parameter help available
- Guard_Period: float or bool: No parameter help available

class ResultData

Response structure. Fields:

- Reliability: int: No parameter help available
- Bursts_Out_Of_Tol: float: No parameter help available
- Nominal_Power: float: No parameter help available
- Gfsk_Power: float: No parameter help available
- Dpsk_Power: float: No parameter help available
- Dpsk_Gfsk_Power: float: No parameter help available
- Guard_Period: float: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>:HDR:PVTime:AVERage
value: CalculateStruct = driver.bluetooth.measurement.hdr.powerVsTime.average.
↪calculate()
```

No command help available

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → ResultData

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:HDR:PVTime:AVERage
value: ResultData = driver.bluetooth.measurement.hdr.powerVsTime.average.fetch()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

read() → ResultData

```
# SCPI: READ:BLUetooth:MEASurement<Instance>:HDR:PVTime:AVERage
value: ResultData = driver.bluetooth.measurement.hdr.powerVsTime.average.read()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

6.1.1.3.4.2 Current

SCPI Commands :

```
READ:BLUetooth:MEASurement<Instance>:HDR:PVTime:CURRent
FETCh:BLUetooth:MEASurement<Instance>:HDR:PVTime:CURRent
CALCulate:BLUetooth:MEASurement<Instance>:HDR:PVTime:CURRent
```

class CurrentCls

Current commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Bursts_Out_Of_Tol: float or bool: No parameter help available
- Nominal_Power: float or bool: No parameter help available
- Gfsk_Power: float or bool: No parameter help available
- Dpsk_Power: float or bool: No parameter help available
- Dpsk_Gfsk_Power: float or bool: No parameter help available
- Guard_Period: float or bool: No parameter help available

class ResultData

Response structure. Fields:

- Reliability: int: No parameter help available
- Bursts_Out_Of_Tol: float: No parameter help available
- Nominal_Power: float: No parameter help available
- Gfsk_Power: float: No parameter help available
- Dpsk_Power: float: No parameter help available
- Dpsk_Gfsk_Power: float: No parameter help available
- Guard_Period: float: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>:HDR:PVTime:CURRent
value: CalculateStruct = driver.bluetooth.measurement.hdr.powerVsTime.current.
↪calculate()
```

No command help available

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → ResultData

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:HDR:PVTime:CURRent
value: ResultData = driver.bluetooth.measurement.hdr.powerVsTime.current.fetch()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

read() → ResultData

```
# SCPI: READ:BLUetooth:MEASurement<Instance>:HDR:PVTime:CURRent
value: ResultData = driver.bluetooth.measurement.hdr.powerVsTime.current.read()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

6.1.1.3.4.3 Maximum

SCPI Commands :

```
READ:BLUetooth:MEASurement<Instance>:HDR:PVTime:MAXimum
FETCh:BLUetooth:MEASurement<Instance>:HDR:PVTime:MAXimum
CALCulate:BLUetooth:MEASurement<Instance>:HDR:PVTime:MAXimum
```

class MaximumCls

Maximum commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Bursts_Out_Of_Tol: float or bool: No parameter help available
- Nominal_Power: float or bool: No parameter help available
- Gfsk_Power: float or bool: No parameter help available
- Dpsk_Power: float or bool: No parameter help available
- Dpsk_Gfsk_Power: float or bool: No parameter help available
- Guard_Period: float or bool: No parameter help available

class ResultData

Response structure. Fields:

- Reliability: int: No parameter help available
- Bursts_Out_Of_Tol: float: No parameter help available
- Nominal_Power: float: No parameter help available
- Gfsk_Power: float: No parameter help available
- Dpsk_Power: float: No parameter help available

- Dpsk_Gfsk_Power: float: No parameter help available
- Guard_Period: float: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>:HDR:PVTime:MAXimum
value: CalculateStruct = driver.bluetooth.measurement.hdr.powerVsTime.maximum.
↪calculate()
```

No command help available

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → ResultData

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:HDR:PVTime:MAXimum
value: ResultData = driver.bluetooth.measurement.hdr.powerVsTime.maximum.fetch()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

read() → ResultData

```
# SCPI: READ:BLUetooth:MEASurement<Instance>:HDR:PVTime:MAXimum
value: ResultData = driver.bluetooth.measurement.hdr.powerVsTime.maximum.read()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

6.1.1.3.4.4 Minimum

SCPI Commands :

```
READ:BLUetooth:MEASurement<Instance>:HDR:PVTime:MINimum
FETCh:BLUetooth:MEASurement<Instance>:HDR:PVTime:MINimum
CALCulate:BLUetooth:MEASurement<Instance>:HDR:PVTime:MINimum
```

class MinimumCls

Minimum commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Bursts_Out_Of_Tol: float or bool: No parameter help available
- Nominal_Power: float or bool: No parameter help available
- Gfsk_Power: float or bool: No parameter help available
- Dpsk_Power: float or bool: No parameter help available

- Dpsk_Gfsk_Power: float or bool: No parameter help available
- Guard_Period: float or bool: No parameter help available

class ResultData

Response structure. Fields:

- Reliability: int: No parameter help available
- Bursts_Out_Of_Tol: float: No parameter help available
- Nominal_Power: float: No parameter help available
- Gfsk_Power: float: No parameter help available
- Dpsk_Power: float: No parameter help available
- Dpsk_Gfsk_Power: float: No parameter help available
- Guard_Period: float: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>:HDR:PVTime:MINimum
value: CalculateStruct = driver.bluetooth.measurement.hdr.powerVsTime.minimum.
↪calculate()
```

No command help available

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → ResultData

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:HDR:PVTime:MINimum
value: ResultData = driver.bluetooth.measurement.hdr.powerVsTime.minimum.fetch()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

read() → ResultData

```
# SCPI: READ:BLUetooth:MEASurement<Instance>:HDR:PVTime:MINimum
value: ResultData = driver.bluetooth.measurement.hdr.powerVsTime.minimum.read()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

6.1.1.3.5 Sgacp

class SgacpCls

Sgacp commands group definition. 3 total commands, 1 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.bluetooth.measurement.hdr.sgacp.clone()
```

Subgroups

6.1.1.3.5.1 Current

SCPI Commands :

```
READ:BLUetooth:MEASurement<Instance>:HDR:SGACp[:CURRent]
FETCh:BLUetooth:MEASurement<Instance>:HDR:SGACp[:CURRent]
CALCulate:BLUetooth:MEASurement<Instance>:HDR:SGACp[:CURRent]
```

class CurrentCls

Current commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Bursts_Out_Of_Tol: float or bool: No parameter help available
- Nominal_Power: float or bool: No parameter help available
- Ptx_Ref: float or bool: No parameter help available

class ResultData

Response structure. Fields:

- Reliability: int: No parameter help available
- Bursts_Out_Of_Tol: float: No parameter help available
- Nominal_Power: float: No parameter help available
- Ptx_Ref: float: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>:HDR:SGACp[:CURRent]
value: CalculateStruct = driver.bluetooth.measurement.hdr.sgacp.current.
↪calculate()
```

No command help available

return

structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → ResultData

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:HDR:SGACp[:CURRent]
value: ResultData = driver.bluetooth.measurement.hdr.sgacp.current.fetch()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

read() → ResultData

```
# SCPI: READ:BLUetooth:MEASurement<Instance>:HDR:SGACp[:CURRent]
value: ResultData = driver.bluetooth.measurement.hdr.sgacp.current.read()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

6.1.1.3.6 State

SCPI Command :

```
FETCh:BLUetooth:MEASurement<Instance>:HDR:STATe
```

class StateCls

State commands group definition. 2 total commands, 1 Subgroups, 1 group commands

fetch(*timeout: float = None*, *target_main_state: ResourceState = None*, *target_sync_state: SyncState = None*) → ResourceState

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:HDR:STATe
value: enums.ResourceState = driver.bluetooth.measurement.hdr.state.
↪fetch(timeout = 1.0, target_main_state = enums.ResourceState.ACTive, target_
↪sync_state = enums.SyncState.ADJusted)
```

No command help available

param timeout
No help available

param target_main_state
No help available

param target_sync_state
No help available

return
meas_state: No help available

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.bluetooth.measurement.hdr.state.clone()
```

Subgroups

6.1.1.3.6.1 All

SCPI Command :

```
FETCh:BLUetooth:MEASurement<Instance>:HDR:STATe:ALL
```

class AllCls

All commands group definition. 1 total commands, 0 Subgroups, 1 group commands

fetch(*timeout: float = None*, *target_main_state: ResourceState = None*, *target_sync_state: SyncState = None*) → ResourceState

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:HDR:STATe:ALL
value: enums.ResourceState = driver.bluetooth.measurement.hdr.state.all.
↪fetch(timeout = 1.0, target_main_state = enums.ResourceState.ACTive, target_
↪sync_state = enums.SyncState.ADJusted)
```

No command help available

param timeout
No help available

param target_main_state
No help available

param target_sync_state
No help available

return
meas_states: No help available

6.1.1.3.7 Trace

class TraceCls

Trace commands group definition. 34 total commands, 7 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.bluetooth.measurement.hdr.trace.clone()
```

Subgroups

6.1.1.3.7.1 DevMagnitude

class DevMagnitudeCls

DevMagnitude commands group definition. 6 total commands, 3 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.bluetooth.measurement.hdr.trace.devMagnitude.clone()
```

Subgroups

6.1.1.3.7.2 Average

SCPI Commands :

```
READ:BLUetooth:MEASurement<Instance>:HDR:TRACe:DEVMagnitude:AVERage
FETCh:BLUetooth:MEASurement<Instance>:HDR:TRACe:DEVMagnitude:AVERage
```

class AverageCls

Average commands group definition. 2 total commands, 0 Subgroups, 2 group commands

fetch() → List[float]

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:HDR:TRACe:DEVMagnitude:AVERage
value: List[float] = driver.bluetooth.measurement.hdr.trace.devMagnitude.
↪average.fetch()
```

No command help available

Suppressed linked return values: reliability

return
devm: No help available

read() → List[float]

```
# SCPI: READ:BLUetooth:MEASurement<Instance>:HDR:TRACe:DEVMagnitude:AVERage
value: List[float] = driver.bluetooth.measurement.hdr.trace.devMagnitude.
↪average.read()
```

No command help available

Suppressed linked return values: reliability

return
devm: No help available

6.1.1.3.7.3 Current

SCPI Commands :

```
READ:BLUetooth:MEASurement<Instance>:HDR:TRACe:DEVMagnitude:CURRent
FETCh:BLUetooth:MEASurement<Instance>:HDR:TRACe:DEVMagnitude:CURRent
```

class CurrentCls

Current commands group definition. 2 total commands, 0 Subgroups, 2 group commands

fetch() → List[float]

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:HDR:TRACe:DEVMagnitude:CURRent
value: List[float] = driver.bluetooth.measurement.hdr.trace.devMagnitude.
↪current.fetch()
```

No command help available

Suppressed linked return values: reliability

return
devm: No help available

read() → List[float]

```
# SCPI: READ:BLUetooth:MEASurement<Instance>:HDR:TRACe:DEVMagnitude:CURRent
value: List[float] = driver.bluetooth.measurement.hdr.trace.devMagnitude.
↪current.read()
```

No command help available

Suppressed linked return values: reliability

return
devm: No help available

6.1.1.3.7.4 Maximum

SCPI Commands :

```
READ:BLUetooth:MEASurement<Instance>:HDR:TRACe:DEVMagnitude:MAXimum
FETCh:BLUetooth:MEASurement<Instance>:HDR:TRACe:DEVMagnitude:MAXimum
```

class MaximumCls

Maximum commands group definition. 2 total commands, 0 Subgroups, 2 group commands

fetch() → List[float]

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:HDR:TRACe:DEVMagnitude:MAXimum
value: List[float] = driver.bluetooth.measurement.hdr.trace.devMagnitude.
↪maximum.fetch()
```

No command help available

Suppressed linked return values: reliability

return
devm: No help available

read() → List[float]

```
# SCPI: READ:BLUetooth:MEASurement<Instance>:HDR:TRACe:DEVMagnitude:MAXimum
value: List[float] = driver.bluetooth.measurement.hdr.trace.devMagnitude.
↪maximum.read()
```

No command help available

Suppressed linked return values: reliability

return
devm: No help available

6.1.1.3.7.5 IqAbs

SCPI Commands :

```
FETCh:BLUetooth:MEASurement<Instance>:HDR:TRACe:IQABs
READ:BLUetooth:MEASurement<Instance>:HDR:TRACe:IQABs
```

class IqAbsCls

IqAbs commands group definition. 2 total commands, 0 Subgroups, 2 group commands

class ResultData

Response structure. Fields:

- Reliability: int: No parameter help available
- Iphase: List[float]: No parameter help available
- Qphase: List[float]: No parameter help available

fetch() → ResultData

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:HDR:TRACe:IQABs
value: ResultData = driver.bluetooth.measurement.hdr.trace.iqAbs.fetch()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

read() → ResultData

```
# SCPI: READ:BLUetooth:MEASurement<Instance>:HDR:TRACe:IQABs
value: ResultData = driver.bluetooth.measurement.hdr.trace.iqAbs.read()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

6.1.1.3.7.6 IqDifference

SCPI Commands :

```
FETCh:BLUetooth:MEASurement<Instance>:HDR:TRACe:IQDiff
READ:BLUetooth:MEASurement<Instance>:HDR:TRACe:IQDiff
```

class IqDifferenceCls

IqDifference commands group definition. 2 total commands, 0 Subgroups, 2 group commands

class ResultData

Response structure. Fields:

- Reliability: int: No parameter help available
- Iphase: List[float]: No parameter help available
- Qphase: List[float]: No parameter help available

fetch() → ResultData

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:HDR:TRACe:IQDiff
value: ResultData = driver.bluetooth.measurement.hdr.trace.iqDifference.fetch()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

read() → ResultData

```
# SCPI: READ:BLUetooth:MEASurement<Instance>:HDR:TRACe:IQDiff
value: ResultData = driver.bluetooth.measurement.hdr.trace.iqDifference.read()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

6.1.1.3.7.7 IqError

SCPI Commands :

```
READ:BLUetooth:MEASurement<Instance>:HDR:TRACe:IQERr
FETCh:BLUetooth:MEASurement<Instance>:HDR:TRACe:IQERr
```

class IqErrorCls

IqError commands group definition. 2 total commands, 0 Subgroups, 2 group commands

class ResultData

Response structure. Fields:

- Reliability: int: No parameter help available
- Iphase: List[float]: No parameter help available
- Qphase: List[float]: No parameter help available

fetch() → ResultData

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:HDR:TRACe:IQERr
value: ResultData = driver.bluetooth.measurement.hdr.trace.iqError.fetch()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

read() → ResultData

```
# SCPI: READ:BLUetooth:MEASurement<Instance>:HDR:TRACe:IQERr
value: ResultData = driver.bluetooth.measurement.hdr.trace.iqError.read()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

6.1.1.3.7.8 Pdifference

class PdifferenceCls

Pdifference commands group definition. 6 total commands, 3 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.bluetooth.measurement.hdr.trace.pdifference.clone()
```

Subgroups

6.1.1.3.7.9 Average

SCPI Commands :

```
READ:BLUetooth:MEASurement<Instance>:HDR:TRACe:PDIFference:AVERage
FETCh:BLUetooth:MEASurement<Instance>:HDR:TRACe:PDIFference:AVERage
```

class AverageCls

Average commands group definition. 2 total commands, 0 Subgroups, 2 group commands

fetch() → List[float]

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:HDR:TRACe:PDIFference:AVERage
value: List[float] = driver.bluetooth.measurement.hdr.trace.pdifference.average.
↪fetch()
```

No command help available

Suppressed linked return values: reliability

return
phase_difference: No help available

read() → List[float]

```
# SCPI: READ:BLUetooth:MEASurement<Instance>:HDR:TRACe:PDIFference:AVERage
value: List[float] = driver.bluetooth.measurement.hdr.trace.pdifference.average.
↪read()
```

No command help available

Suppressed linked return values: reliability

return
phase_difference: No help available

6.1.1.3.7.10 Current

SCPI Commands :

```
READ:BLUetooth:MEASurement<Instance>:HDR:TRACe:PDIFference:CURRent
FETCh:BLUetooth:MEASurement<Instance>:HDR:TRACe:PDIFference:CURRent
```

class CurrentCls

Current commands group definition. 2 total commands, 0 Subgroups, 2 group commands

fetch() → List[float]

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:HDR:TRACe:PDIFference:CURRent
value: List[float] = driver.bluetooth.measurement.hdr.trace.pdifference.current.
↪fetch()
```

No command help available

Suppressed linked return values: reliability

return
phase_difference: No help available

read() → List[float]

```
# SCPI: READ:BLUetooth:MEASurement<Instance>:HDR:TRACe:PDIFference:CURRent
value: List[float] = driver.bluetooth.measurement.hdr.trace.pdifference.current.
↪read()
```

No command help available

Suppressed linked return values: reliability

return
phase_difference: No help available

6.1.1.3.7.11 Maximum

SCPI Commands :

```
READ:BLUetooth:MEASurement<Instance>:HDR:TRACe:PDIFference:MAXimum
FETCh:BLUetooth:MEASurement<Instance>:HDR:TRACe:PDIFference:MAXimum
```

class MaximumCls

Maximum commands group definition. 2 total commands, 0 Subgroups, 2 group commands

fetch() → List[float]

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:HDR:TRACe:PDIFference:MAXimum
value: List[float] = driver.bluetooth.measurement.hdr.trace.pdifference.maximum.
↪fetch()
```

No command help available

Suppressed linked return values: reliability

return
phase_difference: No help available

read() → List[float]

```
# SCPI: READ:BLUetooth:MEASurement<Instance>:HDR:TRACe:PDIFference:MAXimum
value: List[float] = driver.bluetooth.measurement.hdr.trace.pdifference.maximum.
↪read()
```

No command help available

Suppressed linked return values: reliability

return
phase_difference: No help available

6.1.1.3.7.12 PowerVsTime

class PowerVsTimeCls

PowerVsTime commands group definition. 8 total commands, 4 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.bluetooth.measurement.hdr.trace.powerVsTime.clone()
```

Subgroups

6.1.1.3.7.13 Average

SCPI Commands :

```
READ:BLUetooth:MEASurement<Instance>:HDR:TRACe:PVTime:AVERage
FETCh:BLUetooth:MEASurement<Instance>:HDR:TRACe:PVTime:AVERage
```

class AverageCls

Average commands group definition. 2 total commands, 0 Subgroups, 2 group commands

fetch() → List[float]

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:HDR:TRACe:PVTime:AVERage
value: List[float] = driver.bluetooth.measurement.hdr.trace.powerVsTime.average.
↪fetch()
```

No command help available

Suppressed linked return values: reliability

return
pvt_trace: No help available

read() → List[float]

```
# SCPI: READ:BLUetooth:MEASurement<Instance>:HDR:TRACe:PVTime:AVERage
value: List[float] = driver.bluetooth.measurement.hdr.trace.powerVsTime.average.
↪read()
```

No command help available

Suppressed linked return values: reliability

return
pvt_trace: No help available

6.1.1.3.7.14 Current

SCPI Commands :

```
READ:BLUetooth:MEASurement<Instance>:HDR:TRACe:PVTime:CURRent
FETCh:BLUetooth:MEASurement<Instance>:HDR:TRACe:PVTime:CURRent
```

class CurrentCls

Current commands group definition. 2 total commands, 0 Subgroups, 2 group commands

fetch() → List[float]

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:HDR:TRACe:PVTime:CURRent
value: List[float] = driver.bluetooth.measurement.hdr.trace.powerVsTime.current.
↪fetch()
```

No command help available

Suppressed linked return values: reliability

return
pvt_trace: No help available

read() → List[float]

```
# SCPI: READ:BLUetooth:MEASurement<Instance>:HDR:TRACe:PVTime:CURRent
value: List[float] = driver.bluetooth.measurement.hdr.trace.powerVsTime.current.
↪read()
```

No command help available

Suppressed linked return values: reliability

return
pvt_trace: No help available

6.1.1.3.7.15 Maximum

SCPI Commands :

```
READ:BLUetooth:MEASurement<Instance>:HDR:TRACe:PVTime:MAXimum
FETCh:BLUetooth:MEASurement<Instance>:HDR:TRACe:PVTime:MAXimum
```

class MaximumCls

Maximum commands group definition. 2 total commands, 0 Subgroups, 2 group commands

fetch() → List[float]

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:HDR:TRACe:PVTime:MAXimum
value: List[float] = driver.bluetooth.measurement.hdr.trace.powerVsTime.maximum.
↪fetch()
```

No command help available

Suppressed linked return values: reliability

return
pvt_trace: No help available

read() → List[float]

```
# SCPI: READ:BLUetooth:MEASurement<Instance>:HDR:TRACe:PVTime:MAXimum
value: List[float] = driver.bluetooth.measurement.hdr.trace.powerVsTime.maximum.
↪read()
```

No command help available

Suppressed linked return values: reliability

return
pvt_trace: No help available

6.1.1.3.7.16 Minimum

SCPI Commands :

```
READ:BLUetooth:MEASurement<Instance>:HDR:TRACe:PVTime:MINimum
FETCh:BLUetooth:MEASurement<Instance>:HDR:TRACe:PVTime:MINimum
```

class MinimumCls

Minimum commands group definition. 2 total commands, 0 Subgroups, 2 group commands

fetch() → List[float]

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:HDR:TRACe:PVTime:MINimum
value: List[float] = driver.bluetooth.measurement.hdr.trace.powerVsTime.minimum.
↪fetch()
```

No command help available

Suppressed linked return values: reliability

return
pvt_trace: No help available

read() → List[float]

```
# SCPI: READ:BLUetooth:MEASurement<Instance>:HDR:TRACe:PVTime:MINimum
value: List[float] = driver.bluetooth.measurement.hdr.trace.powerVsTime.minimum.
↪read()
```

No command help available

Suppressed linked return values: reliability

return
pvt_trace: No help available

6.1.1.3.7.17 Sgacp

class SgacpCls

Sgacp commands group definition. 8 total commands, 4 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.bluetooth.measurement.hdr.trace.sgacp.clone()
```

Subgroups

6.1.1.3.7.18 Average

SCPI Commands :

```
READ:BLUetooth:MEASurement<Instance>:HDR:TRACe:SGACp:AVERage
FETCh:BLUetooth:MEASurement<Instance>:HDR:TRACe:SGACp:AVERage
```

class AverageCls

Average commands group definition. 2 total commands, 0 Subgroups, 2 group commands

fetch() → List[float]

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:HDR:TRACe:SGACp:AVERage
value: List[float] = driver.bluetooth.measurement.hdr.trace.sgacp.average.
↪fetch()
```

No command help available

Suppressed linked return values: reliability

return
sgacp_trace: No help available

read() → List[float]

```
# SCPI: READ:BLUetooth:MEASurement<Instance>:HDR:TRACe:SGACp:AVERage
value: List[float] = driver.bluetooth.measurement.hdr.trace.sgacp.average.read()
```

No command help available

Suppressed linked return values: reliability

return
sgacp_trace: No help available

6.1.1.3.7.19 Current

SCPI Commands :

```
READ:BLUetooth:MEASurement<Instance>:HDR:TRACe:SGACp:CURRent
FETCh:BLUetooth:MEASurement<Instance>:HDR:TRACe:SGACp:CURRent
```

class CurrentCls

Current commands group definition. 2 total commands, 0 Subgroups, 2 group commands

fetch() → List[float]

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:HDR:TRACe:SGACp:CURRent
value: List[float] = driver.bluetooth.measurement.hdr.trace.sgacp.current.
↪fetch()
```

No command help available

Suppressed linked return values: reliability

return
sgacp_trace: No help available

read() → List[float]

```
# SCPI: READ:BLUetooth:MEASurement<Instance>:HDR:TRACe:SGACp:CURRent
value: List[float] = driver.bluetooth.measurement.hdr.trace.sgacp.current.read()
```

No command help available

Suppressed linked return values: reliability

return
sgacp_trace: No help available

6.1.1.3.7.20 Maximum

SCPI Commands :

```
READ:BLUetooth:MEASurement<Instance>:HDR:TRACe:SGACp:MAXimum
FETCh:BLUetooth:MEASurement<Instance>:HDR:TRACe:SGACp:MAXimum
```

class MaximumCls

Maximum commands group definition. 2 total commands, 0 Subgroups, 2 group commands

fetch() → List[float]

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:HDR:TRACe:SGACp:MAXimum
value: List[float] = driver.bluetooth.measurement.hdr.trace.sgacp.maximum.
↪fetch()
```

No command help available

Suppressed linked return values: reliability

return
sgacp_trace: No help available

read() → List[float]

```
# SCPI: READ:BLUetooth:MEASurement<Instance>:HDR:TRACe:SGACp:MAXimum
value: List[float] = driver.bluetooth.measurement.hdr.trace.sgacp.maximum.read()
```

No command help available

Suppressed linked return values: reliability

return
sgacp_trace: No help available

6.1.1.3.7.21 Ptx

SCPI Commands :

```
READ:BLUetooth:MEASurement<Instance>:HDR:TRACe:SGACp[:PTX]
FETCh:BLUetooth:MEASurement<Instance>:HDR:TRACe:SGACp[:PTX]
```

class PtxCls

Ptx commands group definition. 2 total commands, 0 Subgroups, 2 group commands

fetch() → List[float]

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:HDR:TRACe:SGACp[:PTX]
value: List[float] = driver.bluetooth.measurement.hdr.trace.sgacp.ptx.fetch()
```

No command help available

Suppressed linked return values: reliability

return
sgacp_trace: No help available

read() → List[float]

```
# SCPI: READ:BLUetooth:MEASurement<Instance>:HDR:TRACe:SGACp[:PTX]
value: List[float] = driver.bluetooth.measurement.hdr.trace.sgacp.ptx.read()
```

No command help available

Suppressed linked return values: reliability

return
sgacp_trace: No help available

6.1.1.4 Hdrp

SCPI Commands :

```
ABORt:BLUetooth:MEASurement<Instance>:HDRP
STOP:BLUetooth:MEASurement<Instance>:HDRP
INITiate:BLUetooth:MEASurement<Instance>:HDRP
```

class HdrpCls

Hdrp commands group definition. 66 total commands, 6 Subgroups, 3 group commands

abort() → None

```
# SCPI: ABORt:BLUetooth:MEASurement<Instance>:HDRP
driver.bluetooth.measurement.hdrp.abort()
```

No command help available

abort_with_opc(*opc_timeout_ms: int = -1*) → None

```
# SCPI: ABORt:BLUetooth:MEASurement<Instance>:HDRP
driver.bluetooth.measurement.hdrp.abort_with_opc()
```

No command help available

Same as abort, but waits for the operation to complete before continuing further. Use the RsCMPX_BluetoothMeas.utilities.opc_timeout_set() to set the timeout value.

param opc_timeout_ms
Maximum time to wait in milliseconds, valid only for this call.

initiate() → None

```
# SCPI: INITiate:BLUetooth:MEASurement<Instance>:HDRP
driver.bluetooth.measurement.hdrp.initiate()
```

No command help available

initiate_with_opc(*opc_timeout_ms: int = -1*) → None

```
# SCPI: INITiate:BLUetooth:MEASurement<Instance>:HDRP
driver.bluetooth.measurement.hdrp.initiate_with_opc()
```

No command help available

Same as initiate, but waits for the operation to complete before continuing further. Use the RsCMPX_BluetoothMeas.utilities.opc_timeout_set() to set the timeout value.

param opc_timeout_ms
Maximum time to wait in milliseconds, valid only for this call.

stop() → None

```
# SCPI: STOP:BLUetooth:MEASurement<Instance>:HDRP
driver.bluetooth.measurement.hdrp.stop()
```

No command help available

stop_with_opc(*opc_timeout_ms: int = -1*) → None

```
# SCPI: STOP:BLUetooth:MEASurement<Instance>:HDRP
driver.bluetooth.measurement.hdrp.stop_with_opc()
```

No command help available

Same as stop, but waits for the operation to complete before continuing further. Use the RsCMPX_BluetoothMeas.utilities.opc_timeout_set() to set the timeout value.

param opc_timeout_ms
Maximum time to wait in milliseconds, valid only for this call.

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.bluetooth.measurement.hdrp.clone()
```

Subgroups

6.1.1.4.1 InputSignal

class InputSignalCls

InputSignal commands group definition. 2 total commands, 1 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.bluetooth.measurement.hdrp.inputSignal.clone()
```

Subgroups

6.1.1.4.1.1 Adetected

class AdetectedCls

Adetected commands group definition. 2 total commands, 2 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.bluetooth.measurement.hdrp.inputSignal.adetected.clone()
```

Subgroups

6.1.1.4.1.2 Pcoding

SCPI Command :

```
FETCh:BLUetooth:MEASurement<Instance>:HDRP:ISIGnal:ADETected:PCODing
```

class PcodingCls

Pcoding commands group definition. 1 total commands, 0 Subgroups, 1 group commands

fetch() → PayloadCoding

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:HDRP:ISIGnal:ADETected:PCODing
value: enums.PayloadCoding = driver.bluetooth.measurement.hdrp.inputSignal.
↪adetected.pcoding.fetch()
```

No command help available

Suppressed linked return values: reliability

return
payload_coding: No help available

6.1.1.4.1.3 Plength

SCPI Command :

```
FETCh:BLUetooth:MEASurement<Instance>:HDRP:ISIGnal:ADETected:PLENgth
```

class PlengthCls

Plength commands group definition. 1 total commands, 0 Subgroups, 1 group commands

fetch() → int

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:HDRP:ISIGnal:ADETected:PLENgth
value: int = driver.bluetooth.measurement.hdrp.inputSignal.adetected.plength.
↪fetch()
```

No command help available

Suppressed linked return values: reliability

return
payload_length: No help available

6.1.1.4.2 Modulation

class ModulationCls

Modulation commands group definition. 12 total commands, 4 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.bluetooth.measurement.hdrp.modulation.clone()
```

Subgroups

6.1.1.4.2.1 Average

SCPI Commands :

```
READ:BLUetooth:MEASurement<Instance>:HDRP:MODulation:AVERage
FETCh:BLUetooth:MEASurement<Instance>:HDRP:MODulation:AVERage
CALCulate:BLUetooth:MEASurement<Instance>:HDRP:MODulation:AVERage
```

class AverageCls

Average commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Bursts_Out_Of_Tol: float or bool: No parameter help available
- Nominal_Power: float or bool: No parameter help available
- Wi: float or bool: No parameter help available
- W_0_Wi: float or bool: No parameter help available
- W_0_Max: float or bool: No parameter help available
- Rms_Evm: float or bool: No parameter help available
- Peak_Evm: float or bool: No parameter help available
- Symbol_Rate_Error: int: No parameter help available

class FetchStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Bursts_Out_Of_Tol: float: No parameter help available
- Nominal_Power: float: No parameter help available
- Wi: float: No parameter help available
- W_0_Wi: float: No parameter help available
- W_0_Max: float: No parameter help available
- Rms_Evm: float: No parameter help available
- Peak_Evm: float: No parameter help available
- Symbol_Rate_Error: int: No parameter help available

class ReadStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Bursts_Out_Of_Tol: int: No parameter help available
- Nominal_Power: float: No parameter help available
- Wi: float: No parameter help available
- W_0_Wi: float: No parameter help available
- W_0_Max: float: No parameter help available
- Rms_Evm: float: No parameter help available
- Peak_Evm: float: No parameter help available
- Symbol_Rate_Error: int: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>:HDRP:MODulation:AVERage
value: CalculateStruct = driver.bluetooth.measurement.hdrp.modulation.average.
↪calculate()
```

No command help available

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → FetchStruct

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:HDRP:MODulation:AVERage
value: FetchStruct = driver.bluetooth.measurement.hdrp.modulation.average.
↪fetch()
```

No command help available

return
structure: for return value, see the help for FetchStruct structure arguments.

read() → ReadStruct

```
# SCPI: READ:BLUetooth:MEASurement<Instance>:HDRP:MODulation:AVERage
value: ReadStruct = driver.bluetooth.measurement.hdrp.modulation.average.read()
```

No command help available

return
structure: for return value, see the help for ReadStruct structure arguments.

6.1.1.4.2.2 Current

SCPI Commands :

```
READ:BLUetooth:MEASurement<Instance>:HDRP:MODulation:CURRent
FETCh:BLUetooth:MEASurement<Instance>:HDRP:MODulation:CURRent
CALCulate:BLUetooth:MEASurement<Instance>:HDRP:MODulation:CURRent
```

class CurrentCls

Current commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Bursts_Out_Of_Tol: float or bool: No parameter help available
- Nominal_Power: float or bool: No parameter help available
- Wi: float or bool: No parameter help available
- W_0_Wi: float or bool: No parameter help available
- W_0_Max: float or bool: No parameter help available
- Rms_Evm: float or bool: No parameter help available

- Peak_Evm: float or bool: No parameter help available
- P_99_Evm: float or bool: No parameter help available
- Symbol_Rate_Error: int: No parameter help available

class FetchStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Bursts_Out_Of_Tol: float: No parameter help available
- Nominal_Power: float: No parameter help available
- Wi: float: No parameter help available
- W_0_Wi: float: No parameter help available
- W_0_Max: float: No parameter help available
- Rms_Evm: float: No parameter help available
- Peak_Evm: float: No parameter help available
- P_99_Devm: float: No parameter help available
- Symbol_Rate_Error: int: No parameter help available

class ReadStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Bursts_Out_Of_Tol: float: No parameter help available
- Nominal_Power: float: No parameter help available
- Wi: float: No parameter help available
- W_0_Wi: float: No parameter help available
- W_0_Max: float: No parameter help available
- Rms_Evm: float: No parameter help available
- Peak_Evm: float: No parameter help available
- P_99_Evm: float: No parameter help available
- Symbol_Rate_Error: int: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>:HDRP:MODulation:CURRent
value: CalculateStruct = driver.bluetooth.measurement.hdrp.modulation.current.
↪calculate()
```

No command help available

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → FetchStruct

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:HDRP:MODulation:CURRent
value: FetchStruct = driver.bluetooth.measurement.hdrp.modulation.current.
↪fetch()
```

No command help available

return
structure: for return value, see the help for FetchStruct structure arguments.

read() → ReadStruct

```
# SCPI: READ:BLUetooth:MEASurement<Instance>:HDRP:MODulation:CURRent
value: ReadStruct = driver.bluetooth.measurement.hdrp.modulation.current.read()
```

No command help available

return
structure: for return value, see the help for ReadStruct structure arguments.

6.1.1.4.2.3 Maximum

SCPI Commands :

```
READ:BLUetooth:MEASurement<Instance>:HDRP:MODulation:MAXimum
FETCh:BLUetooth:MEASurement<Instance>:HDRP:MODulation:MAXimum
CALCulate:BLUetooth:MEASurement<Instance>:HDRP:MODulation:MAXimum
```

class MaximumCls

Maximum commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Bursts_Out_Of_Tol: float or bool: No parameter help available
- Nominal_Power: float or bool: No parameter help available
- Wi: float or bool: No parameter help available
- W_0_Wi: float or bool: No parameter help available
- W_0_Max: float or bool: No parameter help available
- Rms_Evm: float or bool: No parameter help available
- Peak_Evm: float or bool: No parameter help available
- Symbol_Rate_Error: int: No parameter help available

class ResultData

Response structure. Fields:

- Reliability: int: No parameter help available
- Bursts_Out_Of_Tol: float: No parameter help available
- Nominal_Power: float: No parameter help available

- Wi: float: No parameter help available
- W_0_Wi: float: No parameter help available
- W_0_Max: float: No parameter help available
- Rms_Evm: float: No parameter help available
- Peak_Evm: float: No parameter help available
- Symbol_Rate_Error: int: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>:HDRP:MODulation:MAXimum
value: CalculateStruct = driver.bluetooth.measurement.hdrp.modulation.maximum.
↪calculate()
```

No command help available

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → ResultData

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:HDRP:MODulation:MAXimum
value: ResultData = driver.bluetooth.measurement.hdrp.modulation.maximum.fetch()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

read() → ResultData

```
# SCPI: READ:BLUetooth:MEASurement<Instance>:HDRP:MODulation:MAXimum
value: ResultData = driver.bluetooth.measurement.hdrp.modulation.maximum.read()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

6.1.1.4.2.4 StandardDev

SCPI Commands :

```
READ:BLUetooth:MEASurement<Instance>:HDRP:MODulation:SDEViation
FETCh:BLUetooth:MEASurement<Instance>:HDRP:MODulation:SDEViation
CALCulate:BLUetooth:MEASurement<Instance>:HDRP:MODulation:SDEViation
```

class StandardDevCls

StandardDev commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: No parameter help available

- Bursts_Out_Of_Tol: float or bool: No parameter help available
- Nominal_Power: float or bool: No parameter help available
- Wi: float or bool: No parameter help available
- W_0_Wi: float or bool: No parameter help available
- W_0_Max: float or bool: No parameter help available
- Rms_Evm: float or bool: No parameter help available
- Peak_Evm: float or bool: No parameter help available
- Symbol_Rate_Error: int: No parameter help available

class FetchStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Bursts_Out_Of_Tol: float: No parameter help available
- Nominal_Power: float: No parameter help available
- Wi: float: No parameter help available
- W_0_Wi: float: No parameter help available
- W_0_Max: float: No parameter help available
- Rms_Evm: float: No parameter help available
- Peak_Evm: float: No parameter help available
- Symbol_Error_Rate: int: No parameter help available

class ReadStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Bursts_Out_Of_Tol: float: No parameter help available
- Nominal_Power: float: No parameter help available
- Wi: float: No parameter help available
- W_0_Wi: float: No parameter help available
- W_0_Max: float: No parameter help available
- Rms_Evm: float: No parameter help available
- Peak_Evm: float: No parameter help available
- Symbol_Rate_Error: int: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>:HDRP:MODulation:SDEViation
value: CalculateStruct = driver.bluetooth.measurement.hdrp.modulation.
↪standardDev.calculate()
```

No command help available

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → FetchStruct

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:HDRP:MODulation:SDEViation
value: FetchStruct = driver.bluetooth.measurement.hdrp.modulation.standardDev.
↪fetch()
```

No command help available

return
structure: for return value, see the help for FetchStruct structure arguments.

read() → ReadStruct

```
# SCPI: READ:BLUetooth:MEASurement<Instance>:HDRP:MODulation:SDEViation
value: ReadStruct = driver.bluetooth.measurement.hdrp.modulation.standardDev.
↪read()
```

No command help available

return
structure: for return value, see the help for ReadStruct structure arguments.

6.1.1.4.3 PowerVsTime

class PowerVsTimeCls

PowerVsTime commands group definition. 12 total commands, 4 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.bluetooth.measurement.hdrp.powerVsTime.clone()
```

Subgroups

6.1.1.4.3.1 Average

SCPI Commands :

```
READ:BLUetooth:MEASurement<Instance>:HDRP:PVTime:AVERage
FETCh:BLUetooth:MEASurement<Instance>:HDRP:PVTime:AVERage
CALCulate:BLUetooth:MEASurement<Instance>:HDRP:PVTime:AVERage
```

class AverageCls

Average commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Bursts_Out_Of_Tol: float or bool: No parameter help available
- Average_Power: float or bool: No parameter help available

class ResultData

Response structure. Fields:

- Reliability: int: No parameter help available
- Bursts_Out_Of_Tol: float: No parameter help available
- Average_Power: float: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>:HDRP:PVTime:AVERage
value: CalculateStruct = driver.bluetooth.measurement.hdrp.powerVsTime.average.
↪calculate()
```

No command help available

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → ResultData

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:HDRP:PVTime:AVERage
value: ResultData = driver.bluetooth.measurement.hdrp.powerVsTime.average.
↪fetch()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

read() → ResultData

```
# SCPI: READ:BLUetooth:MEASurement<Instance>:HDRP:PVTime:AVERage
value: ResultData = driver.bluetooth.measurement.hdrp.powerVsTime.average.read()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

6.1.1.4.3.2 Current

SCPI Commands :

```
READ:BLUetooth:MEASurement<Instance>:HDRP:PVTime:CURRent
FETCh:BLUetooth:MEASurement<Instance>:HDRP:PVTime:CURRent
CALCulate:BLUetooth:MEASurement<Instance>:HDRP:PVTime:CURRent
```

class CurrentCls

Current commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Bursts_Out_Of_Tol: float or bool: No parameter help available

- Average_Power: float or bool: No parameter help available

class ResultData

Response structure. Fields:

- Reliability: int: No parameter help available
- Bursts_Out_Of_Tol: float: No parameter help available
- Average_Power: float: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>:HDRP:PVTime:CURRent
value: CalculateStruct = driver.bluetooth.measurement.hdrp.powerVsTime.current.
↪calculate()
```

No command help available

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → ResultData

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:HDRP:PVTime:CURRent
value: ResultData = driver.bluetooth.measurement.hdrp.powerVsTime.current.
↪fetch()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

read() → ResultData

```
# SCPI: READ:BLUetooth:MEASurement<Instance>:HDRP:PVTime:CURRent
value: ResultData = driver.bluetooth.measurement.hdrp.powerVsTime.current.read()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

6.1.1.4.3.3 Maximum

SCPI Commands :

```
READ:BLUetooth:MEASurement<Instance>:HDRP:PVTime:MAXimum
FETCh:BLUetooth:MEASurement<Instance>:HDRP:PVTime:MAXimum
CALCulate:BLUetooth:MEASurement<Instance>:HDRP:PVTime:MAXimum
```

class MaximumCls

Maximum commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: No parameter help available

- Bursts_Out_Of_Tol: float or bool: No parameter help available
- Average_Power: float or bool: No parameter help available

class ResultData

Response structure. Fields:

- Reliability: int: No parameter help available
- Bursts_Out_Of_Tol: float: No parameter help available
- Average_Power: float: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>:HDRP:PVTime:MAXimum
value: CalculateStruct = driver.bluetooth.measurement.hdrp.powerVsTime.maximum.
↪calculate()
```

No command help available

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → ResultData

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:HDRP:PVTime:MAXimum
value: ResultData = driver.bluetooth.measurement.hdrp.powerVsTime.maximum.
↪fetch()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

read() → ResultData

```
# SCPI: READ:BLUetooth:MEASurement<Instance>:HDRP:PVTime:MAXimum
value: ResultData = driver.bluetooth.measurement.hdrp.powerVsTime.maximum.read()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

6.1.1.4.3.4 Minimum

SCPI Commands :

```
READ:BLUetooth:MEASurement<Instance>:HDRP:PVTime:MINimum
FETCh:BLUetooth:MEASurement<Instance>:HDRP:PVTime:MINimum
CALCulate:BLUetooth:MEASurement<Instance>:HDRP:PVTime:MINimum
```

class MinimumCls

Minimum commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Bursts_Out_Of_Tol: float or bool: No parameter help available
- Average_Power: float or bool: No parameter help available

class ResultData

Response structure. Fields:

- Reliability: int: No parameter help available
- Bursts_Out_Of_Tol: float: No parameter help available
- Average_Power: float: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>:HDRP:PVTime:MINimum
value: CalculateStruct = driver.bluetooth.measurement.hdrp.powerVsTime.minimum.
↪calculate()
```

No command help available

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → ResultData

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:HDRP:PVTime:MINimum
value: ResultData = driver.bluetooth.measurement.hdrp.powerVsTime.minimum.
↪fetch()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

read() → ResultData

```
# SCPI: READ:BLUetooth:MEASurement<Instance>:HDRP:PVTime:MINimum
value: ResultData = driver.bluetooth.measurement.hdrp.powerVsTime.minimum.read()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

6.1.1.4.4 Sacp

class SacpCls

Sacp commands group definition. 3 total commands, 1 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.bluetooth.measurement.hdrp.sacp.clone()
```

Subgroups

6.1.1.4.4.1 Current

SCPI Commands :

```
READ:BLUetooth:MEASurement<Instance>:HDRP:SACP[:CURRent]
FETCh:BLUetooth:MEASurement<Instance>:HDRP:SACP[:CURRent]
CALCulate:BLUetooth:MEASurement<Instance>:HDRP:SACP[:CURRent]
```

class CurrentCls

Current commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Bursts_Out_Of_Tol: float or bool: No parameter help available
- Nominal_Power: float or bool: No parameter help available
- Ptx_Ref: float or bool: No parameter help available

class ResultData

Response structure. Fields:

- Reliability: int: No parameter help available
- Bursts_Out_Of_Tol: float: No parameter help available
- Nominal_Power: float: No parameter help available
- Ptx_Ref: float: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>:HDRP:SACP[:CURRent]
value: CalculateStruct = driver.bluetooth.measurement.hdrp.sacp.current.
↪calculate()
```

No command help available

return

structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → ResultData

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:HDRP:SACP[:CURRent]
value: ResultData = driver.bluetooth.measurement.hdrp.sacp.current.fetch()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

read() → ResultData

```
# SCPI: READ:BLUetooth:MEASurement<Instance>:HDRP:SACP[:CURRent]
value: ResultData = driver.bluetooth.measurement.hdrp.sacp.current.read()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

6.1.1.4.5 State

SCPI Command :

```
FETCh:BLUetooth:MEASurement<Instance>:HDRP:STATe
```

class StateCls

State commands group definition. 2 total commands, 1 Subgroups, 1 group commands

fetch(*timeout: float = None*, *target_main_state: TargetMainState = None*, *target_sync_state: SyncState = None*) → ResourceState

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:HDRP:STATe
value: enums.ResourceState = driver.bluetooth.measurement.hdrp.state.
↪fetch(timeout = 1.0, target_main_state = enums.TargetMainState.OFF, target_
↪sync_state = enums.SyncState.ADJusted)
```

No command help available

param timeout
No help available

param target_main_state
No help available

param target_sync_state
No help available

return
main_state: No help available

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.bluetooth.measurement.hdrp.state.clone()
```

Subgroups

6.1.1.4.5.1 All

SCPI Command :

```
FETCh:BLUetooth:MEASurement<Instance>:HDRP:STATe:ALL
```

class AllCls

All commands group definition. 1 total commands, 0 Subgroups, 1 group commands

class FetchStruct

Response structure. Fields:

- Main_State: enums.ResourceState: No parameter help available
- Sync_State: enums.SyncState: No parameter help available
- Resource_State: enums.ResourceState: No parameter help available

fetch(*timeout: float = None*, *target_main_state: TargetMainState = None*, *target_sync_state: SyncState = None*) → FetchStruct

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:HDRP:STATe:ALL
value: FetchStruct = driver.bluetooth.measurement.hdrp.state.all.fetch(timeout␣
↪= 1.0, target_main_state = enums.TargetMainState.OFF, target_sync_state =␣
↪enums.SyncState.ADJusted)
```

No command help available

param timeout
No help available

param target_main_state
No help available

param target_sync_state
No help available

return
structure: for return value, see the help for FetchStruct structure arguments.

6.1.1.4.6 Trace

class TraceCls

Trace commands group definition. 32 total commands, 5 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.bluetooth.measurement.hdrp.trace.clone()
```

Subgroups

6.1.1.4.6.1 EvMagnitude

class EvMagnitudeCls

EvMagnitude commands group definition. 12 total commands, 2 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.bluetooth.measurement.hdrp.trace.evMagnitude.clone()
```

Subgroups

6.1.1.4.6.2 Absolute

class AbsoluteCls

Absolute commands group definition. 6 total commands, 3 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.bluetooth.measurement.hdrp.trace.evMagnitude.absolute.clone()
```

Subgroups

6.1.1.4.6.3 Average

SCPI Commands :

```
READ:BLUetooth:MEASurement<Instance>:HDRP:TRACe:EVMagnitude:ABSolute:AVERage
FETCh:BLUetooth:MEASurement<Instance>:HDRP:TRACe:EVMagnitude:ABSolute:AVERage
```

class AverageCls

Average commands group definition. 2 total commands, 0 Subgroups, 2 group commands

fetch() → List[float]

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:HDRP:TRACe:EVMagnitude:ABSolute:AVERage
value: List[float] = driver.bluetooth.measurement.hdrp.trace.evMagnitude.
↪absolute.average.fetch()
```

No command help available

Suppressed linked return values: reliability

return
evm: No help available

read() → List[float]

```
# SCPI: READ:BLUetooth:MEASurement<Instance>
↪:HDRP:TRACe:EVMagnitude:ABSolute:AVERage
value: List[float] = driver.bluetooth.measurement.hdrp.trace.evMagnitude.
↪absolute.average.read()
```

No command help available

Suppressed linked return values: reliability

return
evm: No help available

6.1.1.4.6.4 Current

SCPI Commands :

```
READ:BLUetooth:MEASurement<Instance>:HDRP:TRACe:EVMagnitude:ABSolute:CURRent
FETCh:BLUetooth:MEASurement<Instance>:HDRP:TRACe:EVMagnitude:ABSolute:CURRent
```

class CurrentCls

Current commands group definition. 2 total commands, 0 Subgroups, 2 group commands

fetch() → List[float]

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:HDRP:TRACe:EVMagnitude:ABSolute:CURRent
value: List[float] = driver.bluetooth.measurement.hdrp.trace.evMagnitude.
↪absolute.current.fetch()
```

No command help available

Suppressed linked return values: reliability

return
evm: No help available

read() → List[float]

```
# SCPI: READ:BLUetooth:MEASurement<Instance>
↪:HDRP:TRACe:EVMagnitude:ABSolute:CURRent
value: List[float] = driver.bluetooth.measurement.hdrp.trace.evMagnitude.
↪absolute.current.read()
```

No command help available

Suppressed linked return values: reliability

return
evm: No help available

6.1.1.4.6.5 Maximum

SCPI Commands :

```
READ:BLUetooth:MEASurement<Instance>:HDRP:TRACe:EVMagnitude:ABSolute:MAXimum
FETCh:BLUetooth:MEASurement<Instance>:HDRP:TRACe:EVMagnitude:ABSolute:MAXimum
```

class MaximumCls

Maximum commands group definition. 2 total commands, 0 Subgroups, 2 group commands

fetch() → List[float]

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:HDRP:TRACe:EVMagnitude:ABSolute:MAXimum
value: List[float] = driver.bluetooth.measurement.hdrp.trace.evMagnitude.
↪absolute.maximum.fetch()
```

No command help available

Suppressed linked return values: reliability

return
evm: No help available

read() → List[float]

```
# SCPI: READ:BLUetooth:MEASurement<Instance>
↪:HDRP:TRACe:EVMagnitude:ABSolute:MAXimum
value: List[float] = driver.bluetooth.measurement.hdrp.trace.evMagnitude.
↪absolute.maximum.read()
```

No command help available

Suppressed linked return values: reliability

return
evm: No help available

6.1.1.4.6.6 Offset

class OffsetCls

Offset commands group definition. 6 total commands, 3 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.bluetooth.measurement.hdrp.trace.evMagnitude.offset.clone()
```

Subgroups

6.1.1.4.6.7 Average

SCPI Commands :

```
READ:BLUetooth:MEASurement<Instance>:HDRP:TRACe:EVMagnitude:OFFSet:AVERage
FETCh:BLUetooth:MEASurement<Instance>:HDRP:TRACe:EVMagnitude:OFFSet:AVERage
```

class AverageCls

Average commands group definition. 2 total commands, 0 Subgroups, 2 group commands

fetch() → List[float]

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:HDRP:TRACe:EVMagnitude:OFFSet:AVERage
value: List[float] = driver.bluetooth.measurement.hdrp.trace.evMagnitude.offset.
↪average.fetch()
```

No command help available

Suppressed linked return values: reliability

return
evm: No help available

read() → List[float]

```
# SCPI: READ:BLUetooth:MEASurement<Instance>
↪:HDRP:TRACe:EVMagnitude:OFFSet:AVERage
value: List[float] = driver.bluetooth.measurement.hdrp.trace.evMagnitude.offset.
↪average.read()
```

No command help available

Suppressed linked return values: reliability

return
evm: No help available

6.1.1.4.6.8 Current

SCPI Commands :

```
READ:BLUetooth:MEASurement<Instance>:HDRP:TRACe:EVMagnitude:OFFSet:CURRent
FETCh:BLUetooth:MEASurement<Instance>:HDRP:TRACe:EVMagnitude:OFFSet:CURRent
```

class CurrentCls

Current commands group definition. 2 total commands, 0 Subgroups, 2 group commands

fetch() → List[float]

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:HDRP:TRACe:EVMagnitude:OFFSet:CURRent
value: List[float] = driver.bluetooth.measurement.hdrp.trace.evMagnitude.offset.
↪current.fetch()
```

No command help available

Suppressed linked return values: reliability

return
evm: No help available

read() → List[float]

```
# SCPI: READ:BLUetooth:MEASurement<Instance>
↪:HDRP:TRACe:EVMagnitude:OFFSet:CURRent
value: List[float] = driver.bluetooth.measurement.hdrp.trace.evMagnitude.offset.
↪current.read()
```

No command help available

Suppressed linked return values: reliability

return
evm: No help available

6.1.1.4.6.9 Maximum

SCPI Commands :

```
READ:BLUetooth:MEASurement<Instance>:HDRP:TRACe:EVMagnitude:OFFSet:MAXimum
FETCh:BLUetooth:MEASurement<Instance>:HDRP:TRACe:EVMagnitude:OFFSet:MAXimum
```

class MaximumCls

Maximum commands group definition. 2 total commands, 0 Subgroups, 2 group commands

fetch() → List[float]

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:HDRP:TRACe:EVMagnitude:OFFSet:MAXimum
value: List[float] = driver.bluetooth.measurement.hdrp.trace.evMagnitude.offset.
↪maximum.fetch()
```

No command help available

Suppressed linked return values: reliability

return
evm: No help available

read() → List[float]

```
# SCPI: READ:BLUetooth:MEASurement<Instance>
↪:HDRP:TRACe:EVMagnitude:OFFSet:MAXimum
value: List[float] = driver.bluetooth.measurement.hdrp.trace.evMagnitude.offset.
↪maximum.read()
```

No command help available

Suppressed linked return values: reliability

return
evm: No help available

6.1.1.4.6.10 IqAbs

SCPI Commands :

```
FETCh:BLUetooth:MEASurement<Instance>:HDRP:TRACe:IQABs
READ:BLUetooth:MEASurement<Instance>:HDRP:TRACe:IQABs
```

class IqAbsCls

IqAbs commands group definition. 2 total commands, 0 Subgroups, 2 group commands

class ResultData

Response structure. Fields:

- Reliability: int: No parameter help available
- Iphase: List[float]: No parameter help available
- Qphase: List[float]: No parameter help available

fetch() → ResultData

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:HDRP:TRACe:IQABs
value: ResultData = driver.bluetooth.measurement.hdrp.trace.iqAbs.fetch()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

read() → ResultData

```
# SCPI: READ:BLUetooth:MEASurement<Instance>:HDRP:TRACe:IQABs
value: ResultData = driver.bluetooth.measurement.hdrp.trace.iqAbs.read()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

6.1.1.4.6.11 IqOffset

SCPI Commands :

```
READ:BLUetooth:MEASurement<Instance>:HDRP:TRACe:IQOFfset
FETCh:BLUetooth:MEASurement<Instance>:HDRP:TRACe:IQOFfset
```

class IqOffsetCls

IqOffset commands group definition. 2 total commands, 0 Subgroups, 2 group commands

class ResultData

Response structure. Fields:

- Reliability: int: No parameter help available
- Iphase: List[float]: No parameter help available
- Qphase: List[float]: No parameter help available

fetch() → ResultData

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:HDRP:TRACe:IQOFfset
value: ResultData = driver.bluetooth.measurement.hdrp.trace.iqOffset.fetch()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

read() → ResultData

```
# SCPI: READ:BLUetooth:MEASurement<Instance>:HDRP:TRACe:IQOFfset
value: ResultData = driver.bluetooth.measurement.hdrp.trace.iqOffset.read()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

6.1.1.4.6.12 PowerVsTime

class PowerVsTimeCls

PowerVsTime commands group definition. 8 total commands, 4 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.bluetooth.measurement.hdrp.trace.powerVsTime.clone()
```

Subgroups

6.1.1.4.6.13 Average

SCPI Commands :

```
READ:BLUetooth:MEASurement<Instance>:HDRP:TRACe:PVTime:AVERage
FETCh:BLUetooth:MEASurement<Instance>:HDRP:TRACe:PVTime:AVERage
```

class AverageCls

Average commands group definition. 2 total commands, 0 Subgroups, 2 group commands

fetch() → List[float]

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:HDRP:TRACe:PVTime:AVERage
value: List[float] = driver.bluetooth.measurement.hdrp.trace.powerVsTime.
↪average.fetch()
```

No command help available

Suppressed linked return values: reliability

return
pvt_trace: No help available

read() → List[float]

```
# SCPI: READ:BLUetooth:MEASurement<Instance>:HDRP:TRACe:PVTime:AVERage
value: List[float] = driver.bluetooth.measurement.hdrp.trace.powerVsTime.
↪average.read()
```

No command help available

Suppressed linked return values: reliability

return
pvt_trace: No help available

6.1.1.4.6.14 Current

SCPI Commands :

```
READ:BLUetooth:MEASurement<Instance>:HDRP:TRACe:PVTime:CURRent
FETCh:BLUetooth:MEASurement<Instance>:HDRP:TRACe:PVTime:CURRent
```

class CurrentCls

Current commands group definition. 2 total commands, 0 Subgroups, 2 group commands

fetch() → List[float]

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:HDRP:TRACe:PVTime:CURRent
value: List[float] = driver.bluetooth.measurement.hdrp.trace.powerVsTime.
↪current.fetch()
```

No command help available

Suppressed linked return values: reliability

return
pvt_trace: No help available

read() → List[float]

```
# SCPI: READ:BLUetooth:MEASurement<Instance>:HDRP:TRACe:PVTime:CURRent
value: List[float] = driver.bluetooth.measurement.hdrp.trace.powerVsTime.
↪current.read()
```

No command help available

Suppressed linked return values: reliability

return
pvt_trace: No help available

6.1.1.4.6.15 Maximum

SCPI Commands :

```
READ:BLUetooth:MEASurement<Instance>:HDRP:TRACe:PVTime:MAXimum
FETCh:BLUetooth:MEASurement<Instance>:HDRP:TRACe:PVTime:MAXimum
```

class MaximumCls

Maximum commands group definition. 2 total commands, 0 Subgroups, 2 group commands

fetch() → List[float]

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:HDRP:TRACe:PVTime:MAXimum
value: List[float] = driver.bluetooth.measurement.hdrp.trace.powerVsTime.
↪maximum.fetch()
```

No command help available

Suppressed linked return values: reliability

return
pvt_trace: No help available

read() → List[float]

```
# SCPI: READ:BLUetooth:MEASurement<Instance>:HDRP:TRACe:PVTime:MAXimum
value: List[float] = driver.bluetooth.measurement.hdrp.trace.powerVsTime.
↪maximum.read()
```

No command help available

Suppressed linked return values: reliability

return
pvt_trace: No help available

6.1.1.4.6.16 Minimum

SCPI Commands :

```
READ:BLUetooth:MEASurement<Instance>:HDRP:TRACe:PVTime:MINimum
FETCh:BLUetooth:MEASurement<Instance>:HDRP:TRACe:PVTime:MINimum
```

class MinimumCls

Minimum commands group definition. 2 total commands, 0 Subgroups, 2 group commands

fetch() → List[float]

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:HDRP:TRACe:PVTime:MINimum
value: List[float] = driver.bluetooth.measurement.hdrp.trace.powerVsTime.
↪minimum.fetch()
```

No command help available

Suppressed linked return values: reliability

return
pvt_trace: No help available

read() → List[float]

```
# SCPI: READ:BLUetooth:MEASurement<Instance>:HDRP:TRACe:PVTime:MINimum
value: List[float] = driver.bluetooth.measurement.hdrp.trace.powerVsTime.
↪minimum.read()
```

No command help available

Suppressed linked return values: reliability

return
pvt_trace: No help available

6.1.1.4.6.17 Sacp

class SacpCls

Sacp commands group definition. 8 total commands, 4 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.bluetooth.measurement.hdrp.trace.sacp.clone()
```

Subgroups

6.1.1.4.6.18 Average

SCPI Commands :

```
READ:BLUetooth:MEASurement<Instance>:HDRP:TRACe:SACP:AVERage
FETCh:BLUetooth:MEASurement<Instance>:HDRP:TRACe:SACP:AVERage
```

class AverageCls

Average commands group definition. 2 total commands, 0 Subgroups, 2 group commands

fetch() → List[float]

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:HDRP:TRACe:SACP:AVERage
value: List[float] = driver.bluetooth.measurement.hdrp.trace.sacp.average.
↪fetch()
```

No command help available

Suppressed linked return values: reliability

return
sacp_trace: No help available

read() → List[float]

```
# SCPI: READ:BLUetooth:MEASurement<Instance>:HDRP:TRACe:SACP:AVERage
value: List[float] = driver.bluetooth.measurement.hdrp.trace.sacp.average.read()
```

No command help available

Suppressed linked return values: reliability

return
sacp_trace: No help available

6.1.1.4.6.19 Current

SCPI Commands :

```
READ:BLUetooth:MEASurement<Instance>:HDRP:TRACe:SACP:CURRent
FETCh:BLUetooth:MEASurement<Instance>:HDRP:TRACe:SACP:CURRent
```

class CurrentCls

Current commands group definition. 2 total commands, 0 Subgroups, 2 group commands

fetch() → List[float]

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:HDRP:TRACe:SACP:CURRent
value: List[float] = driver.bluetooth.measurement.hdrp.trace.sacp.current.
↪fetch()
```

No command help available

Suppressed linked return values: reliability

return
sacp_trace: No help available

read() → List[float]

```
# SCPI: READ:BLUetooth:MEASurement<Instance>:HDRP:TRACe:SACP:CURRent
value: List[float] = driver.bluetooth.measurement.hdrp.trace.sacp.current.read()
```

No command help available

Suppressed linked return values: reliability

return
sacp_trace: No help available

6.1.1.4.6.20 Maximum

SCPI Commands :

```
READ:BLUetooth:MEASurement<Instance>:HDRP:TRACe:SACP:MAXimum
FETCh:BLUetooth:MEASurement<Instance>:HDRP:TRACe:SACP:MAXimum
```

class MaximumCls

Maximum commands group definition. 2 total commands, 0 Subgroups, 2 group commands

fetch() → List[float]

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:HDRP:TRACe:SACP:MAXimum
value: List[float] = driver.bluetooth.measurement.hdrp.trace.sacp.maximum.
↪fetch()
```

No command help available

Suppressed linked return values: reliability

return
sacp_trace: No help available

read() → List[float]

```
# SCPI: READ:BLUetooth:MEASurement<Instance>:HDRP:TRACe:SACP:MAXimum
value: List[float] = driver.bluetooth.measurement.hdrp.trace.sacp.maximum.read()
```

No command help available

Suppressed linked return values: reliability

return
sacp_trace: No help available

6.1.1.4.6.21 Ptx

SCPI Commands :

```
READ:BLUetooth:MEASurement<Instance>:HDRP:TRACe:SACP[:PTX]
FETCh:BLUetooth:MEASurement<Instance>:HDRP:TRACe:SACP[:PTX]
```

class PtxCls

Ptx commands group definition. 2 total commands, 0 Subgroups, 2 group commands

fetch() → List[float]

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:HDRP:TRACe:SACP[:PTX]
value: List[float] = driver.bluetooth.measurement.hdrp.trace.sacp.ptx.fetch()
```

No command help available

Suppressed linked return values: reliability

return
sacp_trace: No help available

read() → List[float]

```
# SCPI: READ:BLUetooth:MEASurement<Instance>:HDRP:TRACe:SACP[:PTX]
value: List[float] = driver.bluetooth.measurement.hdrp.trace.sacp.ptx.read()
```

No command help available

Suppressed linked return values: reliability

return
sacp_trace: No help available

6.1.1.5 InputSignal

class InputSignalCls

InputSignal commands group definition. 39 total commands, 1 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.bluetooth.measurement.inputSignal.clone()
```

Subgroups

6.1.1.5.1 Adetected

class AdetectedCls

Adetected commands group definition. 39 total commands, 9 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.bluetooth.measurement.inputSignal.adetected.clone()
```

Subgroups

6.1.1.5.1.1 Aaddress

class AaddressCls

Aaddress commands group definition. 1 total commands, 1 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.bluetooth.measurement.inputSignal.adetected.aaddress.clone()
```

Subgroups

6.1.1.5.1.2 LowEnergy

class LowEnergyCls

LowEnergy commands group definition. 1 total commands, 1 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.bluetooth.measurement.inputSignal.adetected.aaddress.lowEnergy.clone()
```

Subgroups

6.1.1.5.1.3 Le1M

SCPI Command :

```
FETCh:BLUetooth:MEASurement<Instance>:ISIGnal:ADETected:AADDress:LENergy[:LE1M]
```

class Le1MCls

Le1M commands group definition. 1 total commands, 0 Subgroups, 1 group commands

fetch() → str

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:ISIGnal:ADETected:AADDress:LENergy[:LE1M]
value: str = driver.bluetooth.measurement.inputSignal.adetected.aaddress.
↪lowEnergy.le1M.fetch()
```

Returns the detected access address of the advertiser for LE 1M PHY. A result is available after the CMP180 has auto-detected a packet (method RsCMPX_BluetoothMeas.Configure.Bluetooth.Measurement.InputSignal.dmode AUTO) .

Suppressed linked return values: reliability

return
adv_address: No help available

6.1.1.5.1.4 Coding

class CodingCls

Coding commands group definition. 1 total commands, 1 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.bluetooth.measurement.inputSignal.adetected.coding.clone()
```

Subgroups

6.1.1.5.1.5 LowEnergy

class LowEnergyCls

LowEnergy commands group definition. 1 total commands, 1 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.bluetooth.measurement.inputSignal.adetected.coding.lowEnergy.clone()
```

Subgroups

6.1.1.5.1.6 Lrange

SCPI Command :

```
FETCh:BLUetooth:MEASurement<Instance>:ISIGnal:ADETected:CODing:LENergy:LRANge
```

class LrangeCls

Lrange commands group definition. 1 total commands, 0 Subgroups, 1 group commands

fetch() → CodingScheme

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:ISIGnal:ADETected:CODing:LENergy:LRANge
value: enums.CodingScheme = driver.bluetooth.measurement.inputSignal.adetected.
↪coding.lowEnergy.lrange.fetch()
```

Returns the detected forward error correction coding for LE coded PHY.

Suppressed linked return values: reliability

return
coding: Coding S = 8 or S = 2

6.1.1.5.1.7 Cte

class CteCls

Cte commands group definition. 14 total commands, 2 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.bluetooth.measurement.inputSignal.adetected.cte.clone()
```

Subgroups

6.1.1.5.1.8 LowEnergy

class LowEnergyCls

LowEnergy commands group definition. 4 total commands, 2 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.bluetooth.measurement.inputSignal.adetected.cte.lowEnergy.clone()
```

Subgroups

6.1.1.5.1.9 Le1M

class Le1MCls

Le1M commands group definition. 2 total commands, 2 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.bluetooth.measurement.inputSignal.adetected.cte.lowEnergy.le1M.clone()
```

Subgroups

6.1.1.5.1.10 TypePy

SCPI Command :

```
FETCh:BLUetooth:MEASurement<Instance>:ISIGnal:ADETected:CTE:LENergy:LE1M:TYPE
```

class TypePyCls

TypePy commands group definition. 1 total commands, 0 Subgroups, 1 group commands

fetch() → CtePacketType

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:ISIGnal:ADETected:CTE:LENergy:LE1M:TYPE
value: enums.CtePacketType = driver.bluetooth.measurement.inputSignal.adetected.
↪cte.lowEnergy.le1M.typePy.fetch()
```

Returns the detected CTE type. A result is available after the CMP180 has auto-detected a packet (method RsCMPX_BluetoothMeas.Configure.Bluetooth.Measurement.InputSignal.dmodeAUTO) . Commands for uncoded LE 1M PHY (..:LE1M..) and LE 2M PHY (..:LE2M..) are available.

Suppressed linked return values: reliability

return

packet_type: AOD1us, AOD2us: CTE type angle of departure, 1 µs or 2 µs slot AOAus, AOA2us: CTE type angle of arrival, 2 µs slot AOA1us: CTE type angle of arrival, 1 µs slot

6.1.1.5.1.11 Units

SCPI Command :

```
FETCh:BLUetooth:MEASurement<Instance>:ISIGnal:ADETected:CTE:LENergy:LE1M:UNITs
```

class UnitsCls

Units commands group definition. 1 total commands, 0 Subgroups, 1 group commands

fetch() → int

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:ISIGnal:ADETected:CTE:LENergy:LE1M:UNITs
value: int = driver.bluetooth.measurement.inputSignal.adetected.cte.lowEnergy.
↪le1M.units.fetch()
```

Returns the detected number of CTE units. A result is available after the CMP180 has auto-detected a packet (method RsCMPX_BluetoothMeas.Configure.Bluetooth.Measurement.InputSignal.dmode AUTO) . Commands for uncoded LE 1M PHY (..:LE1M..) and LE 2M PHY (..:LE2M..) are available.

Suppressed linked return values: reliability

return
duration_units: No help available

6.1.1.5.1.12 Le2M

class Le2MCls

Le2M commands group definition. 2 total commands, 2 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.bluetooth.measurement.inputSignal.adetected.cte.lowEnergy.le2M.clone()
```

Subgroups

6.1.1.5.1.13 TypePy

SCPI Command :

```
FETCh:BLUetooth:MEASurement<Instance>:ISIGnal:ADETected:CTE:LENergy:LE2M:TYPE
```

class TypePyCls

TypePy commands group definition. 1 total commands, 0 Subgroups, 1 group commands

fetch() → CtePacketType

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:ISIGnal:ADETected:CTE:LENergy:LE2M:TYPE
value: enums.CtePacketType = driver.bluetooth.measurement.inputSignal.adetected.
↪cte.lowEnergy.le2M.typePy.fetch()
```

Returns the detected CTE type. A result is available after the CMP180 has auto-detected a packet (method RsCMPX_BluetoothMeas.Configure.Bluetooth.Measurement.InputSignal.dmodeAUTO) . Commands for uncoded LE 1M PHY (..:LE1M..) and LE 2M PHY (..:LE2M..) are available.

Suppressed linked return values: reliability

return
packet_type: AOD1us, AOD2us: CTE type angle of departure, 1 µs or 2 µs slot AOAus, AOA2us: CTE type angle of arrival, 2 µs slot AOA1us: CTE type angle of arrival, 1 µs slot

6.1.1.5.1.14 Units

SCPI Command :

```
FETCh:BLUetooth:MEASurement<Instance>:ISIGnal:ADETected:CTE:LENergy:LE2M:UNITs
```

class UnitsCls

Units commands group definition. 1 total commands, 0 Subgroups, 1 group commands

fetch() → int

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:ISIGnal:ADETected:CTE:LENergy:LE2M:UNITs
value: int = driver.bluetooth.measurement.inputSignal.adetected.cte.lowEnergy.
↪le2M.units.fetch()
```

Returns the detected number of CTE units. A result is available after the CMP180 has auto-detected a packet (method RsCMPX_BluetoothMeas.Configure.Bluetooth.Measurement.InputSignal.dmode AUTO) . Commands for uncoded LE 1M PHY (..:LE1M..) and LE 2M PHY (..:LE2M..) are available.

Suppressed linked return values: reliability

return
duration_units: No help available

6.1.1.5.1.15 Qhsl

class QhslCls

Qhsl commands group definition. 10 total commands, 5 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.bluetooth.measurement.inputSignal.adetected.cte.qhsl.clone()
```

Subgroups

6.1.1.5.1.16 P2Q

class P2QCls

P2Q commands group definition. 2 total commands, 2 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.bluetooth.measurement.inputSignal.adetected.cte.qhsl.p2Q.clone()
```

Subgroups

6.1.1.5.1.17 TypePy

SCPI Command :

```
FETCh:BLUetooth:MEASurement<Instance>:ISIGnal:ADETected:CTE:QHSL:P2Q:TYPE
```

class TypePyCls

TypePy commands group definition. 1 total commands, 0 Subgroups, 1 group commands

fetch() → CtePacketType

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:ISIGnal:ADETected:CTE:QHSL:P2Q:TYPE
value: enums.CtePacketType = driver.bluetooth.measurement.inputSignal.adetected.
↪cte.qhsl.p2Q.typePy.fetch()
```

No command help available

Suppressed linked return values: reliability

return
packet_type: No help available

6.1.1.5.1.18 Units

SCPI Command :

```
FETCh:BLUetooth:MEASurement<Instance>:ISIGnal:ADETected:CTE:QHSL:P2Q:UNITs
```

class UnitsCls

Units commands group definition. 1 total commands, 0 Subgroups, 1 group commands

fetch() → int

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:ISIGnal:ADETected:CTE:QHSL:P2Q:UNITs
value: int = driver.bluetooth.measurement.inputSignal.adetected.cte.qhsl.p2Q.
↪units.fetch()
```

No command help available

Suppressed linked return values: reliability

return
duration_units: No help available

6.1.1.5.1.19 P3Q

class P3QCls

P3Q commands group definition. 2 total commands, 2 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.bluetooth.measurement.inputSignal.adetected.cte.qhsl.p3Q.clone()
```

Subgroups

6.1.1.5.1.20 TypePy

SCPI Command :

```
FETCh:BLUetooth:MEASurement<Instance>:ISIGnal:ADETected:CTE:QHSL:P3Q:TYPE
```

class TypePyCls

TypePy commands group definition. 1 total commands, 0 Subgroups, 1 group commands

fetch() → CtePacketType

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:ISIGnal:ADETected:CTE:QHSL:P3Q:TYPE
value: enums.CtePacketType = driver.bluetooth.measurement.inputSignal.adetected.
↪cte.qhsl.p3Q.typePy.fetch()
```

No command help available

Suppressed linked return values: reliability

return
packet_type: No help available

6.1.1.5.1.21 Units

SCPI Command :

```
FETCh:BLUetooth:MEASurement<Instance>:ISIGnal:ADETected:CTE:QHSL:P3Q:UNITs
```

class UnitsCls

Units commands group definition. 1 total commands, 0 Subgroups, 1 group commands

fetch() → int

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:ISIGnal:ADETected:CTE:QHSL:P3Q:UNITs
value: int = driver.bluetooth.measurement.inputSignal.adetected.cte.qhsl.p3Q.
↪units.fetch()
```

No command help available

Suppressed linked return values: reliability

return
duration_units: No help available

6.1.1.5.1.22 P4Q

class P4QCls

P4Q commands group definition. 2 total commands, 2 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.bluetooth.measurement.inputSignal.adetected.cte.qhsl.p4Q.clone()
```

Subgroups

6.1.1.5.1.23 TypePy

SCPI Command :

```
FETCh:BLUetooth:MEASurement<Instance>:ISIGnal:ADETected:CTE:QHSL:P4Q:TYPE
```

class TypePyCls

TypePy commands group definition. 1 total commands, 0 Subgroups, 1 group commands

fetch() → CtePacketType

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:ISIGnal:ADETected:CTE:QHSL:P4Q:TYPE
value: enums.CtePacketType = driver.bluetooth.measurement.inputSignal.adetected.
↪cte.qhsl.p4Q.typePy.fetch()
```

No command help available

Suppressed linked return values: reliability

return
packet_type: No help available

6.1.1.5.1.24 Units

SCPI Command :

```
FETCh:BLUetooth:MEASurement<Instance>:ISIGnal:ADETected:CTE:QHSL:P4Q:UNITs
```

class UnitsCls

Units commands group definition. 1 total commands, 0 Subgroups, 1 group commands

fetch() → int

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:ISIGnal:ADETected:CTE:QHSL:P4Q:UNITs
value: int = driver.bluetooth.measurement.inputSignal.adetected.cte.qhsl.p4Q.
↪units.fetch()
```

No command help available

Suppressed linked return values: reliability

return
duration_units: No help available

6.1.1.5.1.25 P5Q

class P5QCls

P5Q commands group definition. 2 total commands, 2 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.bluetooth.measurement.inputSignal.adetected.cte.qhsl.p5Q.clone()
```

Subgroups

6.1.1.5.1.26 TypePy

SCPI Command :

```
FETCh:BLUetooth:MEASurement<Instance>:ISIGnal:ADETected:CTE:QHSL:P5Q:TYPE
```

class TypePyCls

TypePy commands group definition. 1 total commands, 0 Subgroups, 1 group commands

fetch() → CtePacketType

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:ISIGnal:ADETected:CTE:QHSL:P5Q:TYPE
value: enums.CtePacketType = driver.bluetooth.measurement.inputSignal.adetected.
↪cte.qhsl.p5Q.typePy.fetch()
```

No command help available

Suppressed linked return values: reliability

return
packet_type: No help available

6.1.1.5.1.27 Units

SCPI Command :

```
FETCh:BLUetooth:MEASurement<Instance>:ISIGnal:ADETected:CTE:QHSL:P5Q:UNITs
```

class UnitsCls

Units commands group definition. 1 total commands, 0 Subgroups, 1 group commands

fetch() → int

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:ISIGnal:ADETected:CTE:QHSL:P5Q:UNITs
value: int = driver.bluetooth.measurement.inputSignal.adetected.cte.qhsl.p5Q.
↪units.fetch()
```

No command help available

Suppressed linked return values: reliability

return
duration_units: No help available

6.1.1.5.1.28 P6Q

class P6QCls

P6Q commands group definition. 2 total commands, 2 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.bluetooth.measurement.inputSignal.adetected.cte.qhsl.p6Q.clone()
```

Subgroups

6.1.1.5.1.29 TypePy

SCPI Command :

```
FETCh:BLUetooth:MEASurement<Instance>:ISIGnal:ADETected:CTE:QHSL:P6Q:TYPE
```

class TypePyCls

TypePy commands group definition. 1 total commands, 0 Subgroups, 1 group commands

fetch() → CtePacketType

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:ISIGnal:ADETected:CTE:QHSL:P6Q:TYPE
value: enums.CtePacketType = driver.bluetooth.measurement.inputSignal.adetected.
↪cte.qhsl.p6Q.typePy.fetch()
```

No command help available

Suppressed linked return values: reliability

return
packet_type: No help available

6.1.1.5.1.30 Units

SCPI Command :

```
FETCh:BLUetooth:MEASurement<Instance>:ISIGnal:ADETected:CTE:QHSL:P6Q:UNITs
```

class UnitsCls

Units commands group definition. 1 total commands, 0 Subgroups, 1 group commands

fetch() → int

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:ISIGnal:ADETected:CTE:QHSL:P6Q:UNITs
value: int = driver.bluetooth.measurement.inputSignal.adetected.cte.qhsl.p6Q.
↪units.fetch()
```

No command help available

Suppressed linked return values: reliability

return
duration_units: No help available

6.1.1.5.1.31 NoSlots

class NoSlotsCls

NoSlots commands group definition. 2 total commands, 2 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.bluetooth.measurement.inputSignal.adetected.noSlots.clone()
```

Subgroups

6.1.1.5.1.32 Brate

SCPI Command :

```
FETCh:BLUetooth:MEASurement<Instance>:ISIGnal:ADETected:NOSLots:BRATe
```

class BrateCls

Brate commands group definition. 1 total commands, 0 Subgroups, 1 group commands

fetch() → int

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:ISIGnal:ADETected:NOSLots:BRATe
value: int = driver.bluetooth.measurement.inputSignal.adetected.noSlots.brate.
↪fetch()
```

No command help available

Suppressed linked return values: reliability

return
no_of_off_slots: No help available

6.1.1.5.1.33 Edrate

SCPI Command :

```
FETCh:BLUetooth:MEASurement<Instance>:ISIGnal:ADETected:NOSLots:EDRate
```

class EdrateCls

Edrate commands group definition. 1 total commands, 0 Subgroups, 1 group commands

fetch() → int

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:ISIGnal:ADETected:NOSLots:EDRate
value: int = driver.bluetooth.measurement.inputSignal.adetected.noSlots.edrate.
↪fetch()
```

No command help available

Suppressed linked return values: reliability

return
no_of_off_slots: No help available

6.1.1.5.1.34 Pattern

class PatternCls

Pattern commands group definition. 4 total commands, 2 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.bluetooth.measurement.inputSignal.adetected.pattern.clone()
```

Subgroups

6.1.1.5.1.35 Brate

SCPI Command :

```
FETCh:BLUetooth:MEASurement<Instance>:ISIGnal:ADETected:PATTern[:BRATe]
```

class BrateCls

Brate commands group definition. 1 total commands, 0 Subgroups, 1 group commands

fetch() → DetectedPatternType

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:ISIGnal:ADETected:PATTern[:BRATe]
value: enums.DetectedPatternType = driver.bluetooth.measurement.inputSignal.
↪adetected.pattern.brate.fetch()
```

Returns the detected payload pattern type. A result is available after the CMP180 has auto-detected a packet (method RsCMPX_BluetoothMeas.Configure.Bluetooth.Measurement.InputSignal.dmode AUTO) .

Suppressed linked return values: reliability

return
pattern_type: P11: 10101010 P44: 11110000 OTHer: any pattern except P11, P44
ALTernating: the periodical change between the pattern P11 and P44

6.1.1.5.1.36 LowEnergy

class LowEnergyCls

LowEnergy commands group definition. 3 total commands, 3 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.bluetooth.measurement.inputSignal.adetected.pattern.lowEnergy.clone()
```

Subgroups

6.1.1.5.1.37 Le1M

SCPI Command :

```
FETCh:BLUetooth:MEASurement<Instance>:ISIGnal:ADETected:PATTern:LENergy[:LE1M]
```

class Le1MCls

Le1M commands group definition. 1 total commands, 0 Subgroups, 1 group commands

fetch() → DetectedPatternType

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:ISIGnal:ADETected:PATTern:LENergy[:LE1M]
value: enums.DetectedPatternType = driver.bluetooth.measurement.inputSignal.
↪adetected.pattern.lowEnergy.le1M.fetch()
```

Returns the detected payload pattern type. Commands for LE 1M PHY (...:LE1M...) and LE 2M PHY (...:LE2M...) are available. A result is available after the CMP180 has auto-detected a packet (method RsCMPX_BluetoothMeas.Configure.Bluetooth. Measurement.InputSignal.dmode AUTO) .

Suppressed linked return values: reliability

return

pattern_type: P11: 10101010 P44: 11110000 OTHer: any pattern except P11, P44 ALTernating: the periodical change between the pattern P11 and P44

6.1.1.5.1.38 Le2M

SCPI Command :

```
FETCh:BLUetooth:MEASurement<Instance>:ISIGnal:ADETected:PATTern:LENergy:LE2M
```

class Le2MCls

Le2M commands group definition. 1 total commands, 0 Subgroups, 1 group commands

fetch() → DetectedPatternType

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:ISIGnal:ADETected:PATTern:LENergy:LE2M
value: enums.DetectedPatternType = driver.bluetooth.measurement.inputSignal.
↪adetected.pattern.lowEnergy.le2M.fetch()
```

Returns the detected payload pattern type. Commands for LE 1M PHY (...:LE1M...) and LE 2M PHY (...:LE2M...) are available. A result is available after the CMP180 has auto-detected a packet (method RsCMPX_BluetoothMeas.Configure.Bluetooth. Measurement.InputSignal.dmode AUTO) .

Suppressed linked return values: reliability

return
pattern_type: P11: 10101010 P44: 11110000 OTHer: any pattern except P11, P44 ALTernating: the periodical change between the pattern P11 and P44

6.1.1.5.1.39 Lrange

SCPI Command :

```
FETCh:BLUetooth:MEASurement<Instance>:ISIGnal:ADETected:PATTern:LENergy:LRANge
```

class LrangeCls

Lrange commands group definition. 1 total commands, 0 Subgroups, 1 group commands

fetch() → LeRangePaternType

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:ISIGnal:ADETected:PATTern:LENergy:LRANge
value: enums.LeRangePaternType = driver.bluetooth.measurement.inputSignal.
↪adetected.pattern.lowEnergy.lrange.fetch()
```

Returns the detected payload pattern type for LE coded PHY. A result is available after the CMP180 has auto-detected a packet (method RsCMPX_BluetoothMeas.Configure.Bluetooth.Measurement.InputSignal.dmode AUTO) .

Suppressed linked return values: reliability

return
pattern_type: ALL0: '00000000' ALL1: '11111111' P11: '10101010' P44: '11110000' OTHer: any pattern except ALL0, ALL1, P11, P44, PRBS9 PRBS9: Pseudo random binary sequence of nine bits

6.1.1.5.1.40 PduType

class PduTypeCls

PduType commands group definition. 1 total commands, 1 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.bluetooth.measurement.inputSignal.adetected.pduType.clone()
```

Subgroups

6.1.1.5.1.41 LowEnergy

class LowEnergyCls

LowEnergy commands group definition. 1 total commands, 1 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.bluetooth.measurement.inputSignal.adetected.pduType.lowEnergy.clone()
```

Subgroups

6.1.1.5.1.42 Le1M

SCPI Command :

```
FETCh:BLUetooth:MEASurement<Instance>:ISIGnal:ADETected:PDUType:LENergy[:LE1M]
```

class Le1MCls

Le1M commands group definition. 1 total commands, 0 Subgroups, 1 group commands

class FetchStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Current_Pdu_Type: enums.PduType: No parameter help available
- Previous_Pdu_Type: enums.PduType: No parameter help available

fetch() → FetchStruct

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:ISIGnal:ADETected:PDUType:LENergy[:LE1M]
value: FetchStruct = driver.bluetooth.measurement.inputSignal.adetected.pduType.
↪lowEnergy.le1M.fetch()
```

No command help available

return

structure: for return value, see the help for FetchStruct structure arguments.

6.1.1.5.1.43 Plength

class PlengthCls

Plength commands group definition. 10 total commands, 4 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.bluetooth.measurement.inputSignal.adetected.plength.clone()
```

Subgroups

6.1.1.5.1.44 Brate

SCPI Command :

```
FETCh:BLUetooth:MEASurement<Instance>:ISIGnal:ADETected:PLENgth:BRATe
```

class BrateCls

Brate commands group definition. 1 total commands, 0 Subgroups, 1 group commands

fetch() → int

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:ISIGnal:ADETected:PLENgth:BRATe
value: int = driver.bluetooth.measurement.inputSignal.adetected.plength.brate.
↪fetch()
```

Returns the detected BR payload length. A result is available after the CMP180 has auto-detected a packet (method RsCMPX_BluetoothMeas.Configure.Bluetooth.Measurement.InputSignal.dmode AUTO) .

Suppressed linked return values: reliability

return
payload_length: No help available

6.1.1.5.1.45 Edrate

SCPI Command :

```
FETCh:BLUetooth:MEASurement<Instance>:ISIGnal:ADETected:PLENgth:EDRate
```

class EdrateCls

Edrate commands group definition. 1 total commands, 0 Subgroups, 1 group commands

fetch() → int

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:ISIGnal:ADETected:PLENgth:EDRate
value: int = driver.bluetooth.measurement.inputSignal.adetected.plength.edrate.
↪fetch()
```

Returns the detected EDR payload length. A result is available after the CMP180 has auto-detected a packet (method RsCMPX_BluetoothMeas.Configure.Bluetooth.Measurement.InputSignal.dmode AUTO) .

Suppressed linked return values: reliability

return
payload_length: No help available

6.1.1.5.1.46 LowEnergy

class LowEnergyCls

LowEnergy commands group definition. 3 total commands, 3 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.bluetooth.measurement.inputSignal.adetected.plength.lowEnergy.clone()
```

Subgroups

6.1.1.5.1.47 Le1M

SCPI Command :

```
FETCh:BLUetooth:MEASurement<Instance>:ISIGnal:ADETected:PLENgth:LENergy[:LE1M]
```

class Le1MCls

Le1M commands group definition. 1 total commands, 0 Subgroups, 1 group commands

fetch() → int

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:ISIGnal:ADETected:PLENgth:LENergy[:LE1M]
value: int = driver.bluetooth.measurement.inputSignal.adetected.plength.
↪lowEnergy.le1M.fetch()
```

Returns the detected payload length. Commands for uncoded LE 1M PHY (..:LE1M..) , LE 2M PHY (..:LE2M..) , and LE coded PHY (..:LRANge..) are available. A result is available after the CMP180 has auto-detected a packet (method RsCMPX_BluetoothMeas.Configure.Bluetooth.Measurement.InputSignal.dmode AUTO) .

Suppressed linked return values: reliability

return
payload_length: No help available

6.1.1.5.1.48 Le2M

SCPI Command :

```
FETCh:BLUetooth:MEASurement<Instance>:ISIGnal:ADETected:PLENgth:LENergy:LE2M
```

class Le2MCls

Le2M commands group definition. 1 total commands, 0 Subgroups, 1 group commands

fetch() → int

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:ISIGnal:ADETected:PLENgth:LENergy:LE2M
value: int = driver.bluetooth.measurement.inputSignal.adetected.plength.
↪lowEnergy.le2M.fetch()
```

Returns the detected payload length. Commands for uncoded LE 1M PHY (..:LE1M..) , LE 2M PHY (..:LE2M..) , and LE coded PHY (..:LRANge..) are available. A result is available after the CMP180 has auto-detected a packet (method RsCMPX_BluetoothMeas.Configure.Bluetooth.Measurement.InputSignal.dmode AUTO) .

Suppressed linked return values: reliability

return
payload_length: No help available

6.1.1.5.1.49 Lrange

SCPI Command :

```
FETCh:BLUetooth:MEASurement<Instance>:ISIGnal:ADETected:PLENgth:LENergy:LRANge
```

class LrangeCls

Lrange commands group definition. 1 total commands, 0 Subgroups, 1 group commands

fetch() → int

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:ISIGnal:ADETected:PLENgth:LENergy:LRANge
value: int = driver.bluetooth.measurement.inputSignal.adetected.plength.
↪lowEnergy.lrange.fetch()
```

Returns the detected payload length. Commands for uncoded LE 1M PHY (..:LE1M..) , LE 2M PHY (..:LE2M..) , and LE coded PHY (..:LRANge..) are available. A result is available after the CMP180 has auto-detected a packet (method RsCMPX_BluetoothMeas.Configure.Bluetooth.Measurement.InputSignal.dmode AUTO) .

Suppressed linked return values: reliability

return
payload_length: No help available

6.1.1.5.1.50 Qhsl

class QhslCls

Qhsl commands group definition. 5 total commands, 5 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.bluetooth.measurement.inputSignal.adetected.plength.qhsl.clone()
```

Subgroups

6.1.1.5.1.51 P2Q

SCPI Command :

```
FETCh:BLUetooth:MEASurement<Instance>:ISIGnal:ADETected:PLENgth:QHSL:P2Q
```

class P2QCls

P2Q commands group definition. 1 total commands, 0 Subgroups, 1 group commands

fetch() → int

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:ISIGnal:ADETected:PLENgth:QHSL:P2Q
value: int = driver.bluetooth.measurement.inputSignal.adetected.plength.qhsl.
↪p2Q.fetch()
```

No command help available

Suppressed linked return values: reliability

return
payload_length: No help available

6.1.1.5.1.52 P3Q

SCPI Command :

```
FETCh:BLUetooth:MEASurement<Instance>:ISIGnal:ADETected:PLENgth:QHSL:P3Q
```

class P3QCls

P3Q commands group definition. 1 total commands, 0 Subgroups, 1 group commands

fetch() → int

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:ISIGnal:ADETected:PLENgth:QHSL:P3Q
value: int = driver.bluetooth.measurement.inputSignal.adetected.plength.qhsl.
↪p3Q.fetch()
```

No command help available

Suppressed linked return values: reliability

return
payload_length: No help available

6.1.1.5.1.53 P4Q

SCPI Command :

```
FETCh:BLUetooth:MEASurement<Instance>:ISIGnal:ADETected:PLENgth:QHSL:P4Q
```

class P4QCls

P4Q commands group definition. 1 total commands, 0 Subgroups, 1 group commands

fetch() → int

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:ISIGnal:ADETected:PLENgth:QHSL:P4Q
value: int = driver.bluetooth.measurement.inputSignal.adetected.plength.qhsl.
↪p4Q.fetch()
```

No command help available

Suppressed linked return values: reliability

return
payload_length: No help available

6.1.1.5.1.54 P5Q

SCPI Command :

```
FETCh:BLUetooth:MEASurement<Instance>:ISIGnal:ADETected:PLENgth:QHSL:P5Q
```

class P5QCls

P5Q commands group definition. 1 total commands, 0 Subgroups, 1 group commands

fetch() → int

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:ISIGnal:ADETected:PLENgth:QHSL:P5Q
value: int = driver.bluetooth.measurement.inputSignal.adetected.plength.qhsl.
↪p5Q.fetch()
```

No command help available

Suppressed linked return values: reliability

return
payload_length: No help available

6.1.1.5.1.55 P6Q

SCPI Command :

```
FETCh:BLUetooth:MEASurement<Instance>:ISIGnal:ADETected:PLENgth:QHSL:P6Q
```

class P6QCls

P6Q commands group definition. 1 total commands, 0 Subgroups, 1 group commands

fetch() → int

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:ISIGnal:ADETected:PLENgth:QHSL:P6Q
value: int = driver.bluetooth.measurement.inputSignal.adetected.plength.qhsl.
↪p6Q.fetch()
```

No command help available

Suppressed linked return values: reliability

return
payload_length: No help available

6.1.1.5.1.56 Ptype

class PtypeCls

Ptype commands group definition. 5 total commands, 3 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.bluetooth.measurement.inputSignal.adetected.ptype.clone()
```

Subgroups

6.1.1.5.1.57 Brate

SCPI Command :

```
FETCh:BLUetooth:MEASurement<Instance>:ISIGnal:ADETected:PTYPe:BRATe
```

class BrateCls

Brate commands group definition. 1 total commands, 0 Subgroups, 1 group commands

fetch() → BrPacketType

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:ISIGnal:ADETected:PTYPe:BRATe
value: enums.BrPacketType = driver.bluetooth.measurement.inputSignal.adetected.
↪ptype.brate.fetch()
```

Returns the detected BR packet type. A result is available after the CMP180 has auto-detected a packet (method RsCMPX_BluetoothMeas.Configure.Bluetooth.Measurement.InputSignal.dmode AUTO) .

Suppressed linked return values: reliability

return
packet_type: No help available

6.1.1.5.1.58 Edrate

SCPI Command :

```
FETCh:BLUetooth:MEASurement<Instance>:ISIGnal:ADETected:PTYPe:EDRate
```

class EdrateCls

Edrate commands group definition. 1 total commands, 0 Subgroups, 1 group commands

fetch() → EdrPacketType

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:ISIGnal:ADETected:PTYPe:EDRate
value: enums.EdrPacketType = driver.bluetooth.measurement.inputSignal.adetected.
↪ptype.edrate.fetch()
```

Returns the detected EDR packet type. A result is available after the CMP180 has auto-detected a packet (method RsCMPX_BluetoothMeas.Configure.Bluetooth.Measurement.InputSignal.dmode AUTO) .

Suppressed linked return values: reliability

return
packet_type: 2-DH1, 2-DH3, 2-DH5, 3-DH1, 3-DH3, or 3-DH5 packets

6.1.1.5.1.59 LowEnergy

class LowEnergyCls

LowEnergy commands group definition. 3 total commands, 3 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.bluetooth.measurement.inputSignal.adetected.ptype.lowEnergy.clone()
```

Subgroups

6.1.1.5.1.60 Le1M

SCPI Command :

```
FETCh:BLUetooth:MEASurement<Instance>:ISIGnal:ADETected:PTYPe:LENergy[:LE1M]
```

class Le1MCls

Le1M commands group definition. 1 total commands, 0 Subgroups, 1 group commands

fetch() → LePacketType

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:ISIGnal:ADETected:PTYPe:LENergy[:LE1M]
value: enums.LePacketType = driver.bluetooth.measurement.inputSignal.adetected.
↪ptype.lowEnergy.le1M.fetch()
```

Returns the detected packet type for LE 1M PHY (uncoded) . A result is available after the CMP180 has auto-detected a packet (method RsCMPX_BluetoothMeas.Configure.Bluetooth.Measurement.InputSignal.dmode AUTO) .

Suppressed linked return values: reliability

return
packet_type: RFPHytest: LE test packet (direct test mode) ADVertiser: air interface packet with advertising channel PDU

6.1.1.5.1.61 Le2M

SCPI Command :

```
FETCh:BLUetooth:MEASurement<Instance>:ISIGnal:ADETected:PTYPe:LENergy:LE2M
```

class Le2MCls

Le2M commands group definition. 1 total commands, 0 Subgroups, 1 group commands

fetch() → LePacketType

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:ISIGnal:ADETected:PTYPe:LENergy:LE2M
value: enums.LePacketType = driver.bluetooth.measurement.inputSignal.adetected.
↪ptype.lowEnergy.le2M.fetch()
```

Returns the detected packet type for LE 2M PHY. A result is available after the CMP180 has auto-detected a packet (method RsCMPX_BluetoothMeas.Configure.Bluetooth.Measurement.InputSignal.dmode AUTO) .

Suppressed linked return values: reliability

return
packet_type: RFPHytest: LE test packet (direct test mode) ADVertiser: air interface packet with advertising channel PDU

6.1.1.5.1.62 Lrange

SCPI Command :

```
FETCh:BLUetooth:MEASurement<Instance>:ISIGnal:ADETected:PTYPe:LENergy:LRANge
```

class LrangeCls

Lrange commands group definition. 1 total commands, 0 Subgroups, 1 group commands

fetch() → LePacketType

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:ISIGnal:ADETected:PTYPe:LENergy:LRANge
value: enums.LePacketType = driver.bluetooth.measurement.inputSignal.adetected.
↪ptype.lowEnergy.lrange.fetch()
```

Returns the detected packet type for LE coded PHY. A result is available after the CMP180 has auto-detected a packet (method RsCMPX_BluetoothMeas.Configure.Bluetooth.Measurement.InputSignal.dmode AUTO) .

Suppressed linked return values: reliability

return
packet_type: RFPHytest: LE test packet (direct test mode) ADVertiser: air interface packet with advertising channel PDU

6.1.1.5.1.63 Qhsl

class QhslCls

Qhsl commands group definition. 1 total commands, 1 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.bluetooth.measurement.inputSignal.adetected.qhsl.clone()
```

Subgroups

6.1.1.5.1.64 Phy

SCPI Command :

```
FETCh:BLUetooth:MEASurement<Instance>:ISIGnal:ADETected:QHSL:PHY
```

class PhyCls

Phy commands group definition. 1 total commands, 0 Subgroups, 1 group commands

fetch() → DetectedPhyType

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:ISIGnal:ADETected:QHSL:PHY
value: enums.DetectedPhyType = driver.bluetooth.measurement.inputSignal.
↪adetected.qhsl.phy.fetch()
```

No command help available

Suppressed linked return values: reliability

return
detected_phy_type: No help available

6.1.1.6 MultiEval

SCPI Commands :

```
STOP:BLUetooth:MEASurement<Instance>:MEValuation
ABORt:BLUetooth:MEASurement<Instance>:MEValuation
INITiate:BLUetooth:MEASurement<Instance>:MEValuation
```

class MultiEvalCls

MultiEval commands group definition. 608 total commands, 10 Subgroups, 3 group commands

abort(*opc_timeout_ms: int = -1*) → None

```
# SCPI: ABORt:BLUetooth:MEASurement<Instance>:MEValuation
driver.bluetooth.measurement.multiEval.abort()


    INTRO_CMD_HELP: Starts, stops or aborts the measurement:

    - INITiate... starts or restarts the measurement. The measurement enters␣
↪the RUN state.
    - STOP... halts the measurement immediately. The measurement enters the RDY␣
↪state. Measurement results are kept. The resources remain allocated to the␣
↪measurement.
    - ABORt... halts the measurement immediately. The measurement enters the␣
↪OFF state. All measurement values are set to NAV. Allocated resources are␣
↪released.
```

Use FETCh...STATe? to query the current measurement state.

param opc_timeout_ms
Maximum time to wait in milliseconds, valid only for this call.

initiate(*opc_timeout_ms: int = -1*) → None

```
# SCPI: INITiate:BLUetooth:MEASurement<Instance>:MEValuation
driver.bluetooth.measurement.multiEval.initiate()


    INTRO_CMD_HELP: Starts, stops or aborts the measurement:

    - INITiate... starts or restarts the measurement. The measurement enters␣
↪the RUN state.
```

(continues on next page)

(continued from previous page)

```
    - STOP... halts the measurement immediately. The measurement enters the RDY␣
↪state. Measurement results are kept. The resources remain allocated to the␣
↪measurement.
    - ABORt... halts the measurement immediately. The measurement enters the␣
↪OFF state. All measurement values are set to NAV. Allocated resources are␣
↪released.
```

Use FETCh...STATe? to query the current measurement state.

param opc_timeout_ms
Maximum time to wait in milliseconds, valid only for this call.

stop() → None

```
# SCPI: STOP:BLUetooth:MEASurement<Instance>:MEValuation
driver.bluetooth.measurement.multiEval.stop()


    INTRO_CMD_HELP: Starts, stops or aborts the measurement:

    - INITiate... starts or restarts the measurement. The measurement enters␣
↪the RUN state.
    - STOP... halts the measurement immediately. The measurement enters the RDY␣
↪state. Measurement results are kept. The resources remain allocated to the␣
↪measurement.
    - ABORt... halts the measurement immediately. The measurement enters the␣
↪OFF state. All measurement values are set to NAV. Allocated resources are␣
↪released.
```

Use FETCh...STATe? to query the current measurement state.

stop_with_opc(*opc_timeout_ms: int = -1*) → None

```
# SCPI: STOP:BLUetooth:MEASurement<Instance>:MEValuation
driver.bluetooth.measurement.multiEval.stop_with_opc()


    INTRO_CMD_HELP: Starts, stops or aborts the measurement:

    - INITiate... starts or restarts the measurement. The measurement enters␣
↪the RUN state.
    - STOP... halts the measurement immediately. The measurement enters the RDY␣
↪state. Measurement results are kept. The resources remain allocated to the␣
↪measurement.
    - ABORt... halts the measurement immediately. The measurement enters the␣
↪OFF state. All measurement values are set to NAV. Allocated resources are␣
↪released.
```

Use FETCh...STATe? to query the current measurement state.

Same as stop, but waits for the operation to complete before continuing further. Use the RsCMPX_BluetoothMeas.utilities.opc_timeout_set() to set the timeout value.

param opc_timeout_ms
Maximum time to wait in milliseconds, valid only for this call.

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.bluetooth.measurement.multiEval.clone()
```

Subgroups

6.1.1.6.1 Frange

class FrangeCls

Frange commands group definition. 3 total commands, 1 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.bluetooth.measurement.multiEval.frange.clone()
```

Subgroups

6.1.1.6.1.1 Brate

class BrateCls

Brate commands group definition. 3 total commands, 1 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.bluetooth.measurement.multiEval.frange.brate.clone()
```

Subgroups

6.1.1.6.1.2 Current

SCPI Commands :

```
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:FRANge:BRATe:CURRent
READ:BLUetooth:MEASurement<Instance>:MEValuation:FRANge:BRATe:CURRent
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:FRANge:BRATe:CURRent
```

class CurrentCls

Current commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: 'Reliability indicator'

- Out_Of_Tol: float or bool: Out of tolerance result, i.e. the percentage of measurement intervals of the statistic count ([CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Scount#Frange CMDLINKRESOLVED]) exceeding the specified limit. Additional ON/OFF enables/disables the out of tolerance evaluation.
- Nominal_Power: float or bool: Average power during the carrier-on state
- Fl: float or bool: Lowest frequency at which the spectral power density drops below the specified threshold.
- Fh: float or bool: Highest frequency at which the spectral power density drops below the specified threshold.

class FetchStruct

Response structure. Fields:

- Reliability: int: 'Reliability indicator'
- Out_Of_Tol: float: Out of tolerance result, i.e. the percentage of measurement intervals of the statistic count ([CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Scount#Frange CMDLINKRESOLVED]) exceeding the specified limit. Additional ON/OFF enables/disables the out of tolerance evaluation.
- Nominal_Power: float: Average power during the carrier-on state
- Fl: float: Lowest frequency at which the spectral power density drops below the specified threshold.
- Fh: float: Highest frequency at which the spectral power density drops below the specified threshold.

class ReadStruct

Response structure. Fields:

- Reliability: int: 'Reliability indicator'
- Out_Of_Tol: float or bool: Out of tolerance result, i.e. the percentage of measurement intervals of the statistic count ([CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Scount#Frange CMDLINKRESOLVED]) exceeding the specified limit. Additional ON/OFF enables/disables the out of tolerance evaluation.
- Nominal_Power: float: Average power during the carrier-on state
- Fl: float: Lowest frequency at which the spectral power density drops below the specified threshold.
- Fh: float: Highest frequency at which the spectral power density drops below the specified threshold.

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:FRANge:BRATe:CURRent
value: CalculateStruct = driver.bluetooth.measurement.multiEval.frange.brate.
↪current.calculate()
```

Returns the Frequency Range results for BR. The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → FetchStruct

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:MEValuation:FRANge:BRATe:CURRent
value: FetchStruct = driver.bluetooth.measurement.multiEval.frange.brate.
↪current.fetch()
```

Returns the Frequency Range results for BR. The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for FetchStruct structure arguments.

read() → ReadStruct

```
# SCPI: READ:BLUetooth:MEASurement<Instance>:MEValuation:FRANge:BRATe:CURRent
value: ReadStruct = driver.bluetooth.measurement.multiEval.frange.brate.current.
↪read()
```

Returns the Frequency Range results for BR. The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for ReadStruct structure arguments.

6.1.1.6.2 ListPy

class ListPyCls

ListPy commands group definition. 20 total commands, 1 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.bluetooth.measurement.multiEval.listPy.clone()
```

Subgroups

6.1.1.6.2.1 Segment

RepCap Settings

```
# Range: S1 .. S128
rc = driver.bluetooth.measurement.multiEval.listPy.segment.repcap_segment_get()
driver.bluetooth.measurement.multiEval.listPy.segment.repcap_segment_set(repcap.Segment.
↪S1)
```

class SegmentCls

Segment commands group definition. 20 total commands, 5 Subgroups, 0 group commands Repeated Capability: Segment, default value after init: Segment.S1

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.bluetooth.measurement.multiEval.listPy.segment.clone()
```

Subgroups

6.1.1.6.2.2 Modulation

class ModulationCls

Modulation commands group definition. 13 total commands, 7 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.bluetooth.measurement.multiEval.listPy.segment.modulation.clone()
```

Subgroups

6.1.1.6.2.3 Average

SCPI Command :

```
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:LIST:SEGMent<nr>:MODulation:AVERage
```

class AverageCls

Average commands group definition. 2 total commands, 1 Subgroups, 1 group commands

class FetchStruct

Response structure. Fields:

- Reliability: int: 'Reliability indicator'
- Seg_Reliability: int: Reliability indicator for the segment. The meaning of the returned values is the same as for the common reliability indicator, see previous parameter.
- Out_Of_Tol: float: Percentage of measured bursts with failed limit check
- Nominal_Power: float: Average power during the carrier-on state
- Freq_Acc_Or_Init_Freq_Error: float: Frequency accuracy (BR, LE) or initial center frequency error i (EDR)
- Freq_Drift: float: Frequency drift (BR, LE)
- Max_Drift_Rate: float: Maximal drift rate (BR, LE)
- Delta_F_1_Avg: float: Frequency deviation results (BR, LE)
- Delta_F_1_Min: float: Frequency deviation results (BR, LE)
- Delta_F_1_Max: float: Frequency deviation results (BR, LE)
- Delta_F_2_Avg: float: Frequency deviation results (BR, LE)

- Delta_F_2_Min: float: Frequency deviation results (BR, LE)
- Delta_F_2_Max: float: Frequency deviation results (BR, LE)
- Omegai_Omega_0: float: Overall uncompensated frequency error (EDR)
- Omega_0_Max: float: Maximum compensated frequency error (EDR)
- Rms_Devm: float: RMS DEVM (EDR)
- Peak_Devm: float: Peak DEVM (EDR)
- Freq_Offset: float: Frequency offset (LE)
- Initial_Freq_Drift: float: Initial frequency drift (LE)

fetch(*segment=Segment.Default*) → FetchStruct

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:MEValuation:LIST:SEGMent<nr>
↪:MODulation:AVERage
value: FetchStruct = driver.bluetooth.measurement.multiEval.listPy.segment.
↪modulation.average.fetch(segment = repcap.Segment.Default)
```

Returns modulation single value results for segment<no> in list mode. The command returns all parameters listed below, independent of the selected list mode setup. However, only for some of the parameters measured values are available. For the other parameters, only an indicator is returned (e.g. NAV) .

param segment
optional repeated capability selector. Default value: S1 (settable in the interface 'Segment')

return
structure: for return value, see the help for FetchStruct structure arguments.

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.bluetooth.measurement.multiEval.listPy.segment.modulation.average.clone()
```

Subgroups

6.1.1.6.2.4 Extended

SCPI Command :

```
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:LIST:SEGMent<nr>
↪:MODulation:AVERage:EXTended
```

class ExtendedCls

Extended commands group definition. 1 total commands, 0 Subgroups, 1 group commands

class FetchStruct

Response structure. Fields:

- Reliability: int: 'Reliability indicator'

- Seg_Reliability: int: Reliability indicator for the segment. The meaning of the returned values is the same as for the common reliability indicator, see previous parameter.
- Out_Of_Tol: float: Percentage of measured bursts with failed limit check
- Nominal_Power: float: Average power during the carrier-on state
- Freq_Acc_Or_Init_Freq_Error: float: Frequency accuracy (BR, LE) or initial center frequency error i (EDR)
- Freq_Drift: float: Frequency drift (BR, LE)
- Max_Drift_Rate: float: Maximal drift rate (BR, LE)
- Delta_F_1_Avg: float: Frequency deviation results (BR, LE)
- Delta_F_1_Min: float: Frequency deviation results (BR, LE)
- Delta_F_1_Max: float: Frequency deviation results (BR, LE)
- Delta_F_2_Avg: float: Frequency deviation results (BR, LE)
- Delta_F_2_Min: float: Frequency deviation results (BR, LE)
- Delta_F_2_Max: float: Frequency deviation results (BR, LE)
- Omegai_Omega_0: float: No parameter help available
- Omega_0_Max: float: Maximum compensated frequency error (EDR)
- Rms_Devm: float: RMS DEVM (EDR)
- Peak_Devm: float: Peak DEVM (EDR)
- Freq_Offset: float: Frequency offset (LE)
- Init_Freq_Drift: float: Initial frequency drift (LE)
- Cte_Freq_Drift: float: Frequency drift of CTE portion
- Cte_Mx_Drift_Rate: float: No parameter help available
- Cte_Freq_Offset: float: Frequency offset of CTE portion
- Cte_Int_Frq_Drift: float: No parameter help available

fetch(*segment=Segment.Default*) → FetchStruct

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:MEValuation:LIST:SEGMent<nr>
↪:MODulation:AVERage:EXTended
value: FetchStruct = driver.bluetooth.measurement.multiEval.listPy.segment.
↪modulation.average.extended.fetch(segment = repcap.Segment.Default)
```

Returns modulation single average value results for segment<no> in list mode including Bluetooth version 5.0 and higher. The command returns all parameters listed below, independent of the selected list mode setup. However, only for some of the parameters measured values are available. For the other parameters, only an indicator is returned (e.g. NAV) .

param segment
optional repeated capability selector. Default value: S1 (settable in the interface 'Segment')

return
structure: for return value, see the help for FetchStruct structure arguments.

6.1.1.6.2.5 Current

SCPI Command :

```
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:LIST:SEGMent<nr>:MODulation:CURRent
```

class CurrentCls

Current commands group definition. 2 total commands, 1 Subgroups, 1 group commands

class FetchStruct

Response structure. Fields:

- Reliability: int: 'Reliability indicator'
- Seg_Reliability: int: Reliability indicator for the segment. The meaning of the returned values is the same as for the common reliability indicator, see previous parameter.
- Out_Of_Tol: float: Percentage of measured bursts with failed limit check
- Nominal_Power: float: Average power during the carrier-on state
- Freq_Acc_Or_Init_Freq_Error: float: Frequency accuracy (BR, LE) or initial center frequency error i (EDR)
- Freq_Drift: float: Frequency drift (BR, LE)
- Max_Drift_Rate: float: Maximal drift rate (BR, LE)
- Delta_F_1_Avg: float: Frequency deviation results (BR, LE)
- Delta_F_1_Min: float: Frequency deviation results (BR, LE)
- Delta_F_1_Max: float: Frequency deviation results (BR, LE)
- Delta_F_2_Avg: float: Frequency deviation results (BR, LE)
- Delta_F_2_Min: float: Frequency deviation results (BR, LE)
- Delta_F_2_Max: float: Frequency deviation results (BR, LE)
- Delta_F_299_P: float: Frequency deviation value f2 above which 99.9% of all measured f2 values occur (BR, LE) .
- Omegai_Omega_0: float: Overall uncompensated frequency error (EDR)
- Omega_0_Max: float: Maximum compensated frequency error (EDR)
- Rms_Devm: float: RMS DEVM (EDR)
- Peak_Devm: float: Peak DEVM (EDR)
- P_99_Devm: float: DEVM value below which 99% of all measured DEVM values occur (EDR) .
- Freq_Offset: float: Frequency offset (LE)
- Initial_Freq_Drift: float: Initial frequency drift (LE)

fetch(*segment=Segment.Default*) → FetchStruct

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:MEValuation:LIST:SEGMent<nr>
↪:MODulation:CURRent
value: FetchStruct = driver.bluetooth.measurement.multiEval.listPy.segment.
↪modulation.current.fetch(segment = repcap.Segment.Default)
```

Returns modulation single current value results for segment<no> in list mode. The command returns all parameters listed below, independent of the selected list mode setup. However, only for some of the parameters measured values are available. For the other parameters, only an indicator is returned (e.g. NAV) .

param segment
optional repeated capability selector. Default value: S1 (settable in the interface 'Segment')

return
structure: for return value, see the help for FetchStruct structure arguments.

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.bluetooth.measurement.multiEval.listPy.segment.modulation.current.clone()
```

Subgroups

6.1.1.6.2.6 Extended

SCPI Command :

```
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:LIST:SEGMent<nr>
↪:MODulation:CURRent:EXTended
```

class ExtendedCls

Extended commands group definition. 1 total commands, 0 Subgroups, 1 group commands

class FetchStruct

Response structure. Fields:

- Reliability: int: 'Reliability indicator'
- Seg_Reliability: int: Reliability indicator for the segment. The meaning of the returned values is the same as for the common reliability indicator, see previous parameter.
- Out_Of_Tol: float: Percentage of measured bursts with failed limit check
- Nominal_Power: float: Average power during the carrier-on state
- Freq_Acc_Or_Init_Freq_Error: float: Frequency accuracy (BR, LE) or initial center frequency error i (EDR)
- Freq_Drift: float: Frequency drift (BR, LE)
- Max_Drift_Rate: float: Maximal drift rate (BR, LE)
- Delta_F_1_Avg: float: Frequency deviation results (BR, LE)
- Delta_F_1_Min: float: Frequency deviation results (BR, LE)
- Delta_F_1_Max: float: Frequency deviation results (BR, LE)
- Delta_F_2_Avg: float: Frequency deviation results (BR, LE)
- Delta_F_2_Min: float: Frequency deviation results (BR, LE)
- Delta_F_2_Max: float: Frequency deviation results (BR, LE)

- Delta_F_299_P: float: Frequency deviation value f2 above which 99.9% of all measured f2 values occur (BR, LE) .
- Omegai_Omega_0: float: No parameter help available
- Omega_0_Max: float: Maximum compensated frequency error (EDR)
- Rms_Devm: float: RMS DEVM (EDR)
- Peak_Devm: float: Peak DEVM (EDR)
- P_99_Devm: float: DEVM value below which 99% of all measured DEVM values occur (EDR) .
- Freq_Offset: float: Frequency offset (LE)
- Init_Freq_Drift: float: Initial frequency drift (LE)
- Delta_F_199_P: float: Standard deviation of frequency deviation value f1 above which 99.9% of all measured f1 values occur (LE coded PHY) .
- Cte_Freq_Drift: float: Frequency drift of CTE portion
- Cte_Mx_Drift_Rate: float: No parameter help available
- Cte_Freq_Offset: float: Frequency offset of CTE portion
- Cte_Int_Frq_Drift: float: No parameter help available

fetch(*segment=Segment.Default*) → FetchStruct

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:MEValuation:LIST:SEGMent<nr>
↪:MODulation:CURRent:EXTended
value: FetchStruct = driver.bluetooth.measurement.multiEval.listPy.segment.
↪modulation.current.extended.fetch(segment = repcap.Segment.Default)
```

Returns modulation single current value results for segment<no> in list mode including Bluetooth version 5.0 and higher. The command returns all parameters listed below, independent of the selected list mode setup. However, only for some of the parameters measured values are available. For the other parameters, only an indicator is returned (e.g. NAV) .

param segment
optional repeated capability selector. Default value: S1 (settable in the interface 'Segment')

return
structure: for return value, see the help for FetchStruct structure arguments.

6.1.1.6.2.7 Maximum

SCPI Command :

```
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:LIST:SEGMent<nr>:MODulation:MAXimum
```

class MaximumCls

Maximum commands group definition. 2 total commands, 1 Subgroups, 1 group commands

class FetchStruct

Response structure. Fields:

- Reliability: int: 'Reliability indicator'

- Seg_Reliability: int: Reliability indicator for the segment. The meaning of the returned values is the same as for the common reliability indicator, see previous parameter.
- Out_Of_Tol: float: Percentage of measured bursts with failed limit check
- Nominal_Power: float: Average power during the carrier-on state
- Freq_Acc_Or_Init_Freq_Error: float: Frequency accuracy (BR, LE) or initial center frequency error i (EDR)
- Freq_Drift: float: Frequency drift (BR, LE)
- Max_Drift_Rate: float: Maximal drift rate (BR, LE)
- Delta_F_1_Avg: float: Frequency deviation results (BR, LE)
- Delta_F_1_Min: float: Frequency deviation results (BR, LE)
- Delta_F_1_Max: float: Frequency deviation results (BR, LE)
- Delta_F_2_Avg: float: Frequency deviation results (BR, LE)
- Delta_F_2_Min: float: Frequency deviation results (BR, LE)
- Delta_F_2_Max: float: Frequency deviation results (BR, LE)
- Omegai_Omega_0: float: Overall uncompensated frequency error (EDR)
- Omega_0_Max: float: Maximum compensated frequency error (EDR)
- Rms_Devm: float: RMS DEVM (EDR)
- Peak_Devm: float: Peak DEVM (EDR)
- Freq_Offset: float: Frequency offset (LE)
- Initial_Freq_Drift: float: Initial frequency drift (LE)

fetch(*segment=Segment.Default*) → FetchStruct

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:MEValuation:LIST:SEGMent<nr>
↪:MODulation:MAXimum
value: FetchStruct = driver.bluetooth.measurement.multiEval.listPy.segment.
↪modulation.maximum.fetch(segment = repcap.Segment.Default)
```

Returns modulation single value results for segment<no> in list mode. The command returns all parameters listed below, independent of the selected list mode setup. However, only for some of the parameters measured values are available. For the other parameters, only an indicator is returned (e.g. NAV) .

param segment
optional repeated capability selector. Default value: S1 (settable in the interface 'Segment')

return
structure: for return value, see the help for FetchStruct structure arguments.

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.bluetooth.measurement.multiEval.listPy.segment.modulation.maximum.clone()
```

Subgroups

6.1.1.6.2.8 Extended

SCPI Command :

```
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:LIST:SEGMent<nr>
↪:MODulation:MAXimum:EXTended
```

class ExtendedCls

Extended commands group definition. 1 total commands, 0 Subgroups, 1 group commands

class FetchStruct

Response structure. Fields:

- Reliability: int: 'Reliability indicator'
- Seg_Reliability: int: Reliability indicator for the segment. The meaning of the returned values is the same as for the common reliability indicator, see previous parameter.
- Out_Of_Tol: float: Percentage of measured bursts with failed limit check
- Nominal_Power: float: Average power during the carrier-on state
- Freq_Acc_Or_Init_Freq_Error: float: Frequency accuracy (BR, LE) or initial center frequency error i (EDR)
- Freq_Drift: float: Frequency drift (BR, LE)
- Max_Drift_Rate: float: Maximal drift rate (BR, LE)
- Delta_F_1_Avg: float: Frequency deviation results (BR, LE)
- Delta_F_1_Min: float: Frequency deviation results (BR, LE)
- Delta_F_1_Max: float: Frequency deviation results (BR, LE)
- Delta_F_2_Avg: float: Frequency deviation results (BR, LE)
- Delta_F_2_Min: float: Frequency deviation results (BR, LE)
- Delta_F_2_Max: float: Frequency deviation results (BR, LE)
- Omegai_Omega_0: float: No parameter help available
- Omega_0_Max: float: Maximum compensated frequency error (EDR)
- Rms_Devm: float: RMS DEVM (EDR)
- Peak_Devm: float: Peak DEVM (EDR)
- Freq_Offset: float: Frequency offset (LE)
- Init_Freq_Drift: float: Initial frequency drift (LE)
- Cte_Freq_Drift: float: No parameter help available

- Cte_Mx_Drift_Rate: float: No parameter help available
- Cte_Freq_Offset: float: No parameter help available
- Cte_Int_Frq_Drift: float: No parameter help available

fetch(*segment=Segment.Default*) → FetchStruct

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:MEValuation:LIST:SEGMent<nr>
↪:MODulation:MAXimum:EXTended
value: FetchStruct = driver.bluetooth.measurement.multiEval.listPy.segment.
↪modulation.maximum.extended.fetch(segment = repcap.Segment.Default)
```

Returns single maximum modulation results for segment<no> in list mode including Bluetooth version 5.0 and higher. The command returns all parameters listed below, independent of the selected list mode setup. However, only for some of the parameters measured values are available. For the other parameters, only an indicator is returned (e.g. NAV) .

param segment
optional repeated capability selector. Default value: S1 (settable in the interface 'Segment')

return
structure: for return value, see the help for FetchStruct structure arguments.

6.1.1.6.2.9 Minimum

SCPI Command :

```
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:LIST:SEGMent<nr>:MODulation:MINimum
```

class MinimumCls

Minimum commands group definition. 1 total commands, 0 Subgroups, 1 group commands

class FetchStruct

Response structure. Fields:

- Reliability: int: 'Reliability indicator'
- Seg_Reliability: int: Reliability indicator for the segment. The meaning of the returned values is the same as for the common reliability indicator, see previous parameter.
- Out_Of_Tol: float: Percentage of measured bursts with failed limit check
- Nominal_Power: float: Average power during the carrier-on state
- Delta_F_1_Avg: float: Frequency deviation results (BR, LE)
- Delta_F_1_Min: float: Frequency deviation results (BR, LE)
- Delta_F_1_Max: float: Frequency deviation results (BR, LE)
- Delta_F_2_Avg: float: Frequency deviation results (BR, LE)
- Delta_F_2_Min: float: Frequency deviation results (BR, LE)
- Delta_F_2_Max: float: Frequency deviation results (BR, LE)

fetch(*segment=Segment.Default*) → FetchStruct

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:MEValuation:LIST:SEGMent<nr>
↪:MODulation:MINimum
value: FetchStruct = driver.bluetooth.measurement.multiEval.listPy.segment.
↪modulation.minimum.fetch(segment = repcap.Segment.Default)
```

Returns modulation single value results for segment<no> in list mode. The command returns all parameters listed below, independent of the selected list mode setup. However, only for some of the parameters measured values are available. For the other parameters, only an indicator is returned (e.g. NAV) .

param segment
optional repeated capability selector. Default value: S1 (settable in the interface 'Segment')

return
structure: for return value, see the help for FetchStruct structure arguments.

6.1.1.6.2.10 StandardDev

SCPI Command :

```
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:LIST:SEGMent<nr>:MODulation:SDEViation
```

class StandardDevCls

StandardDev commands group definition. 2 total commands, 1 Subgroups, 1 group commands

class FetchStruct

Response structure. Fields:

- Reliability: int: 'Reliability indicator'
- Seg_Reliability: int: Reliability indicator for the segment. The meaning of the returned values is the same as for the common reliability indicator, see previous parameter.
- Out_Of_Tol: float: Percentage of measured bursts with failed limit check
- Nominal_Power: float: Standard deviation of average power during the carrier-on state
- Freq_Acc_Or_Init_Freq_Error: float: Standard deviation of frequency accuracy (BR, LE) or initial center frequency error i (EDR)
- Freq_Drift: float: Standard deviation of frequency drift (BR, LE)
- Max_Drift_Rate: float: Standard deviation of maximal drift rate (BR, LE)
- Delta_F_299_P: float: Standard deviation of frequency deviation value f2 above which 99.9% of all measured f2 values occur (BR, LE) .
- Omegai_Omega_0: float: Standard deviation of overall uncompensated frequency error (EDR)
- Omega_0_Max: float: Standard deviation of maximum compensated frequency error (EDR)
- Rms_Devm: float: Standard deviation of RMS DEVM (EDR)
- Peak_Devm: float: Standard deviation of peak DEVM (EDR)
- P_99_Devm: float: Standard deviation of DEVM value below which 99% of all measured DEVM values occur (EDR) .
- Freq_Offset: float: Standard deviation of frequency offset (LE)

- Initial_Freq_Drift: float: Standard deviation of initial frequency drift (LE)

fetch(*segment=Segment.Default*) → FetchStruct

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:MEValuation:LIST:SEGMent<nr>
↪:MODulation:SDEViation
value: FetchStruct = driver.bluetooth.measurement.multiEval.listPy.segment.
↪modulation.standardDev.fetch(segment = repcap.Segment.Default)
```

Returns modulation single value results for segment<no> in list mode. The command returns all parameters listed below, independent of the selected list mode setup. However, only for some of the parameters measured values are available. For the other parameters, only an indicator is returned (e.g. NAV) .

param segment
optional repeated capability selector. Default value: S1 (settable in the interface 'Segment')

return
structure: for return value, see the help for FetchStruct structure arguments.

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.bluetooth.measurement.multiEval.listPy.segment.modulation.standardDev.
↪clone()
```

Subgroups

6.1.1.6.2.11 Extended

SCPI Command :

```
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:LIST:SEGMent<nr>
↪:MODulation:SDEViation:EXTended
```

class ExtendedCls

Extended commands group definition. 1 total commands, 0 Subgroups, 1 group commands

class FetchStruct

Response structure. Fields:

- Reliability: int: 'Reliability indicator'
- Seg_Reliability: int: Reliability indicator for the segment. The meaning of the returned values is the same as for the common reliability indicator, see previous parameter.
- Out_Of_Tol: float: Percentage of measured bursts with failed limit check
- Nominal_Power: float: Standard deviation of average power during the carrier-on state
- Freq_Acc_Or_Init_Freq_Error: float: Standard deviation of frequency accuracy (BR, LE) or initial center frequency error i (EDR)
- Freq_Drift: float: Standard deviation of frequency drift (BR, LE)
- Max_Drift_Rate: float: Standard deviation of maximal drift rate (BR, LE)

- Delta_F_299_P: float: Standard deviation of frequency deviation value f2 above which 99.9% of all measured f2 values occur (BR, LE) .
- Omegai_Omega_0: float: No parameter help available
- Omega_0_Max: float: Standard deviation of maximum compensated frequency error (EDR)
- Rms_Devm: float: Standard deviation of RMS DEVM (EDR)
- Peak_Devm: float: Standard deviation of peak DEVM (EDR)
- P_99_Devm: float: Standard deviation of DEVM value below which 99% of all measured DEVM values occur (EDR) .
- Freq_Offset: float: Standard deviation of frequency offset (LE)
- Init_Freq_Drift: float: Standard deviation of initial frequency drift (LE)
- Delta_F_199_P: float: Standard deviation of frequency deviation value f1 above which 99.9% of all measured f1 values occur (LE coded PHY) .
- Cte_Freq_Drift: float: Frequency drift of CTE portion
- Cte_Mx_Drift_Rate: float: No parameter help available
- Cte_Freq_Offset: float: Frequency offset of CTE portion
- Cte_Int_Frq_Drift: float: No parameter help available

fetch(*segment=Segment.Default*) → FetchStruct

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:MEValuation:LIST:SEGMent<nr>
↪:MODulation:SDEViation:EXTended
value: FetchStruct = driver.bluetooth.measurement.multiEval.listPy.segment.
↪modulation.standardDev.extended.fetch(segment = repcap.Segment.Default)
```

Returns modulation single value results for segment<no> in list mode including Bluetooth version 5.0 and higher. The command returns all parameters listed below, independent of the selected list mode setup. However, only for some of the parameters measured values are available. For the other parameters, only an indicator is returned (e.g. NAV) .

param segment
optional repeated capability selector. Default value: S1 (settable in the interface 'Segment')

return
structure: for return value, see the help for FetchStruct structure arguments.

6.1.1.6.2.12 Xmaximum

SCPI Command :

```
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:LIST:SEGMent<nr>:MODulation:XMAXimum
```

class XmaximumCls

Xmaximum commands group definition. 2 total commands, 1 Subgroups, 1 group commands

class FetchStruct

Response structure. Fields:

- Reliability: int: 'Reliability indicator'

- Seg_Reliability: int: Reliability indicator for the segment. The meaning of the returned values is the same as for the common reliability indicator, see previous parameter.
- Out_Of_Tol: float: Percentage of measured bursts with failed limit check
- Nominal_Power: float: Average power during the carrier-on state
- Freq_Acc_Or_Init_Freq_Error: float: Frequency accuracy (BR, LE) or initial center frequency error i (EDR)
- Freq_Drift: float: Frequency drift (BR, LE)
- Max_Drift_Rate: float: Maximal drift rate (BR, LE)
- Freq_Offset: float: Frequency offset (LE)
- Initial_Freq_Drift: float: Initial frequency drift (LE)

fetch(*segment=Segment.Default*) → FetchStruct

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:MEValuation:LIST:SEGMent<nr>
↪:MODulation:XMAXimum
value: FetchStruct = driver.bluetooth.measurement.multiEval.listPy.segment.
↪modulation.xmaximum.fetch(segment = repcap.Segment.Default)
```

Returns modulation single extreme minimum and maximum (xmin, xmax) value results for segment<no> in list mode. The command returns all parameters listed below, independent of the selected list mode setup. However, only for some of the parameters measured values are available. For the other parameters, only an indicator is returned (e.g. NAV) .

param segment
optional repeated capability selector. Default value: S1 (settable in the interface 'Segment')

return
structure: for return value, see the help for FetchStruct structure arguments.

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.bluetooth.measurement.multiEval.listPy.segment.modulation.xmaximum.
↪clone()
```

Subgroups

6.1.1.6.2.13 Extended

SCPI Command :

```
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:LIST:SEGMent<nr>
↪:MODulation:XMAXimum:EXTended
```

class ExtendedCls

Extended commands group definition. 1 total commands, 0 Subgroups, 1 group commands

class FetchStruct

Response structure. Fields:

- Reliability: int: 'Reliability indicator'
- Seg_Reliability: int: Reliability indicator for the segment. The meaning of the returned values is the same as for the common reliability indicator, see previous parameter.
- Out_Of_Tol: float: Percentage of measured bursts with failed limit check
- Nominal_Power: float: Average power during the carrier-on state
- Freq_Acc_Or_Init_Freq_Error: float: Frequency accuracy (BR, LE) or initial center frequency error i (EDR)
- Freq_Drift: float: Frequency drift (BR, LE)
- Max_Drift_Rate: float: Maximal drift rate (BR, LE)
- Freq_Offset: float: Frequency offset (LE)
- Init_Freq_Drift: float: Initial frequency drift (LE)
- Cte_Freq_Drift: float: Frequency drift of CTE portion
- Cte_Mx_Drift_Rate: float: No parameter help available
- Cte_Freq_Offset: float: Frequency offset of CTE portion
- Cte_Int_Frq_Drift: float: No parameter help available

fetch(*segment=Segment.Default*) → FetchStruct

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:MEValuation:LIST:SEGMent<nr>
↪:MODulation:XMAXimum:EXTended
value: FetchStruct = driver.bluetooth.measurement.multiEval.listPy.segment.
↪modulation.xmaximum.extended.fetch(segment = repcap.Segment.Default)
```

Returns single extreme minimum and maximum (xmin and xmax) value modulation results for segment<no> in list mode including Bluetooth version 5.0 and higher. The command returns all parameters listed below, independent of the selected list mode setup. However, only for some of the parameters measured values are available. For the other parameters, only an indicator is returned (e.g. NAV) .

param segment
optional repeated capability selector. Default value: S1 (settable in the interface 'Segment')

return
structure: for return value, see the help for FetchStruct structure arguments.

6.1.1.6.2.14 Xminimum

SCPI Command :

```
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:LIST:SEGMent<nr>:MODulation:XMINimum
```

class XminimumCls

Xminimum commands group definition. 2 total commands, 1 Subgroups, 1 group commands

class FetchStruct

Response structure. Fields:

- Reliability: int: 'Reliability indicator'
- Seg_Reliability: int: Reliability indicator for the segment. The meaning of the returned values is the same as for the common reliability indicator, see previous parameter.
- Out_Of_Tol: float: Percentage of measured bursts with failed limit check
- Nominal_Power: float: Average power during the carrier-on state
- Freq_Acc_Or_Init_Freq_Error: float: Frequency accuracy (BR, LE) or initial center frequency error i (EDR)
- Freq_Drift: float: Frequency drift (BR, LE)
- Max_Drift_Rate: float: Maximal drift rate (BR, LE)
- Freq_Offset: float: Frequency offset (LE)
- Initial_Freq_Drift: float: Initial frequency drift (LE)

fetch(*segment=Segment.Default*) → FetchStruct

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:MEValuation:LIST:SEGMent<nr>
↪:MODulation:XMINimum
value: FetchStruct = driver.bluetooth.measurement.multiEval.listPy.segment.
↪modulation.xminimum.fetch(segment = repcap.Segment.Default)
```

Returns modulation single extreme minimum and maximum (xmin, xmax) value results for segment<no> in list mode. The command returns all parameters listed below, independent of the selected list mode setup. However, only for some of the parameters measured values are available. For the other parameters, only an indicator is returned (e.g. NAV) .

param segment
optional repeated capability selector. Default value: S1 (settable in the interface 'Segment')

return
structure: for return value, see the help for FetchStruct structure arguments.

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.bluetooth.measurement.multiEval.listPy.segment.modulation.xminimum.
↪clone()
```

Subgroups

6.1.1.6.2.15 Extended

SCPI Command :

```
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:LIST:SEGMent<nr>
↪:MODulation:XMINimum:EXTended
```

class ExtendedCls

Extended commands group definition. 1 total commands, 0 Subgroups, 1 group commands

class FetchStruct

Response structure. Fields:

- Reliability: int: 'Reliability indicator'
- Seg_Reliability: int: Reliability indicator for the segment. The meaning of the returned values is the same as for the common reliability indicator, see previous parameter.
- Out_Of_Tol: float: Percentage of measured bursts with failed limit check
- Nominal_Power: float: Average power during the carrier-on state
- Freq_Acc_Or_Init_Freq_Error: float: Frequency accuracy (BR, LE) or initial center frequency error i (EDR)
- Freq_Drift: float: Frequency drift (BR, LE)
- Max_Drift_Rate: float: Maximal drift rate (BR, LE)
- Freq_Offset: float: Frequency offset (LE)
- Init_Freq_Drift: float: Initial frequency drift (LE)
- Cte_Freq_Drift: float: Frequency drift of CTE portion
- Cte_Mx_Drift_Rate: float: No parameter help available
- Cte_Freq_Offset: float: Frequency offset of CTE portion
- Cte_Int_Frq_Drift: float: No parameter help available

fetch(*segment=Segment.Default*) → FetchStruct

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:MEValuation:LIST:SEGMent<nr>
↪:MODulation:XMINimum:EXTended
value: FetchStruct = driver.bluetooth.measurement.multiEval.listPy.segment.
↪modulation.xminimum.extended.fetch(segment = repcap.Segment.Default)
```

Returns single extreme minimum and maximum (xmin and xmax) value modulation results for segment<no> in list mode including Bluetooth version 5.0 and higher. The command returns all parameters listed below, independent of the selected list mode setup. However, only for some of the parameters measured values are available. For the other parameters, only an indicator is returned (e.g. NAV) .

param segment
optional repeated capability selector. Default value: S1 (settable in the interface 'Segment')

return
structure: for return value, see the help for FetchStruct structure arguments.

6.1.1.6.2.16 Pencoding

SCPI Command :

```
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:LIST:SEGMent<nr>:PENCoding
```

class PencodingCls

Pencoding commands group definition. 1 total commands, 0 Subgroups, 1 group commands

class FetchStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Seg_Reliability: int: No parameter help available
- Out_Of_Tol: float: No parameter help available
- Nominal_Power: float: No parameter help available
- Bit_Error_Rate: float: No parameter help available
- Packets_0_Errors: float: No parameter help available

fetch(*segment=Segment.Default*) → FetchStruct

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:MEValuation:LIST:SEGMent<nr>
↪:PENCoding
value: FetchStruct = driver.bluetooth.measurement.multiEval.listPy.segment.
↪pencoding.fetch(segment = repcap.Segment.Default)
```

No command help available

param segment
optional repeated capability selector. Default value: S1 (settable in the interface 'Segment')

return
structure: for return value, see the help for FetchStruct structure arguments.

6.1.1.6.2.17 PowerVsTime

class PowerVsTimeCls

PowerVsTime commands group definition. 4 total commands, 4 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.bluetooth.measurement.multiEval.listPy.segment.powerVsTime.clone()
```

Subgroups

6.1.1.6.2.18 Average

SCPI Command :

```
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:LIST:SEGMent<nr>:PVTime:AVERage
```

class AverageCls

Average commands group definition. 1 total commands, 0 Subgroups, 1 group commands

class FetchStruct

Response structure. Fields:

- Reliability: int: 'Reliability indicator'
- Seg_Reliability: int: Reliability indicator for the segment. The meaning of the returned values is the same as for the common reliability indicator, see previous parameter.
- Out_Of_Tol: float: Percentage of measured bursts with failed limit check
- Nominal_Power: float: Average power during the carrier-on state
- Peak_Power: float: Peak power
- Leakage_Power: float: Leakage power (BR, LE)
- Gfsk_Power: float: Average power within the GFSK modulated part of the burst (EDR)
- Dpsk_Power: float: Average power within the DPSK modulated part of the burst (EDR)
- Dpsk_Minus_Gfsk: float: Difference between the 8_DPSKPower and 7_GFSKPower (EDR)
- Guard_Period: float: Length of the guard band between the packet header and the EDR synchronization sequence (EDR)
- Peak_Minus_Avg: float: Difference between the peak power and the average power in the burst (LE)

fetch(*segment=Segment.Default*) → FetchStruct

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:MEValuation:LIST:SEGMent<nr>
↪:PVTime:AVERage
value: FetchStruct = driver.bluetooth.measurement.multiEval.listPy.segment.
↪powerVsTime.average.fetch(segment = repcap.Segment.Default)
```

Returns statistical power vs time single value results for segment<no> in list mode. The command returns all parameters listed below, independent of the selected list mode setup. However, only for some of the parameters measured values are available. For the other parameters, only an indicator is returned (e.g. NAV) .

param segment
optional repeated capability selector. Default value: S1 (settable in the interface 'Segment')

return
structure: for return value, see the help for FetchStruct structure arguments.

6.1.1.6.2.19 Current

SCPI Command :

```
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:LIST:SEGMent<nr>:PVTime:CURRent
```

class CurrentCls

Current commands group definition. 1 total commands, 0 Subgroups, 1 group commands

class FetchStruct

Response structure. Fields:

- Reliability: int: 'Reliability indicator'
- Seg_Reliability: int: Reliability indicator for the segment. The meaning of the returned values is the same as for the common reliability indicator, see previous parameter.
- Out_Of_Tol: float: Percentage of measured bursts with failed limit check
- Nominal_Power: float: Average power during the carrier-on state
- Peak_Power: float: Peak power
- Leakage_Power: float: Leakage power (BR, LE)
- Gfsk_Power: float: Average power within the GFSK modulated part of the burst (EDR)
- Dpsk_Power: float: Average power within the DPSK modulated part of the burst (EDR)
- Dpsk_Minus_Gfsk: float: Difference between the 8_DPSKPower and 7_GFSKPower (EDR)
- Guard_Period: float: Length of the guard band between the packet header and the EDR synchronization sequence (EDR)
- Peak_Minus_Avg: float: Difference between the peak power and the average power in the burst (LE)

fetch(*segment=Segment.Default*) → FetchStruct

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:MEValuation:LIST:SEGMent<nr>
↪:PVTime:CURRent
value: FetchStruct = driver.bluetooth.measurement.multiEval.listPy.segment.
↪powerVsTime.current.fetch(segment = repcap.Segment.Default)
```

Returns statistical power vs time single value results for segment<no> in list mode. The command returns all parameters listed below, independent of the selected list mode setup. However, only for some of the parameters measured values are available. For the other parameters, only an indicator is returned (e.g. NAV) .

param segment
optional repeated capability selector. Default value: S1 (settable in the interface 'Segment')

return
structure: for return value, see the help for FetchStruct structure arguments.

6.1.1.6.2.20 Maximum

SCPI Command :

```
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:LIST:SEGMent<nr>:PVTime:MAXimum
```

class MaximumCls

Maximum commands group definition. 1 total commands, 0 Subgroups, 1 group commands

class FetchStruct

Response structure. Fields:

- Reliability: int: 'Reliability indicator'
- Seg_Reliability: int: Reliability indicator for the segment. The meaning of the returned values is the same as for the common reliability indicator, see previous parameter.
- Out_Of_Tol: float: Percentage of measured bursts with failed limit check
- Nominal_Power: float: Average power during the carrier-on state
- Peak_Power: float: Peak power
- Leakage_Power: float: Leakage power (BR, LE)
- Gfsk_Power: float: Average power within the GFSK modulated part of the burst (EDR)
- Dpsk_Power: float: Average power within the DPSK modulated part of the burst (EDR)
- Dpsk_Minus_Gfsk: float: Difference between the 8_DPSKPower and 7_GFSKPower (EDR)
- Guard_Period: float: Length of the guard band between the packet header and the EDR synchronization sequence (EDR)
- Peak_Minus_Avg: float: Difference between the peak power and the average power in the burst (LE)

fetch(*segment=Segment.Default*) → FetchStruct

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:MEValuation:LIST:SEGMent<nr>
↪:PVTime:MAXimum
value: FetchStruct = driver.bluetooth.measurement.multiEval.listPy.segment.
↪powerVsTime.maximum.fetch(segment = repcap.Segment.Default)
```

Returns statistical power vs time single value results for segment<no> in list mode. The command returns all parameters listed below, independent of the selected list mode setup. However, only for some of the parameters measured values are available. For the other parameters, only an indicator is returned (e.g. NAV) .

param segment
optional repeated capability selector. Default value: S1 (settable in the interface 'Segment')

return
structure: for return value, see the help for FetchStruct structure arguments.

6.1.1.6.2.21 Minimum

SCPI Command :

```
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:LIST:SEGMent<nr>:PVTime:MINimum
```

class MinimumCls

Minimum commands group definition. 1 total commands, 0 Subgroups, 1 group commands

class FetchStruct

Response structure. Fields:

- Reliability: int: 'Reliability indicator'
- Seg_Reliability: int: Reliability indicator for the segment. The meaning of the returned values is the same as for the common reliability indicator, see previous parameter.
- Out_Of_Tol: float: Percentage of measured bursts with failed limit check
- Nominal_Power: float: Average power during the carrier-on state
- Peak_Power: float: Peak power
- Leakage_Power: float: Leakage power (BR, LE)
- Gfsk_Power: float: Average power within the GFSK modulated part of the burst (EDR)
- Dpsk_Power: float: Average power within the DPSK modulated part of the burst (EDR)
- Dpsk_Minus_Gfsk: float: Difference between the 8_DPSKPower and 7_GFSKPower (EDR)
- Guard_Period: float: Length of the guard band between the packet header and the EDR synchronization sequence (EDR)
- Peak_Minus_Avg: float: Difference between the peak power and the average power in the burst (LE)

fetch(*segment=Segment.Default*) → FetchStruct

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:MEValuation:LIST:SEGMent<nr>
↪:PVTime:MINimum
value: FetchStruct = driver.bluetooth.measurement.multiEval.listPy.segment.
↪powerVsTime.minimum.fetch(segment = repcap.Segment.Default)
```

Returns statistical power vs time single value results for segment<no> in list mode. The command returns all parameters listed below, independent of the selected list mode setup. However, only for some of the parameters measured values are available. For the other parameters, only an indicator is returned (e.g. NAV) .

param segment
optional repeated capability selector. Default value: S1 (settable in the interface 'Segment')

return
structure: for return value, see the help for FetchStruct structure arguments.

6.1.1.6.2.22 Sacp

class SacpCls

Sacp commands group definition. 1 total commands, 1 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.bluetooth.measurement.multiEval.listPy.segment.sacp.clone()
```

Subgroups

6.1.1.6.2.23 Ptx

SCPI Command :

```
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:LIST:SEGMent<nr>:SACP[:PTX]
```

class PtxCls

Ptx commands group definition. 1 total commands, 0 Subgroups, 1 group commands

class FetchStruct

Response structure. Fields:

- Reliability: int: 'Reliability indicator'
- Seg_Reliability: int: Reliability indicator for the segment. The meaning of the returned values is the same as for the common reliability indicator, see previous parameter.
- Out_Of_Tol: float: Percentage of measured bursts with failed limit check
- Nominal_Power: float: Average power during the carrier-on state
- No_Of_Exceptions: int: The number of exceptions, whose power is above Exception PTX.
- Ptx_Ref: float: Reference power obtained within the center channel (EDR)
- Ptx_N_26_Ch_N_1_Abs: float: No parameter help available
- Ptx_N_26_Ch_P_1_Abs: float: No parameter help available
- Ptx_N_26_Ch_N_1_Rel: float: No parameter help available
- Ptx_N_26_Ch_P_1_Rel: float: No parameter help available
- Acp: List[float]: No parameter help available

fetch(*segment=Segment.Default*) → FetchStruct

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:MEValuation:LIST:SEGMent<nr>
↪:SACP[:PTX]
value: FetchStruct = driver.bluetooth.measurement.multiEval.listPy.segment.sacp.
↪ptx.fetch(segment = repcap.Segment.Default)
```

Returns spectrum ACP single value results for segment<no> in list mode. The command returns all parameters listed below, independent of the selected list mode setup. However, only for some of the parameters measured values are available. For the other parameters, only an indicator is returned (e.g. NAV) .

param segment
optional repeated capability selector. Default value: S1 (settable in the interface 'Segment')

return
structure: for return value, see the help for FetchStruct structure arguments.

6.1.1.6.2.24 SoBw

class SoBwCls

SoBw commands group definition. 1 total commands, 1 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.bluetooth.measurement.multiEval.listPy.segment.soBw.clone()
```

Subgroups

6.1.1.6.2.25 Maximum

SCPI Command :

```
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:LIST:SEGMent<nr>:SOBW:MAXimum
```

class MaximumCls

Maximum commands group definition. 1 total commands, 0 Subgroups, 1 group commands

class FetchStruct

Response structure. Fields:

- Reliability: int: 'Reliability indicator'
- Seg_Reliability: int: Reliability indicator for the segment. The meaning of the returned values is the same as for the common reliability indicator, see previous parameter.
- Out_Of_Tol: float: Percentage of measured bursts with failed limit check.
- Nominal_Power: float: Average power during the carrier-on state.
- Peak_Emission: float: Peak power within the maximum spectral trace.
- Fl: float: The smallest frequency at which the transmit power drops 20 dB below the peak power.
- Fh: float: The highest frequency at which the transmit power drops 20 dB below the peak power.
- Fh_Min_Fl: float: Difference between the 7_fH and 6_fL

fetch(*segment=Segment.Default*) → FetchStruct

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:MEValuation:LIST:SEGMent<nr>
↪:SOBW:MAXimum
value: FetchStruct = driver.bluetooth.measurement.multiEval.listPy.segment.soBw.
↪maximum.fetch(segment = repcap.Segment.Default)
```

Returns spectrum occupied bandwidth (20 dB bandwidth) single value results for segment<no> in list mode. The 20 dB bandwidth measurement is available for BR bursts only.

param segment
optional repeated capability selector. Default value: S1 (settable in the interface 'Segment')

return
structure: for return value, see the help for FetchStruct structure arguments.

6.1.1.6.3 Modulation

class ModulationCls

Modulation commands group definition. 288 total commands, 6 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.bluetooth.measurement.multiEval.modulation.clone()
```

Subgroups

6.1.1.6.3.1 Brate

class BrateCls

Brate commands group definition. 22 total commands, 8 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.bluetooth.measurement.multiEval.modulation.brate.clone()
```

Subgroups

6.1.1.6.3.2 Average

SCPI Commands :

```
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:BRATe:AVERage
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:BRATe:AVERage
READ:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:BRATe:AVERage
```

class AverageCls

Average commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: 'Reliability indicator'
- Out_Of_Tol: float or bool: Out of tolerance result, i.e. the percentage of measurement intervals of the statistic count ([CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Scount#Modulation CMDLINKRESOLVED]) exceeding the specified limits.
- Delta_F_299_P_9: float or bool: Frequency deviation value f2 above which 99.9% of all measured f2 values occur
- Freq_Accuracy: float or bool: No parameter help available
- Freq_Drift: float or bool: No parameter help available
- Max_Drift: float or bool: No parameter help available
- Freq_Dev_Avg_F_1: float or bool: No parameter help available
- Freq_Dev_Min_F_1: float or bool: No parameter help available
- Freq_Dev_Max_F_1: float or bool: No parameter help available
- Freq_Dev_Avg_F_2: float or bool: No parameter help available
- Freq_Dev_Min_F_2: float or bool: No parameter help available
- Freq_Dev_Max_F_2: float or bool: No parameter help available
- Nominal_Power: float or bool: Average power during the carrier-on state
- Mod_Ratio: enums.ResultStatus2: Modulation ratio f2 avg / f1 avg

class ResultData

Response structure. Fields:

- Reliability: int: 'Reliability indicator'
- Out_Of_Tol: float: Out of tolerance result, i.e. the percentage of measurement intervals of the statistic count ([CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Scount#Modulation CMDLINKRESOLVED]) exceeding the specified limits.
- Delta_F_299_P_9: float: Frequency deviation value f2 above which 99.9% of all measured f2 values occur
- Freq_Accuracy: float: No parameter help available
- Freq_Drift: float: No parameter help available
- Max_Drift: float: No parameter help available
- Freq_Dev_Avg_F_1: float: No parameter help available
- Freq_Dev_Min_F_1: float: No parameter help available
- Freq_Dev_Max_F_1: float: No parameter help available
- Freq_Dev_Avg_F_2: float: No parameter help available
- Freq_Dev_Min_F_2: float: No parameter help available
- Freq_Dev_Max_F_2: float: No parameter help available
- Nominal_Power: float: Average power during the carrier-on state
- Mod_Ratio: float: Modulation ratio f2 avg / f1 avg

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:BRATe:AVERage
value: CalculateStruct = driver.bluetooth.measurement.multiEval.modulation.
↪brate.average.calculate()
```

Returns the current, average, absolute min (xmin) , absolute max (xmax) , and max modulation results for BR packets. The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → ResultData

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:BRATe:AVERage
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.brate.
↪average.fetch()
```

Returns the current, average, absolute min (xmin) , absolute max (xmax) , and max modulation results for BR packets. The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for ResultData structure arguments.

read() → ResultData

```
# SCPI: READ:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:BRATe:AVERage
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.brate.
↪average.read()
```

Returns the current, average, absolute min (xmin) , absolute max (xmax) , and max modulation results for BR packets. The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for ResultData structure arguments.

6.1.1.6.3.3 Current

SCPI Commands :

```
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:BRATe:CURRent
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:BRATe:CURRent
READ:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:BRATe:CURRent
```

class CurrentCls

Current commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: 'Reliability indicator'
- Out_Of_Tol: float or bool: Out of tolerance result, i.e. the percentage of measurement intervals of the statistic count ([CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Scount#Modulation CMDLINKRESOLVED]) exceeding the specified limits.
- Delta_F_299_P_9: float or bool: Frequency deviation value f2 above which 99.9% of all measured f2 values occur
- Freq_Accuracy: float or bool: No parameter help available
- Freq_Drift: float or bool: No parameter help available
- Max_Drift: float or bool: No parameter help available
- Freq_Dev_Avg_F_1: float or bool: No parameter help available
- Freq_Dev_Min_F_1: float or bool: No parameter help available
- Freq_Dev_Max_F_1: float or bool: No parameter help available
- Freq_Dev_Avg_F_2: float or bool: No parameter help available
- Freq_Dev_Min_F_2: float or bool: No parameter help available
- Freq_Dev_Max_F_2: float or bool: No parameter help available
- Nominal_Power: float or bool: Average power during the carrier-on state
- Mod_Ratio: enums.ResultStatus2: Modulation ratio f2 avg / f1 avg

class ResultData

Response structure. Fields:

- Reliability: int: 'Reliability indicator'
- Out_Of_Tol: float: Out of tolerance result, i.e. the percentage of measurement intervals of the statistic count ([CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Scount#Modulation CMDLINKRESOLVED]) exceeding the specified limits.
- Delta_F_299_P_9: float: Frequency deviation value f2 above which 99.9% of all measured f2 values occur
- Freq_Accuracy: float: No parameter help available
- Freq_Drift: float: No parameter help available
- Max_Drift: float: No parameter help available
- Freq_Dev_Avg_F_1: float: No parameter help available
- Freq_Dev_Min_F_1: float: No parameter help available
- Freq_Dev_Max_F_1: float: No parameter help available
- Freq_Dev_Avg_F_2: float: No parameter help available
- Freq_Dev_Min_F_2: float: No parameter help available
- Freq_Dev_Max_F_2: float: No parameter help available
- Nominal_Power: float: Average power during the carrier-on state
- Mod_Ratio: float: Modulation ratio f2 avg / f1 avg

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:BRATe:CURRent
value: CalculateStruct = driver.bluetooth.measurement.multiEval.modulation.
↪brate.current.calculate()
```

Returns the current, average, absolute min (xmin) , absolute max (xmax) , and max modulation results for BR packets. The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → ResultData

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:BRATe:CURRent
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.brate.
↪current.fetch()
```

Returns the current, average, absolute min (xmin) , absolute max (xmax) , and max modulation results for BR packets. The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for ResultData structure arguments.

read() → ResultData

```
# SCPI: READ:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:BRATe:CURRent
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.brate.
↪current.read()
```

Returns the current, average, absolute min (xmin) , absolute max (xmax) , and max modulation results for BR packets. The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for ResultData structure arguments.

6.1.1.6.3.4 Maximum

SCPI Commands :

```
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:BRATe:MAXimum
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:BRATe:MAXimum
READ:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:BRATe:MAXimum
```

class MaximumCls

Maximum commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: 'Reliability indicator'
- Out_Of_Tol: float or bool: Out of tolerance result, i.e. the percentage of measurement intervals of the statistic count ([CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Scount#Modulation CMDLINKRESOLVED]) exceeding the specified limits.
- Delta_F_299_P_9: float or bool: Frequency deviation value f2 above which 99.9% of all measured f2 values occur
- Freq_Accuracy: float or bool: No parameter help available
- Freq_Drift: float or bool: No parameter help available
- Max_Drift: float or bool: No parameter help available
- Freq_Dev_Avg_F_1: float or bool: No parameter help available
- Freq_Dev_Min_F_1: float or bool: No parameter help available
- Freq_Dev_Max_F_1: float or bool: No parameter help available
- Freq_Dev_Avg_F_2: float or bool: No parameter help available
- Freq_Dev_Min_F_2: float or bool: No parameter help available
- Freq_Dev_Max_F_2: float or bool: No parameter help available
- Nominal_Power: float or bool: Average power during the carrier-on state
- Mod_Ratio: enums.ResultStatus2: Modulation ratio f2 avg / f1 avg

class ResultData

Response structure. Fields:

- Reliability: int: 'Reliability indicator'
- Out_Of_Tol: float: Out of tolerance result, i.e. the percentage of measurement intervals of the statistic count ([CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Scount#Modulation CMDLINKRESOLVED]) exceeding the specified limits.
- Delta_F_299_P_9: float: Frequency deviation value f2 above which 99.9% of all measured f2 values occur
- Freq_Accuracy: float: No parameter help available
- Freq_Drift: float: No parameter help available
- Max_Drift: float: No parameter help available
- Freq_Dev_Avg_F_1: float: No parameter help available
- Freq_Dev_Min_F_1: float: No parameter help available
- Freq_Dev_Max_F_1: float: No parameter help available
- Freq_Dev_Avg_F_2: float: No parameter help available
- Freq_Dev_Min_F_2: float: No parameter help available
- Freq_Dev_Max_F_2: float: No parameter help available
- Nominal_Power: float: Average power during the carrier-on state
- Mod_Ratio: float: Modulation ratio f2 avg / f1 avg

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:BRATe:MAXimum
value: CalculateStruct = driver.bluetooth.measurement.multiEval.modulation.
↪brate.maximum.calculate()
```

Returns the current, average, absolute min (xmin) , absolute max (xmax) , and max modulation results for BR packets. The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → ResultData

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:BRATe:MAXimum
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.brate.
↪maximum.fetch()
```

Returns the current, average, absolute min (xmin) , absolute max (xmax) , and max modulation results for BR packets. The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for ResultData structure arguments.

read() → ResultData

```
# SCPI: READ:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:BRATe:MAXimum
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.brate.
↪maximum.read()
```

Returns the current, average, absolute min (xmin) , absolute max (xmax) , and max modulation results for BR packets. The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for ResultData structure arguments.

6.1.1.6.3.5 Minimum

SCPI Commands :

```
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:BRATe:MINimum
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:BRATe:MINimum
READ:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:BRATe:MINimum
```

class MinimumCls

Minimum commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: 'Reliability indicator'
- Out_Of_Tol: float or bool: Out of tolerance result, i.e. the percentage of measurement intervals of the statistic count ([CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Scount#Modulation CMDLINKRESOLVED]) exceeding the specified limits.
- Delta_F_299_P_9: float or bool: Frequency deviation value f2 above which 99.9% of all measured f2 values occur
- Freq_Dev_Avg_F_1: float or bool: No parameter help available
- Freq_Dev_Min_F_1: float or bool: No parameter help available
- Freq_Dev_Max_F_1: float or bool: No parameter help available
- Freq_Dev_Avg_F_2: float or bool: No parameter help available
- Freq_Dev_Min_F_2: float or bool: No parameter help available
- Freq_Dev_Max_F_2: float or bool: No parameter help available
- Nominal_Power: float or bool: Average power during the carrier-on state
- Mod_Ratio: enums.ResultStatus2: Modulation ratio f2 avg / f1 avg

class ResultData

Response structure. Fields:

- Reliability: int: 'Reliability indicator'
- Out_Of_Tol: float: Out of tolerance result, i.e. the percentage of measurement intervals of the statistic count ([CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Scount#Modulation CMDLINKRESOLVED]) exceeding the specified limits.
- Delta_F_299_P_9: float: Frequency deviation value f2 above which 99.9% of all measured f2 values occur
- Freq_Dev_Avg_F_1: float: No parameter help available
- Freq_Dev_Min_F_1: float: No parameter help available
- Freq_Dev_Max_F_1: float: No parameter help available
- Freq_Dev_Avg_F_2: float: No parameter help available
- Freq_Dev_Min_F_2: float: No parameter help available
- Freq_Dev_Max_F_2: float: No parameter help available
- Nominal_Power: float: Average power during the carrier-on state
- Mod_Ratio: float: Modulation ratio f2 avg / f1 avg

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:BRATe:MINimum
value: CalculateStruct = driver.bluetooth.measurement.multiEval.modulation.
↪brate.minimum.calculate()
```

Returns the minimum modulation results for BR packets. The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → ResultData

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:BRATe:MINimum
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.brate.
↪minimum.fetch()
```

Returns the minimum modulation results for BR packets. The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for ResultData structure arguments.

read() → ResultData

```
# SCPI: READ:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:BRATe:MINimum
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.brate.
↪minimum.read()
```

Returns the minimum modulation results for BR packets. The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for ResultData structure arguments.

6.1.1.6.3.6 StandardDev

SCPI Commands :

```
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:BRATe:SDEViation
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:BRATe:SDEViation
READ:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:BRATe:SDEViation
```

class StandardDevCls

StandardDev commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: 'Reliability indicator'
- Out_Of_Tol: float or bool: Out of tolerance result, i.e. the percentage of measurement intervals of the statistic count ([CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Scount#Modulation CMDLINKRESOLVED]) exceeding the specified limits.

- Delta_F_299_P_9: float or bool: Frequency deviation value f2 above which 99.9% of all measured f2 values occur
- Freq_Accuracy: float or bool: No parameter help available
- Freq_Drift: float or bool: No parameter help available
- Max_Drift: float or bool: No parameter help available
- Nominal_Power: float or bool: Average power during the carrier-on state
- Mod_Ratio: enums.ResultStatus2: Modulation ratio f2 avg / f1 avg

class ResultData

Response structure. Fields:

- Reliability: int: 'Reliability indicator'
- Out_Of_Tol: float: Out of tolerance result, i.e. the percentage of measurement intervals of the statistic count ([CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Scount#Modulation CMDLINKRESOLVED]) exceeding the specified limits.
- Delta_F_299_P_9: float: Frequency deviation value f2 above which 99.9% of all measured f2 values occur
- Freq_Accuracy: float: No parameter help available
- Freq_Drift: float: No parameter help available
- Max_Drift: float: No parameter help available
- Nominal_Power: float: Average power during the carrier-on state
- Mod_Ratio: float: Modulation ratio f2 avg / f1 avg

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:BRATe:SDEViation
value: CalculateStruct = driver.bluetooth.measurement.multiEval.modulation.
↪brate.standardDev.calculate()
```

Returns the standard deviation of the modulation results for BR packets. The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → ResultData

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:BRATe:SDEViation
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.brate.
↪standardDev.fetch()
```

Returns the standard deviation of the modulation results for BR packets. The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for ResultData structure arguments.

read() → ResultData

```
# SCPI: READ:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:BRATe:SDEViation
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.brate.
↪standardDev.read()
```

Returns the standard deviation of the modulation results for BR packets. The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for ResultData structure arguments.

6.1.1.6.3.7 Xmaximum

SCPI Commands :

```
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:BRATe:XMAXimum
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:BRATe:XMAXimum
READ:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:BRATe:XMAXimum
```

class XmaximumCls

Xmaximum commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: 'Reliability indicator'
- Out_Of_Tol: float or bool: Out of tolerance result, i.e. the percentage of measurement intervals of the statistic count ([CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Scount#Modulation CMDLINKRESOLVED]) exceeding the specified limits.
- Delta_F_299_P_9: float or bool: Frequency deviation value f2 above which 99.9% of all measured f2 values occur
- Freq_Accuracy: float or bool: No parameter help available
- Freq_Drift: float or bool: No parameter help available
- Max_Drift: float or bool: No parameter help available
- Freq_Dev_Avg_F_1: float or bool: No parameter help available
- Freq_Dev_Min_F_1: float or bool: No parameter help available
- Freq_Dev_Max_F_1: float or bool: No parameter help available
- Freq_Dev_Avg_F_2: float or bool: No parameter help available
- Freq_Dev_Min_F_2: float or bool: No parameter help available
- Freq_Dev_Max_F_2: float or bool: No parameter help available
- Nominal_Power: float or bool: Average power during the carrier-on state
- Mod_Ratio: enums.ResultStatus2: Modulation ratio f2 avg / f1 avg

class ResultData

Response structure. Fields:

- Reliability: int: 'Reliability indicator'
- Out_Of_Tol: float: Out of tolerance result, i.e. the percentage of measurement intervals of the statistic count ([CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Scount#Modulation CMDLINKRESOLVED]) exceeding the specified limits.
- Delta_F_299_P_9: float: Frequency deviation value f2 above which 99.9% of all measured f2 values occur
- Freq_Accuracy: float: No parameter help available
- Freq_Drift: float: No parameter help available
- Max_Drift: float: No parameter help available
- Freq_Dev_Avg_F_1: float: No parameter help available
- Freq_Dev_Min_F_1: float: No parameter help available
- Freq_Dev_Max_F_1: float: No parameter help available
- Freq_Dev_Avg_F_2: float: No parameter help available
- Freq_Dev_Min_F_2: float: No parameter help available
- Freq_Dev_Max_F_2: float: No parameter help available
- Nominal_Power: float: Average power during the carrier-on state
- Mod_Ratio: float: Modulation ratio f2 avg / f1 avg

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:BRATe:XMAXimum
value: CalculateStruct = driver.bluetooth.measurement.multiEval.modulation.
↪brate.xmaximum.calculate()
```

Returns the current, average, absolute min (xmin) , absolute max (xmax) , and max modulation results for BR packets. The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → ResultData

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:BRATe:XMAXimum
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.brate.
↪xmaximum.fetch()
```

Returns the current, average, absolute min (xmin) , absolute max (xmax) , and max modulation results for BR packets. The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for ResultData structure arguments.

read() → ResultData

```
# SCPI: READ:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:BRATe:XMAXimum
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.brate.
↪xmaximum.read()
```

Returns the current, average, absolute min (xmin) , absolute max (xmax) , and max modulation results for BR packets. The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for ResultData structure arguments.

6.1.1.6.3.8 Xminimum

SCPI Commands :

```
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:BRATe:XMINimum
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:BRATe:XMINimum
READ:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:BRATe:XMINimum
```

class XminimumCls

Xminimum commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: 'Reliability indicator'
- Out_Of_Tol: float or bool: Out of tolerance result, i.e. the percentage of measurement intervals of the statistic count ([CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Scount#Modulation CMDLINKRESOLVED]) exceeding the specified limits.
- Delta_F_299_P_9: float or bool: Frequency deviation value f2 above which 99.9% of all measured f2 values occur
- Freq_Accuracy: float or bool: No parameter help available
- Freq_Drift: float or bool: No parameter help available
- Max_Drift: float or bool: No parameter help available
- Freq_Dev_Avg_F_1: float or bool: No parameter help available
- Freq_Dev_Min_F_1: float or bool: No parameter help available
- Freq_Dev_Max_F_1: float or bool: No parameter help available
- Freq_Dev_Avg_F_2: float or bool: No parameter help available
- Freq_Dev_Min_F_2: float or bool: No parameter help available
- Freq_Dev_Max_F_2: float or bool: No parameter help available
- Nominal_Power: float or bool: Average power during the carrier-on state
- Mod_Ratio: enums.ResultStatus2: Modulation ratio f2 avg / f1 avg

class ResultData

Response structure. Fields:

- Reliability: int: 'Reliability indicator'
- Out_Of_Tol: float: Out of tolerance result, i.e. the percentage of measurement intervals of the statistic count ([CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Scount#Modulation CMDLINKRESOLVED]) exceeding the specified limits.
- Delta_F_299_P_9: float: Frequency deviation value f2 above which 99.9% of all measured f2 values occur
- Freq_Accuracy: float: No parameter help available
- Freq_Drift: float: No parameter help available
- Max_Drift: float: No parameter help available
- Freq_Dev_Avg_F_1: float: No parameter help available
- Freq_Dev_Min_F_1: float: No parameter help available
- Freq_Dev_Max_F_1: float: No parameter help available
- Freq_Dev_Avg_F_2: float: No parameter help available
- Freq_Dev_Min_F_2: float: No parameter help available
- Freq_Dev_Max_F_2: float: No parameter help available
- Nominal_Power: float: Average power during the carrier-on state
- Mod_Ratio: float: Modulation ratio f2 avg / f1 avg

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:BRATe:XMINimum
value: CalculateStruct = driver.bluetooth.measurement.multiEval.modulation.
↪brate.xminimum.calculate()
```

Returns the current, average, absolute min (xmin) , absolute max (xmax) , and max modulation results for BR packets. The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → ResultData

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:BRATe:XMINimum
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.brate.
↪xminimum.fetch()
```

Returns the current, average, absolute min (xmin) , absolute max (xmax) , and max modulation results for BR packets. The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for ResultData structure arguments.

read() → ResultData

```
# SCPI: READ:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:BRATe:XMINimum
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.brate.
↪xminimum.read()
```

Returns the current, average, absolute min (xmin) , absolute max (xmax) , and max modulation results for BR packets. The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for ResultData structure arguments.

6.1.1.6.3.9 YieldPy

SCPI Command :

```
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:BRATe:YIELd
```

class YieldPyCls

YieldPy commands group definition. 1 total commands, 0 Subgroups, 1 group commands

fetch() → List[float]

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:BRATe:YIELd
value: List[float] = driver.bluetooth.measurement.multiEval.modulation.brate.
↪yieldPy.fetch()
```

Returns the percentage of auto-detected BR packets with a particular pattern type. A result is available after the CMP180 has auto-detected a packet (method RsCMPX_BluetoothMeas.Configure.Bluetooth.Measurement.InputSignal.dmode AUTO) .

Suppressed linked return values: reliability

return
pattern_yield: Pattern yield for 11110000 patterns, 10101010 patterns, and any other patterns (3 values)

6.1.1.6.3.10 Cte

class CteCls

Cte commands group definition. 36 total commands, 1 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.bluetooth.measurement.multiEval.modulation.cte.clone()
```

Subgroups

6.1.1.6.3.11 LowEnergy

class LowEnergyCls

LowEnergy commands group definition. 36 total commands, 2 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.bluetooth.measurement.multiEval.modulation.cte.lowEnergy.clone()
```

Subgroups

6.1.1.6.3.12 Le1M

class Le1MCls

Le1M commands group definition. 18 total commands, 6 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.bluetooth.measurement.multiEval.modulation.cte.lowEnergy.le1M.clone()
```

Subgroups

6.1.1.6.3.13 Average

SCPI Commands :

```
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:CTE:LENergy:LE1M:AVERage
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:CTE:LENergy:LE1M:AVERage
READ:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:CTE:LENergy:LE1M:AVERage
```

class AverageCls

Average commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: 'Reliability indicator'

- Out_Of_Tol: float or bool: Out of tolerance result, i.e. the percentage of measurement intervals of the statistic count ([CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Scount#Modulation CMDLINKRESOLVED]) exceeding the specified limits, see 'Modulation limits (LE) '.
- Freq_Drift: float or bool: No parameter help available
- Max_Drift: float or bool: No parameter help available
- Freq_Offset: float or bool: No parameter help available
- Init_Freq_Drift: float or bool: No parameter help available

class ResultData

Response structure. Fields:

- Reliability: int: 'Reliability indicator'
- Out_Of_Tol: float: Out of tolerance result, i.e. the percentage of measurement intervals of the statistic count ([CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Scount#Modulation CMDLINKRESOLVED]) exceeding the specified limits, see 'Modulation limits (LE) '.
- Freq_Drift: float: No parameter help available
- Max_Drift: float: No parameter help available
- Freq_Offset: float: No parameter help available
- Init_Freq_Drift: float: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:CTE:LENergy:LE1M:AVERage
value: CalculateStruct = driver.bluetooth.measurement.multiEval.modulation.cte.
↪lowEnergy.le1M.average.calculate()
```

Returns current, average, standard deviation, absolute min (xmin) , absolute max (xmax) , and max CTE modulation results for LE uncoded PHY (LE 1M PHY, LE 2M PHY) . The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → ResultData

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:CTE:LENergy:LE1M:AVERage
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.cte.
↪lowEnergy.le1M.average.fetch()
```

Returns current, average, standard deviation, absolute min (xmin) , absolute max (xmax) , and max CTE modulation results for LE uncoded PHY (LE 1M PHY, LE 2M PHY) . The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for ResultData structure arguments.

read() → ResultData

```
# SCPI: READ:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:CTE:LENergy:LE1M:AVERage
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.cte.
↪lowEnergy.le1M.average.read()
```

Returns current, average, standard deviation, absolute min (xmin) , absolute max (xmax) , and max CTE modulation results for LE uncoded PHY (LE 1M PHY, LE 2M PHY) . The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for ResultData structure arguments.

6.1.1.6.3.14 Current

SCPI Commands :

```
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:CTE:LENergy:LE1M:CURRent
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:CTE:LENergy:LE1M:CURRent
READ:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:CTE:LENergy:LE1M:CURRent
```

class CurrentCls

Current commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: 'Reliability indicator'
- Out_Of_Tol: float or bool: Out of tolerance result, i.e. the percentage of measurement intervals of the statistic count ([CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Scount#Modulation CMDLINKRESOLVED]) exceeding the specified limits, see 'Modulation limits (LE) '.
- Freq_Drift: float or bool: No parameter help available
- Max_Drift: float or bool: No parameter help available
- Freq_Offset: float or bool: No parameter help available
- Init_Freq_Drift: float or bool: No parameter help available

class ResultData

Response structure. Fields:

- Reliability: int: 'Reliability indicator'
- Out_Of_Tol: float: Out of tolerance result, i.e. the percentage of measurement intervals of the statistic count ([CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Scount#Modulation CMDLINKRESOLVED]) exceeding the specified limits, see 'Modulation limits (LE) '.
- Freq_Drift: float: No parameter help available
- Max_Drift: float: No parameter help available
- Freq_Offset: float: No parameter help available

- Init_Freq_Drift: float: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:CTE:LENergy:LE1M:CURRent
value: CalculateStruct = driver.bluetooth.measurement.multiEval.modulation.cte.
↪lowEnergy.le1M.current.calculate()
```

Returns current, average, standard deviation, absolute min (xmin) , absolute max (xmax) , and max CTE modulation results for LE uncoded PHY (LE 1M PHY, LE 2M PHY) . The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → ResultData

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:CTE:LENergy:LE1M:CURRent
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.cte.
↪lowEnergy.le1M.current.fetch()
```

Returns current, average, standard deviation, absolute min (xmin) , absolute max (xmax) , and max CTE modulation results for LE uncoded PHY (LE 1M PHY, LE 2M PHY) . The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for ResultData structure arguments.

read() → ResultData

```
# SCPI: READ:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:CTE:LENergy:LE1M:CURRent
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.cte.
↪lowEnergy.le1M.current.read()
```

Returns current, average, standard deviation, absolute min (xmin) , absolute max (xmax) , and max CTE modulation results for LE uncoded PHY (LE 1M PHY, LE 2M PHY) . The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for ResultData structure arguments.

6.1.1.6.3.15 Maximum

SCPI Commands :

```
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:CTE:LENergy:LE1M:MAXimum
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:CTE:LENergy:LE1M:MAXimum
READ:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:CTE:LENergy:LE1M:MAXimum
```

class MaximumCls

Maximum commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: 'Reliability indicator'
- Out_Of_Tol: float or bool: Out of tolerance result, i.e. the percentage of measurement intervals of the statistic count ([CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Scount#Modulation CMDLINKRESOLVED]) exceeding the specified limits, see 'Modulation limits (LE) '.
- Freq_Drift: float or bool: No parameter help available
- Max_Drift: float or bool: No parameter help available
- Freq_Offset: float or bool: No parameter help available
- Init_Freq_Drift: float or bool: No parameter help available

class ResultData

Response structure. Fields:

- Reliability: int: 'Reliability indicator'
- Out_Of_Tol: float: Out of tolerance result, i.e. the percentage of measurement intervals of the statistic count ([CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Scount#Modulation CMDLINKRESOLVED]) exceeding the specified limits, see 'Modulation limits (LE) '.
- Freq_Drift: float: No parameter help available
- Max_Drift: float: No parameter help available
- Freq_Offset: float: No parameter help available
- Init_Freq_Drift: float: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:CTE:LENergy:LE1M:MAXimum
value: CalculateStruct = driver.bluetooth.measurement.multiEval.modulation.cte.
↪lowEnergy.le1M.maximum.calculate()
```

Returns current, average, standard deviation, absolute min (xmin) , absolute max (xmax) , and max CTE modulation results for LE uncoded PHY (LE 1M PHY, LE 2M PHY) . The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → ResultData

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:CTE:LENergy:LE1M:MAXimum
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.cte.
↪lowEnergy.le1M.maximum.fetch()
```

Returns current, average, standard deviation, absolute min (xmin) , absolute max (xmax) , and max CTE modulation results for LE uncoded PHY (LE 1M PHY, LE 2M PHY) . The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for ResultData structure arguments.

read() → ResultData

```
# SCPI: READ:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:CTE:LENergy:LE1M:MAXimum
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.cte.
↪lowEnergy.le1M.maximum.read()
```

Returns current, average, standard deviation, absolute min (xmin) , absolute max (xmax) , and max CTE modulation results for LE uncoded PHY (LE 1M PHY, LE 2M PHY) . The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for ResultData structure arguments.

6.1.1.6.3.16 StandardDev

SCPI Commands :

```
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:CTE:LENergy:LE1M:SDEViation
CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:CTE:LENergy:LE1M:SDEViation
READ:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:CTE:LENergy:LE1M:SDEViation
```

class StandardDevCls

StandardDev commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: 'Reliability indicator'
- Out_Of_Tol: float or bool: Out of tolerance result, i.e. the percentage of measurement intervals of the statistic count ([CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Scount#Modulation CMDLINKRESOLVED]) exceeding the specified limits, see 'Modulation limits (LE) '.
- Freq_Drift: float or bool: No parameter help available
- Max_Drift: float or bool: No parameter help available
- Freq_Offset: float or bool: No parameter help available

- Init_Freq_Drift: float or bool: No parameter help available

class ResultData

Response structure. Fields:

- Reliability: int: 'Reliability indicator'
- Out_Of_Tol: float: Out of tolerance result, i.e. the percentage of measurement intervals of the statistic count ([CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Scount#Modulation CMDLINKRESOLVED]) exceeding the specified limits, see 'Modulation limits (LE) '.
- Freq_Drift: float: No parameter help available
- Max_Drift: float: No parameter help available
- Freq_Offset: float: No parameter help available
- Init_Freq_Drift: float: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:CTE:LENergy:LE1M:SDEViation
value: CalculateStruct = driver.bluetooth.measurement.multiEval.modulation.cte.
↪lowEnergy.le1M.standardDev.calculate()
```

Returns current, average, standard deviation, absolute min (xmin) , absolute max (xmax) , and max CTE modulation results for LE uncoded PHY (LE 1M PHY, LE 2M PHY) . The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → ResultData

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:CTE:LENergy:LE1M:SDEViation
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.cte.
↪lowEnergy.le1M.standardDev.fetch()
```

Returns current, average, standard deviation, absolute min (xmin) , absolute max (xmax) , and max CTE modulation results for LE uncoded PHY (LE 1M PHY, LE 2M PHY) . The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for ResultData structure arguments.

read() → ResultData

```
# SCPI: READ:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:CTE:LENergy:LE1M:SDEViation
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.cte.
↪lowEnergy.le1M.standardDev.read()
```

Returns current, average, standard deviation, absolute min (xmin) , absolute max (xmax) , and max CTE modulation results for LE uncoded PHY (LE 1M PHY, LE 2M PHY) . The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for ResultData structure arguments.

6.1.1.6.3.17 Xmaximum

SCPI Commands :

```
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:CTE:LENergy:LE1M:XMAXimum
CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:CTE:LENergy:LE1M:XMAXimum
READ:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:CTE:LENergy:LE1M:XMAXimum
```

class XmaximumCls

Xmaximum commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: 'Reliability indicator'
- Out_Of_Tol: float or bool: Out of tolerance result, i.e. the percentage of measurement intervals of the statistic count ([CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Scount#Modulation CMDLINKRESOLVED]) exceeding the specified limits, see 'Modulation limits (LE) '.
- Freq_Drift: float or bool: No parameter help available
- Max_Drift: float or bool: No parameter help available
- Freq_Offset: float or bool: No parameter help available
- Init_Freq_Drift: float or bool: No parameter help available

class ResultData

Response structure. Fields:

- Reliability: int: 'Reliability indicator'
- Out_Of_Tol: float: Out of tolerance result, i.e. the percentage of measurement intervals of the statistic count ([CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Scount#Modulation CMDLINKRESOLVED]) exceeding the specified limits, see 'Modulation limits (LE) '.
- Freq_Drift: float: No parameter help available
- Max_Drift: float: No parameter help available
- Freq_Offset: float: No parameter help available
- Init_Freq_Drift: float: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:CTE:LENergy:LE1M:XMAXimum
value: CalculateStruct = driver.bluetooth.measurement.multiEval.modulation.cte.
↪lowEnergy.le1M.xmaximum.calculate()
```

Returns current, average, standard deviation, absolute min (xmin) , absolute max (xmax) , and max CTE modulation results for LE uncoded PHY (LE 1M PHY, LE 2M PHY) . The values described below are

returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → ResultData

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:CTE:LENergy:LE1M:XMAXimum
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.cte.
↪lowEnergy.le1M.xmaximum.fetch()
```

Returns current, average, standard deviation, absolute min (xmin) , absolute max (xmax) , and max CTE modulation results for LE uncoded PHY (LE 1M PHY, LE 2M PHY) . The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for ResultData structure arguments.

read() → ResultData

```
# SCPI: READ:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:CTE:LENergy:LE1M:XMAXimum
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.cte.
↪lowEnergy.le1M.xmaximum.read()
```

Returns current, average, standard deviation, absolute min (xmin) , absolute max (xmax) , and max CTE modulation results for LE uncoded PHY (LE 1M PHY, LE 2M PHY) . The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for ResultData structure arguments.

6.1.1.6.3.18 Xminimum

SCPI Commands :

```
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:CTE:LENergy:LE1M:XMINimum
CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:CTE:LENergy:LE1M:XMINimum
READ:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:CTE:LENergy:LE1M:XMINimum
```

class XminimumCls

Xminimum commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: 'Reliability indicator'
- Out_Of_Tol: float or bool: Out of tolerance result, i.e. the percentage of measurement intervals of the statistic count ([CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Scount#Modulation CMDLINKRESOLVED]) exceeding the specified limits, see 'Modulation limits (LE) '.

- Freq_Drift: float or bool: No parameter help available
- Max_Drift: float or bool: No parameter help available
- Freq_Offset: float or bool: No parameter help available
- Init_Freq_Drift: float or bool: No parameter help available

class ResultData

Response structure. Fields:

- Reliability: int: 'Reliability indicator'
- Out_Of_Tol: float: Out of tolerance result, i.e. the percentage of measurement intervals of the statistic count ([CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Scount#Modulation CMDLINKRESOLVED]) exceeding the specified limits, see 'Modulation limits (LE) '.
- Freq_Drift: float: No parameter help available
- Max_Drift: float: No parameter help available
- Freq_Offset: float: No parameter help available
- Init_Freq_Drift: float: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:CTE:LENergy:LE1M:XMINimum
value: CalculateStruct = driver.bluetooth.measurement.multiEval.modulation.cte.
↪lowEnergy.le1M.xminimum.calculate()
```

Returns current, average, standard deviation, absolute min (xmin) , absolute max (xmax) , and max CTE modulation results for LE uncoded PHY (LE 1M PHY, LE 2M PHY) . The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → ResultData

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:CTE:LENergy:LE1M:XMINimum
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.cte.
↪lowEnergy.le1M.xminimum.fetch()
```

Returns current, average, standard deviation, absolute min (xmin) , absolute max (xmax) , and max CTE modulation results for LE uncoded PHY (LE 1M PHY, LE 2M PHY) . The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for ResultData structure arguments.

read() → ResultData

```
# SCPI: READ:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:CTE:LENergy:LE1M:XMINimum
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.cte.
↪lowEnergy.le1M.xminimum.read()
```

Returns current, average, standard deviation, absolute min (xmin) , absolute max (xmax) , and max CTE modulation results for LE uncoded PHY (LE 1M PHY, LE 2M PHY) . The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for ResultData structure arguments.

6.1.1.6.3.19 Le2M

class Le2MCls

Le2M commands group definition. 18 total commands, 6 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.bluetooth.measurement.multiEval.modulation.cte.lowEnergy.le2M.clone()
```

Subgroups

6.1.1.6.3.20 Average

SCPI Commands :

```
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:CTE:LENergy:LE2M:AVERage
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:CTE:LENergy:LE2M:AVERage
READ:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:CTE:LENergy:LE2M:AVERage
```

class AverageCls

Average commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: 'Reliability indicator'
- Out_Of_Tol: float or bool: Out of tolerance result, i.e. the percentage of measurement intervals of the statistic count ([CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Scount#Modulation CMDLINKRESOLVED]) exceeding the specified limits, see 'Modulation limits (LE) '.
- Freq_Drift: float or bool: No parameter help available
- Max_Drift: float or bool: No parameter help available
- Freq_Offset: float or bool: No parameter help available
- Init_Freq_Drift: float or bool: No parameter help available

class ResultData

Response structure. Fields:

- Reliability: int: 'Reliability indicator'

- Out_Of_Tol: float: Out of tolerance result, i.e. the percentage of measurement intervals of the statistic count ([CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Scount#Modulation CMDLINKRESOLVED]) exceeding the specified limits, see 'Modulation limits (LE) '.
- Freq_Drift: float: No parameter help available
- Max_Drift: float: No parameter help available
- Freq_Offset: float: No parameter help available
- Init_Freq_Drift: float: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:CTE:LENergy:LE2M:AVERage
value: CalculateStruct = driver.bluetooth.measurement.multiEval.modulation.cte.
↪lowEnergy.le2M.average.calculate()
```

Returns current, average, standard deviation, absolute min (xmin) , absolute max (xmax) , and max CTE modulation results for LE uncoded PHY (LE 1M PHY, LE 2M PHY) . The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → ResultData

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:CTE:LENergy:LE2M:AVERage
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.cte.
↪lowEnergy.le2M.average.fetch()
```

Returns current, average, standard deviation, absolute min (xmin) , absolute max (xmax) , and max CTE modulation results for LE uncoded PHY (LE 1M PHY, LE 2M PHY) . The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for ResultData structure arguments.

read() → ResultData

```
# SCPI: READ:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:CTE:LENergy:LE2M:AVERage
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.cte.
↪lowEnergy.le2M.average.read()
```

Returns current, average, standard deviation, absolute min (xmin) , absolute max (xmax) , and max CTE modulation results for LE uncoded PHY (LE 1M PHY, LE 2M PHY) . The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for ResultData structure arguments.

6.1.1.6.3.21 Current

SCPI Commands :

```
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:CTE:LENergy:LE2M:CURRent
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:CTE:LENergy:LE2M:CURRent
READ:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:CTE:LENergy:LE2M:CURRent
```

class CurrentCls

Current commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: 'Reliability indicator'
- Out_Of_Tol: float or bool: Out of tolerance result, i.e. the percentage of measurement intervals of the statistic count ([CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Scount#Modulation CMDLINKRESOLVED]) exceeding the specified limits, see 'Modulation limits (LE) '.
- Freq_Drift: float or bool: No parameter help available
- Max_Drift: float or bool: No parameter help available
- Freq_Offset: float or bool: No parameter help available
- Init_Freq_Drift: float or bool: No parameter help available

class ResultData

Response structure. Fields:

- Reliability: int: 'Reliability indicator'
- Out_Of_Tol: float: Out of tolerance result, i.e. the percentage of measurement intervals of the statistic count ([CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Scount#Modulation CMDLINKRESOLVED]) exceeding the specified limits, see 'Modulation limits (LE) '.
- Freq_Drift: float: No parameter help available
- Max_Drift: float: No parameter help available
- Freq_Offset: float: No parameter help available
- Init_Freq_Drift: float: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:CTE:LENergy:LE2M:CURRent
value: CalculateStruct = driver.bluetooth.measurement.multiEval.modulation.cte.
↪lowEnergy.le2M.current.calculate()
```

Returns current, average, standard deviation, absolute min (xmin) , absolute max (xmax) , and max CTE modulation results for LE uncoded PHY (LE 1M PHY, LE 2M PHY) . The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → ResultData

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:CTE:LENergy:LE2M:CURRent
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.cte.
↪lowEnergy.le2M.current.fetch()
```

Returns current, average, standard deviation, absolute min (xmin) , absolute max (xmax) , and max CTE modulation results for LE uncoded PHY (LE 1M PHY, LE 2M PHY) . The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for ResultData structure arguments.

read() → ResultData

```
# SCPI: READ:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:CTE:LENergy:LE2M:CURRent
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.cte.
↪lowEnergy.le2M.current.read()
```

Returns current, average, standard deviation, absolute min (xmin) , absolute max (xmax) , and max CTE modulation results for LE uncoded PHY (LE 1M PHY, LE 2M PHY) . The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for ResultData structure arguments.

6.1.1.6.3.22 Maximum

SCPI Commands :

```
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:CTE:LENergy:LE2M:MAXimum
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:CTE:LENergy:LE2M:MAXimum
READ:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:CTE:LENergy:LE2M:MAXimum
```

class MaximumCls

Maximum commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: 'Reliability indicator'
- Out_Of_Tol: float or bool: Out of tolerance result, i.e. the percentage of measurement intervals of the statistic count ([CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Scount#Modulation CMDLINKRESOLVED]) exceeding the specified limits, see 'Modulation limits (LE) '.
- Freq_Drift: float or bool: No parameter help available
- Max_Drift: float or bool: No parameter help available
- Freq_Offset: float or bool: No parameter help available

- Init_Freq_Drift: float or bool: No parameter help available

class ResultData

Response structure. Fields:

- Reliability: int: 'Reliability indicator'
- Out_Of_Tol: float: Out of tolerance result, i.e. the percentage of measurement intervals of the statistic count ([CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Scount#Modulation CMDLINKRESOLVED]) exceeding the specified limits, see 'Modulation limits (LE) '.
- Freq_Drift: float: No parameter help available
- Max_Drift: float: No parameter help available
- Freq_Offset: float: No parameter help available
- Init_Freq_Drift: float: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:CTE:LENergy:LE2M:MAXimum
value: CalculateStruct = driver.bluetooth.measurement.multiEval.modulation.cte.
↪lowEnergy.le2M.maximum.calculate()
```

Returns current, average, standard deviation, absolute min (xmin) , absolute max (xmax) , and max CTE modulation results for LE uncoded PHY (LE 1M PHY, LE 2M PHY) . The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → ResultData

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:CTE:LENergy:LE2M:MAXimum
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.cte.
↪lowEnergy.le2M.maximum.fetch()
```

Returns current, average, standard deviation, absolute min (xmin) , absolute max (xmax) , and max CTE modulation results for LE uncoded PHY (LE 1M PHY, LE 2M PHY) . The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for ResultData structure arguments.

read() → ResultData

```
# SCPI: READ:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:CTE:LENergy:LE2M:MAXimum
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.cte.
↪lowEnergy.le2M.maximum.read()
```

Returns current, average, standard deviation, absolute min (xmin) , absolute max (xmax) , and max CTE modulation results for LE uncoded PHY (LE 1M PHY, LE 2M PHY) . The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for ResultData structure arguments.

6.1.1.6.3.23 StandardDev

SCPI Commands :

```
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:CTE:LENergy:LE2M:SDEViation
CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:CTE:LENergy:LE2M:SDEViation
READ:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:CTE:LENergy:LE2M:SDEViation
```

class StandardDevCls

StandardDev commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: 'Reliability indicator'
- Out_Of_Tol: float or bool: Out of tolerance result, i.e. the percentage of measurement intervals of the statistic count ([CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Scount#Modulation CMDLINKRESOLVED]) exceeding the specified limits, see 'Modulation limits (LE) '.
- Freq_Drift: float or bool: No parameter help available
- Max_Drift: float or bool: No parameter help available
- Freq_Offset: float or bool: No parameter help available
- Init_Freq_Drift: float or bool: No parameter help available

class ResultData

Response structure. Fields:

- Reliability: int: 'Reliability indicator'
- Out_Of_Tol: float: Out of tolerance result, i.e. the percentage of measurement intervals of the statistic count ([CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Scount#Modulation CMDLINKRESOLVED]) exceeding the specified limits, see 'Modulation limits (LE) '.
- Freq_Drift: float: No parameter help available
- Max_Drift: float: No parameter help available
- Freq_Offset: float: No parameter help available
- Init_Freq_Drift: float: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:CTE:LENergy:LE2M:SDEViation
value: CalculateStruct = driver.bluetooth.measurement.multiEval.modulation.cte.
↪lowEnergy.le2M.standardDev.calculate()
```

Returns current, average, standard deviation, absolute min (xmin) , absolute max (xmax) , and max CTE modulation results for LE uncoded PHY (LE 1M PHY, LE 2M PHY) . The values described below are

returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → ResultData

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:CTE:LENergy:LE2M:SDEViation
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.cte.
↪lowEnergy.le2M.standardDev.fetch()
```

Returns current, average, standard deviation, absolute min (xmin) , absolute max (xmax) , and max CTE modulation results for LE uncoded PHY (LE 1M PHY, LE 2M PHY) . The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for ResultData structure arguments.

read() → ResultData

```
# SCPI: READ:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:CTE:LENergy:LE2M:SDEViation
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.cte.
↪lowEnergy.le2M.standardDev.read()
```

Returns current, average, standard deviation, absolute min (xmin) , absolute max (xmax) , and max CTE modulation results for LE uncoded PHY (LE 1M PHY, LE 2M PHY) . The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for ResultData structure arguments.

6.1.1.6.3.24 Xmaximum

SCPI Commands :

```
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:CTE:LENergy:LE2M:XMAXimum
CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:CTE:LENergy:LE2M:XMAXimum
READ:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:CTE:LENergy:LE2M:XMAXimum
```

class XmaximumCls

Xmaximum commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: 'Reliability indicator'
- Out_Of_Tol: float or bool: Out of tolerance result, i.e. the percentage of measurement intervals of the statistic count ([CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Scount#Modulation CMDLINKRESOLVED]) exceeding the specified limits, see 'Modulation limits (LE) '.

- Freq_Drift: float or bool: No parameter help available
- Max_Drift: float or bool: No parameter help available
- Freq_Offset: float or bool: No parameter help available
- Init_Freq_Drift: float or bool: No parameter help available

class ResultData

Response structure. Fields:

- Reliability: int: 'Reliability indicator'
- Out_Of_Tol: float: Out of tolerance result, i.e. the percentage of measurement intervals of the statistic count ([CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Scount#Modulation CMDLINKRESOLVED]) exceeding the specified limits, see 'Modulation limits (LE) '.
- Freq_Drift: float: No parameter help available
- Max_Drift: float: No parameter help available
- Freq_Offset: float: No parameter help available
- Init_Freq_Drift: float: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:CTE:LENergy:LE2M:XMAXimum
value: CalculateStruct = driver.bluetooth.measurement.multiEval.modulation.cte.
↪lowEnergy.le2M.xmaximum.calculate()
```

Returns current, average, standard deviation, absolute min (xmin) , absolute max (xmax) , and max CTE modulation results for LE uncoded PHY (LE 1M PHY, LE 2M PHY) . The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → ResultData

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:CTE:LENergy:LE2M:XMAXimum
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.cte.
↪lowEnergy.le2M.xmaximum.fetch()
```

Returns current, average, standard deviation, absolute min (xmin) , absolute max (xmax) , and max CTE modulation results for LE uncoded PHY (LE 1M PHY, LE 2M PHY) . The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for ResultData structure arguments.

read() → ResultData

```
# SCPI: READ:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:CTE:LENergy:LE2M:XMAXimum
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.cte.
↪lowEnergy.le2M.xmaximum.read()
```

Returns current, average, standard deviation, absolute min (xmin) , absolute max (xmax) , and max CTE modulation results for LE uncoded PHY (LE 1M PHY, LE 2M PHY) . The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for ResultData structure arguments.

6.1.1.6.3.25 Xminimum

SCPI Commands :

```
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:CTE:LENergy:LE2M:XMINimum
CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:CTE:LENergy:LE2M:XMINimum
READ:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:CTE:LENergy:LE2M:XMINimum
```

class XminimumCls

Xminimum commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: 'Reliability indicator'
- Out_Of_Tol: float or bool: Out of tolerance result, i.e. the percentage of measurement intervals of the statistic count ([CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Scount#Modulation CMDLINKRESOLVED]) exceeding the specified limits, see 'Modulation limits (LE) '.
- Freq_Drift: float or bool: No parameter help available
- Max_Drift: float or bool: No parameter help available
- Freq_Offset: float or bool: No parameter help available
- Init_Freq_Drift: float or bool: No parameter help available

class ResultData

Response structure. Fields:

- Reliability: int: 'Reliability indicator'
- Out_Of_Tol: float: Out of tolerance result, i.e. the percentage of measurement intervals of the statistic count ([CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Scount#Modulation CMDLINKRESOLVED]) exceeding the specified limits, see 'Modulation limits (LE) '.
- Freq_Drift: float: No parameter help available
- Max_Drift: float: No parameter help available
- Freq_Offset: float: No parameter help available
- Init_Freq_Drift: float: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:CTE:LENergy:LE2M:XMINimum
value: CalculateStruct = driver.bluetooth.measurement.multiEval.modulation.cte.
↪lowEnergy.le2M.xminimum.calculate()
```

Returns current, average, standard deviation, absolute min (xmin) , absolute max (xmax) , and max CTE modulation results for LE uncoded PHY (LE 1M PHY, LE 2M PHY) . The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → ResultData

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:CTE:LENergy:LE2M:XMINimum
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.cte.
↪lowEnergy.le2M.xminimum.fetch()
```

Returns current, average, standard deviation, absolute min (xmin) , absolute max (xmax) , and max CTE modulation results for LE uncoded PHY (LE 1M PHY, LE 2M PHY) . The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for ResultData structure arguments.

read() → ResultData

```
# SCPI: READ:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:CTE:LENergy:LE2M:XMINimum
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.cte.
↪lowEnergy.le2M.xminimum.read()
```

Returns current, average, standard deviation, absolute min (xmin) , absolute max (xmax) , and max CTE modulation results for LE uncoded PHY (LE 1M PHY, LE 2M PHY) . The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for ResultData structure arguments.

6.1.1.6.3.26 Edrate

class EdrateCls

Edrate commands group definition. 14 total commands, 4 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.bluetooth.measurement.multiEval.modulation.edrate.clone()
```

Subgroups

6.1.1.6.3.27 Average

SCPI Commands :

```
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:EDRate:AVERage
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:EDRate:AVERage
READ:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:EDRate:AVERage
```

class AverageCls

Average commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: 'Reliability indicator'
- Out_Of_Tol: float or bool: Out of tolerance result, i.e. the percentage of measurement intervals of the statistic count ([CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Scount#Modulation CMDLINKRESOLVED]) exceeding the specified limits.
- Omega_I: float or bool: Initial center frequency error
- Omega_Iplus_Omega_0_Max: float or bool: Maximum compensated frequency error
- Omega_0_Max: float or bool: Maximum compensated frequency error
- Rms_Devm: float or bool: Differential EVM results
- Peak_Devm: float or bool: No parameter help available
- P_99_Devm: float or bool: No parameter help available
- Nominal_Power: float or bool: Average power during the carrier-on state

class ResultData

Response structure. Fields:

- Reliability: int: 'Reliability indicator'
- Out_Of_Tol: float: Out of tolerance result, i.e. the percentage of measurement intervals of the statistic count ([CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Scount#Modulation CMDLINKRESOLVED]) exceeding the specified limits.
- Omega_I: float: Initial center frequency error
- Omega_Iplus_Omega_0_Max: float: Maximum compensated frequency error
- Omega_0_Max: float: Maximum compensated frequency error
- Rms_Devm: float: Differential EVM results
- Peak_Devm: float: No parameter help available

- P_99_Devm: float: No parameter help available
- Nominal_Power: float: Average power during the carrier-on state

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:EDRate:AVERage
value: CalculateStruct = driver.bluetooth.measurement.multiEval.modulation.
↪edrate.average.calculate()
```

Returns the modulation results for EDR packets. The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → ResultData

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:EDRate:AVERage
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.edrate.
↪average.fetch()
```

Returns the modulation results for EDR packets. The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for ResultData structure arguments.

read() → ResultData

```
# SCPI: READ:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:EDRate:AVERage
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.edrate.
↪average.read()
```

Returns the modulation results for EDR packets. The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for ResultData structure arguments.

6.1.1.6.3.28 Current

SCPI Commands :

```
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:EDRate:CURRent
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:EDRate:CURRent
READ:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:EDRate:CURRent
```

class CurrentCls

Current commands group definition. 5 total commands, 1 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: 'Reliability indicator'
- Out_Of_Tol: float or bool: Out of tolerance result, i.e. the percentage of measurement intervals of the statistic count ([CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Scount#Modulation CMDLINKRESOLVED]) exceeding the specified limits.
- Omega_I: float or bool: Initial center frequency error
- Omega_Iplus_Omega_0_Max: float or bool: Maximum compensated frequency error
- Omega_0_Max: float or bool: Maximum compensated frequency error
- Rms_Devm: float or bool: Differential EVM results
- Peak_Devm: float or bool: No parameter help available
- P_99_Devm: float or bool: No parameter help available
- Nominal_Power: float or bool: Average power during the carrier-on state

class ResultData

Response structure. Fields:

- Reliability: int: 'Reliability indicator'
- Out_Of_Tol: float: Out of tolerance result, i.e. the percentage of measurement intervals of the statistic count ([CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Scount#Modulation CMDLINKRESOLVED]) exceeding the specified limits.
- Omega_I: float: Initial center frequency error
- Omega_Iplus_Omega_0_Max: float: Maximum compensated frequency error
- Omega_0_Max: float: Maximum compensated frequency error
- Rms_Devm: float: Differential EVM results
- Peak_Devm: float: No parameter help available
- P_99_Devm: float: No parameter help available
- Nominal_Power: float: Average power during the carrier-on state

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:EDRate:CURRent
value: CalculateStruct = driver.bluetooth.measurement.multiEval.modulation.
↪edrate.current.calculate()
```

Returns the modulation results for EDR packets. The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → ResultData

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:EDRate:CURRent
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.edrate.
↪current.fetch()
```

Returns the modulation results for EDR packets. The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for ResultData structure arguments.

read() → ResultData

```
# SCPI: READ:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:EDRate:CURRent
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.edrate.
↪current.read()
```

Returns the modulation results for EDR packets. The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for ResultData structure arguments.

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.bluetooth.measurement.multiEval.modulation.edrate.current.clone()
```

Subgroups

6.1.1.6.3.29 Extended

SCPI Commands :

```
READ:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:EDRate:CURRent:EXTended
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:EDRate:CURRent:EXTended
```

class ExtendedCls

Extended commands group definition. 2 total commands, 0 Subgroups, 2 group commands

class ResultData

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float: No parameter help available
- Omega_I: float: No parameter help available
- Omega_Iplus_Omega_0_Max: float: No parameter help available

- Omega_0_Max: float: No parameter help available
- Rms_Devm: float: No parameter help available
- Peak_Devm: float: No parameter help available
- P_99_Devm: float: No parameter help available
- Nominal_Power: float: No parameter help available
- P_99_Devm_Per: float: No parameter help available

fetch() → ResultData

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:EDRate:CURRent:EXTended
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.edrate.
↪current.extended.fetch()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

read() → ResultData

```
# SCPI: READ:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:EDRate:CURRent:EXTended
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.edrate.
↪current.extended.read()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

6.1.1.6.3.30 Maximum

SCPI Commands :

```
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:EDRate:MAXimum
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:EDRate:MAXimum
READ:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:EDRate:MAXimum
```

class MaximumCls

Maximum commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: 'Reliability indicator'
- Out_Of_Tol: float or bool: Out of tolerance result, i.e. the percentage of measurement intervals of the statistic count ([CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Scount#Modulation CMDLINKRESOLVED]) exceeding the specified limits.
- Omega_I: float or bool: Initial center frequency error

- Omega_Iplus_Omega_0_Max: float or bool: Maximum compensated frequency error
- Omega_0_Max: float or bool: Maximum compensated frequency error
- Rms_Devm: float or bool: Differential EVM results
- Peak_Devm: float or bool: No parameter help available
- P_99_Devm: float or bool: No parameter help available
- Nominal_Power: float or bool: Average power during the carrier-on state

class ResultData

Response structure. Fields:

- Reliability: int: 'Reliability indicator'
- Out_Of_Tol: float: Out of tolerance result, i.e. the percentage of measurement intervals of the statistic count ([CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Scount#Modulation CMDLINKRESOLVED]) exceeding the specified limits.
- Omega_I: float: Initial center frequency error
- Omega_Iplus_Omega_0_Max: float: Maximum compensated frequency error
- Omega_0_Max: float: Maximum compensated frequency error
- Rms_Devm: float: Differential EVM results
- Peak_Devm: float: No parameter help available
- P_99_Devm: float: No parameter help available
- Nominal_Power: float: Average power during the carrier-on state

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:EDRate:MAXimum
value: CalculateStruct = driver.bluetooth.measurement.multiEval.modulation.
↪edrate.maximum.calculate()
```

Returns the modulation results for EDR packets. The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → ResultData

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:EDRate:MAXimum
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.edrate.
↪maximum.fetch()
```

Returns the modulation results for EDR packets. The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for ResultData structure arguments.

read() → ResultData

```
# SCPI: READ:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:EDRate:MAXimum
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.edrate.
↪maximum.read()
```

Returns the modulation results for EDR packets. The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for ResultData structure arguments.

6.1.1.6.3.31 StandardDev

SCPI Commands :

```
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:EDRate:SDEViation
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:EDRate:SDEViation
READ:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:EDRate:SDEViation
```

class StandardDevCls

StandardDev commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: 'Reliability indicator'
- Out_Of_Tol: float or bool: Out of tolerance result, i.e. the percentage of measurement intervals of the statistic count ([CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Scount#Modulation CMDLINKRESOLVED]) exceeding the specified limits.
- Omega_I: float or bool: Standard deviation of the initial center frequency error
- Omega_Iplus_Omega_0_Max: float or bool: Standard deviation of the maximum compensated frequency error
- Omega_0_Max: float or bool: Standard deviation of the maximum compensated frequency error
- Rms_Devm: float or bool: Standard deviation of the differential EVM results
- Peak_Devm: float or bool: No parameter help available
- P_99_Devm: float or bool: No parameter help available
- Nominal_Power: float or bool: Standard deviation of the average power during the carrier-on state

class ResultData

Response structure. Fields:

- Reliability: int: 'Reliability indicator'
- Out_Of_Tol: float: Out of tolerance result, i.e. the percentage of measurement intervals of the statistic count ([CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Scount#Modulation CMDLINKRESOLVED]) exceeding the specified limits.

- Omega_I: float: Standard deviation of the initial center frequency error
- Omega_Iplus_Omega_0_Max: float: Standard deviation of the maximum compensated frequency error
- Omega_0_Max: float: Standard deviation of the maximum compensated frequency error
- Rms_Devm: float: Standard deviation of the differential EVM results
- Peak_Devm: float: No parameter help available
- P_99_Devm: float: No parameter help available
- Nominal_Power: float: Standard deviation of the average power during the carrier-on state

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:EDRate:SDEViation
value: CalculateStruct = driver.bluetooth.measurement.multiEval.modulation.
↪edrate.standardDev.calculate()
```

Returns the standard deviation of the modulation results for EDR packets. The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → ResultData

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:EDRate:SDEViation
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.edrate.
↪standardDev.fetch()
```

Returns the standard deviation of the modulation results for EDR packets. The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for ResultData structure arguments.

read() → ResultData

```
# SCPI: READ:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:EDRate:SDEViation
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.edrate.
↪standardDev.read()
```

Returns the standard deviation of the modulation results for EDR packets. The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for ResultData structure arguments.

6.1.1.6.3.32 LowEnergy

class LowEnergyCls

LowEnergy commands group definition. 68 total commands, 3 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.bluetooth.measurement.multiEval.modulation.lowEnergy.clone()
```

Subgroups

6.1.1.6.3.33 Le1M

class Le1MCls

Le1M commands group definition. 22 total commands, 8 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.bluetooth.measurement.multiEval.modulation.lowEnergy.le1M.clone()
```

Subgroups

6.1.1.6.3.34 Average

SCPI Commands :

```
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:LENergy[:LE1M]:AVERage
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:LENergy[:LE1M]:AVERage
READ:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:LENergy[:LE1M]:AVERage
```

class AverageCls

Average commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: 'Reliability indicator'
- Out_Of_Tol: float or bool: Out of tolerance result, i.e. the percentage of measurement intervals of the statistic count ([CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Scount#Modulation CMDLINKRESOLVED]) exceeding the specified limits, see 'Modulation limits (LE) '.
- Delta_F_299_P_9: float or bool: Frequency deviation value f2 above which 99.9% of all measured f2 values occur
- Freq_Accuracy: float or bool: No parameter help available

- Freq_Drift: float or bool: No parameter help available
- Max_Drift: float or bool: No parameter help available
- Freq_Dev_Avg_F_1: float or bool: No parameter help available
- Freq_Dev_Min_F_1: float or bool: No parameter help available
- Freq_Dev_Max_F_1: float or bool: No parameter help available
- Freq_Dev_Avg_F_2: float or bool: No parameter help available
- Freq_Dev_Min_F_2: float or bool: No parameter help available
- Freq_Dev_Max_F_2: float or bool: No parameter help available
- Nominal_Power: float or bool: Average power during the carrier-on state
- Mod_Ratio: enums.ResultStatus2: Modulation ratio f2 avg / f1 avg
- Freq_Offset: float or bool: No parameter help available
- Init_Freq_Drift: float or bool: No parameter help available

class ResultData

Response structure. Fields:

- Reliability: int: 'Reliability indicator'
- Out_Of_Tol: float: Out of tolerance result, i.e. the percentage of measurement intervals of the statistic count ([CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Scount#Modulation CMDLINKRESOLVED]) exceeding the specified limits, see 'Modulation limits (LE) '.
- Delta_F_299_P_9: float: Frequency deviation value f2 above which 99.9% of all measured f2 values occur
- Freq_Accuracy: float: No parameter help available
- Freq_Drift: float: No parameter help available
- Max_Drift: float: No parameter help available
- Freq_Dev_Avg_F_1: float: No parameter help available
- Freq_Dev_Min_F_1: float: No parameter help available
- Freq_Dev_Max_F_1: float: No parameter help available
- Freq_Dev_Avg_F_2: float: No parameter help available
- Freq_Dev_Min_F_2: float: No parameter help available
- Freq_Dev_Max_F_2: float: No parameter help available
- Nominal_Power: float: Average power during the carrier-on state
- Mod_Ratio: float: Modulation ratio f2 avg / f1 avg
- Freq_Offset: float: No parameter help available
- Init_Freq_Drift: float: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:LENergy[:LE1M]:AVERage
value: CalculateStruct = driver.bluetooth.measurement.multiEval.modulation.
↪lowEnergy.le1M.average.calculate()
```

Returns current, average, absolute min (xmin) , absolute max (xmax) , and max modulation results for LE uncoded PHY (LE 1M PHY, LE 2M PHY) , see 'Square TX Measurement - modulation statistics'. The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → ResultData

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:LENergy[:LE1M]:AVERage
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.lowEnergy.
↪le1M.average.fetch()
```

Returns current, average, absolute min (xmin) , absolute max (xmax) , and max modulation results for LE uncoded PHY (LE 1M PHY, LE 2M PHY) , see 'Square TX Measurement - modulation statistics'. The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for ResultData structure arguments.

read() → ResultData

```
# SCPI: READ:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:LENergy[:LE1M]:AVERage
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.lowEnergy.
↪le1M.average.read()
```

Returns current, average, absolute min (xmin) , absolute max (xmax) , and max modulation results for LE uncoded PHY (LE 1M PHY, LE 2M PHY) , see 'Square TX Measurement - modulation statistics'. The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for ResultData structure arguments.

6.1.1.6.3.35 Current

SCPI Commands :

```
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:LENergy[:LE1M]:CURRent
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:LENergy[:LE1M]:CURRent
READ:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:LENergy[:LE1M]:CURRent
```

class CurrentCls

Current commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: 'Reliability indicator'

- Out_Of_Tol: float or bool: Out of tolerance result, i.e. the percentage of measurement intervals of the statistic count ([CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Scount#Modulation CMDLINKRESOLVED]) exceeding the specified limits, see 'Modulation limits (LE) '.
- Delta_F_299_P_9: float or bool: Frequency deviation value f2 above which 99.9% of all measured f2 values occur
- Freq_Accuracy: float or bool: No parameter help available
- Freq_Drift: float or bool: No parameter help available
- Max_Drift: float or bool: No parameter help available
- Freq_Dev_Avg_F_1: float or bool: No parameter help available
- Freq_Dev_Min_F_1: float or bool: No parameter help available
- Freq_Dev_Max_F_1: float or bool: No parameter help available
- Freq_Dev_Avg_F_2: float or bool: No parameter help available
- Freq_Dev_Min_F_2: float or bool: No parameter help available
- Freq_Dev_Max_F_2: float or bool: No parameter help available
- Nominal_Power: float or bool: Average power during the carrier-on state
- Mod_Ratio: enums.ResultStatus2: Modulation ratio f2 avg / f1 avg
- Freq_Offset: float or bool: No parameter help available
- Init_Freq_Drift: float or bool: No parameter help available

class ResultData

Response structure. Fields:

- Reliability: int: 'Reliability indicator'
- Out_Of_Tol: float: Out of tolerance result, i.e. the percentage of measurement intervals of the statistic count ([CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Scount#Modulation CMDLINKRESOLVED]) exceeding the specified limits, see 'Modulation limits (LE) '.
- Delta_F_299_P_9: float: Frequency deviation value f2 above which 99.9% of all measured f2 values occur
- Freq_Accuracy: float: No parameter help available
- Freq_Drift: float: No parameter help available
- Max_Drift: float: No parameter help available
- Freq_Dev_Avg_F_1: float: No parameter help available
- Freq_Dev_Min_F_1: float: No parameter help available
- Freq_Dev_Max_F_1: float: No parameter help available
- Freq_Dev_Avg_F_2: float: No parameter help available
- Freq_Dev_Min_F_2: float: No parameter help available
- Freq_Dev_Max_F_2: float: No parameter help available
- Nominal_Power: float: Average power during the carrier-on state
- Mod_Ratio: float: Modulation ratio f2 avg / f1 avg
- Freq_Offset: float: No parameter help available

- Init_Freq_Drift: float: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:LENergy[:LE1M]:CURRent
value: CalculateStruct = driver.bluetooth.measurement.multiEval.modulation.
↪lowEnergy.le1M.current.calculate()
```

Returns current, average, absolute min (xmin) , absolute max (xmax) , and max modulation results for LE uncoded PHY (LE 1M PHY, LE 2M PHY) , see 'Square TX Measurement - modulation statistics'. The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → ResultData

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:LENergy[:LE1M]:CURRent
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.lowEnergy.
↪le1M.current.fetch()
```

Returns current, average, absolute min (xmin) , absolute max (xmax) , and max modulation results for LE uncoded PHY (LE 1M PHY, LE 2M PHY) , see 'Square TX Measurement - modulation statistics'. The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for ResultData structure arguments.

read() → ResultData

```
# SCPI: READ:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:LENergy[:LE1M]:CURRent
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.lowEnergy.
↪le1M.current.read()
```

Returns current, average, absolute min (xmin) , absolute max (xmax) , and max modulation results for LE uncoded PHY (LE 1M PHY, LE 2M PHY) , see 'Square TX Measurement - modulation statistics'. The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for ResultData structure arguments.

6.1.1.6.3.36 Maximum

SCPI Commands :

```
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:LENergy[:LE1M]:MAXimum
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:LENergy[:LE1M]:MAXimum
READ:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:LENergy[:LE1M]:MAXimum
```

class MaximumCls

Maximum commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: 'Reliability indicator'
- Out_Of_Tol: float or bool: Out of tolerance result, i.e. the percentage of measurement intervals of the statistic count ([CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Scount#Modulation CMDLINKRESOLVED]) exceeding the specified limits, see 'Modulation limits (LE) '.
- Delta_F_299_P_9: float or bool: Frequency deviation value f2 above which 99.9% of all measured f2 values occur
- Freq_Accuracy: float or bool: No parameter help available
- Freq_Drift: float or bool: No parameter help available
- Max_Drift: float or bool: No parameter help available
- Freq_Dev_Avg_F_1: float or bool: No parameter help available
- Freq_Dev_Min_F_1: float or bool: No parameter help available
- Freq_Dev_Max_F_1: float or bool: No parameter help available
- Freq_Dev_Avg_F_2: float or bool: No parameter help available
- Freq_Dev_Min_F_2: float or bool: No parameter help available
- Freq_Dev_Max_F_2: float or bool: No parameter help available
- Nominal_Power: float or bool: Average power during the carrier-on state
- Mod_Ratio: enums.ResultStatus2: Modulation ratio f2 avg / f1 avg
- Freq_Offset: float or bool: No parameter help available
- Init_Freq_Drift: float or bool: No parameter help available

class ResultData

Response structure. Fields:

- Reliability: int: 'Reliability indicator'
- Out_Of_Tol: float: Out of tolerance result, i.e. the percentage of measurement intervals of the statistic count ([CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Scount#Modulation CMDLINKRESOLVED]) exceeding the specified limits, see 'Modulation limits (LE) '.
- Delta_F_299_P_9: float: Frequency deviation value f2 above which 99.9% of all measured f2 values occur
- Freq_Accuracy: float: No parameter help available

- Freq_Drift: float: No parameter help available
- Max_Drift: float: No parameter help available
- Freq_Dev_Avg_F_1: float: No parameter help available
- Freq_Dev_Min_F_1: float: No parameter help available
- Freq_Dev_Max_F_1: float: No parameter help available
- Freq_Dev_Avg_F_2: float: No parameter help available
- Freq_Dev_Min_F_2: float: No parameter help available
- Freq_Dev_Max_F_2: float: No parameter help available
- Nominal_Power: float: Average power during the carrier-on state
- Mod_Ratio: float: Modulation ratio f2 avg / f1 avg
- Freq_Offset: float: No parameter help available
- Init_Freq_Drift: float: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:LENergy[:LE1M]:MAXimum
value: CalculateStruct = driver.bluetooth.measurement.multiEval.modulation.
↪lowEnergy.le1M.maximum.calculate()
```

Returns current, average, absolute min (xmin) , absolute max (xmax) , and max modulation results for LE uncoded PHY (LE 1M PHY, LE 2M PHY) , see 'Square TX Measurement - modulation statistics'. The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → ResultData

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:LENergy[:LE1M]:MAXimum
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.lowEnergy.
↪le1M.maximum.fetch()
```

Returns current, average, absolute min (xmin) , absolute max (xmax) , and max modulation results for LE uncoded PHY (LE 1M PHY, LE 2M PHY) , see 'Square TX Measurement - modulation statistics'. The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for ResultData structure arguments.

read() → ResultData

```
# SCPI: READ:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:LENergy[:LE1M]:MAXimum
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.lowEnergy.
↪le1M.maximum.read()
```

Returns current, average, absolute min (xmin) , absolute max (xmax) , and max modulation results for LE uncoded PHY (LE 1M PHY, LE 2M PHY) , see 'Square TX Measurement - modulation statistics'. The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for ResultData structure arguments.

6.1.1.6.3.37 Minimum

SCPI Commands :

```
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:LENergy[:LE1M]:MINimum
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:LENergy[:LE1M]:MINimum
READ:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:LENergy[:LE1M]:MINimum
```

`class MinimumCls`

Minimum commands group definition. 3 total commands, 0 Subgroups, 3 group commands

`class CalculateStruct`

Response structure. Fields:

- Reliability: int: 'Reliability indicator'
- Out_Of_Tol: float or bool: Out of tolerance result, i.e. the percentage of measurement intervals of the statistic count ([CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Scount#Modulation CMDLINKRESOLVED]) exceeding the specified limits, see 'Modulation limits (LE) '.
- Delta_F_299_P_9: float or bool: Frequency deviation value f2 above which 99.9% of all measured f2 values occur
- Freq_Dev_Avg_F_1: float or bool: No parameter help available
- Freq_Dev_Min_F_1: float or bool: No parameter help available
- Freq_Dev_Max_F_1: float or bool: No parameter help available
- Freq_Dev_Avg_F_2: float or bool: No parameter help available
- Freq_Dev_Min_F_2: float or bool: No parameter help available
- Freq_Dev_Max_F_2: float or bool: No parameter help available
- Nominal_Power: float or bool: Average power during the carrier-on state
- Mod_Ratio: enums.ResultStatus2: Modulation ratio f2 avg / f1 avg

`class ResultData`

Response structure. Fields:

- Reliability: int: 'Reliability indicator'
- Out_Of_Tol: float: Out of tolerance result, i.e. the percentage of measurement intervals of the statistic count ([CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Scount#Modulation CMDLINKRESOLVED]) exceeding the specified limits, see 'Modulation limits (LE) '.
- Delta_F_299_P_9: float: Frequency deviation value f2 above which 99.9% of all measured f2 values occur
- Freq_Dev_Avg_F_1: float: No parameter help available

- Freq_Dev_Min_F_1: float: No parameter help available
- Freq_Dev_Max_F_1: float: No parameter help available
- Freq_Dev_Avg_F_2: float: No parameter help available
- Freq_Dev_Min_F_2: float: No parameter help available
- Freq_Dev_Max_F_2: float: No parameter help available
- Nominal_Power: float: Average power during the carrier-on state
- Mod_Ratio: float: Modulation ratio f2 avg / f1 avg

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:LENergy[:LE1M]:MINimum
value: CalculateStruct = driver.bluetooth.measurement.multiEval.modulation.
↪lowEnergy.le1M.minimum.calculate()
```

Returns the minimum modulation results for LE uncoded PHY (LE 1M PHY, LE 2M PHY) , see 'Square TX Measurement - modulation statistics'. The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → ResultData

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:LENergy[:LE1M]:MINimum
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.lowEnergy.
↪le1M.minimum.fetch()
```

Returns the minimum modulation results for LE uncoded PHY (LE 1M PHY, LE 2M PHY) , see 'Square TX Measurement - modulation statistics'. The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for ResultData structure arguments.

read() → ResultData

```
# SCPI: READ:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:LENergy[:LE1M]:MINimum
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.lowEnergy.
↪le1M.minimum.read()
```

Returns the minimum modulation results for LE uncoded PHY (LE 1M PHY, LE 2M PHY) , see 'Square TX Measurement - modulation statistics'. The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for ResultData structure arguments.

6.1.1.6.3.38 StandardDev

SCPI Commands :

```
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:LENergy[:LE1M]:SDEViation
CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:LENergy[:LE1M]:SDEViation
READ:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:LENergy[:LE1M]:SDEViation
```

class StandardDevCls

StandardDev commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: 'Reliability indicator'
- Out_Of_Tol: float or bool: Out of tolerance result, i.e. the percentage of measurement intervals of the statistic count ([CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Scount#Modulation CMDLINKRESOLVED]) exceeding the specified limits, see 'Modulation limits (LE) '.
- Delta_F_299_P_9: float or bool: Frequency deviation value f2 above which 99.9% of all measured f2 values occur
- Freq_Accuracy: float or bool: No parameter help available
- Freq_Drift: float or bool: No parameter help available
- Max_Drift: float or bool: No parameter help available
- Nominal_Power: float or bool: Average power during the carrier-on state
- Mod_Ratio: enums.ResultStatus2: Modulation ratio f2 avg / f1 avg
- Freq_Offset: float or bool: No parameter help available
- Init_Freq_Drift: float or bool: No parameter help available

class ResultData

Response structure. Fields:

- Reliability: int: 'Reliability indicator'
- Out_Of_Tol: float: Out of tolerance result, i.e. the percentage of measurement intervals of the statistic count ([CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Scount#Modulation CMDLINKRESOLVED]) exceeding the specified limits, see 'Modulation limits (LE) '.
- Delta_F_299_P_9: float: Frequency deviation value f2 above which 99.9% of all measured f2 values occur
- Freq_Accuracy: float: No parameter help available
- Freq_Drift: float: No parameter help available
- Max_Drift: float: No parameter help available
- Nominal_Power: float: Average power during the carrier-on state
- Mod_Ratio: float: Modulation ratio f2 avg / f1 avg
- Freq_Offset: float: No parameter help available
- Init_Freq_Drift: float: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:LENergy[:LE1M]:SDEViation
value: CalculateStruct = driver.bluetooth.measurement.multiEval.modulation.
↪lowEnergy.le1M.standardDev.calculate()
```

Returns the standard deviation of the modulation results for LE uncoded PHY (LE 1M PHY, LE 2M PHY) , see 'Square TX Measurement - modulation statistics'. The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → ResultData

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:LENergy[:LE1M]:SDEViation
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.lowEnergy.
↪le1M.standardDev.fetch()
```

Returns the standard deviation of the modulation results for LE uncoded PHY (LE 1M PHY, LE 2M PHY) , see 'Square TX Measurement - modulation statistics'. The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for ResultData structure arguments.

read() → ResultData

```
# SCPI: READ:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:LENergy[:LE1M]:SDEViation
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.lowEnergy.
↪le1M.standardDev.read()
```

Returns the standard deviation of the modulation results for LE uncoded PHY (LE 1M PHY, LE 2M PHY) , see 'Square TX Measurement - modulation statistics'. The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for ResultData structure arguments.

6.1.1.6.3.39 Xmaximum

SCPI Commands :

```
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:LENergy[:LE1M]:XMAXimum
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:LENergy[:LE1M]:XMAXimum
READ:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:LENergy[:LE1M]:XMAXimum
```

class XmaximumCls

Xmaximum commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: 'Reliability indicator'
- Out_Of_Tol: float or bool: Out of tolerance result, i.e. the percentage of measurement intervals of the statistic count ([CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Scount#Modulation CMDLINKRESOLVED]) exceeding the specified limits, see 'Modulation limits (LE) '.
- Delta_F_299_P_9: float or bool: Frequency deviation value f2 above which 99.9% of all measured f2 values occur
- Freq_Accuracy: float or bool: No parameter help available
- Freq_Drift: float or bool: No parameter help available
- Max_Drift: float or bool: No parameter help available
- Freq_Dev_Avg_F_1: float or bool: No parameter help available
- Freq_Dev_Min_F_1: float or bool: No parameter help available
- Freq_Dev_Max_F_1: float or bool: No parameter help available
- Freq_Dev_Avg_F_2: float or bool: No parameter help available
- Freq_Dev_Min_F_2: float or bool: No parameter help available
- Freq_Dev_Max_F_2: float or bool: No parameter help available
- Nominal_Power: float or bool: Average power during the carrier-on state
- Mod_Ratio: enums.ResultStatus2: Modulation ratio f2 avg / f1 avg
- Freq_Offset: float or bool: No parameter help available
- Init_Freq_Drift: float or bool: No parameter help available

class ResultData

Response structure. Fields:

- Reliability: int: 'Reliability indicator'
- Out_Of_Tol: float: Out of tolerance result, i.e. the percentage of measurement intervals of the statistic count ([CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Scount#Modulation CMDLINKRESOLVED]) exceeding the specified limits, see 'Modulation limits (LE) '.
- Delta_F_299_P_9: float: Frequency deviation value f2 above which 99.9% of all measured f2 values occur
- Freq_Accuracy: float: No parameter help available
- Freq_Drift: float: No parameter help available
- Max_Drift: float: No parameter help available
- Freq_Dev_Avg_F_1: float: No parameter help available
- Freq_Dev_Min_F_1: float: No parameter help available
- Freq_Dev_Max_F_1: float: No parameter help available
- Freq_Dev_Avg_F_2: float: No parameter help available
- Freq_Dev_Min_F_2: float: No parameter help available
- Freq_Dev_Max_F_2: float: No parameter help available

- Nominal_Power: float: Average power during the carrier-on state
- Mod_Ratio: float: Modulation ratio f2 avg / f1 avg
- Freq_Offset: float: No parameter help available
- Init_Freq_Drift: float: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:LENergy[:LE1M]:XMAXimum
value: CalculateStruct = driver.bluetooth.measurement.multiEval.modulation.
↪lowEnergy.le1M.xmaximum.calculate()
```

Returns current, average, absolute min (xmin) , absolute max (xmax) , and max modulation results for LE uncoded PHY (LE 1M PHY, LE 2M PHY) , see 'Square TX Measurement - modulation statistics'. The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → ResultData

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:LENergy[:LE1M]:XMAXimum
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.lowEnergy.
↪le1M.xmaximum.fetch()
```

Returns current, average, absolute min (xmin) , absolute max (xmax) , and max modulation results for LE uncoded PHY (LE 1M PHY, LE 2M PHY) , see 'Square TX Measurement - modulation statistics'. The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for ResultData structure arguments.

read() → ResultData

```
# SCPI: READ:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:LENergy[:LE1M]:XMAXimum
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.lowEnergy.
↪le1M.xmaximum.read()
```

Returns current, average, absolute min (xmin) , absolute max (xmax) , and max modulation results for LE uncoded PHY (LE 1M PHY, LE 2M PHY) , see 'Square TX Measurement - modulation statistics'. The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for ResultData structure arguments.

6.1.1.6.3.40 Xminimum

SCPI Commands :

```
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:LENergy[:LE1M]:XMINimum
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:LENergy[:LE1M]:XMINimum
READ:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:LENergy[:LE1M]:XMINimum
```

class XminimumCls

Xminimum commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: 'Reliability indicator'
- Out_Of_Tol: float or bool: Out of tolerance result, i.e. the percentage of measurement intervals of the statistic count ([CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Scount#Modulation CMDLINKRESOLVED]) exceeding the specified limits, see 'Modulation limits (LE) '.
- Delta_F_299_P_9: float or bool: Frequency deviation value f2 above which 99.9% of all measured f2 values occur
- Freq_Accuracy: float or bool: No parameter help available
- Freq_Drift: float or bool: No parameter help available
- Max_Drift: float or bool: No parameter help available
- Freq_Dev_Avg_F_1: float or bool: No parameter help available
- Freq_Dev_Min_F_1: float or bool: No parameter help available
- Freq_Dev_Max_F_1: float or bool: No parameter help available
- Freq_Dev_Avg_F_2: float or bool: No parameter help available
- Freq_Dev_Min_F_2: float or bool: No parameter help available
- Freq_Dev_Max_F_2: float or bool: No parameter help available
- Nominal_Power: float or bool: Average power during the carrier-on state
- Mod_Ratio: enums.ResultStatus2: Modulation ratio f2 avg / f1 avg
- Freq_Offset: float or bool: No parameter help available
- Init_Freq_Drift: float or bool: No parameter help available

class ResultData

Response structure. Fields:

- Reliability: int: 'Reliability indicator'
- Out_Of_Tol: float: Out of tolerance result, i.e. the percentage of measurement intervals of the statistic count ([CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Scount#Modulation CMDLINKRESOLVED]) exceeding the specified limits, see 'Modulation limits (LE) '.
- Delta_F_299_P_9: float: Frequency deviation value f2 above which 99.9% of all measured f2 values occur
- Freq_Accuracy: float: No parameter help available

- Freq_Drift: float: No parameter help available
- Max_Drift: float: No parameter help available
- Freq_Dev_Avg_F_1: float: No parameter help available
- Freq_Dev_Min_F_1: float: No parameter help available
- Freq_Dev_Max_F_1: float: No parameter help available
- Freq_Dev_Avg_F_2: float: No parameter help available
- Freq_Dev_Min_F_2: float: No parameter help available
- Freq_Dev_Max_F_2: float: No parameter help available
- Nominal_Power: float: Average power during the carrier-on state
- Mod_Ratio: float: Modulation ratio f2 avg / f1 avg
- Freq_Offset: float: No parameter help available
- Init_Freq_Drift: float: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:LENergy[:LE1M]:XMINimum
value: CalculateStruct = driver.bluetooth.measurement.multiEval.modulation.
↪lowEnergy.le1M.xminimum.calculate()
```

Returns current, average, absolute min (xmin) , absolute max (xmax) , and max modulation results for LE uncoded PHY (LE 1M PHY, LE 2M PHY) , see 'Square TX Measurement - modulation statistics'. The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → ResultData

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:LENergy[:LE1M]:XMINimum
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.lowEnergy.
↪le1M.xminimum.fetch()
```

Returns current, average, absolute min (xmin) , absolute max (xmax) , and max modulation results for LE uncoded PHY (LE 1M PHY, LE 2M PHY) , see 'Square TX Measurement - modulation statistics'. The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for ResultData structure arguments.

read() → ResultData

```
# SCPI: READ:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:LENergy[:LE1M]:XMINimum
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.lowEnergy.
↪le1M.xminimum.read()
```

Returns current, average, absolute min (xmin) , absolute max (xmax) , and max modulation results for LE uncoded PHY (LE 1M PHY, LE 2M PHY) , see 'Square TX Measurement - modulation statistics'. The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for ResultData structure arguments.

6.1.1.6.3.41 YieldPy

SCPI Command :

```
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:LENergy[:LE1M]:YIELd
```

class YieldPyCls

YieldPy commands group definition. 1 total commands, 0 Subgroups, 1 group commands

fetch() → List[float]

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:LENergy[:LE1M]:YIELd
value: List[float] = driver.bluetooth.measurement.multiEval.modulation.
↪lowEnergy.le1M.yieldPy.fetch()
```

Returns the percentage of auto-detected LE packets with a particular pattern type. Commands for uncoded LE 1M PHY (.. :LE1M..) and LE 2M PHY (..:LE2M..) are available. A result is available after the CMP180 has auto-detected a packet (method RsCMPX_BluetoothMeas.Configure.Bluetooth.Measurement.InputSignal.dmode AUTO) .

Suppressed linked return values: reliability

return
pattern_yield: Pattern yield for 11110000 patterns, 10101010 patterns, and any other patterns (3 values)

6.1.1.6.3.42 Le2M

class Le2MCls

Le2M commands group definition. 22 total commands, 8 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.bluetooth.measurement.multiEval.modulation.lowEnergy.le2M.clone()
```

Subgroups

6.1.1.6.3.43 Average

SCPI Commands :

```
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:LENergy:LE2M:AVERage
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:LENergy:LE2M:AVERage
READ:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:LENergy:LE2M:AVERage
```

class AverageCls

Average commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: 'Reliability indicator'
- Out_Of_Tol: float or bool: Out of tolerance result, i.e. the percentage of measurement intervals of the statistic count ([CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Scount#Modulation CMDLINKRESOLVED]) exceeding the specified limits, see 'Modulation limits (LE) '.
- Delta_F_299_P_9: float or bool: Frequency deviation value f2 above which 99.9% of all measured f2 values occur
- Freq_Accuracy: float or bool: No parameter help available
- Freq_Drift: float or bool: No parameter help available
- Max_Drift: float or bool: No parameter help available
- Freq_Dev_Avg_F_1: float or bool: No parameter help available
- Freq_Dev_Min_F_1: float or bool: No parameter help available
- Freq_Dev_Max_F_1: float or bool: No parameter help available
- Freq_Dev_Avg_F_2: float or bool: No parameter help available
- Freq_Dev_Min_F_2: float or bool: No parameter help available
- Freq_Dev_Max_F_2: float or bool: No parameter help available
- Nominal_Power: float or bool: Average power during the carrier-on state
- Mod_Ratio: enums.ResultStatus2: Modulation ratio f2 avg / f1 avg
- Freq_Offset: float or bool: No parameter help available
- Init_Freq_Drift: float or bool: No parameter help available

class ResultData

Response structure. Fields:

- Reliability: int: 'Reliability indicator'
- Out_Of_Tol: float: Out of tolerance result, i.e. the percentage of measurement intervals of the statistic count ([CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Scount#Modulation CMDLINKRESOLVED]) exceeding the specified limits, see 'Modulation limits (LE) '.
- Delta_F_299_P_9: float: Frequency deviation value f2 above which 99.9% of all measured f2 values occur

- Freq_Accuracy: float: No parameter help available
- Freq_Drift: float: No parameter help available
- Max_Drift: float: No parameter help available
- Freq_Dev_Avg_F_1: float: No parameter help available
- Freq_Dev_Min_F_1: float: No parameter help available
- Freq_Dev_Max_F_1: float: No parameter help available
- Freq_Dev_Avg_F_2: float: No parameter help available
- Freq_Dev_Min_F_2: float: No parameter help available
- Freq_Dev_Max_F_2: float: No parameter help available
- Nominal_Power: float: Average power during the carrier-on state
- Mod_Ratio: float: Modulation ratio f2 avg / f1 avg
- Freq_Offset: float: No parameter help available
- Init_Freq_Drift: float: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:LENergy:LE2M:AVERage
value: CalculateStruct = driver.bluetooth.measurement.multiEval.modulation.
↪lowEnergy.le2M.average.calculate()
```

Returns current, average, absolute min (xmin) , absolute max (xmax) , and max modulation results for LE uncoded PHY (LE 1M PHY, LE 2M PHY) , see 'Square TX Measurement - modulation statistics'. The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → ResultData

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:LENergy:LE2M:AVERage
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.lowEnergy.
↪le2M.average.fetch()
```

Returns current, average, absolute min (xmin) , absolute max (xmax) , and max modulation results for LE uncoded PHY (LE 1M PHY, LE 2M PHY) , see 'Square TX Measurement - modulation statistics'. The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for ResultData structure arguments.

read() → ResultData

```
# SCPI: READ:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:LENergy:LE2M:AVERage
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.lowEnergy.
↪le2M.average.read()
```

Returns current, average, absolute min (xmin) , absolute max (xmax) , and max modulation results for LE uncoded PHY (LE 1M PHY, LE 2M PHY) , see 'Square TX Measurement - modulation statistics'. The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for ResultData structure arguments.

6.1.1.6.3.44 Current

SCPI Commands :

```
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:LENergy:LE2M:CURRent
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:LENergy:LE2M:CURRent
READ:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:LENergy:LE2M:CURRent
```

`class CurrentCls`

Current commands group definition. 3 total commands, 0 Subgroups, 3 group commands

`class CalculateStruct`

Response structure. Fields:

- Reliability: int: 'Reliability indicator'
- Out_Of_Tol: float or bool: Out of tolerance result, i.e. the percentage of measurement intervals of the statistic count ([CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Scount#Modulation CMDLINKRESOLVED]) exceeding the specified limits, see 'Modulation limits (LE) '.
- Delta_F_299_P_9: float or bool: Frequency deviation value f2 above which 99.9% of all measured f2 values occur
- Freq_Accuracy: float or bool: No parameter help available
- Freq_Drift: float or bool: No parameter help available
- Max_Drift: float or bool: No parameter help available
- Freq_Dev_Avg_F_1: float or bool: No parameter help available
- Freq_Dev_Min_F_1: float or bool: No parameter help available
- Freq_Dev_Max_F_1: float or bool: No parameter help available
- Freq_Dev_Avg_F_2: float or bool: No parameter help available
- Freq_Dev_Min_F_2: float or bool: No parameter help available
- Freq_Dev_Max_F_2: float or bool: No parameter help available
- Nominal_Power: float or bool: Average power during the carrier-on state
- Mod_Ratio: enums.ResultStatus2: Modulation ratio f2 avg / f1 avg
- Freq_Offset: float or bool: No parameter help available
- Init_Freq_Drift: float or bool: No parameter help available

`class ResultData`

Response structure. Fields:

- Reliability: int: 'Reliability indicator'

- Out_Of_Tol: float: Out of tolerance result, i.e. the percentage of measurement intervals of the statistic count ([CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Scount#Modulation CMDLINKRESOLVED]) exceeding the specified limits, see 'Modulation limits (LE) '.
- Delta_F_299_P_9: float: Frequency deviation value f2 above which 99.9% of all measured f2 values occur
- Freq_Accuracy: float: No parameter help available
- Freq_Drift: float: No parameter help available
- Max_Drift: float: No parameter help available
- Freq_Dev_Avg_F_1: float: No parameter help available
- Freq_Dev_Min_F_1: float: No parameter help available
- Freq_Dev_Max_F_1: float: No parameter help available
- Freq_Dev_Avg_F_2: float: No parameter help available
- Freq_Dev_Min_F_2: float: No parameter help available
- Freq_Dev_Max_F_2: float: No parameter help available
- Nominal_Power: float: Average power during the carrier-on state
- Mod_Ratio: float: Modulation ratio f2 avg / f1 avg
- Freq_Offset: float: No parameter help available
- Init_Freq_Drift: float: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:LENergy:LE2M:CURRent
value: CalculateStruct = driver.bluetooth.measurement.multiEval.modulation.
↪lowEnergy.le2M.current.calculate()
```

Returns current, average, absolute min (xmin) , absolute max (xmax) , and max modulation results for LE uncoded PHY (LE 1M PHY, LE 2M PHY) , see 'Square TX Measurement - modulation statistics'. The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → ResultData

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:LENergy:LE2M:CURRent
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.lowEnergy.
↪le2M.current.fetch()
```

Returns current, average, absolute min (xmin) , absolute max (xmax) , and max modulation results for LE uncoded PHY (LE 1M PHY, LE 2M PHY) , see 'Square TX Measurement - modulation statistics'. The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for ResultData structure arguments.

read() → ResultData

```
# SCPI: READ:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:LENergy:LE2M:CURRent
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.lowEnergy.
↪le2M.current.read()
```

Returns current, average, absolute min (xmin) , absolute max (xmax) , and max modulation results for LE uncoded PHY (LE 1M PHY, LE 2M PHY) , see 'Square TX Measurement - modulation statistics'. The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for ResultData structure arguments.

6.1.1.6.3.45 Maximum

SCPI Commands :

```
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:LENergy:LE2M:MAXimum
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:LENergy:LE2M:MAXimum
READ:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:LENergy:LE2M:MAXimum
```

class MaximumCls

Maximum commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: 'Reliability indicator'
- Out_Of_Tol: float or bool: Out of tolerance result, i.e. the percentage of measurement intervals of the statistic count ([CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Scount#Modulation CMDLINKRESOLVED]) exceeding the specified limits, see 'Modulation limits (LE) '.
- Delta_F_299_P_9: float or bool: Frequency deviation value f2 above which 99.9% of all measured f2 values occur
- Freq_Accuracy: float or bool: No parameter help available
- Freq_Drift: float or bool: No parameter help available
- Max_Drift: float or bool: No parameter help available
- Freq_Dev_Avg_F_1: float or bool: No parameter help available
- Freq_Dev_Min_F_1: float or bool: No parameter help available
- Freq_Dev_Max_F_1: float or bool: No parameter help available
- Freq_Dev_Avg_F_2: float or bool: No parameter help available
- Freq_Dev_Min_F_2: float or bool: No parameter help available
- Freq_Dev_Max_F_2: float or bool: No parameter help available
- Nominal_Power: float or bool: Average power during the carrier-on state
- Mod_Ratio: enums.ResultStatus2: Modulation ratio f2 avg / f1 avg

- Freq_Offset: float or bool: No parameter help available
- Init_Freq_Drift: float or bool: No parameter help available

class ResultData

Response structure. Fields:

- Reliability: int: 'Reliability indicator'
- Out_Of_Tol: float: Out of tolerance result, i.e. the percentage of measurement intervals of the statistic count ([CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Scount#Modulation CMDLINKRESOLVED]) exceeding the specified limits, see 'Modulation limits (LE) '.
- Delta_F_299_P_9: float: Frequency deviation value f2 above which 99.9% of all measured f2 values occur
- Freq_Accuracy: float: No parameter help available
- Freq_Drift: float: No parameter help available
- Max_Drift: float: No parameter help available
- Freq_Dev_Avg_F_1: float: No parameter help available
- Freq_Dev_Min_F_1: float: No parameter help available
- Freq_Dev_Max_F_1: float: No parameter help available
- Freq_Dev_Avg_F_2: float: No parameter help available
- Freq_Dev_Min_F_2: float: No parameter help available
- Freq_Dev_Max_F_2: float: No parameter help available
- Nominal_Power: float: Average power during the carrier-on state
- Mod_Ratio: float: Modulation ratio f2 avg / f1 avg
- Freq_Offset: float: No parameter help available
- Init_Freq_Drift: float: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:LENergy:LE2M:MAXimum
value: CalculateStruct = driver.bluetooth.measurement.multiEval.modulation.
↪lowEnergy.le2M.maximum.calculate()
```

Returns current, average, absolute min (xmin) , absolute max (xmax) , and max modulation results for LE uncoded PHY (LE 1M PHY, LE 2M PHY) , see 'Square TX Measurement - modulation statistics'. The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → ResultData

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:LENergy:LE2M:MAXimum
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.lowEnergy.
↪le2M.maximum.fetch()
```

Returns current, average, absolute min (xmin) , absolute max (xmax) , and max modulation results for LE uncoded PHY (LE 1M PHY, LE 2M PHY) , see 'Square TX Measurement - modulation statistics'. The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for ResultData structure arguments.

read() → ResultData

```
# SCPI: READ:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:LENergy:LE2M:MAXimum
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.lowEnergy.
↪le2M.maximum.read()
```

Returns current, average, absolute min (xmin) , absolute max (xmax) , and max modulation results for LE uncoded PHY (LE 1M PHY, LE 2M PHY) , see 'Square TX Measurement - modulation statistics'. The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for ResultData structure arguments.

6.1.1.6.3.46 Minimum

SCPI Commands :

```
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:LENergy:LE2M:MINimum
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:LENergy:LE2M:MINimum
READ:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:LENergy:LE2M:MINimum
```

class MinimumCls

Minimum commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: 'Reliability indicator'
- Out_Of_Tol: float or bool: Out of tolerance result, i.e. the percentage of measurement intervals of the statistic count ([CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Scount#Modulation CMDLINKRESOLVED]) exceeding the specified limits, see 'Modulation limits (LE) '.
- Delta_F_299_P_9: float or bool: Frequency deviation value f2 above which 99.9% of all measured f2 values occur
- Freq_Dev_Avg_F_1: float or bool: No parameter help available
- Freq_Dev_Min_F_1: float or bool: No parameter help available
- Freq_Dev_Max_F_1: float or bool: No parameter help available
- Freq_Dev_Avg_F_2: float or bool: No parameter help available
- Freq_Dev_Min_F_2: float or bool: No parameter help available
- Freq_Dev_Max_F_2: float or bool: No parameter help available

- Nominal_Power: float or bool: Average power during the carrier-on state
- Mod_Ratio: enums.ResultStatus2: Modulation ratio f2 avg / f1 avg

class ResultData

Response structure. Fields:

- Reliability: int: 'Reliability indicator'
- Out_Of_Tol: float: Out of tolerance result, i.e. the percentage of measurement intervals of the statistic count ([CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Scount#Modulation CMDLINKRESOLVED]) exceeding the specified limits, see 'Modulation limits (LE) '.
- Delta_F_299_P_9: float: Frequency deviation value f2 above which 99.9% of all measured f2 values occur
- Freq_Dev_Avg_F_1: float: No parameter help available
- Freq_Dev_Min_F_1: float: No parameter help available
- Freq_Dev_Max_F_1: float: No parameter help available
- Freq_Dev_Avg_F_2: float: No parameter help available
- Freq_Dev_Min_F_2: float: No parameter help available
- Freq_Dev_Max_F_2: float: No parameter help available
- Nominal_Power: float: Average power during the carrier-on state
- Mod_Ratio: float: Modulation ratio f2 avg / f1 avg

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:LENergy:LE2M:MINimum
value: CalculateStruct = driver.bluetooth.measurement.multiEval.modulation.
↪lowEnergy.le2M.minimum.calculate()
```

Returns the minimum modulation results for LE uncoded PHY (LE 1M PHY, LE 2M PHY) , see 'Square TX Measurement - modulation statistics'. The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → ResultData

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:LENergy:LE2M:MINimum
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.lowEnergy.
↪le2M.minimum.fetch()
```

Returns the minimum modulation results for LE uncoded PHY (LE 1M PHY, LE 2M PHY) , see 'Square TX Measurement - modulation statistics'. The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for ResultData structure arguments.

read() → ResultData

```
# SCPI: READ:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:LENergy:LE2M:MINimum
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.lowEnergy.
↪le2M.minimum.read()
```

Returns the minimum modulation results for LE uncoded PHY (LE 1M PHY, LE 2M PHY) , see 'Square TX Measurement - modulation statistics'. The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for ResultData structure arguments.

6.1.1.6.3.47 StandardDev

SCPI Commands :

```
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:LENergy:LE2M:SDEViation
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:LENergy:LE2M:SDEViation
READ:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:LENergy:LE2M:SDEViation
```

class StandardDevCls

StandardDev commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: 'Reliability indicator'
- Out_Of_Tol: float or bool: Out of tolerance result, i.e. the percentage of measurement intervals of the statistic count ([CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Scount#Modulation CMDLINKRESOLVED]) exceeding the specified limits, see 'Modulation limits (LE) '.
- Delta_F_299_P_9: float or bool: Frequency deviation value f2 above which 99.9% of all measured f2 values occur
- Freq_Accuracy: float or bool: No parameter help available
- Freq_Drift: float or bool: No parameter help available
- Max_Drift: float or bool: No parameter help available
- Nominal_Power: float or bool: Average power during the carrier-on state
- Mod_Ratio: enums.ResultStatus2: Modulation ratio f2 avg / f1 avg
- Freq_Offset: float or bool: No parameter help available
- Init_Freq_Drift: float or bool: No parameter help available

class ResultData

Response structure. Fields:

- Reliability: int: 'Reliability indicator'
- Out_Of_Tol: float: Out of tolerance result, i.e. the percentage of measurement intervals of the statistic count ([CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Scount#Modulation CMDLINKRESOLVED]) exceeding the specified limits, see 'Modulation limits (LE) '.

- Delta_F_299_P_9: float: Frequency deviation value f2 above which 99.9% of all measured f2 values occur
- Freq_Accuracy: float: No parameter help available
- Freq_Drift: float: No parameter help available
- Max_Drift: float: No parameter help available
- Nominal_Power: float: Average power during the carrier-on state
- Mod_Ratio: float: Modulation ratio f2 avg / f1 avg
- Freq_Offset: float: No parameter help available
- Init_Freq_Drift: float: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:LENergy:LE2M:SDEViation
value: CalculateStruct = driver.bluetooth.measurement.multiEval.modulation.
↪lowEnergy.le2M.standardDev.calculate()
```

Returns the standard deviation of the modulation results for LE uncoded PHY (LE 1M PHY, LE 2M PHY) , see 'Square TX Measurement - modulation statistics'. The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → ResultData

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:LENergy:LE2M:SDEViation
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.lowEnergy.
↪le2M.standardDev.fetch()
```

Returns the standard deviation of the modulation results for LE uncoded PHY (LE 1M PHY, LE 2M PHY) , see 'Square TX Measurement - modulation statistics'. The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for ResultData structure arguments.

read() → ResultData

```
# SCPI: READ:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:LENergy:LE2M:SDEViation
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.lowEnergy.
↪le2M.standardDev.read()
```

Returns the standard deviation of the modulation results for LE uncoded PHY (LE 1M PHY, LE 2M PHY) , see 'Square TX Measurement - modulation statistics'. The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for ResultData structure arguments.

6.1.1.6.3.48 Xmaximum

SCPI Commands :

```
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:LENergy:LE2M:XMAXimum
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:LENergy:LE2M:XMAXimum
READ:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:LENergy:LE2M:XMAXimum
```

class XmaximumCls

Xmaximum commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: 'Reliability indicator'
- Out_Of_Tol: float or bool: Out of tolerance result, i.e. the percentage of measurement intervals of the statistic count ([CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Scount#Modulation CMDLINKRESOLVED]) exceeding the specified limits, see 'Modulation limits (LE) '.
- Delta_F_299_P_9: float or bool: Frequency deviation value f2 above which 99.9% of all measured f2 values occur
- Freq_Accuracy: float or bool: No parameter help available
- Freq_Drift: float or bool: No parameter help available
- Max_Drift: float or bool: No parameter help available
- Freq_Dev_Avg_F_1: float or bool: No parameter help available
- Freq_Dev_Min_F_1: float or bool: No parameter help available
- Freq_Dev_Max_F_1: float or bool: No parameter help available
- Freq_Dev_Avg_F_2: float or bool: No parameter help available
- Freq_Dev_Min_F_2: float or bool: No parameter help available
- Freq_Dev_Max_F_2: float or bool: No parameter help available
- Nominal_Power: float or bool: Average power during the carrier-on state
- Mod_Ratio: enums.ResultStatus2: Modulation ratio f2 avg / f1 avg
- Freq_Offset: float or bool: No parameter help available
- Init_Freq_Drift: float or bool: No parameter help available

class ResultData

Response structure. Fields:

- Reliability: int: 'Reliability indicator'
- Out_Of_Tol: float: Out of tolerance result, i.e. the percentage of measurement intervals of the statistic count ([CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Scount#Modulation CMDLINKRESOLVED]) exceeding the specified limits, see 'Modulation limits (LE) '.
- Delta_F_299_P_9: float: Frequency deviation value f2 above which 99.9% of all measured f2 values occur
- Freq_Accuracy: float: No parameter help available

- Freq_Drift: float: No parameter help available
- Max_Drift: float: No parameter help available
- Freq_Dev_Avg_F_1: float: No parameter help available
- Freq_Dev_Min_F_1: float: No parameter help available
- Freq_Dev_Max_F_1: float: No parameter help available
- Freq_Dev_Avg_F_2: float: No parameter help available
- Freq_Dev_Min_F_2: float: No parameter help available
- Freq_Dev_Max_F_2: float: No parameter help available
- Nominal_Power: float: Average power during the carrier-on state
- Mod_Ratio: float: Modulation ratio f2 avg / f1 avg
- Freq_Offset: float: No parameter help available
- Init_Freq_Drift: float: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:LENergy:LE2M:XMAXimum
value: CalculateStruct = driver.bluetooth.measurement.multiEval.modulation.
↪lowEnergy.le2M.xmaximum.calculate()
```

Returns current, average, absolute min (xmin) , absolute max (xmax) , and max modulation results for LE uncoded PHY (LE 1M PHY, LE 2M PHY) , see 'Square TX Measurement - modulation statistics'. The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → ResultData

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:LENergy:LE2M:XMAXimum
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.lowEnergy.
↪le2M.xmaximum.fetch()
```

Returns current, average, absolute min (xmin) , absolute max (xmax) , and max modulation results for LE uncoded PHY (LE 1M PHY, LE 2M PHY) , see 'Square TX Measurement - modulation statistics'. The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for ResultData structure arguments.

read() → ResultData

```
# SCPI: READ:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:LENergy:LE2M:XMAXimum
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.lowEnergy.
↪le2M.xmaximum.read()
```

Returns current, average, absolute min (xmin) , absolute max (xmax) , and max modulation results for LE uncoded PHY (LE 1M PHY, LE 2M PHY) , see 'Square TX Measurement - modulation statistics'. The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for ResultData structure arguments.

6.1.1.6.3.49 Xminimum

SCPI Commands :

```
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:LENergy:LE2M:XMINimum
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:LENergy:LE2M:XMINimum
READ:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:LENergy:LE2M:XMINimum
```

class XminimumCls

Xminimum commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: 'Reliability indicator'
- Out_Of_Tol: float or bool: Out of tolerance result, i.e. the percentage of measurement intervals of the statistic count ([CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Scount#Modulation CMDLINKRESOLVED]) exceeding the specified limits, see 'Modulation limits (LE) '.
- Delta_F_299_P_9: float or bool: Frequency deviation value f2 above which 99.9% of all measured f2 values occur
- Freq_Accuracy: float or bool: No parameter help available
- Freq_Drift: float or bool: No parameter help available
- Max_Drift: float or bool: No parameter help available
- Freq_Dev_Avg_F_1: float or bool: No parameter help available
- Freq_Dev_Min_F_1: float or bool: No parameter help available
- Freq_Dev_Max_F_1: float or bool: No parameter help available
- Freq_Dev_Avg_F_2: float or bool: No parameter help available
- Freq_Dev_Min_F_2: float or bool: No parameter help available
- Freq_Dev_Max_F_2: float or bool: No parameter help available
- Nominal_Power: float or bool: Average power during the carrier-on state
- Mod_Ratio: enums.ResultStatus2: Modulation ratio f2 avg / f1 avg
- Freq_Offset: float or bool: No parameter help available
- Init_Freq_Drift: float or bool: No parameter help available

class ResultData

Response structure. Fields:

- Reliability: int: 'Reliability indicator'

- Out_Of_Tol: float: Out of tolerance result, i.e. the percentage of measurement intervals of the statistic count ([CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Scount#Modulation CMDLINKRESOLVED]) exceeding the specified limits, see 'Modulation limits (LE) '.
- Delta_F_299_P_9: float: Frequency deviation value f2 above which 99.9% of all measured f2 values occur
- Freq_Accuracy: float: No parameter help available
- Freq_Drift: float: No parameter help available
- Max_Drift: float: No parameter help available
- Freq_Dev_Avg_F_1: float: No parameter help available
- Freq_Dev_Min_F_1: float: No parameter help available
- Freq_Dev_Max_F_1: float: No parameter help available
- Freq_Dev_Avg_F_2: float: No parameter help available
- Freq_Dev_Min_F_2: float: No parameter help available
- Freq_Dev_Max_F_2: float: No parameter help available
- Nominal_Power: float: Average power during the carrier-on state
- Mod_Ratio: float: Modulation ratio f2 avg / f1 avg
- Freq_Offset: float: No parameter help available
- Init_Freq_Drift: float: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:LENergy:LE2M:XMINimum
value: CalculateStruct = driver.bluetooth.measurement.multiEval.modulation.
↪lowEnergy.le2M.xminimum.calculate()
```

Returns current, average, absolute min (xmin) , absolute max (xmax) , and max modulation results for LE uncoded PHY (LE 1M PHY, LE 2M PHY) , see 'Square TX Measurement - modulation statistics'. The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → ResultData

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:LENergy:LE2M:XMINimum
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.lowEnergy.
↪le2M.xminimum.fetch()
```

Returns current, average, absolute min (xmin) , absolute max (xmax) , and max modulation results for LE uncoded PHY (LE 1M PHY, LE 2M PHY) , see 'Square TX Measurement - modulation statistics'. The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for ResultData structure arguments.

read() → ResultData

```
# SCPI: READ:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:LENergy:LE2M:XMINimum
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.lowEnergy.
↪le2M.xminimum.read()
```

Returns current, average, absolute min (xmin) , absolute max (xmax) , and max modulation results for LE uncoded PHY (LE 1M PHY, LE 2M PHY) , see 'Square TX Measurement - modulation statistics'. The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for ResultData structure arguments.

6.1.1.6.3.50 YieldPy

SCPI Command :

```
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:LENergy:LE2M:YIELd
```

class YieldPyCls

YieldPy commands group definition. 1 total commands, 0 Subgroups, 1 group commands

fetch() → List[float]

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:LENergy:LE2M:YIELd
value: List[float] = driver.bluetooth.measurement.multiEval.modulation.
↪lowEnergy.le2M.yieldPy.fetch()
```

Returns the percentage of auto-detected LE packets with a particular pattern type. Commands for uncoded LE 1M PHY (.. :LE1M..) and LE 2M PHY (..:LE2M..) are available. A result is available after the CMP180 has auto-detected a packet (method RsCMPX_BluetoothMeas.Configure.Bluetooth.Measurement.InputSignal.dmode AUTO) .

Suppressed linked return values: reliability

return
pattern_yield: Pattern yield for 11110000 patterns, 10101010 patterns, and any other patterns (3 values)

6.1.1.6.3.51 Lrange

class LrangeCls

Lrange commands group definition. 24 total commands, 8 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.bluetooth.measurement.multiEval.modulation.lowEnergy.lrange.clone()
```

Subgroups

6.1.1.6.3.52 Average

SCPI Commands :

```
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:LENergy:LRANge:AVERage
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:LENergy:LRANge:AVERage
READ:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:LENergy:LRANge:AVERage
```

class AverageCls

Average commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: 'Reliability indicator'
- Out_Of_Tol: float or bool: Out of tolerance result, i.e. the percentage of measurement intervals of the statistic count ([CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Scount#Modulation CMDLINKRESOLVED]) exceeding the specified limits, see 'Modulation limits (LE) '.
- Delta_F_199_P_9: float or bool: No parameter help available
- Freq_Accuracy: float or bool: No parameter help available
- Freq_Drift: float or bool: No parameter help available
- Max_Drift: float or bool: No parameter help available
- Freq_Dev_Avg_F_1: float or bool: No parameter help available
- Freq_Dev_Min_F_1: float or bool: No parameter help available
- Freq_Dev_Max_F_1: float or bool: No parameter help available
- Nominal_Power: float or bool: No parameter help available
- Freq_Offset: float or bool: No parameter help available

class ResultData

Response structure. Fields:

- Reliability: int: 'Reliability indicator'
- Out_Of_Tol: float: Out of tolerance result, i.e. the percentage of measurement intervals of the statistic count ([CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Scount#Modulation CMDLINKRESOLVED]) exceeding the specified limits, see 'Modulation limits (LE) '.
- Delta_F_199_P_9: float: No parameter help available
- Freq_Accuracy: float: No parameter help available
- Freq_Drift: float: No parameter help available

- Max_Drift: float: No parameter help available
- Freq_Dev_Avg_F_1: float: No parameter help available
- Freq_Dev_Min_F_1: float: No parameter help available
- Freq_Dev_Max_F_1: float: No parameter help available
- Nominal_Power: float: No parameter help available
- Freq_Offset: float: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:LENergy:LRANge:AVERage
value: CalculateStruct = driver.bluetooth.measurement.multiEval.modulation.
↪lowEnergy.lrange.average.calculate()
```

Returns the current, average, absolute min (xmin) , absolute max (xmax) , and max modulation results for LE coded PHY, see 'Square TX Measurement - modulation statistics'. The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → ResultData

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:LENergy:LRANge:AVERage
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.lowEnergy.
↪lrange.average.fetch()
```

Returns the current, average, absolute min (xmin) , absolute max (xmax) , and max modulation results for LE coded PHY, see 'Square TX Measurement - modulation statistics'. The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for ResultData structure arguments.

read() → ResultData

```
# SCPI: READ:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:LENergy:LRANge:AVERage
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.lowEnergy.
↪lrange.average.read()
```

Returns the current, average, absolute min (xmin) , absolute max (xmax) , and max modulation results for LE coded PHY, see 'Square TX Measurement - modulation statistics'. The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for ResultData structure arguments.

6.1.1.6.3.53 Current

SCPI Commands :

```
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:LENergy:LRANge:CURRent
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:LENergy:LRANge:CURRent
READ:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:LENergy:LRANge:CURRent
```

`class CurrentCls`

Current commands group definition. 3 total commands, 0 Subgroups, 3 group commands

`class CalculateStruct`

Response structure. Fields:

- Reliability: int: 'Reliability indicator'
- Out_Of_Tol: float or bool: Out of tolerance result, i.e. the percentage of measurement intervals of the statistic count ([CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Scount#Modulation CMDLINKRESOLVED]) exceeding the specified limits, see 'Modulation limits (LE) '.
- Delta_F_199_P_9: float or bool: No parameter help available
- Freq_Accuracy: float or bool: No parameter help available
- Freq_Drift: float or bool: No parameter help available
- Max_Drift: float or bool: No parameter help available
- Freq_Dev_Avg_F_1: float or bool: No parameter help available
- Freq_Dev_Min_F_1: float or bool: No parameter help available
- Freq_Dev_Max_F_1: float or bool: No parameter help available
- Nominal_Power: float or bool: No parameter help available
- Freq_Offset: float or bool: No parameter help available

`class ResultData`

Response structure. Fields:

- Reliability: int: 'Reliability indicator'
- Out_Of_Tol: float: Out of tolerance result, i.e. the percentage of measurement intervals of the statistic count ([CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Scount#Modulation CMDLINKRESOLVED]) exceeding the specified limits, see 'Modulation limits (LE) '.
- Delta_F_199_P_9: float: No parameter help available
- Freq_Accuracy: float: No parameter help available
- Freq_Drift: float: No parameter help available
- Max_Drift: float: No parameter help available
- Freq_Dev_Avg_F_1: float: No parameter help available
- Freq_Dev_Min_F_1: float: No parameter help available
- Freq_Dev_Max_F_1: float: No parameter help available
- Nominal_Power: float: No parameter help available
- Freq_Offset: float: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:LENergy:LRANge:CURRent
value: CalculateStruct = driver.bluetooth.measurement.multiEval.modulation.
↪lowEnergy.lrange.current.calculate()
```

Returns the current, average, absolute min (xmin) , absolute max (xmax) , and max modulation results for LE coded PHY, see 'Square TX Measurement - modulation statistics'. The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → ResultData

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:LENergy:LRANge:CURRent
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.lowEnergy.
↪lrange.current.fetch()
```

Returns the current, average, absolute min (xmin) , absolute max (xmax) , and max modulation results for LE coded PHY, see 'Square TX Measurement - modulation statistics'. The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for ResultData structure arguments.

read() → ResultData

```
# SCPI: READ:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:LENergy:LRANge:CURRent
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.lowEnergy.
↪lrange.current.read()
```

Returns the current, average, absolute min (xmin) , absolute max (xmax) , and max modulation results for LE coded PHY, see 'Square TX Measurement - modulation statistics'. The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for ResultData structure arguments.

6.1.1.6.3.54 Maximum

SCPI Commands :

```
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:LENergy:LRANge:MAXimum
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:LENergy:LRANge:MAXimum
READ:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:LENergy:LRANge:MAXimum
```

class MaximumCls

Maximum commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: 'Reliability indicator'
- Out_Of_Tol: float or bool: Out of tolerance result, i.e. the percentage of measurement intervals of the statistic count ([CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Scount#Modulation CMDLINKRESOLVED]) exceeding the specified limits, see 'Modulation limits (LE) '.
- Delta_F_199_P_9: float or bool: No parameter help available
- Freq_Accuracy: float or bool: No parameter help available
- Freq_Drift: float or bool: No parameter help available
- Max_Drift: float or bool: No parameter help available
- Freq_Dev_Avg_F_1: float or bool: No parameter help available
- Freq_Dev_Min_F_1: float or bool: No parameter help available
- Freq_Dev_Max_F_1: float or bool: No parameter help available
- Nominal_Power: float or bool: No parameter help available
- Freq_Offset: float or bool: No parameter help available

class ResultData

Response structure. Fields:

- Reliability: int: 'Reliability indicator'
- Out_Of_Tol: float: Out of tolerance result, i.e. the percentage of measurement intervals of the statistic count ([CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Scount#Modulation CMDLINKRESOLVED]) exceeding the specified limits, see 'Modulation limits (LE) '.
- Delta_F_199_P_9: float: No parameter help available
- Freq_Accuracy: float: No parameter help available
- Freq_Drift: float: No parameter help available
- Max_Drift: float: No parameter help available
- Freq_Dev_Avg_F_1: float: No parameter help available
- Freq_Dev_Min_F_1: float: No parameter help available
- Freq_Dev_Max_F_1: float: No parameter help available
- Nominal_Power: float: No parameter help available
- Freq_Offset: float: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:LENergy:LRANge:MAXimum
value: CalculateStruct = driver.bluetooth.measurement.multiEval.modulation.
↪lowEnergy.lrange.maximum.calculate()
```

Returns the current, average, absolute min (xmin) , absolute max (xmax) , and max modulation results for LE coded PHY, see 'Square TX Measurement - modulation statistics'. The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → ResultData

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:LENergy:LRANge:MAXimum
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.lowEnergy.
↪lrange.maximum.fetch()
```

Returns the current, average, absolute min (xmin) , absolute max (xmax) , and max modulation results for LE coded PHY, see 'Square TX Measurement - modulation statistics'. The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for ResultData structure arguments.

read() → ResultData

```
# SCPI: READ:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:LENergy:LRANge:MAXimum
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.lowEnergy.
↪lrange.maximum.read()
```

Returns the current, average, absolute min (xmin) , absolute max (xmax) , and max modulation results for LE coded PHY, see 'Square TX Measurement - modulation statistics'. The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for ResultData structure arguments.

6.1.1.6.3.55 Minimum

SCPI Commands :

```
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:LENergy:LRANge:MINimum
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:LENergy:LRANge:MINimum
READ:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:LENergy:LRANge:MINimum
```

class MinimumCls

Minimum commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: 'Reliability indicator'
- Out_Of_Tol: float or bool: Out of tolerance result, i.e. the percentage of measurement intervals of the statistic count ([CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Scount#Modulation CMDLINKRESOLVED]) exceeding the specified limits, see 'Modulation limits (LE) '.
- Delta_F_199_P_9: float or bool: No parameter help available
- Freq_Dev_Avg_F_1: float or bool: No parameter help available

- Freq_Dev_Min_F_1: float or bool: No parameter help available
- Freq_Dev_Max_F_1: float or bool: No parameter help available
- Nominal_Power: float or bool: No parameter help available

class ResultData

Response structure. Fields:

- Reliability: int: 'Reliability indicator'
- Out_Of_Tol: float: Out of tolerance result, i.e. the percentage of measurement intervals of the statistic count ([CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Scount#Modulation CMDLINKRESOLVED]) exceeding the specified limits, see 'Modulation limits (LE) '.
- Delta_F_199_P_9: float: No parameter help available
- Freq_Dev_Avg_F_1: float: No parameter help available
- Freq_Dev_Min_F_1: float: No parameter help available
- Freq_Dev_Max_F_1: float: No parameter help available
- Nominal_Power: float: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:LENergy:LRANge:MINimum
value: CalculateStruct = driver.bluetooth.measurement.multiEval.modulation.
↪lowEnergy.lrange.minimum.calculate()
```

Returns current, average and maximum modulation results for LE coded PHY, see 'Square TX Measurement - modulation statistics'. The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → ResultData

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:LENergy:LRANge:MINimum
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.lowEnergy.
↪lrange.minimum.fetch()
```

Returns current, average and maximum modulation results for LE coded PHY, see 'Square TX Measurement - modulation statistics'. The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for ResultData structure arguments.

read() → ResultData

```
# SCPI: READ:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:LENergy:LRANge:MINimum
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.lowEnergy.
↪lrange.minimum.read()
```

Returns current, average and maximum modulation results for LE coded PHY, see 'Square TX Measurement - modulation statistics'. The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for ResultData structure arguments.

6.1.1.6.3.56 StandardDev

SCPI Commands :

```
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:LENergy:LRANge:SDEViation
CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:LENergy:LRANge:SDEViation
READ:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:LENergy:LRANge:SDEViation
```

class StandardDevCls

StandardDev commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: 'Reliability indicator'
- Out_Of_Tol: float or bool: Out of tolerance result, i.e. the percentage of measurement intervals of the statistic count ([CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Scount#Modulation CMDLINKRESOLVED]) exceeding the specified limits, see 'Modulation limits (LE) '.
- Delta_F_199_P_9: float or bool: No parameter help available
- Freq_Accuracy: float or bool: No parameter help available
- Freq_Drift: float or bool: No parameter help available
- Max_Drift: float or bool: No parameter help available
- Nominal_Power: float or bool: No parameter help available
- Freq_Offset: float or bool: No parameter help available

class ResultData

Response structure. Fields:

- Reliability: int: 'Reliability indicator'
- Out_Of_Tol: float: Out of tolerance result, i.e. the percentage of measurement intervals of the statistic count ([CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Scount#Modulation CMDLINKRESOLVED]) exceeding the specified limits, see 'Modulation limits (LE) '.
- Delta_F_199_P_9: float: No parameter help available
- Freq_Accuracy: float: No parameter help available
- Freq_Drift: float: No parameter help available
- Max_Drift: float: No parameter help available
- Nominal_Power: float: No parameter help available
- Freq_Offset: float: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:LENergy:LRANge:SDEViation
value: CalculateStruct = driver.bluetooth.measurement.multiEval.modulation.
↪lowEnergy.lrange.standardDev.calculate()
```

Returns the standard deviation of the modulation results for LE coded PHY, see 'Square TX Measurement - modulation statistics'. The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → ResultData

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:LENergy:LRANge:SDEViation
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.lowEnergy.
↪lrange.standardDev.fetch()
```

Returns the standard deviation of the modulation results for LE coded PHY, see 'Square TX Measurement - modulation statistics'. The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for ResultData structure arguments.

read() → ResultData

```
# SCPI: READ:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:LENergy:LRANge:SDEViation
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.lowEnergy.
↪lrange.standardDev.read()
```

Returns the standard deviation of the modulation results for LE coded PHY, see 'Square TX Measurement - modulation statistics'. The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for ResultData structure arguments.

6.1.1.6.3.57 StDev

SCPI Commands :

```
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:LENergy:LRANge:STDev
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:LENergy:LRANge:STDev
READ:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:LENergy:LRANge:STDev
```

class StDevCls

StDev commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float or bool: No parameter help available
- Delta_F_199_P_9: float or bool: No parameter help available
- Freq_Accuracy: float or bool: No parameter help available
- Freq_Drift: float or bool: No parameter help available
- Max_Drift: float or bool: No parameter help available
- Nominal_Power: float or bool: No parameter help available
- Freq_Offset: float or bool: No parameter help available

class ResultData

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float: No parameter help available
- Delta_F_199_P_9: float: No parameter help available
- Freq_Accuracy: float: No parameter help available
- Freq_Drift: float: No parameter help available
- Max_Drift: float: No parameter help available
- Nominal_Power: float: No parameter help available
- Freq_Offset: float: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:LENergy:LRANge:STDev
value: CalculateStruct = driver.bluetooth.measurement.multiEval.modulation.
↪lowEnergy.lrange.stDev.calculate()
```

No command help available

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → ResultData

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:LENergy:LRANge:STDev
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.lowEnergy.
↪lrange.stDev.fetch()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

read() → ResultData

```
# SCPI: READ:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:LENergy:LRANge:STDev
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.lowEnergy.
↪lrange.stDev.read()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

6.1.1.6.3.58 Xmaximum

SCPI Commands :

```
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:LENergy:LRANge:XMAXimum
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:LENergy:LRANge:XMAXimum
READ:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:LENergy:LRANge:XMAXimum
```

class XmaximumCls

Xmaximum commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: 'Reliability indicator'
- Out_Of_Tol: float or bool: Out of tolerance result, i.e. the percentage of measurement intervals of the statistic count ([CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Scount#Modulation CMDLINKRESOLVED]) exceeding the specified limits, see 'Modulation limits (LE) '.
- Delta_F_199_P_9: float or bool: No parameter help available
- Freq_Accuracy: float or bool: No parameter help available
- Freq_Drift: float or bool: No parameter help available
- Max_Drift: float or bool: No parameter help available
- Freq_Dev_Avg_F_1: float or bool: No parameter help available
- Freq_Dev_Min_F_1: float or bool: No parameter help available
- Freq_Dev_Max_F_1: float or bool: No parameter help available
- Nominal_Power: float or bool: No parameter help available
- Freq_Offset: float or bool: No parameter help available

class ResultData

Response structure. Fields:

- Reliability: int: 'Reliability indicator'
- Out_Of_Tol: float: Out of tolerance result, i.e. the percentage of measurement intervals of the statistic count ([CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Scount#Modulation CMDLINKRESOLVED]) exceeding the specified limits, see 'Modulation limits (LE) '.

- Delta_F_199_P_9: float: No parameter help available
- Freq_Accuracy: float: No parameter help available
- Freq_Drift: float: No parameter help available
- Max_Drift: float: No parameter help available
- Freq_Dev_Avg_F_1: float: No parameter help available
- Freq_Dev_Min_F_1: float: No parameter help available
- Freq_Dev_Max_F_1: float: No parameter help available
- Nominal_Power: float: No parameter help available
- Freq_Offset: float: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:LENergy:LRANge:XMAXimum
value: CalculateStruct = driver.bluetooth.measurement.multiEval.modulation.
↪lowEnergy.lrange.xmaximum.calculate()
```

Returns the current, average, absolute min (xmin) , absolute max (xmax) , and max modulation results for LE coded PHY, see 'Square TX Measurement - modulation statistics'. The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → ResultData

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:LENergy:LRANge:XMAXimum
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.lowEnergy.
↪lrange.xmaximum.fetch()
```

Returns the current, average, absolute min (xmin) , absolute max (xmax) , and max modulation results for LE coded PHY, see 'Square TX Measurement - modulation statistics'. The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for ResultData structure arguments.

read() → ResultData

```
# SCPI: READ:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:LENergy:LRANge:XMAXimum
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.lowEnergy.
↪lrange.xmaximum.read()
```

Returns the current, average, absolute min (xmin) , absolute max (xmax) , and max modulation results for LE coded PHY, see 'Square TX Measurement - modulation statistics'. The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for ResultData structure arguments.

6.1.1.6.3.59 Xminimum

SCPI Commands :

```
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:LENergy:LRANge:XMINimum
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:LENergy:LRANge:XMINimum
READ:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:LENergy:LRANge:XMINimum
```

class XminimumCls

Xminimum commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: 'Reliability indicator'
- Out_Of_Tol: float or bool: Out of tolerance result, i.e. the percentage of measurement intervals of the statistic count ([CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Scount#Modulation CMDLINKRESOLVED]) exceeding the specified limits, see 'Modulation limits (LE) '.
- Delta_F_199_P_9: float or bool: No parameter help available
- Freq_Accuracy: float or bool: No parameter help available
- Freq_Drift: float or bool: No parameter help available
- Max_Drift: float or bool: No parameter help available
- Freq_Dev_Avg_F_1: float or bool: No parameter help available
- Freq_Dev_Min_F_1: float or bool: No parameter help available
- Freq_Dev_Max_F_1: float or bool: No parameter help available
- Nominal_Power: float or bool: No parameter help available
- Freq_Offset: float or bool: No parameter help available

class ResultData

Response structure. Fields:

- Reliability: int: 'Reliability indicator'
- Out_Of_Tol: float: Out of tolerance result, i.e. the percentage of measurement intervals of the statistic count ([CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Scount#Modulation CMDLINKRESOLVED]) exceeding the specified limits, see 'Modulation limits (LE) '.
- Delta_F_199_P_9: float: No parameter help available
- Freq_Accuracy: float: No parameter help available
- Freq_Drift: float: No parameter help available
- Max_Drift: float: No parameter help available
- Freq_Dev_Avg_F_1: float: No parameter help available
- Freq_Dev_Min_F_1: float: No parameter help available

- Freq_Dev_Max_F_1: float: No parameter help available
- Nominal_Power: float: No parameter help available
- Freq_Offset: float: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:LENergy:LRANge:XMINimum
value: CalculateStruct = driver.bluetooth.measurement.multiEval.modulation.
↪lowEnergy.lrange.xminimum.calculate()
```

Returns the current, average, absolute min (xmin) , absolute max (xmax) , and max modulation results for LE coded PHY, see 'Square TX Measurement - modulation statistics'. The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → ResultData

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:LENergy:LRANge:XMINimum
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.lowEnergy.
↪lrange.xminimum.fetch()
```

Returns the current, average, absolute min (xmin) , absolute max (xmax) , and max modulation results for LE coded PHY, see 'Square TX Measurement - modulation statistics'. The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for ResultData structure arguments.

read() → ResultData

```
# SCPI: READ:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:LENergy:LRANge:XMINimum
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.lowEnergy.
↪lrange.xminimum.read()
```

Returns the current, average, absolute min (xmin) , absolute max (xmax) , and max modulation results for LE coded PHY, see 'Square TX Measurement - modulation statistics'. The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for ResultData structure arguments.

6.1.1.6.3.60 Nmode

class NmodeCls

Nmode commands group definition. 88 total commands, 2 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.bluetooth.measurement.multiEval.modulation.nmode.clone()
```

Subgroups

6.1.1.6.3.61 Classic

class ClassicCls

Classic commands group definition. 21 total commands, 7 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.bluetooth.measurement.multiEval.modulation.nmode.classic.clone()
```

Subgroups

6.1.1.6.3.62 Average

SCPI Commands :

```
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:NMODe:CLASsic:AVERage
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:NMODe:CLASsic:AVERage
READ:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:NMODe:CLASsic:AVERage
```

class AverageCls

Average commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float or bool: No parameter help available
- Freq_Accuracy: float or bool: No parameter help available
- Freq_Drift: float or bool: No parameter help available
- Max_Drift: float or bool: No parameter help available
- Freq_Dev_Avg_F_1: float or bool: No parameter help available
- Freq_Dev_Min_F_1: float or bool: No parameter help available

- Freq_Dev_Max_F_1: float or bool: No parameter help available
- Freq_Dev_Avg_F_2: float or bool: No parameter help available
- Freq_Dev_Min_F_2: float or bool: No parameter help available
- Freq_Dev_Max_F_2: float or bool: No parameter help available

class ResultData

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float: No parameter help available
- Freq_Accuracy: float: No parameter help available
- Freq_Drift: float: No parameter help available
- Max_Drift: float: No parameter help available
- Freq_Dev_Avg_F_1: float: No parameter help available
- Freq_Dev_Min_F_1: float: No parameter help available
- Freq_Dev_Max_F_1: float: No parameter help available
- Freq_Dev_Avg_F_2: float: No parameter help available
- Freq_Dev_Min_F_2: float: No parameter help available
- Freq_Dev_Max_F_2: float: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:NMODe:CLASsic:AVERage
value: CalculateStruct = driver.bluetooth.measurement.multiEval.modulation.
↪nmode.classic.average.calculate()
```

No command help available

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → ResultData

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:NMODe:CLASsic:AVERage
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.nmode.
↪classic.average.fetch()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

read() → ResultData

```
# SCPI: READ:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:NMODe:CLASsic:AVERage
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.nmode.
↪classic.average.read()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

6.1.1.6.3.63 Current

SCPI Commands :

```
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:NMODe:CLASsic:CURRent
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:NMODe:CLASsic:CURRent
READ:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:NMODe:CLASsic:CURRent
```

`class CurrentCls`

Current commands group definition. 3 total commands, 0 Subgroups, 3 group commands

`class CalculateStruct`

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float or bool: No parameter help available
- Freq_Accuracy: float or bool: No parameter help available
- Freq_Drift: float or bool: No parameter help available
- Max_Drift: float or bool: No parameter help available
- Freq_Dev_Avg_F_1: float or bool: No parameter help available
- Freq_Dev_Min_F_1: float or bool: No parameter help available
- Freq_Dev_Max_F_1: float or bool: No parameter help available
- Freq_Dev_Avg_F_2: float or bool: No parameter help available
- Freq_Dev_Min_F_2: float or bool: No parameter help available
- Freq_Dev_Max_F_2: float or bool: No parameter help available

`class ResultData`

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float: No parameter help available
- Freq_Accuracy: float: No parameter help available
- Freq_Drift: float: No parameter help available
- Max_Drift: float: No parameter help available
- Freq_Dev_Avg_F_1: float: No parameter help available
- Freq_Dev_Min_F_1: float: No parameter help available
- Freq_Dev_Max_F_1: float: No parameter help available
- Freq_Dev_Avg_F_2: float: No parameter help available
- Freq_Dev_Min_F_2: float: No parameter help available

- Freq_Dev_Max_F_2: float: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:NMODe:CLASsic:CURRent
value: CalculateStruct = driver.bluetooth.measurement.multiEval.modulation.
↪nmode.classic.current.calculate()
```

No command help available

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → ResultData

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:NMODe:CLASsic:CURRent
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.nmode.
↪classic.current.fetch()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

read() → ResultData

```
# SCPI: READ:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:NMODe:CLASsic:CURRent
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.nmode.
↪classic.current.read()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

6.1.1.6.3.64 Maximum

SCPI Commands :

```
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:NMODe:CLASsic:MAXimum
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:NMODe:CLASsic:MAXimum
READ:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:NMODe:CLASsic:MAXimum
```

class MaximumCls

Maximum commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float or bool: No parameter help available
- Freq_Accuracy: float or bool: No parameter help available

- Freq_Drift: float or bool: No parameter help available
- Max_Drift: float or bool: No parameter help available
- Freq_Dev_Avg_F_1: float or bool: No parameter help available
- Freq_Dev_Min_F_1: float or bool: No parameter help available
- Freq_Dev_Max_F_1: float or bool: No parameter help available
- Freq_Dev_Avg_F_2: float or bool: No parameter help available
- Freq_Dev_Min_F_2: float or bool: No parameter help available
- Freq_Dev_Max_F_2: float or bool: No parameter help available

class ResultData

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float: No parameter help available
- Freq_Accuracy: float: No parameter help available
- Freq_Drift: float: No parameter help available
- Max_Drift: float: No parameter help available
- Freq_Dev_Avg_F_1: float: No parameter help available
- Freq_Dev_Min_F_1: float: No parameter help available
- Freq_Dev_Max_F_1: float: No parameter help available
- Freq_Dev_Avg_F_2: float: No parameter help available
- Freq_Dev_Min_F_2: float: No parameter help available
- Freq_Dev_Max_F_2: float: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:NMODe:CLASsic:MAXimum
value: CalculateStruct = driver.bluetooth.measurement.multiEval.modulation.
↪nmode.classic.maximum.calculate()
```

No command help available

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → ResultData

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:NMODe:CLASsic:MAXimum
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.nmode.
↪classic.maximum.fetch()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

read() → ResultData

```
# SCPI: READ:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:NMODe:CLASsic:MAXimum
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.nmode.
↪classic.maximum.read()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

6.1.1.6.3.65 Minimum

SCPI Commands :

```
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:NMODe:CLASsic:MINimum
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:NMODe:CLASsic:MINimum
READ:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:NMODe:CLASsic:MINimum
```

class MinimumCls

Minimum commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float or bool: No parameter help available
- Freq_Dev_Avg_F_1: float or bool: No parameter help available
- Freq_Dev_Min_F_1: float or bool: No parameter help available
- Freq_Dev_Max_F_1: float or bool: No parameter help available
- Freq_Dev_Avg_F_2: float or bool: No parameter help available
- Freq_Dev_Min_F_2: float or bool: No parameter help available
- Freq_Dev_Max_F_2: float or bool: No parameter help available

class ResultData

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float: No parameter help available
- Freq_Dev_Avg_F_1: float: No parameter help available
- Freq_Dev_Min_F_1: float: No parameter help available
- Freq_Dev_Max_F_1: float: No parameter help available
- Freq_Dev_Avg_F_2: float: No parameter help available
- Freq_Dev_Min_F_2: float: No parameter help available
- Freq_Dev_Max_F_2: float: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:NMODe:CLASsic:MINimum
value: CalculateStruct = driver.bluetooth.measurement.multiEval.modulation.
↪nmode.classic.minimum.calculate()
```

No command help available

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → ResultData

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:NMODe:CLASsic:MINimum
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.nmode.
↪classic.minimum.fetch()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

read() → ResultData

```
# SCPI: READ:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:NMODe:CLASsic:MINimum
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.nmode.
↪classic.minimum.read()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

6.1.1.6.3.66 StandardDev

SCPI Commands :

```
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:NMODe:CLASsic:SDEViation
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:NMODe:CLASsic:SDEViation
READ:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:NMODe:CLASsic:SDEViation
```

class StandardDevCls

StandardDev commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float or bool: No parameter help available
- Freq_Accuracy: float or bool: No parameter help available
- Freq_Drift: float or bool: No parameter help available

- Max_Drift: float or bool: No parameter help available

class ResultData

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float: No parameter help available
- Freq_Accuracy: float: No parameter help available
- Freq_Drift: float: No parameter help available
- Max_Drift: float: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:NMODe:CLASsic:SDEViation
value: CalculateStruct = driver.bluetooth.measurement.multiEval.modulation.
↪nmode.classic.standardDev.calculate()
```

No command help available

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → ResultData

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:NMODe:CLASsic:SDEViation
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.nmode.
↪classic.standardDev.fetch()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

read() → ResultData

```
# SCPI: READ:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:NMODe:CLASsic:SDEViation
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.nmode.
↪classic.standardDev.read()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

6.1.1.6.3.67 Xmaximum

SCPI Commands :

```
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:NMODe:CLASsic:XMAXimum
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:NMODe:CLASsic:XMAXimum
READ:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:NMODe:CLASsic:XMAXimum
```

class XmaximumCls

Xmaximum commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float or bool: No parameter help available
- Freq_Accuracy: float or bool: No parameter help available
- Freq_Drift: float or bool: No parameter help available
- Max_Drift_Rate: float or bool: No parameter help available
- Freq_Dev_Avg_F_1: float or bool: No parameter help available
- Freq_Dev_Min_F_1: float or bool: No parameter help available
- Freq_Dev_Max_F_1: float or bool: No parameter help available
- Freq_Dev_Avg_F_2: float or bool: No parameter help available
- Freq_Dev_Min_F_2: float or bool: No parameter help available
- Freq_Dev_Max_F_2: float or bool: No parameter help available

class ResultData

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float: No parameter help available
- Freq_Accuracy: float: No parameter help available
- Freq_Drift: float: No parameter help available
- Max_Drift: float: No parameter help available
- Freq_Dev_Avg_F_1: float: No parameter help available
- Freq_Dev_Min_F_1: float: No parameter help available
- Freq_Dev_Max_F_1: float: No parameter help available
- Freq_Dev_Avg_F_2: float: No parameter help available
- Freq_Dev_Min_F_2: float: No parameter help available
- Freq_Dev_Max_F_2: float: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:NMODe:CLASsic:XMAXimum
value: CalculateStruct = driver.bluetooth.measurement.multiEval.modulation.
↪nmode.classic.xmaximum.calculate()
```

No command help available

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → ResultData

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:NMODe:CLASsic:XMAXimum
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.nmode.
↪classic.xmaximum.fetch()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

read() → ResultData

```
# SCPI: READ:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:NMODe:CLASsic:XMAXimum
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.nmode.
↪classic.xmaximum.read()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

6.1.1.6.3.68 Xminimum

SCPI Commands :

```
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:NMODe:CLASsic:XMINimum
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:NMODe:CLASsic:XMINimum
READ:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:NMODe:CLASsic:XMINimum
```

class XminimumCls

Xminimum commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float or bool: No parameter help available
- Freq_Accuracy: float or bool: No parameter help available
- Freq_Drift: float or bool: No parameter help available
- Max_Drift: float or bool: No parameter help available

- Freq_Dev_Avg_F_1: float or bool: No parameter help available
- Freq_Dev_Min_F_1: float or bool: No parameter help available
- Freq_Dev_Max_F_1: float or bool: No parameter help available
- Freq_Dev_Avg_F_2: float or bool: No parameter help available
- Freq_Dev_Min_F_2: float or bool: No parameter help available
- Freq_Dev_Max_F_2: float or bool: No parameter help available

class ResultData

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float: No parameter help available
- Freq_Accuracy: float: No parameter help available
- Freq_Drift: float: No parameter help available
- Max_Drift: float: No parameter help available
- Freq_Dev_Avg_F_1: float: No parameter help available
- Freq_Dev_Min_F_1: float: No parameter help available
- Freq_Dev_Max_F_1: float: No parameter help available
- Freq_Dev_Avg_F_2: float: No parameter help available
- Freq_Dev_Min_F_2: float: No parameter help available
- Freq_Dev_Max_F_2: float: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:NMODe:CLASsic:XMINimum
value: CalculateStruct = driver.bluetooth.measurement.multiEval.modulation.
↪nmode.classic.xminimum.calculate()
```

No command help available

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → ResultData

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:NMODe:CLASsic:XMINimum
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.nmode.
↪classic.xminimum.fetch()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

read() → ResultData

```
# SCPI: READ:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:NMODe:CLASsic:XMINimum
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.nmode.
↪classic.xminimum.read()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

6.1.1.6.3.69 LowEnergy

class LowEnergyCls

LowEnergy commands group definition. 67 total commands, 3 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.bluetooth.measurement.multiEval.modulation.nmode.lowEnergy.clone()
```

Subgroups

6.1.1.6.3.70 Le1M

class Le1MCls

Le1M commands group definition. 21 total commands, 7 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.bluetooth.measurement.multiEval.modulation.nmode.lowEnergy.le1M.clone()
```

Subgroups

6.1.1.6.3.71 Average

SCPI Commands :

```
CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:NMODe:LENergy[:LE1M]:AVERage
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:NMODe:LENergy[:LE1M]:AVERage
READ:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:NMODe:LENergy[:LE1M]:AVERage
```

class AverageCls

Average commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float or bool: No parameter help available
- Freq_Accuracy: float or bool: No parameter help available
- Freq_Offset: float or bool: No parameter help available
- Freq_Drift: float or bool: No parameter help available
- Init_Freq_Drift: float or bool: No parameter help available
- Max_Drift: float or bool: No parameter help available
- Freq_Dev_Avg_F_1: float or bool: No parameter help available
- Freq_Dev_Min_F_1: float or bool: No parameter help available
- Freq_Dev_Max_F_1: float or bool: No parameter help available
- Freq_Dev_Avg_F_2: float or bool: No parameter help available
- Freq_Dev_Min_F_2: float or bool: No parameter help available
- Freq_Dev_Max_F_2: float or bool: No parameter help available

class ResultData

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float: No parameter help available
- Freq_Accuracy: float: No parameter help available
- Freq_Offset: float: No parameter help available
- Freq_Drift: float: No parameter help available
- Init_Freq_Drift: float: No parameter help available
- Max_Drift: float: No parameter help available
- Freq_Dev_Avg_F_1: float: No parameter help available
- Freq_Dev_Min_F_1: float: No parameter help available
- Freq_Dev_Max_F_1: float: No parameter help available
- Freq_Dev_Avg_F_2: float: No parameter help available
- Freq_Dev_Min_F_2: float: No parameter help available
- Freq_Dev_Max_F_2: float: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:NMODe:LENergy[:LE1M]:AVERage
value: CalculateStruct = driver.bluetooth.measurement.multiEval.modulation.
↪nmode.lowEnergy.le1M.average.calculate()
```

No command help available

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → ResultData

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:NMODe:LENergy[:LE1M]:AVERage
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.nmode.
↪lowEnergy.le1M.average.fetch()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

read() → ResultData

```
# SCPI: READ:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:NMODe:LENergy[:LE1M]:AVERage
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.nmode.
↪lowEnergy.le1M.average.read()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

6.1.1.6.3.72 Current

SCPI Commands :

```
CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:NMODe:LENergy[:LE1M]:CURRent
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:NMODe:LENergy[:LE1M]:CURRent
READ:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:NMODe:LENergy[:LE1M]:CURRent
```

class CurrentCls

Current commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float or bool: No parameter help available
- Freq_Accuracy: float or bool: No parameter help available
- Freq_Offset: float or bool: No parameter help available
- Freq_Drift: float or bool: No parameter help available
- Init_Freq_Drift: float or bool: No parameter help available
- Max_Drift: float or bool: No parameter help available
- Freq_Dev_Avg_F_1: float or bool: No parameter help available
- Freq_Dev_Min_F_1: float or bool: No parameter help available

- Freq_Dev_Max_F_1: float or bool: No parameter help available
- Freq_Dev_Avg_F_2: float or bool: No parameter help available
- Freq_Dev_Min_F_2: float or bool: No parameter help available
- Freq_Dev_Max_F_2: float or bool: No parameter help available

class ResultData

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float: No parameter help available
- Freq_Accuracy: float: No parameter help available
- Freq_Offset: float: No parameter help available
- Freq_Drift: float: No parameter help available
- Init_Freq_Drift: float: No parameter help available
- Max_Drift: float: No parameter help available
- Freq_Dev_Avg_F_1: float: No parameter help available
- Freq_Dev_Min_F_1: float: No parameter help available
- Freq_Dev_Max_F_1: float: No parameter help available
- Freq_Dev_Avg_F_2: float: No parameter help available
- Freq_Dev_Min_F_2: float: No parameter help available
- Freq_Dev_Max_F_2: float: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:NMODe:LENergy[:LE1M]:CURRent
value: CalculateStruct = driver.bluetooth.measurement.multiEval.modulation.
↪nmode.lowEnergy.le1M.current.calculate()
```

No command help available

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → ResultData

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:NMODe:LENergy[:LE1M]:CURRent
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.nmode.
↪lowEnergy.le1M.current.fetch()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

read() → ResultData

```
# SCPI: READ:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:NMODe:LENergy[:LE1M]:CURRent
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.nmode.
↪lowEnergy.le1M.current.read()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

6.1.1.6.3.73 Maximum

SCPI Commands :

```
CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:NMODe:LENergy[:LE1M]:MAXimum
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:NMODe:LENergy[:LE1M]:MAXimum
READ:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:NMODe:LENergy[:LE1M]:MAXimum
```

class MaximumCls

Maximum commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float or bool: No parameter help available
- Freq_Accuracy: float or bool: No parameter help available
- Freq_Offset: float or bool: No parameter help available
- Freq_Drift: float or bool: No parameter help available
- Init_Freq_Drift: float or bool: No parameter help available
- Max_Drift: float or bool: No parameter help available
- Freq_Dev_Avg_F_1: float or bool: No parameter help available
- Freq_Dev_Min_F_1: float or bool: No parameter help available
- Freq_Dev_Max_F_1: float or bool: No parameter help available
- Freq_Dev_Avg_F_2: float or bool: No parameter help available
- Freq_Dev_Min_F_2: float or bool: No parameter help available
- Freq_Dev_Max_F_2: float or bool: No parameter help available

class FetchStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float: No parameter help available
- Freq_Accuracy: float: No parameter help available
- Freq_Offset: float: No parameter help available

- Freq_Drift: float: No parameter help available
- Init_Freq_Drift: float: No parameter help available
- Max_Drift: float: No parameter help available
- Freq_Dev_Avg_F_1: float: No parameter help available
- Freq_Dev_Min_F_1: float: No parameter help available
- Freq_Dev_Max_F_1: float: No parameter help available
- Freq_Dev_Avg_F_2: float: No parameter help available
- Freq_Dev_Min_F_2: float: No parameter help available
- Freq_Dev_Max_F_2: float: No parameter help available

class ReadStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float: No parameter help available
- Freq_Accuracy: float: No parameter help available
- Freq_Offset: float: No parameter help available
- Freq_Drift: float: No parameter help available
- Init_Freq_Drift: float: No parameter help available
- Max_Drift: float: No parameter help available
- Freq_Dev_Avg_F_1: str: No parameter help available
- Freq_Dev_Min_F_1: float: No parameter help available
- Freq_Dev_Max_F_1: float: No parameter help available
- Freq_Dev_Avg_F_2: float: No parameter help available
- Freq_Dev_Min_F_2: float: No parameter help available
- Freq_Dev_Max_F_2: float: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:NMODe:LENergy[:LE1M]:MAXimum
value: CalculateStruct = driver.bluetooth.measurement.multiEval.modulation.
↪nmode.lowEnergy.le1M.maximum.calculate()
```

No command help available

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → FetchStruct

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:NMODe:LENergy[:LE1M]:MAXimum
value: FetchStruct = driver.bluetooth.measurement.multiEval.modulation.nmode.
↪lowEnergy.le1M.maximum.fetch()
```

No command help available

return
structure: for return value, see the help for FetchStruct structure arguments.

read() → ReadStruct

```
# SCPI: READ:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:NMODe:LENergy[:LE1M]:MAXimum
value: ReadStruct = driver.bluetooth.measurement.multiEval.modulation.nmode.
↪lowEnergy.le1M.maximum.read()
```

No command help available

return
structure: for return value, see the help for ReadStruct structure arguments.

6.1.1.6.3.74 Minimum

SCPI Commands :

```
CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:NMODe:LENergy[:LE1M]:MINimum
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:NMODe:LENergy[:LE1M]:MINimum
READ:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:NMODe:LENergy[:LE1M]:MINimum
```

class MinimumCls

Minimum commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float or bool: No parameter help available
- Freq_Dev_Avg_F_1: float or bool: No parameter help available
- Freq_Dev_Min_F_1: float or bool: No parameter help available
- Freq_Dev_Max_F_1: float or bool: No parameter help available
- Freq_Dev_Avg_F_2: float or bool: No parameter help available
- Freq_Dev_Min_F_2: float or bool: No parameter help available
- Freq_Dev_Max_F_2: float or bool: No parameter help available

class FetchStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float: No parameter help available
- Freq_Dev_Avg_F_1: float: No parameter help available
- Freq_Dev_Min_F_1: float: No parameter help available
- Freq_Dev_Max_F_1: float: No parameter help available

- Freq_Dev_Avg_F_2: float: No parameter help available
- Freq_Dev_Min_F_2: float: No parameter help available
- Freq_Dev_Max_F_2: float: No parameter help available

class ReadStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float: No parameter help available
- Freq_Dev_Avg_F_1: str: No parameter help available
- Freq_Dev_Min_F_1: float: No parameter help available
- Freq_Dev_Max_F_1: float: No parameter help available
- Freq_Dev_Avg_F_2: float: No parameter help available
- Freq_Dev_Min_F_2: float: No parameter help available
- Freq_Dev_Max_F_2: float: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:NMODe:LENergy[:LE1M]:MINimum
value: CalculateStruct = driver.bluetooth.measurement.multiEval.modulation.
↪nmode.lowEnergy.le1M.minimum.calculate()
```

No command help available

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → FetchStruct

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:NMODe:LENergy[:LE1M]:MINimum
value: FetchStruct = driver.bluetooth.measurement.multiEval.modulation.nmode.
↪lowEnergy.le1M.minimum.fetch()
```

No command help available

return
structure: for return value, see the help for FetchStruct structure arguments.

read() → ReadStruct

```
# SCPI: READ:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:NMODe:LENergy[:LE1M]:MINimum
value: ReadStruct = driver.bluetooth.measurement.multiEval.modulation.nmode.
↪lowEnergy.le1M.minimum.read()
```

No command help available

return
structure: for return value, see the help for ReadStruct structure arguments.

6.1.1.6.3.75 StandardDev

SCPI Commands :

```
CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:NMODe:LENergy[:LE1M]:SDEViation
FETCh:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:NMODe:LENergy[:LE1M]:SDEViation
READ:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:NMODe:LENergy[:LE1M]:SDEViation
```

class StandardDevCls

StandardDev commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float or bool: No parameter help available
- Freq_Accuracy: float or bool: No parameter help available
- Freq_Drift: float or bool: No parameter help available
- Max_Drift: float or bool: No parameter help available

class ResultData

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float: No parameter help available
- Freq_Accuracy: float: No parameter help available
- Freq_Drift: float: No parameter help available
- Max_Drift: float: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:NMODe:LENergy[:LE1M]:SDEViation
value: CalculateStruct = driver.bluetooth.measurement.multiEval.modulation.
↪nmode.lowEnergy.le1M.standardDev.calculate()
```

No command help available

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → ResultData

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:NMODe:LENergy[:LE1M]:SDEViation
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.nmode.
↪lowEnergy.le1M.standardDev.fetch()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

read() → ResultData

```
# SCPI: READ:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:NMODe:LENergy[:LE1M]:SDEViation
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.nmode.
↪lowEnergy.le1M.standardDev.read()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

6.1.1.6.3.76 Xmaximum

SCPI Commands :

```
CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:NMODe:LENergy[:LE1M]:XMAXimum
FETCh:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:NMODe:LENergy[:LE1M]:XMAXimum
READ:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:NMODe:LENergy[:LE1M]:XMAXimum
```

class XmaximumCls

Xmaximum commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float or bool: No parameter help available
- Freq_Accuracy: float or bool: No parameter help available
- Freq_Offset: float or bool: No parameter help available
- Freq_Drift: float or bool: No parameter help available
- Init_Freq_Drift: float or bool: No parameter help available
- Max_Drift_Rate: float or bool: No parameter help available
- Freq_Dev_Avg_F_1: float or bool: No parameter help available
- Freq_Dev_Min_F_1: float or bool: No parameter help available
- Freq_Dev_Max_F_1: float or bool: No parameter help available
- Freq_Dev_Avg_F_2: float or bool: No parameter help available
- Freq_Dev_Min_F_2: float or bool: No parameter help available
- Freq_Dev_Max_F_2: float or bool: No parameter help available

class ResultData

Response structure. Fields:

- Reliability: int: No parameter help available

- Out_Of_Tol: float: No parameter help available
- Freq_Accuracy: float: No parameter help available
- Freq_Offset: float: No parameter help available
- Freq_Drift: float: No parameter help available
- Init_Freq_Drift: float: No parameter help available
- Max_Drift: float: No parameter help available
- Freq_Dev_Avg_F_1: float: No parameter help available
- Freq_Dev_Min_F_1: float: No parameter help available
- Freq_Dev_Max_F_1: float: No parameter help available
- Freq_Dev_Avg_F_2: float: No parameter help available
- Freq_Dev_Min_F_2: float: No parameter help available
- Freq_Dev_Max_F_2: float: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:NMODe:LENergy[:LE1M]:XMAXimum
value: CalculateStruct = driver.bluetooth.measurement.multiEval.modulation.
↪nmode.lowEnergy.le1M.xmaximum.calculate()
```

No command help available

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → ResultData

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:NMODe:LENergy[:LE1M]:XMAXimum
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.nmode.
↪lowEnergy.le1M.xmaximum.fetch()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

read() → ResultData

```
# SCPI: READ:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:NMODe:LENergy[:LE1M]:XMAXimum
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.nmode.
↪lowEnergy.le1M.xmaximum.read()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

6.1.1.6.3.77 Xminimum

SCPI Commands :

```
CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:NMODe:LENergy[:LE1M]:XMINimum
FETCh:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:NMODe:LENergy[:LE1M]:XMINimum
READ:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:NMODe:LENergy[:LE1M]:XMINimum
```

class XminimumCls

Xminimum commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float or bool: No parameter help available
- Freq_Accuracy: float or bool: No parameter help available
- Freq_Offset: float or bool: No parameter help available
- Freq_Drift: float or bool: No parameter help available
- Init_Freq_Drift: float or bool: No parameter help available
- Max_Drift: float or bool: No parameter help available
- Freq_Dev_Avg_F_1: float or bool: No parameter help available
- Freq_Dev_Min_F_1: float or bool: No parameter help available
- Freq_Dev_Max_F_1: float or bool: No parameter help available
- Freq_Dev_Avg_F_2: float or bool: No parameter help available
- Freq_Dev_Min_F_2: float or bool: No parameter help available
- Freq_Dev_Max_F_2: float or bool: No parameter help available

class ResultData

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float: No parameter help available
- Freq_Accuracy: float: No parameter help available
- Freq_Offset: float: No parameter help available
- Freq_Drift: float: No parameter help available
- Init_Freq_Drift: float: No parameter help available
- Max_Drift: float: No parameter help available
- Freq_Dev_Avg_F_1: float: No parameter help available
- Freq_Dev_Min_F_1: float: No parameter help available
- Freq_Dev_Max_F_1: float: No parameter help available
- Freq_Dev_Avg_F_2: float: No parameter help available

- Freq_Dev_Min_F_2: float: No parameter help available
- Freq_Dev_Max_F_2: float: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:NMODe:LENergy[:LE1M]:XMINimum
value: CalculateStruct = driver.bluetooth.measurement.multiEval.modulation.
↪nmode.lowEnergy.le1M.xminimum.calculate()
```

No command help available

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → ResultData

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:NMODe:LENergy[:LE1M]:XMINimum
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.nmode.
↪lowEnergy.le1M.xminimum.fetch()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

read() → ResultData

```
# SCPI: READ:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:NMODe:LENergy[:LE1M]:XMINimum
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.nmode.
↪lowEnergy.le1M.xminimum.read()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

6.1.1.6.3.78 Le2M

class Le2MCls

Le2M commands group definition. 22 total commands, 8 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.bluetooth.measurement.multiEval.modulation.nmode.lowEnergy.le2M.clone()
```

Subgroups

6.1.1.6.3.79 Average

SCPI Commands :

```
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:NMODe:LENergy:LE2M:AVERage
CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:NMODe:LENergy:LE2M:AVERage
READ:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:NMODe:LENergy:LE2M:AVERage
```

class AverageCls

Average commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float or bool: No parameter help available
- Delta_F_299_P_9: float or bool: No parameter help available
- Freq_Accuracy: float or bool: No parameter help available
- Freq_Drift: float or bool: No parameter help available
- Max_Drift: float or bool: No parameter help available
- Freq_Dev_Avg_F_1: float or bool: No parameter help available
- Freq_Dev_Min_F_1: float or bool: No parameter help available
- Freq_Dev_Max_F_1: float or bool: No parameter help available
- Freq_Dev_Avg_F_2: float or bool: No parameter help available
- Freq_Dev_Min_F_2: float or bool: No parameter help available
- Freq_Dev_Max_F_2: float or bool: No parameter help available
- Nominal_Power: float or bool: No parameter help available
- Mod_Ratio: enums.ResultStatus2: No parameter help available
- Freq_Offset: float or bool: No parameter help available
- Init_Freq_Drift: float or bool: No parameter help available

class ResultData

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float: No parameter help available
- Delta_F_299_P_9: float: No parameter help available
- Freq_Accuracy: float: No parameter help available
- Freq_Drift: float: No parameter help available
- Max_Drift: float: No parameter help available
- Freq_Dev_Avg_F_1: float: No parameter help available

- Freq_Dev_Min_F_1: float: No parameter help available
- Freq_Dev_Max_F_1: float: No parameter help available
- Freq_Dev_Avg_F_2: float: No parameter help available
- Freq_Dev_Min_F_2: float: No parameter help available
- Freq_Dev_Max_F_2: float: No parameter help available
- Nominal_Power: float: No parameter help available
- Mod_Ratio: float: No parameter help available
- Freq_Offset: float: No parameter help available
- Init_Freq_Drift: float: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:NMODe:LENergy:LE2M:AVERage
value: CalculateStruct = driver.bluetooth.measurement.multiEval.modulation.
↪nmode.lowEnergy.le2M.average.calculate()
```

No command help available

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → ResultData

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:NMODe:LENergy:LE2M:AVERage
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.nmode.
↪lowEnergy.le2M.average.fetch()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

read() → ResultData

```
# SCPI: READ:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:NMODe:LENergy:LE2M:AVERage
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.nmode.
↪lowEnergy.le2M.average.read()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

6.1.1.6.3.80 Current

SCPI Commands :

```
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:NMODe:LENergy:LE2M:CURRent
CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:NMODe:LENergy:LE2M:CURRent
READ:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:NMODe:LENergy:LE2M:CURRent
```

class CurrentCls

Current commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float or bool: No parameter help available
- Delta_F_299_P_9: float or bool: No parameter help available
- Freq_Accuracy: float or bool: No parameter help available
- Freq_Drift: float or bool: No parameter help available
- Max_Drift: float or bool: No parameter help available
- Freq_Dev_Avg_F_1: float or bool: No parameter help available
- Freq_Dev_Min_F_1: float or bool: No parameter help available
- Freq_Dev_Max_F_1: float or bool: No parameter help available
- Freq_Dev_Avg_F_2: float or bool: No parameter help available
- Freq_Dev_Min_F_2: float or bool: No parameter help available
- Freq_Dev_Max_F_2: float or bool: No parameter help available
- Nominal_Power: float or bool: No parameter help available
- Mod_Ratio: enums.ResultStatus2: No parameter help available
- Freq_Offset: float or bool: No parameter help available
- Init_Freq_Drift: float or bool: No parameter help available

class ResultData

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float: No parameter help available
- Delta_F_299_P_9: float: No parameter help available
- Freq_Accuracy: float: No parameter help available
- Freq_Drift: float: No parameter help available
- Max_Drift: float: No parameter help available
- Freq_Dev_Avg_F_1: float: No parameter help available
- Freq_Dev_Min_F_1: float: No parameter help available

- Freq_Dev_Max_F_1: float: No parameter help available
- Freq_Dev_Avg_F_2: float: No parameter help available
- Freq_Dev_Min_F_2: float: No parameter help available
- Freq_Dev_Max_F_2: float: No parameter help available
- Nominal_Power: float: No parameter help available
- Mod_Ratio: float: No parameter help available
- Freq_Offset: float: No parameter help available
- Init_Freq_Drift: float: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:NMODe:LENergy:LE2M:CURRent
value: CalculateStruct = driver.bluetooth.measurement.multiEval.modulation.
↪nmode.lowEnergy.le2M.current.calculate()
```

No command help available

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → ResultData

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:NMODe:LENergy:LE2M:CURRent
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.nmode.
↪lowEnergy.le2M.current.fetch()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

read() → ResultData

```
# SCPI: READ:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:NMODe:LENergy:LE2M:CURRent
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.nmode.
↪lowEnergy.le2M.current.read()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

6.1.1.6.3.81 Maximum

SCPI Commands :

```
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:NMODe:LENergy:LE2M:MAXimum
CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:NMODe:LENergy:LE2M:MAXimum
READ:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:NMODe:LENergy:LE2M:MAXimum
```

class MaximumCls

Maximum commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float or bool: No parameter help available
- Delta_F_299_P_9: float or bool: No parameter help available
- Freq_Accuracy: float or bool: No parameter help available
- Freq_Drift: float or bool: No parameter help available
- Max_Drift: float or bool: No parameter help available
- Freq_Dev_Avg_F_1: float or bool: No parameter help available
- Freq_Dev_Min_F_1: float or bool: No parameter help available
- Freq_Dev_Max_F_1: float or bool: No parameter help available
- Freq_Dev_Avg_F_2: float or bool: No parameter help available
- Freq_Dev_Min_F_2: float or bool: No parameter help available
- Freq_Dev_Max_F_2: float or bool: No parameter help available
- Nominal_Power: float or bool: No parameter help available
- Mod_Ratio: enums.ResultStatus2: No parameter help available
- Freq_Offset: float or bool: No parameter help available
- Init_Freq_Drift: float or bool: No parameter help available

class ResultData

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float: No parameter help available
- Delta_F_299_P_9: float: No parameter help available
- Freq_Accuracy: float: No parameter help available
- Freq_Drift: float: No parameter help available
- Max_Drift: float: No parameter help available
- Freq_Dev_Avg_F_1: float: No parameter help available
- Freq_Dev_Min_F_1: float: No parameter help available

- Freq_Dev_Max_F_1: float: No parameter help available
- Freq_Dev_Avg_F_2: float: No parameter help available
- Freq_Dev_Min_F_2: float: No parameter help available
- Freq_Dev_Max_F_2: float: No parameter help available
- Nominal_Power: float: No parameter help available
- Mod_Ratio: float: No parameter help available
- Freq_Offset: float: No parameter help available
- Init_Freq_Drift: float: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:NMODe:LENergy:LE2M:MAXimum
value: CalculateStruct = driver.bluetooth.measurement.multiEval.modulation.
↪nmode.lowEnergy.le2M.maximum.calculate()
```

No command help available

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → ResultData

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:NMODe:LENergy:LE2M:MAXimum
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.nmode.
↪lowEnergy.le2M.maximum.fetch()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

read() → ResultData

```
# SCPI: READ:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:NMODe:LENergy:LE2M:MAXimum
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.nmode.
↪lowEnergy.le2M.maximum.read()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

6.1.1.6.3.82 Minimum

SCPI Commands :

```
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:NMODe:LENergy:LE2M:MINimum
CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:NMODe:LENergy:LE2M:MINimum
READ:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:NMODe:LENergy:LE2M:MINimum
```

class MinimumCls

Minimum commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float or bool: No parameter help available
- Delta_F_299_P_9: float or bool: No parameter help available
- Freq_Dev_Avg_F_1: float or bool: No parameter help available
- Freq_Dev_Min_F_1: float or bool: No parameter help available
- Freq_Dev_Max_F_1: float or bool: No parameter help available
- Freq_Dev_Avg_F_2: float or bool: No parameter help available
- Freq_Dev_Min_F_2: float or bool: No parameter help available
- Freq_Dev_Max_F_2: float or bool: No parameter help available
- Nominal_Power: float or bool: No parameter help available
- Mod_Ratio: enums.ResultStatus2: No parameter help available

class ResultData

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float: No parameter help available
- Delta_F_299_P_9: float: No parameter help available
- Freq_Dev_Avg_F_1: float: No parameter help available
- Freq_Dev_Min_F_1: float: No parameter help available
- Freq_Dev_Max_F_1: float: No parameter help available
- Freq_Dev_Avg_F_2: float: No parameter help available
- Freq_Dev_Min_F_2: float: No parameter help available
- Freq_Dev_Max_F_2: float: No parameter help available
- Nominal_Power: float: No parameter help available
- Mod_Ratio: float: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:NMODe:LENergy:LE2M:MINimum
value: CalculateStruct = driver.bluetooth.measurement.multiEval.modulation.
↪nmode.lowEnergy.le2M.minimum.calculate()
```

No command help available

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → ResultData

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:NMODe:LENergy:LE2M:MINimum
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.nmode.
↪lowEnergy.le2M.minimum.fetch()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

read() → ResultData

```
# SCPI: READ:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:NMODe:LENergy:LE2M:MINimum
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.nmode.
↪lowEnergy.le2M.minimum.read()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

6.1.1.6.3.83 StandardDev

SCPI Commands :

```
FETCh:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:NMODe:LENergy:LE2M:SDEViation
CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:NMODe:LENergy:LE2M:SDEViation
READ:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:NMODe:LENergy:LE2M:SDEViation
```

class StandardDevCls

StandardDev commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float or bool: No parameter help available
- Delta_F_299_P_9: float or bool: No parameter help available

- Freq_Accuracy: float or bool: No parameter help available
- Freq_Drift: float or bool: No parameter help available
- Max_Drift: float or bool: No parameter help available
- Nominal_Power: float or bool: No parameter help available
- Mod_Ratio: enums.ResultStatus2: No parameter help available
- Freq_Offset: float or bool: No parameter help available
- Init_Freq_Drift: float or bool: No parameter help available

class ResultData

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float: No parameter help available
- Delta_F_299_P_9: float: No parameter help available
- Freq_Accuracy: float: No parameter help available
- Freq_Drift: float: No parameter help available
- Max_Drift: float: No parameter help available
- Nominal_Power: float: No parameter help available
- Mod_Ratio: float: No parameter help available
- Freq_Offset: float: No parameter help available
- Init_Freq_Drift: float: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:NMODe:LENergy:LE2M:SDEViation
value: CalculateStruct = driver.bluetooth.measurement.multiEval.modulation.
↪nmode.lowEnergy.le2M.standardDev.calculate()
```

No command help available

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → ResultData

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:NMODe:LENergy:LE2M:SDEViation
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.nmode.
↪lowEnergy.le2M.standardDev.fetch()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

read() → ResultData

```
# SCPI: READ:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:NMODe:LENergy:LE2M:SDEViation
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.nmode.
↪lowEnergy.le2M.standardDev.read()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

6.1.1.6.3.84 Xmaximum

SCPI Commands :

```
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:NMODe:LENergy:LE2M:XMAXimum
CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:NMODe:LENergy:LE2M:XMAXimum
READ:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:NMODe:LENergy:LE2M:XMAXimum
```

class XmaximumCls

Xmaximum commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float or bool: No parameter help available
- Delta_F_299_P_9: float or bool: No parameter help available
- Freq_Accuracy: float or bool: No parameter help available
- Freq_Drift: float or bool: No parameter help available
- Max_Drift: float or bool: No parameter help available
- Freq_Dev_Avg_F_1: float or bool: No parameter help available
- Freq_Dev_Min_F_1: float or bool: No parameter help available
- Freq_Dev_Max_F_1: float or bool: No parameter help available
- Freq_Dev_Avg_F_2: float or bool: No parameter help available
- Freq_Dev_Min_F_2: float or bool: No parameter help available
- Freq_Dev_Max_F_2: float or bool: No parameter help available
- Nominal_Power: float or bool: No parameter help available
- Mod_Ratio: enums.ResultStatus2: No parameter help available
- Freq_Offset: float or bool: No parameter help available
- Init_Freq_Drift: float or bool: No parameter help available

class ResultData

Response structure. Fields:

- Reliability: int: No parameter help available

- Out_Of_Tol: float: No parameter help available
- Delta_F_299_P_9: float: No parameter help available
- Freq_Accuracy: float: No parameter help available
- Freq_Drift: float: No parameter help available
- Max_Drift: float: No parameter help available
- Freq_Dev_Avg_F_1: float: No parameter help available
- Freq_Dev_Min_F_1: float: No parameter help available
- Freq_Dev_Max_F_1: float: No parameter help available
- Freq_Dev_Avg_F_2: float: No parameter help available
- Freq_Dev_Min_F_2: float: No parameter help available
- Freq_Dev_Max_F_2: float: No parameter help available
- Nominal_Power: float: No parameter help available
- Mod_Ratio: float: No parameter help available
- Freq_Offset: float: No parameter help available
- Init_Freq_Drift: float: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:NMODe:LENergy:LE2M:XMAXimum
value: CalculateStruct = driver.bluetooth.measurement.multiEval.modulation.
↪nmode.lowEnergy.le2M.xmaximum.calculate()
```

No command help available

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → ResultData

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:NMODe:LENergy:LE2M:XMAXimum
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.nmode.
↪lowEnergy.le2M.xmaximum.fetch()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

read() → ResultData

```
# SCPI: READ:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:NMODe:LENergy:LE2M:XMAXimum
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.nmode.
↪lowEnergy.le2M.xmaximum.read()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

6.1.1.6.3.85 Xminimum

SCPI Commands :

```
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:NMODe:LENergy:LE2M:XMINimum
CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:NMODe:LENergy:LE2M:XMINimum
READ:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:NMODe:LENergy:LE2M:XMINimum
```

class XminimumCls

Xminimum commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float or bool: No parameter help available
- Delta_F_299_P_9: float or bool: No parameter help available
- Freq_Accuracy: float or bool: No parameter help available
- Freq_Drift: float or bool: No parameter help available
- Max_Drift: float or bool: No parameter help available
- Freq_Dev_Avg_F_1: float or bool: No parameter help available
- Freq_Dev_Min_F_1: float or bool: No parameter help available
- Freq_Dev_Max_F_1: float or bool: No parameter help available
- Freq_Dev_Avg_F_2: float or bool: No parameter help available
- Freq_Dev_Min_F_2: float or bool: No parameter help available
- Freq_Dev_Max_F_2: float or bool: No parameter help available
- Nominal_Power: float or bool: No parameter help available
- Mod_Ratio: enums.ResultStatus2: No parameter help available
- Freq_Offset: float or bool: No parameter help available
- Init_Freq_Drift: float or bool: No parameter help available

class ResultData

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float: No parameter help available
- Delta_F_299_P_9: float: No parameter help available
- Freq_Accuracy: float: No parameter help available
- Freq_Drift: float: No parameter help available
- Max_Drift: float: No parameter help available

- Freq_Dev_Avg_F_1: float: No parameter help available
- Freq_Dev_Min_F_1: float: No parameter help available
- Freq_Dev_Max_F_1: float: No parameter help available
- Freq_Dev_Avg_F_2: float: No parameter help available
- Freq_Dev_Min_F_2: float: No parameter help available
- Freq_Dev_Max_F_2: float: No parameter help available
- Nominal_Power: float: No parameter help available
- Mod_Ratio: float: No parameter help available
- Freq_Offset: float: No parameter help available
- Init_Freq_Drift: float: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:NMODe:LENergy:LE2M:XMINimum
value: CalculateStruct = driver.bluetooth.measurement.multiEval.modulation.
↪nmode.lowEnergy.le2M.xminimum.calculate()
```

No command help available

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → ResultData

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:NMODe:LENergy:LE2M:XMINimum
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.nmode.
↪lowEnergy.le2M.xminimum.fetch()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

read() → ResultData

```
# SCPI: READ:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:NMODe:LENergy:LE2M:XMINimum
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.nmode.
↪lowEnergy.le2M.xminimum.read()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

6.1.1.6.3.86 YieldPy

SCPI Command :

```
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:NMODe:LENergy:LE2M:YIELd
```

class YieldPyCls

YieldPy commands group definition. 1 total commands, 0 Subgroups, 1 group commands

fetch() → List[float]

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:NMODe:LENergy:LE2M:YIELd
value: List[float] = driver.bluetooth.measurement.multiEval.modulation.nmode.
↪lowEnergy.le2M.yieldPy.fetch()
```

No command help available

Suppressed linked return values: reliability

return
pattern_yield: No help available

6.1.1.6.3.87 Lrange

class LrangeCls

Lrange commands group definition. 24 total commands, 8 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.bluetooth.measurement.multiEval.modulation.nmode.lowEnergy.lrange.clone()
```

Subgroups

6.1.1.6.3.88 Average

SCPI Commands :

```
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:NMODe:LENergy:LRANge:AVERage
CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:NMODe:LENergy:LRANge:AVERage
READ:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:NMODe:LENergy:LRANge:AVERage
```

class AverageCls

Average commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: No parameter help available

- Out_Of_Tol: float or bool: No parameter help available
- Delta_F_199_P_9: float or bool: No parameter help available
- Freq_Accuracy: float or bool: No parameter help available
- Freq_Drift: float or bool: No parameter help available
- Max_Drift: float or bool: No parameter help available
- Freq_Dev_Avg_F_1: float or bool: No parameter help available
- Freq_Dev_Min_F_1: float or bool: No parameter help available
- Freq_Dev_Max_F_1: float or bool: No parameter help available
- Nominal_Power: float or bool: No parameter help available
- Freq_Offset: float or bool: No parameter help available

class ResultData

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float: No parameter help available
- Delta_F_199_P_9: float: No parameter help available
- Freq_Accuracy: float: No parameter help available
- Freq_Drift: float: No parameter help available
- Max_Drift: float: No parameter help available
- Freq_Dev_Avg_F_1: float: No parameter help available
- Freq_Dev_Min_F_1: float: No parameter help available
- Freq_Dev_Max_F_1: float: No parameter help available
- Nominal_Power: float: No parameter help available
- Freq_Offset: float: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:NMODe:LENergy:LRANge:AVERage
value: CalculateStruct = driver.bluetooth.measurement.multiEval.modulation.
↪nmode.lowEnergy.lrange.average.calculate()
```

No command help available

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → ResultData

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:NMODe:LENergy:LRANge:AVERage
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.nmode.
↪lowEnergy.lrange.average.fetch()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

read() → ResultData

```
# SCPI: READ:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:NMODe:LENergy:LRANge:AVERage
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.nmode.
↪lowEnergy.lrange.average.read()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

6.1.1.6.3.89 Current

SCPI Commands :

```
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:NMODe:LENergy:LRANge:CURRent
CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:NMODe:LENergy:LRANge:CURRent
READ:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:NMODe:LENergy:LRANge:CURRent
```

class CurrentCls

Current commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float or bool: No parameter help available
- Delta_F_199_P_9: float or bool: No parameter help available
- Freq_Accuracy: float or bool: No parameter help available
- Freq_Drift: float or bool: No parameter help available
- Max_Drift: float or bool: No parameter help available
- Freq_Dev_Avg_F_1: float or bool: No parameter help available
- Freq_Dev_Min_F_1: float or bool: No parameter help available
- Freq_Dev_Max_F_1: float or bool: No parameter help available
- Nominal_Power: float or bool: No parameter help available
- Freq_Offset: float or bool: No parameter help available

class ResultData

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float: No parameter help available
- Delta_F_199_P_9: float: No parameter help available

- Freq_Accuracy: float: No parameter help available
- Freq_Drift: float: No parameter help available
- Max_Drift: float: No parameter help available
- Freq_Dev_Avg_F_1: float: No parameter help available
- Freq_Dev_Min_F_1: float: No parameter help available
- Freq_Dev_Max_F_1: float: No parameter help available
- Nominal_Power: float: No parameter help available
- Freq_Offset: float: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:NMODe:LENergy:LRANge:CURRent
value: CalculateStruct = driver.bluetooth.measurement.multiEval.modulation.
↪nmode.lowEnergy.lrange.current.calculate()
```

No command help available

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → ResultData

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:NMODe:LENergy:LRANge:CURRent
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.nmode.
↪lowEnergy.lrange.current.fetch()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

read() → ResultData

```
# SCPI: READ:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:NMODe:LENergy:LRANge:CURRent
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.nmode.
↪lowEnergy.lrange.current.read()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

6.1.1.6.3.90 Maximum

SCPI Commands :

```
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:NMODe:LENergy:LRANge:MAXimum
CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:NMODe:LENergy:LRANge:MAXimum
READ:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:NMODe:LENergy:LRANge:MAXimum
```

class MaximumCls

Maximum commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float or bool: No parameter help available
- Delta_F_199_P_9: float or bool: No parameter help available
- Freq_Accuracy: float or bool: No parameter help available
- Freq_Drift: float or bool: No parameter help available
- Max_Drift: float or bool: No parameter help available
- Freq_Dev_Avg_F_1: float or bool: No parameter help available
- Freq_Dev_Min_F_1: float or bool: No parameter help available
- Freq_Dev_Max_F_1: float or bool: No parameter help available
- Nominal_Power: float or bool: No parameter help available
- Freq_Offset: float or bool: No parameter help available

class ResultData

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float: No parameter help available
- Delta_F_199_P_9: float: No parameter help available
- Freq_Accuracy: float: No parameter help available
- Freq_Drift: float: No parameter help available
- Max_Drift: float: No parameter help available
- Freq_Dev_Avg_F_1: float: No parameter help available
- Freq_Dev_Min_F_1: float: No parameter help available
- Freq_Dev_Max_F_1: float: No parameter help available
- Nominal_Power: float: No parameter help available
- Freq_Offset: float: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:NMODe:LENergy:LRANge:MAXimum
value: CalculateStruct = driver.bluetooth.measurement.multiEval.modulation.
↪nmode.lowEnergy.lrange.maximum.calculate()
```

No command help available

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → ResultData

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:NMODe:LENergy:LRANge:MAXimum
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.nmode.
↪lowEnergy.lrange.maximum.fetch()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

read() → ResultData

```
# SCPI: READ:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:NMODe:LENergy:LRANge:MAXimum
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.nmode.
↪lowEnergy.lrange.maximum.read()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

6.1.1.6.3.91 Minimum

SCPI Commands :

```
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:NMODe:LENergy:LRANge:MINimum
CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:NMODe:LENergy:LRANge:MINimum
READ:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:NMODe:LENergy:LRANge:MINimum
```

class MinimumCls

Minimum commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float or bool: No parameter help available
- Delta_F_199_P_9: float or bool: No parameter help available

- Freq_Dev_Avg_F_1: float or bool: No parameter help available
- Freq_Dev_Min_F_1: float or bool: No parameter help available
- Freq_Dev_Max_F_1: float or bool: No parameter help available
- Nominal_Power: float or bool: No parameter help available

class ResultData

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float: No parameter help available
- Delta_F_199_P_9: float: No parameter help available
- Freq_Dev_Avg_F_1: float: No parameter help available
- Freq_Dev_Min_F_1: float: No parameter help available
- Freq_Dev_Max_F_1: float: No parameter help available
- Nominal_Power: float: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:NMODe:LENergy:LRANge:MINimum
value: CalculateStruct = driver.bluetooth.measurement.multiEval.modulation.
↪nmode.lowEnergy.lrange.minimum.calculate()
```

No command help available

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → ResultData

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:NMODe:LENergy:LRANge:MINimum
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.nmode.
↪lowEnergy.lrange.minimum.fetch()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

read() → ResultData

```
# SCPI: READ:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:NMODe:LENergy:LRANge:MINimum
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.nmode.
↪lowEnergy.lrange.minimum.read()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

6.1.1.6.3.92 StandardDev

SCPI Commands :

```
FETCh:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:NMODe:LENergy:LRANge:SDEViation
CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:NMODe:LENergy:LRANge:SDEViation
READ:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:NMODe:LENergy:LRANge:SDEViation
```

class StandardDevCls

StandardDev commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float or bool: No parameter help available
- Delta_F_199_P_9: float or bool: No parameter help available
- Freq_Accuracy: float or bool: No parameter help available
- Freq_Drift: float or bool: No parameter help available
- Max_Drift: float or bool: No parameter help available
- Nominal_Power: float or bool: No parameter help available
- Freq_Offset: float or bool: No parameter help available

class ResultData

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float: No parameter help available
- Delta_F_199_P_9: float: No parameter help available
- Freq_Accuracy: float: No parameter help available
- Freq_Drift: float: No parameter help available
- Max_Drift: float: No parameter help available
- Nominal_Power: float: No parameter help available
- Freq_Offset: float: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:NMODe:LENergy:LRANge:SDEViation
value: CalculateStruct = driver.bluetooth.measurement.multiEval.modulation.
↪nmode.lowEnergy.lrange.standardDev.calculate()
```

No command help available

return

structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → ResultData

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:NMODe:LENergy:LRANge:SDEViation
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.nmode.
↪lowEnergy.lrange.standardDev.fetch()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

read() → ResultData

```
# SCPI: READ:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:NMODe:LENergy:LRANge:SDEViation
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.nmode.
↪lowEnergy.lrange.standardDev.read()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

6.1.1.6.3.93 StDev

SCPI Commands :

```
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:NMODe:LENergy:LRANge:STDev
CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:NMODe:LENergy:LRANge:STDev
READ:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:NMODe:LENergy:LRANge:STDev
```

class StDevCls

StDev commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float or bool: No parameter help available
- Delta_F_199_P_9: float or bool: No parameter help available
- Freq_Accuracy: float or bool: No parameter help available
- Freq_Drift: float or bool: No parameter help available
- Max_Drift: float or bool: No parameter help available
- Nominal_Power: float or bool: No parameter help available
- Freq_Offset: float or bool: No parameter help available

class ResultData

Response structure. Fields:

- Reliability: int: No parameter help available

- Out_Of_Tol: float: No parameter help available
- Delta_F_199_P_9: float: No parameter help available
- Freq_Accuracy: float: No parameter help available
- Freq_Drift: float: No parameter help available
- Max_Drift: float: No parameter help available
- Nominal_Power: float: No parameter help available
- Freq_Offset: float: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:NMODe:LENergy:LRANge:STDev
value: CalculateStruct = driver.bluetooth.measurement.multiEval.modulation.
↪nmode.lowEnergy.lrange.stDev.calculate()
```

No command help available

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → ResultData

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:NMODe:LENergy:LRANge:STDev
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.nmode.
↪lowEnergy.lrange.stDev.fetch()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

read() → ResultData

```
# SCPI: READ:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:NMODe:LENergy:LRANge:STDev
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.nmode.
↪lowEnergy.lrange.stDev.read()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

6.1.1.6.3.94 Xmaximum

SCPI Commands :

```
FETCh:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:NMODe:LENergy:LRANge:XMAXimum
CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:NMODe:LENergy:LRANge:XMAXimum
READ:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:NMODe:LENergy:LRANge:XMAXimum
```

class XmaximumCls

Xmaximum commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float or bool: No parameter help available
- Delta_F_199_P_9: float or bool: No parameter help available
- Freq_Accuracy: float or bool: No parameter help available
- Freq_Drift: float or bool: No parameter help available
- Max_Drift: float or bool: No parameter help available
- Freq_Dev_Avg_F_1: float or bool: No parameter help available
- Freq_Dev_Min_F_1: float or bool: No parameter help available
- Freq_Dev_Max_F_1: float or bool: No parameter help available
- Nominal_Power: float or bool: No parameter help available
- Freq_Offset: float or bool: No parameter help available

class ResultData

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float: No parameter help available
- Delta_F_199_P_9: float: No parameter help available
- Freq_Accuracy: float: No parameter help available
- Freq_Drift: float: No parameter help available
- Max_Drift: float: No parameter help available
- Freq_Dev_Avg_F_1: float: No parameter help available
- Freq_Dev_Min_F_1: float: No parameter help available
- Freq_Dev_Max_F_1: float: No parameter help available
- Nominal_Power: float: No parameter help available
- Freq_Offset: float: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:NMODe:LENergy:LRANge:XMAXimum
value: CalculateStruct = driver.bluetooth.measurement.multiEval.modulation.
↪nmode.lowEnergy.lrange.xmaximum.calculate()
```

No command help available

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → ResultData

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:NMODe:LENergy:LRANge:XMAXimum
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.nmode.
↪lowEnergy.lrange.xmaximum.fetch()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

read() → ResultData

```
# SCPI: READ:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:NMODe:LENergy:LRANge:XMAXimum
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.nmode.
↪lowEnergy.lrange.xmaximum.read()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

6.1.1.6.3.95 Xminimum

SCPI Commands :

```
FETCh:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:NMODe:LENergy:LRANge:XMINimum
CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:NMODe:LENergy:LRANge:XMINimum
READ:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:NMODe:LENergy:LRANge:XMINimum
```

class XminimumCls

Xminimum commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float or bool: No parameter help available
- Delta_F_199_P_9: float or bool: No parameter help available

- Freq_Accuracy: float or bool: No parameter help available
- Freq_Drift: float or bool: No parameter help available
- Max_Drift: float or bool: No parameter help available
- Freq_Dev_Avg_F_1: float or bool: No parameter help available
- Freq_Dev_Min_F_1: float or bool: No parameter help available
- Freq_Dev_Max_F_1: float or bool: No parameter help available
- Nominal_Power: float or bool: No parameter help available
- Freq_Offset: float or bool: No parameter help available

class ResultData

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float: No parameter help available
- Delta_F_199_P_9: float: No parameter help available
- Freq_Accuracy: float: No parameter help available
- Freq_Drift: float: No parameter help available
- Max_Drift: float: No parameter help available
- Freq_Dev_Avg_F_1: float: No parameter help available
- Freq_Dev_Min_F_1: float: No parameter help available
- Freq_Dev_Max_F_1: float: No parameter help available
- Nominal_Power: float: No parameter help available
- Freq_Offset: float: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:NMODe:LENergy:LRANge:XMINimum
value: CalculateStruct = driver.bluetooth.measurement.multiEval.modulation.
↪nmode.lowEnergy.lrange.xminimum.calculate()
```

No command help available

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → ResultData

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:NMODe:LENergy:LRANge:XMINimum
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.nmode.
↪lowEnergy.lrange.xminimum.fetch()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

read() → ResultData

```
# SCPI: READ:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:NMODe:LENergy:LRANge:XMINimum
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.nmode.
↪lowEnergy.lrange.xminimum.read()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

6.1.1.6.3.96 Qhsl

class QhslCls

Qhsl commands group definition. 60 total commands, 5 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.bluetooth.measurement.multiEval.modulation.qhsl.clone()
```

Subgroups

6.1.1.6.3.97 P2Q

class P2QCls

P2Q commands group definition. 12 total commands, 4 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.bluetooth.measurement.multiEval.modulation.qhsl.p2Q.clone()
```

Subgroups

6.1.1.6.3.98 Average

SCPI Commands :

```
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:QHSL:P2Q:AVERage
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:QHSL:P2Q:AVERage
READ:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:QHSL:P2Q:AVERage
```

class AverageCls

Average commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float or bool: No parameter help available
- Omega_I: float or bool: No parameter help available
- Omega_Iplus_Omega_0_Max: float or bool: No parameter help available
- Omega_0_Max: float or bool: No parameter help available
- Rms_Devm: float or bool: No parameter help available
- Peak_Devm: float or bool: No parameter help available
- P_99_Devm: float or bool: No parameter help available
- Nominal_Power: float or bool: No parameter help available

class ResultData

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float: No parameter help available
- Omega_I: float: No parameter help available
- Omega_Iplus_Omega_0_Max: float: No parameter help available
- Omega_0_Max: float: No parameter help available
- Rms_Devm: float: No parameter help available
- Peak_Devm: float: No parameter help available
- P_99_Devm: float: No parameter help available
- Nominal_Power: float: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:QHSL:P2Q:AVERage
value: CalculateStruct = driver.bluetooth.measurement.multiEval.modulation.qhsl.
↪p2Q.average.calculate()
```

No command help available

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → ResultData

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:QHSL:P2Q:AVERage
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.qhsl.p2Q.
↪average.fetch()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

read() → ResultData

```
# SCPI: READ:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:QHSL:P2Q:AVERage
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.qhsl.p2Q.
↪average.read()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

6.1.1.6.3.99 Current

SCPI Commands :

```
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:QHSL:P2Q:CURRent
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:QHSL:P2Q:CURRent
READ:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:QHSL:P2Q:CURRent
```

class CurrentCls

Current commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float or bool: No parameter help available
- Omega_I: float or bool: No parameter help available
- Omega_Iplus_Omega_0_Max: float or bool: No parameter help available
- Omega_0_Max: float or bool: No parameter help available
- Rms_Devm: float or bool: No parameter help available
- Peak_Devm: float or bool: No parameter help available
- P_99_Devm: float or bool: No parameter help available
- Nominal_Power: float or bool: No parameter help available

class ResultData

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float: No parameter help available
- Omega_I: float: No parameter help available
- Omega_Iplus_Omega_0_Max: float: No parameter help available
- Omega_0_Max: float: No parameter help available
- Rms_Devm: float: No parameter help available
- Peak_Devm: float: No parameter help available
- P_99_Devm: float: No parameter help available

- Nominal_Power: float: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:QHSL:P2Q:CURRent
value: CalculateStruct = driver.bluetooth.measurement.multiEval.modulation.qhsl.
↪p2Q.current.calculate()
```

No command help available

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → ResultData

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:QHSL:P2Q:CURRent
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.qhsl.p2Q.
↪current.fetch()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

read() → ResultData

```
# SCPI: READ:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:QHSL:P2Q:CURRent
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.qhsl.p2Q.
↪current.read()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

6.1.1.6.3.100 Maximum

SCPI Commands :

```
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:QHSL:P2Q:MAXimum
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:QHSL:P2Q:MAXimum
READ:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:QHSL:P2Q:MAXimum
```

class MaximumCls

Maximum commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float or bool: No parameter help available
- Omega_I: float or bool: No parameter help available

- Omega_Iplus_Omega_0_Max: float or bool: No parameter help available
- Omega_0_Max: float or bool: No parameter help available
- Rms_Devm: float or bool: No parameter help available
- Peak_Devm: float or bool: No parameter help available
- P_99_Devm: float or bool: No parameter help available
- Nominal_Power: float or bool: No parameter help available

class ResultData

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float: No parameter help available
- Omega_I: float: No parameter help available
- Omega_Iplus_Omega_0_Max: float: No parameter help available
- Omega_0_Max: float: No parameter help available
- Rms_Devm: float: No parameter help available
- Peak_Devm: float: No parameter help available
- P_99_Devm: float: No parameter help available
- Nominal_Power: float: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:QHSL:P2Q:MAXimum
value: CalculateStruct = driver.bluetooth.measurement.multiEval.modulation.qhsl.
↪p2Q.maximum.calculate()
```

No command help available

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → ResultData

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:QHSL:P2Q:MAXimum
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.qhsl.p2Q.
↪maximum.fetch()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

read() → ResultData

```
# SCPI: READ:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:QHSL:P2Q:MAXimum
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.qhsl.p2Q.
↪maximum.read()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

6.1.1.6.3.101 StandardDev

SCPI Commands :

```
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:QHSL:P2Q:SDEViation
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:QHSL:P2Q:SDEViation
READ:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:QHSL:P2Q:SDEViation
```

`class StandardDevCls`

StandardDev commands group definition. 3 total commands, 0 Subgroups, 3 group commands

`class CalculateStruct`

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float or bool: No parameter help available
- Omega_I: float or bool: No parameter help available
- Omega_Iplus_Omega_0_Max: float or bool: No parameter help available
- Omega_0_Max: float or bool: No parameter help available
- Rms_Devm: float or bool: No parameter help available
- Peak_Devm: float or bool: No parameter help available
- P_99_Devm: float or bool: No parameter help available
- Nominal_Power: float or bool: No parameter help available

`class ResultData`

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float: No parameter help available
- Omega_I: float: No parameter help available
- Omega_Iplus_Omega_0_Max: float: No parameter help available
- Omega_0_Max: float: No parameter help available
- Rms_Devm: float: No parameter help available
- Peak_Devm: float: No parameter help available
- P_99_Devm: float: No parameter help available
- Nominal_Power: float: No parameter help available

`calculate()` → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:QHSL:P2Q:SDEViation
value: CalculateStruct = driver.bluetooth.measurement.multiEval.modulation.qhsl.
↪p2Q.standardDev.calculate()
```

No command help available

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → ResultData

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:QHSL:P2Q:SDEViation
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.qhsl.p2Q.
↪standardDev.fetch()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

read() → ResultData

```
# SCPI: READ:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:QHSL:P2Q:SDEViation
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.qhsl.p2Q.
↪standardDev.read()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

6.1.1.6.3.102 P3Q

class P3QCls

P3Q commands group definition. 12 total commands, 4 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.bluetooth.measurement.multiEval.modulation.qhsl.p3Q.clone()
```

Subgroups

6.1.1.6.3.103 Average

SCPI Commands :

```
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:QHSL:P3Q:AVERage
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:QHSL:P3Q:AVERage
READ:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:QHSL:P3Q:AVERage
```

class AverageCls

Average commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float or bool: No parameter help available
- Omega_I: float or bool: No parameter help available
- Omega_Iplus_Omega_0_Max: float or bool: No parameter help available
- Omega_0_Max: float or bool: No parameter help available
- Rms_Devm: float or bool: No parameter help available
- Peak_Devm: float or bool: No parameter help available
- P_99_Devm: float or bool: No parameter help available
- Nominal_Power: float or bool: No parameter help available

class ResultData

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float: No parameter help available
- Omega_I: float: No parameter help available
- Omega_Iplus_Omega_0_Max: float: No parameter help available
- Omega_0_Max: float: No parameter help available
- Rms_Devm: float: No parameter help available
- Peak_Devm: float: No parameter help available
- P_99_Devm: float: No parameter help available
- Nominal_Power: float: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:QHSL:P3Q:AVERage
value: CalculateStruct = driver.bluetooth.measurement.multiEval.modulation.qhsl.
↪p3Q.average.calculate()
```

No command help available

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → ResultData

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:QHSL:P3Q:AVERage
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.qhsl.p3Q.
↪average.fetch()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

read() → ResultData

```
# SCPI: READ:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:QHSL:P3Q:AVERage
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.qhsl.p3Q.
↪average.read()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

6.1.1.6.3.104 Current

SCPI Commands :

```
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:QHSL:P3Q:CURRent
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:QHSL:P3Q:CURRent
READ:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:QHSL:P3Q:CURRent
```

class CurrentCls

Current commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float or bool: No parameter help available
- Omega_I: float or bool: No parameter help available
- Omega_Iplus_Omega_0_Max: float or bool: No parameter help available
- Omega_0_Max: float or bool: No parameter help available
- Rms_Devm: float or bool: No parameter help available
- Peak_Devm: float or bool: No parameter help available
- P_99_Devm: float or bool: No parameter help available
- Nominal_Power: float or bool: No parameter help available

class ResultData

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float: No parameter help available
- Omega_I: float: No parameter help available
- Omega_Iplus_Omega_0_Max: float: No parameter help available
- Omega_0_Max: float: No parameter help available
- Rms_Devm: float: No parameter help available
- Peak_Devm: float: No parameter help available
- P_99_Devm: float: No parameter help available
- Nominal_Power: float: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:QHSL:P3Q:CURRent
value: CalculateStruct = driver.bluetooth.measurement.multiEval.modulation.qhsl.
↪p3Q.current.calculate()
```

No command help available

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → ResultData

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:QHSL:P3Q:CURRent
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.qhsl.p3Q.
↪current.fetch()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

read() → ResultData

```
# SCPI: READ:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:QHSL:P3Q:CURRent
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.qhsl.p3Q.
↪current.read()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

6.1.1.6.3.105 Maximum

SCPI Commands :

```
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:QHSL:P3Q:MAXimum
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:QHSL:P3Q:MAXimum
READ:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:QHSL:P3Q:MAXimum
```

class MaximumCls

Maximum commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float or bool: No parameter help available
- Omega_I: float or bool: No parameter help available
- Omega_Iplus_Omega_0_Max: float or bool: No parameter help available
- Omega_0_Max: float or bool: No parameter help available
- Rms_Devm: float or bool: No parameter help available
- Peak_Devm: float or bool: No parameter help available
- P_99_Devm: float or bool: No parameter help available
- Nominal_Power: float or bool: No parameter help available

class ResultData

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float: No parameter help available
- Omega_I: float: No parameter help available
- Omega_Iplus_Omega_0_Max: float: No parameter help available
- Omega_0_Max: float: No parameter help available
- Rms_Devm: float: No parameter help available
- Peak_Devm: float: No parameter help available
- P_99_Devm: float: No parameter help available
- Nominal_Power: float: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:QHSL:P3Q:MAXimum
value: CalculateStruct = driver.bluetooth.measurement.multiEval.modulation.qhsl.
↪p3Q.maximum.calculate()
```

No command help available

return

structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → ResultData

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:QHSL:P3Q:MAXimum
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.qhsl.p3Q.
↪maximum.fetch()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

read() → ResultData

```
# SCPI: READ:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:QHSL:P3Q:MAXimum
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.qhsl.p3Q.
↪maximum.read()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

6.1.1.6.3.106 StandardDev

SCPI Commands :

```
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:QHSL:P3Q:SDEViation
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:QHSL:P3Q:SDEViation
READ:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:QHSL:P3Q:SDEViation
```

class StandardDevCls

StandardDev commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float or bool: No parameter help available
- Omega_I: float or bool: No parameter help available
- Omega_Iplus_Omega_0_Max: float or bool: No parameter help available
- Omega_0_Max: float or bool: No parameter help available
- Rms_Devm: float or bool: No parameter help available
- Peak_Devm: float or bool: No parameter help available
- P_99_Devm: float or bool: No parameter help available
- Nominal_Power: float or bool: No parameter help available

class ResultData

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float: No parameter help available
- Omega_I: float: No parameter help available
- Omega_Iplus_Omega_0_Max: float: No parameter help available
- Omega_0_Max: float: No parameter help available
- Rms_Devm: float: No parameter help available
- Peak_Devm: float: No parameter help available
- P_99_Devm: float: No parameter help available
- Nominal_Power: float: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:QHSL:P3Q:SDEViation
value: CalculateStruct = driver.bluetooth.measurement.multiEval.modulation.qhsl.
↪p3Q.standardDev.calculate()
```

No command help available

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → ResultData

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:QHSL:P3Q:SDEViation
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.qhsl.p3Q.
↪standardDev.fetch()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

read() → ResultData

```
# SCPI: READ:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:QHSL:P3Q:SDEViation
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.qhsl.p3Q.
↪standardDev.read()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

6.1.1.6.3.107 P4Q

class P4QCls

P4Q commands group definition. 12 total commands, 4 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.bluetooth.measurement.multiEval.modulation.qhsl.p4Q.clone()
```

Subgroups

6.1.1.6.3.108 Average

SCPI Commands :

```
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:QHSL:P4Q:AVERage
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:QHSL:P4Q:AVERage
READ:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:QHSL:P4Q:AVERage
```

class AverageCls

Average commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float or bool: No parameter help available
- Omega_I: float or bool: No parameter help available
- Omega_Iplus_Omega_0_Max: float or bool: No parameter help available
- Omega_0_Max: float or bool: No parameter help available
- Rms_Devm: float or bool: No parameter help available
- Peak_Devm: float or bool: No parameter help available
- P_99_Devm: float or bool: No parameter help available
- Nominal_Power: float or bool: No parameter help available

class ResultData

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float: No parameter help available
- Omega_I: float: No parameter help available
- Omega_Iplus_Omega_0_Max: float: No parameter help available
- Omega_0_Max: float: No parameter help available
- Rms_Devm: float: No parameter help available

- Peak_Devm: float: No parameter help available
- P_99_Devm: float: No parameter help available
- Nominal_Power: float: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:QHSL:P4Q:AVERage
value: CalculateStruct = driver.bluetooth.measurement.multiEval.modulation.qhsl.
↪p4Q.average.calculate()
```

No command help available

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → ResultData

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:QHSL:P4Q:AVERage
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.qhsl.p4Q.
↪average.fetch()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

read() → ResultData

```
# SCPI: READ:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:QHSL:P4Q:AVERage
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.qhsl.p4Q.
↪average.read()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

6.1.1.6.3.109 Current

SCPI Commands :

```
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:QHSL:P4Q:CURRent
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:QHSL:P4Q:CURRent
READ:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:QHSL:P4Q:CURRent
```

class CurrentCls

Current commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: No parameter help available

- Out_Of_Tol: float or bool: No parameter help available
- Omega_I: float or bool: No parameter help available
- Omega_Iplus_Omega_0_Max: float or bool: No parameter help available
- Omega_0_Max: float or bool: No parameter help available
- Rms_Devm: float or bool: No parameter help available
- Peak_Devm: float or bool: No parameter help available
- P_99_Devm: float or bool: No parameter help available
- Nominal_Power: float or bool: No parameter help available

class ResultData

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float: No parameter help available
- Omega_I: float: No parameter help available
- Omega_Iplus_Omega_0_Max: float: No parameter help available
- Omega_0_Max: float: No parameter help available
- Rms_Devm: float: No parameter help available
- Peak_Devm: float: No parameter help available
- P_99_Devm: float: No parameter help available
- Nominal_Power: float: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:QHSL:P4Q:CURRent
value: CalculateStruct = driver.bluetooth.measurement.multiEval.modulation.qhsl.
↪p4Q.current.calculate()
```

No command help available

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → ResultData

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:QHSL:P4Q:CURRent
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.qhsl.p4Q.
↪current.fetch()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

read() → ResultData

```
# SCPI: READ:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:QHSL:P4Q:CURRent
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.qhsl.p4Q.
↪current.read()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

6.1.1.6.3.110 Maximum

SCPI Commands :

```
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:QHSL:P4Q:MAXimum
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:QHSL:P4Q:MAXimum
READ:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:QHSL:P4Q:MAXimum
```

class MaximumCls

Maximum commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float or bool: No parameter help available
- Omega_I: float or bool: No parameter help available
- Omega_Iplus_Omega_0_Max: float or bool: No parameter help available
- Omega_0_Max: float or bool: No parameter help available
- Rms_Devm: float or bool: No parameter help available
- Peak_Devm: float or bool: No parameter help available
- P_99_Devm: float or bool: No parameter help available
- Nominal_Power: float or bool: No parameter help available

class ResultData

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float: No parameter help available
- Omega_I: float: No parameter help available
- Omega_Iplus_Omega_0_Max: float: No parameter help available
- Omega_0_Max: float: No parameter help available
- Rms_Devm: float: No parameter help available
- Peak_Devm: float: No parameter help available
- P_99_Devm: float: No parameter help available
- Nominal_Power: float: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:QHSL:P4Q:MAXimum
value: CalculateStruct = driver.bluetooth.measurement.multiEval.modulation.qhsl.
↪p4Q.maximum.calculate()
```

No command help available

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → ResultData

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:QHSL:P4Q:MAXimum
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.qhsl.p4Q.
↪maximum.fetch()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

read() → ResultData

```
# SCPI: READ:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:QHSL:P4Q:MAXimum
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.qhsl.p4Q.
↪maximum.read()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

6.1.1.6.3.111 StandardDev

SCPI Commands :

```
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:QHSL:P4Q:SDEViation
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:QHSL:P4Q:SDEViation
READ:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:QHSL:P4Q:SDEViation
```

class StandardDevCls

StandardDev commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float or bool: No parameter help available
- Omega_I: float or bool: No parameter help available
- Omega_Iplus_Omega_0_Max: float or bool: No parameter help available

- Omega_0_Max: float or bool: No parameter help available
- Rms_Devm: float or bool: No parameter help available
- Peak_Devm: float or bool: No parameter help available
- P_99_Devm: float or bool: No parameter help available
- Nominal_Power: float or bool: No parameter help available

class ResultData

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float: No parameter help available
- Omega_I: float: No parameter help available
- Omega_Iplus_Omega_0_Max: float: No parameter help available
- Omega_0_Max: float: No parameter help available
- Rms_Devm: float: No parameter help available
- Peak_Devm: float: No parameter help available
- P_99_Devm: float: No parameter help available
- Nominal_Power: float: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:QHSL:P4Q:SDEViation
value: CalculateStruct = driver.bluetooth.measurement.multiEval.modulation.qhsl.
↪p4Q.standardDev.calculate()
```

No command help available

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → ResultData

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:QHSL:P4Q:SDEViation
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.qhsl.p4Q.
↪standardDev.fetch()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

read() → ResultData

```
# SCPI: READ:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:QHSL:P4Q:SDEViation
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.qhsl.p4Q.
↪standardDev.read()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

6.1.1.6.3.112 P5Q

class P5QCls

P5Q commands group definition. 12 total commands, 4 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.bluetooth.measurement.multiEval.modulation.qhsl.p5Q.clone()
```

Subgroups

6.1.1.6.3.113 Average

SCPI Commands :

```
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:QHSL:P5Q:AVERage
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:QHSL:P5Q:AVERage
READ:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:QHSL:P5Q:AVERage
```

class AverageCls

Average commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float or bool: No parameter help available
- Omega_I: float or bool: No parameter help available
- Omega_Iplus_Omega_0_Max: float or bool: No parameter help available
- Omega_0_Max: float or bool: No parameter help available
- Rms_Devm: float or bool: No parameter help available
- Peak_Devm: float or bool: No parameter help available
- P_99_Devm: float or bool: No parameter help available
- Nominal_Power: float or bool: No parameter help available

class ResultData

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float: No parameter help available
- Omega_I: float: No parameter help available
- Omega_Iplus_Omega_0_Max: float: No parameter help available

- Omega_0_Max: float: No parameter help available
- Rms_Devm: float: No parameter help available
- Peak_Devm: float: No parameter help available
- P_99_Devm: float: No parameter help available
- Nominal_Power: float: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:QHSL:P5Q:AVERage
value: CalculateStruct = driver.bluetooth.measurement.multiEval.modulation.qhsl.
↪p5Q.average.calculate()
```

No command help available

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → ResultData

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:QHSL:P5Q:AVERage
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.qhsl.p5Q.
↪average.fetch()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

read() → ResultData

```
# SCPI: READ:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:QHSL:P5Q:AVERage
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.qhsl.p5Q.
↪average.read()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

6.1.1.6.3.114 Current

SCPI Commands :

```
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:QHSL:P5Q:CURRent
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:QHSL:P5Q:CURRent
READ:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:QHSL:P5Q:CURRent
```

class CurrentCls

Current commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float or bool: No parameter help available
- Omega_I: float or bool: No parameter help available
- Omega_Iplus_Omega_0_Max: float or bool: No parameter help available
- Omega_0_Max: float or bool: No parameter help available
- Rms_Devm: float or bool: No parameter help available
- Peak_Devm: float or bool: No parameter help available
- P_99_Devm: float or bool: No parameter help available
- Nominal_Power: float or bool: No parameter help available

class ResultData

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float: No parameter help available
- Omega_I: float: No parameter help available
- Omega_Iplus_Omega_0_Max: float: No parameter help available
- Omega_0_Max: float: No parameter help available
- Rms_Devm: float: No parameter help available
- Peak_Devm: float: No parameter help available
- P_99_Devm: float: No parameter help available
- Nominal_Power: float: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:QHSL:P5Q:CURRent
value: CalculateStruct = driver.bluetooth.measurement.multiEval.modulation.qhsl.
↪p5Q.current.calculate()
```

No command help available

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → ResultData

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:QHSL:P5Q:CURRent
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.qhsl.p5Q.
↪current.fetch()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

read() → ResultData

```
# SCPI: READ:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:QHSL:P5Q:CURRent
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.qhsl.p5Q.
↪current.read()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

6.1.1.6.3.115 Maximum

SCPI Commands :

```
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:QHSL:P5Q:MAXimum
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:QHSL:P5Q:MAXimum
READ:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:QHSL:P5Q:MAXimum
```

class MaximumCls

Maximum commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float or bool: No parameter help available
- Omega_I: float or bool: No parameter help available
- Omega_Iplus_Omega_0_Max: float or bool: No parameter help available
- Omega_0_Max: float or bool: No parameter help available
- Rms_Devm: float or bool: No parameter help available
- Peak_Devm: float or bool: No parameter help available
- P_99_Devm: float or bool: No parameter help available
- Nominal_Power: float or bool: No parameter help available

class ResultData

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float: No parameter help available
- Omega_I: float: No parameter help available
- Omega_Iplus_Omega_0_Max: float: No parameter help available
- Omega_0_Max: float: No parameter help available
- Rms_Devm: float: No parameter help available
- Peak_Devm: float: No parameter help available
- P_99_Devm: float: No parameter help available

- Nominal_Power: float: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:QHSL:P5Q:MAXimum
value: CalculateStruct = driver.bluetooth.measurement.multiEval.modulation.qhsl.
↪p5Q.maximum.calculate()
```

No command help available

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → ResultData

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:QHSL:P5Q:MAXimum
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.qhsl.p5Q.
↪maximum.fetch()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

read() → ResultData

```
# SCPI: READ:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:QHSL:P5Q:MAXimum
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.qhsl.p5Q.
↪maximum.read()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

6.1.1.6.3.116 StandardDev

SCPI Commands :

```
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:QHSL:P5Q:SDEViation
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:QHSL:P5Q:SDEViation
READ:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:QHSL:P5Q:SDEViation
```

class StandardDevCls

StandardDev commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float or bool: No parameter help available
- Omega_I: float or bool: No parameter help available

- Omega_Iplus_Omega_0_Max: float or bool: No parameter help available
- Omega_0_Max: float or bool: No parameter help available
- Rms_Devm: float or bool: No parameter help available
- Peak_Devm: float or bool: No parameter help available
- P_99_Devm: float or bool: No parameter help available
- Nominal_Power: float or bool: No parameter help available

class ResultData

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float: No parameter help available
- Omega_I: float: No parameter help available
- Omega_Iplus_Omega_0_Max: float: No parameter help available
- Omega_0_Max: float: No parameter help available
- Rms_Devm: float: No parameter help available
- Peak_Devm: float: No parameter help available
- P_99_Devm: float: No parameter help available
- Nominal_Power: float: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:QHSL:P5Q:SDEViation
value: CalculateStruct = driver.bluetooth.measurement.multiEval.modulation.qhsl.
↪p5Q.standardDev.calculate()
```

No command help available

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → ResultData

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:QHSL:P5Q:SDEViation
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.qhsl.p5Q.
↪standardDev.fetch()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

read() → ResultData

```
# SCPI: READ:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:QHSL:P5Q:SDEViation
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.qhsl.p5Q.
↪standardDev.read()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

6.1.1.6.3.117 P6Q

class P6QCls

P6Q commands group definition. 12 total commands, 4 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.bluetooth.measurement.multiEval.modulation.qhsl.p6Q.clone()
```

Subgroups

6.1.1.6.3.118 Average

SCPI Commands :

```
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:QHSL:P6Q:AVERage
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:QHSL:P6Q:AVERage
READ:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:QHSL:P6Q:AVERage
```

class AverageCls

Average commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float or bool: No parameter help available
- Omega_I: float or bool: No parameter help available
- Omega_Iplus_Omega_0_Max: float or bool: No parameter help available
- Omega_0_Max: float or bool: No parameter help available
- Rms_Devm: float or bool: No parameter help available
- Peak_Devm: float or bool: No parameter help available
- P_99_Devm: float or bool: No parameter help available
- Nominal_Power: float or bool: No parameter help available

class ResultData

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float: No parameter help available
- Omega_I: float: No parameter help available

- Omega_Iplus_Omega_0_Max: float: No parameter help available
- Omega_0_Max: float: No parameter help available
- Rms_Devm: float: No parameter help available
- Peak_Devm: float: No parameter help available
- P_99_Devm: float: No parameter help available
- Nominal_Power: float: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:QHSL:P6Q:AVERage
value: CalculateStruct = driver.bluetooth.measurement.multiEval.modulation.qhsl.
↪p6Q.average.calculate()
```

No command help available

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → ResultData

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:QHSL:P6Q:AVERage
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.qhsl.p6Q.
↪average.fetch()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

read() → ResultData

```
# SCPI: READ:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:QHSL:P6Q:AVERage
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.qhsl.p6Q.
↪average.read()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

6.1.1.6.3.119 Current

SCPI Commands :

```
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:QHSL:P6Q:CURRent
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:QHSL:P6Q:CURRent
READ:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:QHSL:P6Q:CURRent
```

class CurrentCls

Current commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float or bool: No parameter help available
- Omega_I: float or bool: No parameter help available
- Omega_Iplus_Omega_0_Max: float or bool: No parameter help available
- Omega_0_Max: float or bool: No parameter help available
- Rms_Devm: float or bool: No parameter help available
- Peak_Devm: float or bool: No parameter help available
- P_99_Devm: float or bool: No parameter help available
- Nominal_Power: float or bool: No parameter help available

class ResultData

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float: No parameter help available
- Omega_I: float: No parameter help available
- Omega_Iplus_Omega_0_Max: float: No parameter help available
- Omega_0_Max: float: No parameter help available
- Rms_Devm: float: No parameter help available
- Peak_Devm: float: No parameter help available
- P_99_Devm: float: No parameter help available
- Nominal_Power: float: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:QHSL:P6Q:CURRent
value: CalculateStruct = driver.bluetooth.measurement.multiEval.modulation.qhsl.
↪p6Q.current.calculate()
```

No command help available

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → ResultData

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:QHSL:P6Q:CURRent
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.qhsl.p6Q.
↪current.fetch()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

read() → ResultData

```
# SCPI: READ:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:QHSL:P6Q:CURRent
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.qhsl.p6Q.
↪current.read()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

6.1.1.6.3.120 Maximum

SCPI Commands :

```
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:QHSL:P6Q:MAXimum
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:QHSL:P6Q:MAXimum
READ:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:QHSL:P6Q:MAXimum
```

class MaximumCls

Maximum commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float or bool: No parameter help available
- Omega_I: float or bool: No parameter help available
- Omega_Iplus_Omega_0_Max: float or bool: No parameter help available
- Omega_0_Max: float or bool: No parameter help available
- Rms_Devm: float or bool: No parameter help available
- Peak_Devm: float or bool: No parameter help available
- P_99_Devm: float or bool: No parameter help available
- Nominal_Power: float or bool: No parameter help available

class ResultData

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float: No parameter help available
- Omega_I: float: No parameter help available
- Omega_Iplus_Omega_0_Max: float: No parameter help available
- Omega_0_Max: float: No parameter help available
- Rms_Devm: float: No parameter help available

- Peak_Devm: float: No parameter help available
- P_99_Devm: float: No parameter help available
- Nominal_Power: float: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:QHSL:P6Q:MAXimum
value: CalculateStruct = driver.bluetooth.measurement.multiEval.modulation.qhsl.
↪p6Q.maximum.calculate()
```

No command help available

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → ResultData

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:QHSL:P6Q:MAXimum
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.qhsl.p6Q.
↪maximum.fetch()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

read() → ResultData

```
# SCPI: READ:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:QHSL:P6Q:MAXimum
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.qhsl.p6Q.
↪maximum.read()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

6.1.1.6.3.121 StandardDev

SCPI Commands :

```
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:QHSL:P6Q:SDEViation
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:QHSL:P6Q:SDEViation
READ:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:QHSL:P6Q:SDEViation
```

class StandardDevCls

StandardDev commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: No parameter help available

- Out_Of_Tol: float or bool: No parameter help available
- Omega_I: float or bool: No parameter help available
- Omega_Iplus_Omega_0_Max: float or bool: No parameter help available
- Omega_0_Max: float or bool: No parameter help available
- Rms_Devm: float or bool: No parameter help available
- Peak_Devm: float or bool: No parameter help available
- P_99_Devm: float or bool: No parameter help available
- Nominal_Power: float or bool: No parameter help available

class ResultData

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float: No parameter help available
- Omega_I: float: No parameter help available
- Omega_Iplus_Omega_0_Max: float: No parameter help available
- Omega_0_Max: float: No parameter help available
- Rms_Devm: float: No parameter help available
- Peak_Devm: float: No parameter help available
- P_99_Devm: float: No parameter help available
- Nominal_Power: float: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:QHSL:P6Q:SDEViation
value: CalculateStruct = driver.bluetooth.measurement.multiEval.modulation.qhsl.
↪p6Q.standardDev.calculate()
```

No command help available

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → ResultData

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:QHSL:P6Q:SDEViation
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.qhsl.p6Q.
↪standardDev.fetch()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

read() → ResultData

```
# SCPI: READ:BLUetooth:MEASurement<Instance>
↪:MEValuation:MODulation:QHSL:P6Q:SDEViation
value: ResultData = driver.bluetooth.measurement.multiEval.modulation.qhsl.p6Q.
↪standardDev.read()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

6.1.1.6.4 Pencoding

class PencodingCls

Pencoding commands group definition. 7 total commands, 2 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.bluetooth.measurement.multiEval.pencoding.clone()
```

Subgroups

6.1.1.6.4.1 Edrate

class EdrateCls

Edrate commands group definition. 4 total commands, 1 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.bluetooth.measurement.multiEval.pencoding.edrate.clone()
```

Subgroups

6.1.1.6.4.2 Current

SCPI Commands :

```
READ:BLUetooth:MEASurement<Instance>:MEValuation:PENCoding:EDRate:CURRent
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:PENCoding:EDRate:CURRent
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:PENCoding:EDRate:CURRent
```

class CurrentCls

Current commands group definition. 4 total commands, 1 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Nominal_Power: float or bool: No parameter help available
- Bit_Error_Rate: float or bool: No parameter help available
- Packets_0_Errors: float or bool: No parameter help available

class FetchStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Bit_Error_Rate: float: No parameter help available
- Packets_0_Errors: float: No parameter help available
- Nominal_Power: float: No parameter help available

class ReadStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Nominal_Power: float: No parameter help available
- Bit_Error_Rate: float: No parameter help available
- Packets_0_Errors: float: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:PENCoding:EDRate:CURRent
value: CalculateStruct = driver.bluetooth.measurement.multiEval.pencoding.
↪edrate.current.calculate()
```

No command help available

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → FetchStruct

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:MEValuation:PENCoding:EDRate:CURRent
value: FetchStruct = driver.bluetooth.measurement.multiEval.pencoding.edrate.
↪current.fetch()
```

No command help available

return
structure: for return value, see the help for FetchStruct structure arguments.

read() → ReadStruct

```
# SCPI: READ:BLUetooth:MEASurement<Instance>
↪:MEValuation:PENCoding:EDRate:CURRent
value: ReadStruct = driver.bluetooth.measurement.multiEval.pencoding.edrate.
↪current.read()
```

No command help available

return
structure: for return value, see the help for ReadStruct structure arguments.

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.bluetooth.measurement.multiEval.pencoding.edrate.current.clone()
```

Subgroups

6.1.1.6.4.3 C

SCPI Command :

```
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:PENCoding:EDRate:CURRent:C
```

class CCls

C commands group definition. 1 total commands, 0 Subgroups, 1 group commands

class FetchStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Nominal_Power: float: No parameter help available
- Bit_Error_Rate: float: No parameter help available
- Packets_0_Errors: float: No parameter help available

fetch() → FetchStruct

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:MEValuation:PENCoding:EDRate:CURRent:C
value: FetchStruct = driver.bluetooth.measurement.multiEval.pencoding.edrate.
↪current.c.fetch()
```

No command help available

return
structure: for return value, see the help for FetchStruct structure arguments.

6.1.1.6.4.4 Ssequence

class SsequenceCls

Ssequence commands group definition. 3 total commands, 1 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.bluetooth.measurement.multiEval.pencoding.ssequence.clone()
```

Subgroups

6.1.1.6.4.5 Edrate

class EdrateCls

Edrate commands group definition. 3 total commands, 1 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.bluetooth.measurement.multiEval.pencoding.ssequence.edrate.clone()
```

Subgroups

6.1.1.6.4.6 Current

SCPI Commands :

```
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:PENCoding:SSEQuence:EDRate:CURRent
READ:BLUetooth:MEASurement<Instance>:MEValuation:PENCoding:SSEQuence:EDRate:CURRent
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:PENCoding:SSEQuence:EDRate:CURRent
```

class CurrentCls

Current commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Nominal_Power: float or bool: No parameter help available
- Sync_Bit_Errors: float or bool: No parameter help available
- Trailer_Bit_Errs: float or bool: No parameter help available

class ResultData

Response structure. Fields:

- Reliability: int: No parameter help available
- Nominal_Power: float: No parameter help available
- Sync_Bit_Errors: int: No parameter help available
- Trailer_Bit_Errs: int: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:PENCoding:SSEQuence:EDRate:CURRent
value: CalculateStruct = driver.bluetooth.measurement.multiEval.pencoding.
↪ssequence.edrate.current.calculate()
```

No command help available

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → ResultData

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:MEValuation:PENCoding:SSEQuence:EDRate:CURRent
value: ResultData = driver.bluetooth.measurement.multiEval.pencoding.ssequence.
↪edrate.current.fetch()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

read() → ResultData

```
# SCPI: READ:BLUetooth:MEASurement<Instance>
↪:MEValuation:PENCoding:SSEQuence:EDRate:CURRent
value: ResultData = driver.bluetooth.measurement.multiEval.pencoding.ssequence.
↪edrate.current.read()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

6.1.1.6.5 PowerVsTime

class PowerVsTimeCls

PowerVsTime commands group definition. 168 total commands, 5 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.bluetooth.measurement.multiEval.powerVsTime.clone()
```

Subgroups

6.1.1.6.5.1 Brate

class BrateCls

Brate commands group definition. 12 total commands, 4 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.bluetooth.measurement.multiEval.powerVsTime.brate.clone()
```

Subgroups

6.1.1.6.5.2 Average

SCPI Commands :

```
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:BRATe:AVERage
READ:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:BRATe:AVERage
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:BRATe:AVERage
```

class AverageCls

Average commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: 'Reliability indicator'
- Out_Of_Tol: float or bool: Out of tolerance result, i.e. the percentage of measurement intervals of the statistic count ([CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Scount#PowerVsTime CMDLINKRESOLVED]) exceeding the specified limits.
- Nominal_Power: float or bool: Average power during the carrier-on state
- Peak_Power: float or bool: Peak power during the carrier-on state
- Leakage_Power: float or bool: Average power during the carrier-off state
- Packet_Timing: float or bool: Time between the expected and actual start of the first symbol of the Bluetooth burst
- Gfsk_Power: float or bool: Average power within the access code and header portion of the BR burst (first 126 symbols) .

class FetchStruct

Response structure. Fields:

- Reliability: int: 'Reliability indicator'
- Out_Of_Tol: float: Out of tolerance result, i.e. the percentage of measurement intervals of the statistic count ([CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Scount#PowerVsTime CMDLINKRESOLVED]) exceeding the specified limits.

- Nominal_Power: float: Average power during the carrier-on state
- Peak_Power: float: Peak power during the carrier-on state
- Leakage_Power: float: Average power during the carrier-off state
- Packet_Timing: float: Time between the expected and actual start of the first symbol of the Bluetooth burst
- Gfsk_Power: float: Average power within the access code and header portion of the BR burst (first 126 symbols) .

class ReadStruct

Response structure. Fields:

- Reliability: int: 'Reliability indicator'
- Out_Of_Tol: float or bool: Out of tolerance result, i.e. the percentage of measurement intervals of the statistic count ([CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Scount#PowerVsTime CMDLINKRESOLVED]) exceeding the specified limits.
- Nominal_Power: float: Average power during the carrier-on state
- Peak_Power: float: Peak power during the carrier-on state
- Leakage_Power: float: Average power during the carrier-off state
- Packet_Timing: float: Time between the expected and actual start of the first symbol of the Bluetooth burst
- Gfsk_Power: float: Average power within the access code and header portion of the BR burst (first 126 symbols) .

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:PVTime:BRATe:AVERage
value: CalculateStruct = driver.bluetooth.measurement.multiEval.powerVsTime.
↪brate.average.calculate()
```

Returns the power results for BR packets. The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → FetchStruct

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:BRATe:AVERage
value: FetchStruct = driver.bluetooth.measurement.multiEval.powerVsTime.brate.
↪average.fetch()
```

Returns the power results for BR packets. The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for FetchStruct structure arguments.

read() → ReadStruct

```
# SCPI: READ:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:BRATe:AVERage
value: ReadStruct = driver.bluetooth.measurement.multiEval.powerVsTime.brate.
↪average.read()
```

Returns the power results for BR packets. The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for ReadStruct structure arguments.

6.1.1.6.5.3 Current

SCPI Commands :

```
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:BRATe:CURRent
READ:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:BRATe:CURRent
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:BRATe:CURRent
```

class CurrentCls

Current commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: 'Reliability indicator'
- Out_Of_Tol: float or bool: Out of tolerance result, i.e. the percentage of measurement intervals of the statistic count ([CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Scount#PowerVsTime CMDLINKRESOLVED]) exceeding the specified limits.
- Nominal_Power: float or bool: Average power during the carrier-on state
- Peak_Power: float or bool: Peak power during the carrier-on state
- Leakage_Power: float or bool: Average power during the carrier-off state
- Packet_Timing: float or bool: Time between the expected and actual start of the first symbol of the Bluetooth burst
- Gfsk_Power: float or bool: Average power within the access code and header portion of the BR burst (first 126 symbols) .

class FetchStruct

Response structure. Fields:

- Reliability: int: 'Reliability indicator'
- Out_Of_Tol: float: Out of tolerance result, i.e. the percentage of measurement intervals of the statistic count ([CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Scount#PowerVsTime CMDLINKRESOLVED]) exceeding the specified limits.
- Nominal_Power: float: Average power during the carrier-on state
- Peak_Power: float: Peak power during the carrier-on state
- Leakage_Power: float: Average power during the carrier-off state

- Packet_Timing: float: Time between the expected and actual start of the first symbol of the Bluetooth burst
- Gfsk_Power: float: Average power within the access code and header portion of the BR burst (first 126 symbols) .

class ReadStruct

Response structure. Fields:

- Reliability: int: 'Reliability indicator'
- Out_Of_Tol: float or bool: Out of tolerance result, i.e. the percentage of measurement intervals of the statistic count ([CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Scount#PowerVsTime CMDLINKRESOLVED]) exceeding the specified limits.
- Nominal_Power: float: Average power during the carrier-on state
- Peak_Power: float: Peak power during the carrier-on state
- Leakage_Power: float: Average power during the carrier-off state
- Packet_Timing: float: Time between the expected and actual start of the first symbol of the Bluetooth burst
- Gfsk_Power: float: Average power within the access code and header portion of the BR burst (first 126 symbols) .

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:PVTime:BRATe:CURRent
value: CalculateStruct = driver.bluetooth.measurement.multiEval.powerVsTime.
↪brate.current.calculate()
```

Returns the power results for BR packets. The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → FetchStruct

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:BRATe:CURRent
value: FetchStruct = driver.bluetooth.measurement.multiEval.powerVsTime.brate.
↪current.fetch()
```

Returns the power results for BR packets. The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for FetchStruct structure arguments.

read() → ReadStruct

```
# SCPI: READ:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:BRATe:CURRent
value: ReadStruct = driver.bluetooth.measurement.multiEval.powerVsTime.brate.
↪current.read()
```

Returns the power results for BR packets. The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for ReadStruct structure arguments.

6.1.1.6.5.4 Maximum

SCPI Commands :

```
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:BRATe:MAXimum
READ:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:BRATe:MAXimum
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:BRATe:MAXimum
```

class MaximumCls

Maximum commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: 'Reliability indicator'
- Out_Of_Tol: float or bool: Out of tolerance result, i.e. the percentage of measurement intervals of the statistic count ([CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Scount#PowerVsTime CMDLINKRESOLVED]) exceeding the specified limits.
- Nominal_Power: float or bool: Average power during the carrier-on state
- Peak_Power: float or bool: Peak power during the carrier-on state
- Leakage_Power: float or bool: Average power during the carrier-off state
- Packet_Timing: float or bool: Time between the expected and actual start of the first symbol of the Bluetooth burst
- Gfsk_Power: float or bool: Average power within the access code and header portion of the BR burst (first 126 symbols) .

class FetchStruct

Response structure. Fields:

- Reliability: int: 'Reliability indicator'
- Out_Of_Tol: float: Out of tolerance result, i.e. the percentage of measurement intervals of the statistic count ([CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Scount#PowerVsTime CMDLINKRESOLVED]) exceeding the specified limits.
- Nominal_Power: float: Average power during the carrier-on state
- Peak_Power: float: Peak power during the carrier-on state
- Leakage_Power: float: Average power during the carrier-off state
- Packet_Timing: float: Time between the expected and actual start of the first symbol of the Bluetooth burst
- Gfsk_Power: float: Average power within the access code and header portion of the BR burst (first 126 symbols) .

class ReadStruct

Response structure. Fields:

- Reliability: int: 'Reliability indicator'

- Out_Of_Tol: float or bool: Out of tolerance result, i.e. the percentage of measurement intervals of the statistic count ([CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Scount#PowerVsTime CMDLINKRESOLVED]) exceeding the specified limits.
- Nominal_Power: float: Average power during the carrier-on state
- Peak_Power: float: Peak power during the carrier-on state
- Leakage_Power: float: Average power during the carrier-off state
- Packet_Timing: float: Time between the expected and actual start of the first symbol of the Bluetooth burst
- Gfsk_Power: float: Average power within the access code and header portion of the BR burst (first 126 symbols) .

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:PVTime:BRATe:MAXimum
value: CalculateStruct = driver.bluetooth.measurement.multiEval.powerVsTime.
↪brate.maximum.calculate()
```

Returns the power results for BR packets. The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → FetchStruct

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:BRATe:MAXimum
value: FetchStruct = driver.bluetooth.measurement.multiEval.powerVsTime.brate.
↪maximum.fetch()
```

Returns the power results for BR packets. The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for FetchStruct structure arguments.

read() → ReadStruct

```
# SCPI: READ:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:BRATe:MAXimum
value: ReadStruct = driver.bluetooth.measurement.multiEval.powerVsTime.brate.
↪maximum.read()
```

Returns the power results for BR packets. The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for ReadStruct structure arguments.

6.1.1.6.5.5 Minimum

SCPI Commands :

```
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:BRATe:MINimum
READ:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:BRATe:MINimum
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:BRATe:MINimum
```

class MinimumCls

Minimum commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: 'Reliability indicator'
- Out_Of_Tol: float or bool: Out of tolerance result, i.e. the percentage of measurement intervals of the statistic count ([CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Scount#PowerVsTime CMDLINKRESOLVED]) exceeding the specified limits.
- Nominal_Power: float or bool: Average power during the carrier-on state
- Peak_Power: float or bool: Peak power during the carrier-on state
- Leakage_Power: float or bool: Average power during the carrier-off state
- Packet_Timing: enums.ResultStatus2: Time between the expected and actual start of the first symbol of the Bluetooth burst
- Gfsk_Power: float or bool: Average power within the access code and header portion of the BR burst (first 126 symbols) .

class FetchStruct

Response structure. Fields:

- Reliability: int: 'Reliability indicator'
- Out_Of_Tol: float: Out of tolerance result, i.e. the percentage of measurement intervals of the statistic count ([CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Scount#PowerVsTime CMDLINKRESOLVED]) exceeding the specified limits.
- Nominal_Power: float: Average power during the carrier-on state
- Peak_Power: float: Peak power during the carrier-on state
- Leakage_Power: float: Average power during the carrier-off state
- Packet_Timing: float: Time between the expected and actual start of the first symbol of the Bluetooth burst
- Gfsk_Power: float: Average power within the access code and header portion of the BR burst (first 126 symbols) .

class ReadStruct

Response structure. Fields:

- Reliability: int: 'Reliability indicator'

- Out_Of_Tol: float or bool: Out of tolerance result, i.e. the percentage of measurement intervals of the statistic count ([CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Scount#PowerVsTime CMDLINKRESOLVED]) exceeding the specified limits.
- Nominal_Power: float: Average power during the carrier-on state
- Peak_Power: float: Peak power during the carrier-on state
- Leakage_Power: float: Average power during the carrier-off state
- Packet_Timing: float: Time between the expected and actual start of the first symbol of the Bluetooth burst
- Gfsk_Power: float: Average power within the access code and header portion of the BR burst (first 126 symbols) .

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:PVTime:BRATe:MINimum
value: CalculateStruct = driver.bluetooth.measurement.multiEval.powerVsTime.
↪brate.minimum.calculate()
```

Returns the power results for BR packets. The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → FetchStruct

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:BRATe:MINimum
value: FetchStruct = driver.bluetooth.measurement.multiEval.powerVsTime.brate.
↪minimum.fetch()
```

Returns the power results for BR packets. The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for FetchStruct structure arguments.

read() → ReadStruct

```
# SCPI: READ:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:BRATe:MINimum
value: ReadStruct = driver.bluetooth.measurement.multiEval.powerVsTime.brate.
↪minimum.read()
```

Returns the power results for BR packets. The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for ReadStruct structure arguments.

6.1.1.6.5.6 Edrate

class EdrateCls

Edrate commands group definition. 12 total commands, 4 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.bluetooth.measurement.multiEval.powerVsTime.edrate.clone()
```

Subgroups

6.1.1.6.5.7 Average

SCPI Commands :

```
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:EDRate:AVERage
READ:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:EDRate:AVERage
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:EDRate:AVERage
```

class AverageCls

Average commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: 'Reliability indicator'
- Out_Of_Tol: float or bool: Out of tolerance result, i.e. the percentage of measurement intervals of the statistic count ([CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Scount#PowerVsTime CMDLINKRESOLVED]) exceeding the specified limits.
- Nominal_Power: float or bool: Average power during the carrier-on state
- Gfsk_Power: float or bool: Average power in the GFSK portion of the burst
- Dpsk_Power: float or bool: Average power in the DPSK portion of the burst
- Dpsk_Minus_Gfsk: float or bool: Difference between DPSK and GFSK power
- Guard_Period: float or bool: Length of the guard band between the packet header and the synchronization sequence
- Packet_Timing: float or bool: Time between the expected and actual start of the first symbol of the Bluetooth burst
- Peak_Power: float or bool: Maximum power within the whole burst. The result is only available via remote command.

class FetchStruct

Response structure. Fields:

- Reliability: int: 'Reliability indicator'

- Out_Of_Tol: float: Out of tolerance result, i.e. the percentage of measurement intervals of the statistic count ([CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Scount#PowerVsTime CMDLINKRESOLVED]) exceeding the specified limits.
- Nominal_Power: float: Average power during the carrier-on state
- Gfsk_Power: float: Average power in the GFSK portion of the burst
- Dpsk_Power: float: Average power in the DPSK portion of the burst
- Dpsk_Minus_Gfsk: float: Difference between DPSK and GFSK power
- Guard_Period: float: Length of the guard band between the packet header and the synchronization sequence
- Packet_Timing: float: Time between the expected and actual start of the first symbol of the Bluetooth burst
- Peak_Power: float: Maximum power within the whole burst. The result is only available via remote command.

class ReadStruct

Response structure. Fields:

- Reliability: int: 'Reliability indicator'
- Out_Of_Tol: float or bool: Out of tolerance result, i.e. the percentage of measurement intervals of the statistic count ([CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Scount#PowerVsTime CMDLINKRESOLVED]) exceeding the specified limits.
- Nominal_Power: float: Average power during the carrier-on state
- Gfsk_Power: float: Average power in the GFSK portion of the burst
- Dpsk_Power: float: Average power in the DPSK portion of the burst
- Dpsk_Minus_Gfsk: float: Difference between DPSK and GFSK power
- Guard_Period: float: Length of the guard band between the packet header and the synchronization sequence
- Packet_Timing: float: Time between the expected and actual start of the first symbol of the Bluetooth burst
- Peak_Power: float: Maximum power within the whole burst. The result is only available via remote command.

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:PVTime:EDRate:AVERage
value: CalculateStruct = driver.bluetooth.measurement.multiEval.powerVsTime.
↪edrate.average.calculate()
```

Returns the power results for EDR packets. The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return

structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → FetchStruct

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:EDRate:AVERage
value: FetchStruct = driver.bluetooth.measurement.multiEval.powerVsTime.edrate.
↪average.fetch()
```

Returns the power results for EDR packets. The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for FetchStruct structure arguments.

read() → ReadStruct

```
# SCPI: READ:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:EDRate:AVERage
value: ReadStruct = driver.bluetooth.measurement.multiEval.powerVsTime.edrate.
↪average.read()
```

Returns the power results for EDR packets. The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for ReadStruct structure arguments.

6.1.1.6.5.8 Current

SCPI Commands :

```
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:EDRate:CURRent
READ:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:EDRate:CURRent
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:EDRate:CURRent
```

class CurrentCls

Current commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: 'Reliability indicator'
- Out_Of_Tol: float or bool: Out of tolerance result, i.e. the percentage of measurement intervals of the statistic count ([CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Scount#PowerVsTime CMDLINKRESOLVED]) exceeding the specified limits.
- Nominal_Power: float or bool: Average power during the carrier-on state
- Gfsk_Power: float or bool: Average power in the GFSK portion of the burst
- Dpsk_Power: float or bool: Average power in the DPSK portion of the burst
- Dpsk_Minus_Gfsk: float or bool: Difference between DPSK and GFSK power
- Guard_Period: float or bool: Length of the guard band between the packet header and the synchronization sequence
- Packet_Timing: float or bool: Time between the expected and actual start of the first symbol of the Bluetooth burst

- Peak_Power: float or bool: Maximum power within the whole burst. The result is only available via remote command.

class FetchStruct

Response structure. Fields:

- Reliability: int: 'Reliability indicator'
- Out_Of_Tol: float: Out of tolerance result, i.e. the percentage of measurement intervals of the statistic count ([CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Scount#PowerVsTime CMDLINKRESOLVED]) exceeding the specified limits.
- Nominal_Power: float: Average power during the carrier-on state
- Gfsk_Power: float: Average power in the GFSK portion of the burst
- Dpsk_Power: float: Average power in the DPSK portion of the burst
- Dpsk_Minus_Gfsk: float: Difference between DPSK and GFSK power
- Guard_Period: float: Length of the guard band between the packet header and the synchronization sequence
- Packet_Timing: float: Time between the expected and actual start of the first symbol of the Bluetooth burst
- Peak_Power: float: Maximum power within the whole burst. The result is only available via remote command.

class ReadStruct

Response structure. Fields:

- Reliability: int: 'Reliability indicator'
- Out_Of_Tol: float or bool: Out of tolerance result, i.e. the percentage of measurement intervals of the statistic count ([CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Scount#PowerVsTime CMDLINKRESOLVED]) exceeding the specified limits.
- Nominal_Power: float: Average power during the carrier-on state
- Gfsk_Power: float: Average power in the GFSK portion of the burst
- Dpsk_Power: float: Average power in the DPSK portion of the burst
- Dpsk_Minus_Gfsk: float: Difference between DPSK and GFSK power
- Guard_Period: float: Length of the guard band between the packet header and the synchronization sequence
- Packet_Timing: float: Time between the expected and actual start of the first symbol of the Bluetooth burst
- Peak_Power: float: Maximum power within the whole burst. The result is only available via remote command.

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:PVTime:EDRate:CURRent
value: CalculateStruct = driver.bluetooth.measurement.multiEval.powerVsTime.
↪edrate.current.calculate()
```

Returns the power results for EDR packets. The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → FetchStruct

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:EDRate:CURRent
value: FetchStruct = driver.bluetooth.measurement.multiEval.powerVsTime.edrate.
↪current.fetch()
```

Returns the power results for EDR packets. The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for FetchStruct structure arguments.

read() → ReadStruct

```
# SCPI: READ:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:EDRate:CURRent
value: ReadStruct = driver.bluetooth.measurement.multiEval.powerVsTime.edrate.
↪current.read()
```

Returns the power results for EDR packets. The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for ReadStruct structure arguments.

6.1.1.6.5.9 Maximum

SCPI Commands :

```
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:EDRate:MAXimum
READ:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:EDRate:MAXimum
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:EDRate:MAXimum
```

class MaximumCls

Maximum commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: 'Reliability indicator'
- Out_Of_Tol: float or bool: Out of tolerance result, i.e. the percentage of measurement intervals of the statistic count ([CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Scount#PowerVsTime CMDLINKRESOLVED]) exceeding the specified limits.
- Nominal_Power: float or bool: Average power during the carrier-on state
- Gfsk_Power: float or bool: Average power in the GFSK portion of the burst
- Dpsk_Power: float or bool: Average power in the DPSK portion of the burst
- Dpsk_Minus_Gfsk: float or bool: Difference between DPSK and GFSK power

- Guard_Period: float or bool: Length of the guard band between the packet header and the synchronization sequence
- Packet_Timing: enums.ResultStatus2: Time between the expected and actual start of the first symbol of the Bluetooth burst
- Peak_Power: float or bool: Maximum power within the whole burst. The result is only available via remote command.

`class FetchStruct`

Response structure. Fields:

- Reliability: int: 'Reliability indicator'
- Out_Of_Tol: float: Out of tolerance result, i.e. the percentage of measurement intervals of the statistic count ([CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Scount#PowerVsTime CMDLINKRESOLVED]) exceeding the specified limits.
- Nominal_Power: float: Average power during the carrier-on state
- Gfsk_Power: float: Average power in the GFSK portion of the burst
- Dpsk_Power: float: Average power in the DPSK portion of the burst
- Dpsk_Minus_Gfsk: float: Difference between DPSK and GFSK power
- Guard_Period: float: Length of the guard band between the packet header and the synchronization sequence
- Packet_Timing: float: Time between the expected and actual start of the first symbol of the Bluetooth burst
- Peak_Power: float: Maximum power within the whole burst. The result is only available via remote command.

`class ReadStruct`

Response structure. Fields:

- Reliability: int: 'Reliability indicator'
- Out_Of_Tol: float or bool: Out of tolerance result, i.e. the percentage of measurement intervals of the statistic count ([CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Scount#PowerVsTime CMDLINKRESOLVED]) exceeding the specified limits.
- Nominal_Power: float: Average power during the carrier-on state
- Gfsk_Power: float: Average power in the GFSK portion of the burst
- Dpsk_Power: float: Average power in the DPSK portion of the burst
- Dpsk_Minus_Gfsk: float: Difference between DPSK and GFSK power
- Guard_Period: float: Length of the guard band between the packet header and the synchronization sequence
- Packet_Timing: float: Time between the expected and actual start of the first symbol of the Bluetooth burst
- Peak_Power: float: Maximum power within the whole burst. The result is only available via remote command.

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:PVTime:EDRate:MAXimum
value: CalculateStruct = driver.bluetooth.measurement.multiEval.powerVsTime.
↪edrate.maximum.calculate()
```

Returns the power results for EDR packets. The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → FetchStruct

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:EDRate:MAXimum
value: FetchStruct = driver.bluetooth.measurement.multiEval.powerVsTime.edrate.
↪maximum.fetch()
```

Returns the power results for EDR packets. The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for FetchStruct structure arguments.

read() → ReadStruct

```
# SCPI: READ:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:EDRate:MAXimum
value: ReadStruct = driver.bluetooth.measurement.multiEval.powerVsTime.edrate.
↪maximum.read()
```

Returns the power results for EDR packets. The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for ReadStruct structure arguments.

6.1.1.6.5.10 Minimum

SCPI Commands :

```
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:EDRate:MINimum
READ:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:EDRate:MINimum
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:EDRate:MINimum
```

class MinimumCls

Minimum commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: 'Reliability indicator'
- Out_Of_Tol: float or bool: Out of tolerance result, i.e. the percentage of measurement intervals of the statistic count ([CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Scount#PowerVsTime CMDLINKRESOLVED]) exceeding the specified limits.

- Nominal_Power: float or bool: Average power during the carrier-on state
- Gfsk_Power: float or bool: Average power in the GFSK portion of the burst
- Dpsk_Power: float or bool: Average power in the DPSK portion of the burst
- Dpsk_Minus_Gfsk: float or bool: Difference between DPSK and GFSK power
- Guard_Period: float or bool: Length of the guard band between the packet header and the synchronization sequence
- Packet_Timing: enums.ResultStatus2: Time between the expected and actual start of the first symbol of the Bluetooth burst
- Peak_Power: float or bool: Maximum power within the whole burst. The result is only available via remote command.

class FetchStruct

Response structure. Fields:

- Reliability: int: 'Reliability indicator'
- Out_Of_Tol: float: Out of tolerance result, i.e. the percentage of measurement intervals of the statistic count ([CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Scount#PowerVsTime CMDLINKRESOLVED]) exceeding the specified limits.
- Nominal_Power: float: Average power during the carrier-on state
- Gfsk_Power: float: Average power in the GFSK portion of the burst
- Dpsk_Power: float: Average power in the DPSK portion of the burst
- Dpsk_Minus_Gfsk: float: Difference between DPSK and GFSK power
- Guard_Period: float: Length of the guard band between the packet header and the synchronization sequence
- Packet_Timing: float: Time between the expected and actual start of the first symbol of the Bluetooth burst
- Peak_Power: float: Maximum power within the whole burst. The result is only available via remote command.

class ReadStruct

Response structure. Fields:

- Reliability: int: 'Reliability indicator'
- Out_Of_Tol: float or bool: Out of tolerance result, i.e. the percentage of measurement intervals of the statistic count ([CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Scount#PowerVsTime CMDLINKRESOLVED]) exceeding the specified limits.
- Nominal_Power: float: Average power during the carrier-on state
- Gfsk_Power: float: Average power in the GFSK portion of the burst
- Dpsk_Power: float: Average power in the DPSK portion of the burst
- Dpsk_Minus_Gfsk: float: Difference between DPSK and GFSK power
- Guard_Period: float: Length of the guard band between the packet header and the synchronization sequence
- Packet_Timing: float: Time between the expected and actual start of the first symbol of the Bluetooth burst

- Peak_Power: float: Maximum power within the whole burst. The result is only available via remote command.

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:PVTime:EDRate:MINimum
value: CalculateStruct = driver.bluetooth.measurement.multiEval.powerVsTime.
↪edrate.minimum.calculate()
```

Returns the power results for EDR packets. The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → FetchStruct

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:EDRate:MINimum
value: FetchStruct = driver.bluetooth.measurement.multiEval.powerVsTime.edrate.
↪minimum.fetch()
```

Returns the power results for EDR packets. The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for FetchStruct structure arguments.

read() → ReadStruct

```
# SCPI: READ:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:EDRate:MINimum
value: ReadStruct = driver.bluetooth.measurement.multiEval.powerVsTime.edrate.
↪minimum.read()
```

Returns the power results for EDR packets. The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for ReadStruct structure arguments.

6.1.1.6.5.11 LowEnergy

class LowEnergyCls

LowEnergy commands group definition. 36 total commands, 3 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.bluetooth.measurement.multiEval.powerVsTime.lowEnergy.clone()
```

Subgroups

6.1.1.6.5.12 Le1M

class Le1MCls

Le1M commands group definition. 12 total commands, 4 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.bluetooth.measurement.multiEval.powerVsTime.lowEnergy.le1M.clone()
```

Subgroups

6.1.1.6.5.13 Average

SCPI Commands :

```
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:LENergy[:LE1M]:AVERage
READ:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:LENergy[:LE1M]:AVERage
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:LENergy[:LE1M]:AVERage
```

class AverageCls

Average commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: 'Reliability indicator'
- Out_Of_Tol: float or bool: Out of tolerance result, i.e. the percentage of measurement intervals of the statistic count ([CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Scount#PowerVsTime CMDLINKRESOLVED]) exceeding the specified limits, see [CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Limit.LowEnergy.Le1M.PowerVsTime#set CMDLINKRESOLVED].
- Nominal_Power: float or bool: Average power during the carrier-on state
- Peak_Power: float or bool: Peak power during the carrier-on state
- Leakage_Power: float or bool: Average power during the carrier-off state
- Peak_Min_Avg_Pow: float or bool: Peak power minus average power

class FetchStruct

Response structure. Fields:

- Reliability: int: 'Reliability indicator'
- Out_Of_Tol: float: Out of tolerance result, i.e. the percentage of measurement intervals of the statistic count ([CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Scount#PowerVsTime CMDLINKRESOLVED]) exceeding the specified limits, see [CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Limit.LowEnergy.Le1M.PowerVsTime#set CMDLINKRESOLVED].

- Nominal_Power: float: Average power during the carrier-on state
- Peak_Power: float: Peak power during the carrier-on state
- Leakage_Power: float: Average power during the carrier-off state
- Peak_Min_Avg_Pow: float: Peak power minus average power

class ReadStruct

Response structure. Fields:

- Reliability: int: 'Reliability indicator'
- Out_Of_Tol: float or bool: Out of tolerance result, i.e. the percentage of measurement intervals of the statistic count ([CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Scount#PowerVsTime CMDLINKRESOLVED]) exceeding the specified limits, see [CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Limit.LowEnergy.Le1M.PowerVsTime#set CMDLINKRESOLVED].
- Nominal_Power: float: Average power during the carrier-on state
- Peak_Power: float: Peak power during the carrier-on state
- Leakage_Power: float: Average power during the carrier-off state
- Peak_Min_Avg_Pow: float: Peak power minus average power

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:PVTime:LENergy[:LE1M]:AVERage
value: CalculateStruct = driver.bluetooth.measurement.multiEval.powerVsTime.
↪lowEnergy.le1M.average.calculate()
```

Returns the power results for LE 1M PHY (uncoded) . The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → FetchStruct

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:MEValuation:PVTime:LENergy[:LE1M]:AVERage
value: FetchStruct = driver.bluetooth.measurement.multiEval.powerVsTime.
↪lowEnergy.le1M.average.fetch()
```

Returns the power results for LE 1M PHY (uncoded) . The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for FetchStruct structure arguments.

read() → ReadStruct

```
# SCPI: READ:BLUetooth:MEASurement<Instance>
↪:MEValuation:PVTime:LENergy[:LE1M]:AVERage
```

(continues on next page)

(continued from previous page)

```
value: ReadStruct = driver.bluetooth.measurement.multiEval.powerVsTime.
↪lowEnergy.le1M.average.read()
```

Returns the power results for LE 1M PHY (uncoded) . The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for ReadStruct structure arguments.

6.1.1.6.5.14 Current

SCPI Commands :

```
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:LENergy[:LE1M]:CURRent
READ:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:LENergy[:LE1M]:CURRent
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:LENergy[:LE1M]:CURRent
```

class CurrentCls

Current commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: 'Reliability indicator'
- Out_Of_Tol: float or bool: Out of tolerance result, i.e. the percentage of measurement intervals of the statistic count ([CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Scount#PowerVsTime CMDLINKRESOLVED]) exceeding the specified limits, see [CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Limit.LowEnergy.Le1M.PowerVsTime#set CMDLINKRESOLVED].
- Nominal_Power: float or bool: Average power during the carrier-on state
- Peak_Power: float or bool: Peak power during the carrier-on state
- Leakage_Power: float or bool: Average power during the carrier-off state
- Peak_Min_Avg_Pow: float or bool: Peak power minus average power

class FetchStruct

Response structure. Fields:

- Reliability: int: 'Reliability indicator'
- Out_Of_Tol: float: Out of tolerance result, i.e. the percentage of measurement intervals of the statistic count ([CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Scount#PowerVsTime CMDLINKRESOLVED]) exceeding the specified limits, see [CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Limit.LowEnergy.Le1M.PowerVsTime#set CMDLINKRESOLVED].
- Nominal_Power: float: Average power during the carrier-on state
- Peak_Power: float: Peak power during the carrier-on state
- Leakage_Power: float: Average power during the carrier-off state

- Peak_Min_Avg_Pow: float: Peak power minus average power

class ReadStruct

Response structure. Fields:

- Reliability: int: 'Reliability indicator'
- Out_Of_Tol: float or bool: Out of tolerance result, i.e. the percentage of measurement intervals of the statistic count ([CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Scount#PowerVsTime CMDLINKRESOLVED]) exceeding the specified limits, see [CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Limit.LowEnergy.Le1M.PowerVsTime#set CMDLINKRESOLVED].
- Nominal_Power: float: Average power during the carrier-on state
- Peak_Power: float: Peak power during the carrier-on state
- Leakage_Power: float: Average power during the carrier-off state
- Peak_Min_Avg_Pow: float: Peak power minus average power

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:PVTime:LENergy[:LE1M]:CURRent
value: CalculateStruct = driver.bluetooth.measurement.multiEval.powerVsTime.
↪lowEnergy.le1M.current.calculate()
```

Returns the power results for LE 1M PHY (uncoded) . The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → FetchStruct

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:MEValuation:PVTime:LENergy[:LE1M]:CURRent
value: FetchStruct = driver.bluetooth.measurement.multiEval.powerVsTime.
↪lowEnergy.le1M.current.fetch()
```

Returns the power results for LE 1M PHY (uncoded) . The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for FetchStruct structure arguments.

read() → ReadStruct

```
# SCPI: READ:BLUetooth:MEASurement<Instance>
↪:MEValuation:PVTime:LENergy[:LE1M]:CURRent
value: ReadStruct = driver.bluetooth.measurement.multiEval.powerVsTime.
↪lowEnergy.le1M.current.read()
```

Returns the power results for LE 1M PHY (uncoded) . The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for ReadStruct structure arguments.

6.1.1.6.5.15 Maximum

SCPI Commands :

```
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:LENergy[:LE1M]:MAXimum
READ:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:LENergy[:LE1M]:MAXimum
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:LENergy[:LE1M]:MAXimum
```

class MaximumCls

Maximum commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: 'Reliability indicator'
- Out_Of_Tol: float or bool: Out of tolerance result, i.e. the percentage of measurement intervals of the statistic count ([CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Scount#PowerVsTime CMDLINKRESOLVED]) exceeding the specified limits, see [CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Limit.LowEnergy.Le1M.PowerVsTime#set CMDLINKRESOLVED].
- Nominal_Power: float or bool: Average power during the carrier-on state
- Peak_Power: float or bool: Peak power during the carrier-on state
- Leakage_Power: float or bool: Average power during the carrier-off state
- Peak_Min_Avg_Pow: float or bool: Peak power minus average power

class FetchStruct

Response structure. Fields:

- Reliability: int: 'Reliability indicator'
- Out_Of_Tol: float: Out of tolerance result, i.e. the percentage of measurement intervals of the statistic count ([CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Scount#PowerVsTime CMDLINKRESOLVED]) exceeding the specified limits, see [CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Limit.LowEnergy.Le1M.PowerVsTime#set CMDLINKRESOLVED].
- Nominal_Power: float: Average power during the carrier-on state
- Peak_Power: float: Peak power during the carrier-on state
- Leakage_Power: float: Average power during the carrier-off state
- Peak_Min_Avg_Pow: float: Peak power minus average power

class ReadStruct

Response structure. Fields:

- Reliability: int: 'Reliability indicator'

- Out_Of_Tol: float or bool: Out of tolerance result, i.e. the percentage of measurement intervals of the statistic count ([CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Scount#PowerVsTime CMDLINKRESOLVED]) exceeding the specified limits, see [CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Limit.LowEnergy.Le1M.PowerVsTime#set CMDLINKRESOLVED].
- Nominal_Power: float: Average power during the carrier-on state
- Peak_Power: float: Peak power during the carrier-on state
- Leakage_Power: float: Average power during the carrier-off state
- Peak_Min_Avg_Pow: float: Peak power minus average power

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:PVTime:LENergy[:LE1M]:MAXimum
value: CalculateStruct = driver.bluetooth.measurement.multiEval.powerVsTime.
↪lowEnergy.le1M.maximum.calculate()
```

Returns the power results for LE 1M PHY (uncoded) . The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → FetchStruct

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:MEValuation:PVTime:LENergy[:LE1M]:MAXimum
value: FetchStruct = driver.bluetooth.measurement.multiEval.powerVsTime.
↪lowEnergy.le1M.maximum.fetch()
```

Returns the power results for LE 1M PHY (uncoded) . The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for FetchStruct structure arguments.

read() → ReadStruct

```
# SCPI: READ:BLUetooth:MEASurement<Instance>
↪:MEValuation:PVTime:LENergy[:LE1M]:MAXimum
value: ReadStruct = driver.bluetooth.measurement.multiEval.powerVsTime.
↪lowEnergy.le1M.maximum.read()
```

Returns the power results for LE 1M PHY (uncoded) . The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for ReadStruct structure arguments.

6.1.1.6.5.16 Minimum

SCPI Commands :

```
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:LENergy[:LE1M]:MINimum
READ:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:LENergy[:LE1M]:MINimum
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:LENergy[:LE1M]:MINimum
```

class MinimumCls

Minimum commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: 'Reliability indicator'
- Out_Of_Tol: float or bool: Out of tolerance result, i.e. the percentage of measurement intervals of the statistic count ([CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Scount#PowerVsTime CMDLINKRESOLVED]) exceeding the specified limits, see [CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Limit.LowEnergy.Le1M.PowerVsTime#set CMDLINKRESOLVED].
- Nominal_Power: float or bool: Average power during the carrier-on state
- Peak_Power: float or bool: Peak power during the carrier-on state
- Leakage_Power: float or bool: Average power during the carrier-off state
- Peak_Min_Avg_Pow: float or bool: Peak power minus average power

class FetchStruct

Response structure. Fields:

- Reliability: int: 'Reliability indicator'
- Out_Of_Tol: float: Out of tolerance result, i.e. the percentage of measurement intervals of the statistic count ([CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Scount#PowerVsTime CMDLINKRESOLVED]) exceeding the specified limits, see [CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Limit.LowEnergy.Le1M.PowerVsTime#set CMDLINKRESOLVED].
- Nominal_Power: float: Average power during the carrier-on state
- Peak_Power: float: Peak power during the carrier-on state
- Leakage_Power: float: Average power during the carrier-off state
- Peak_Min_Avg_Pow: float: Peak power minus average power

class ReadStruct

Response structure. Fields:

- Reliability: int: 'Reliability indicator'
- Out_Of_Tol: float or bool: Out of tolerance result, i.e. the percentage of measurement intervals of the statistic count ([CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Scount#PowerVsTime CMDLINKRESOLVED]) exceeding the specified limits, see [CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Limit.LowEnergy.Le1M.PowerVsTime#set CMDLINKRESOLVED].

- Nominal_Power: float: Average power during the carrier-on state
- Peak_Power: float: Peak power during the carrier-on state
- Leakage_Power: float: Average power during the carrier-off state
- Peak_Min_Avg_Pow: float: Peak power minus average power

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:PVTime:LENergy[:LE1M]:MINimum
value: CalculateStruct = driver.bluetooth.measurement.multiEval.powerVsTime.
↪lowEnergy.le1M.minimum.calculate()
```

Returns the power results for LE 1M PHY (uncoded) . The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → FetchStruct

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:MEValuation:PVTime:LENergy[:LE1M]:MINimum
value: FetchStruct = driver.bluetooth.measurement.multiEval.powerVsTime.
↪lowEnergy.le1M.minimum.fetch()
```

Returns the power results for LE 1M PHY (uncoded) . The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for FetchStruct structure arguments.

read() → ReadStruct

```
# SCPI: READ:BLUetooth:MEASurement<Instance>
↪:MEValuation:PVTime:LENergy[:LE1M]:MINimum
value: ReadStruct = driver.bluetooth.measurement.multiEval.powerVsTime.
↪lowEnergy.le1M.minimum.read()
```

Returns the power results for LE 1M PHY (uncoded) . The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for ReadStruct structure arguments.

6.1.1.6.5.17 Le2M

class Le2MCls

Le2M commands group definition. 12 total commands, 4 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.bluetooth.measurement.multiEval.powerVsTime.lowEnergy.le2M.clone()
```

Subgroups

6.1.1.6.5.18 Average

SCPI Commands :

```
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:LENergy:LE2M:AVERage
READ:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:LENergy:LE2M:AVERage
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:LENergy:LE2M:AVERage
```

class AverageCls

Average commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: 'Reliability indicator'
- Out_Of_Tol: float or bool: Out of tolerance result, i.e. the percentage of measurement intervals of the statistic count ([CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Scount#PowerVsTime CMDLINKRESOLVED]) exceeding the specified limits, see [CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Limit.LowEnergy.Le2M.PowerVsTime#set CMDLINKRESOLVED] and [CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Limit.LowEnergy.Lrange.Power' CMDLINKRESOLVED].
- Nominal_Power: float or bool: Average power during the carrier-on state
- Peak_Power: float or bool: Peak power during the carrier-on state
- Leakage_Power: float or bool: Average power during the carrier-off state
- Peak_Min_Avg_Pow: float or bool: Peak power minus average power

class FetchStruct

Response structure. Fields:

- Reliability: int: 'Reliability indicator'
- Out_Of_Tol: float: Out of tolerance result, i.e. the percentage of measurement intervals of the statistic count ([CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Scount#PowerVsTime CMDLINKRESOLVED]) exceeding the specified limits, see [CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Limit.LowEnergy.Le2M.PowerVsTime#set CMDLINKRESOLVED] and [CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Limit.LowEnergy.Lrange.Power' CMDLINKRESOLVED].

- Nominal_Power: float: Average power during the carrier-on state
- Peak_Power: float: Peak power during the carrier-on state
- Leakage_Power: float: Average power during the carrier-off state
- Peak_Min_Avg_Pow: float: Peak power minus average power

class ReadStruct

Response structure. Fields:

- Reliability: int: 'Reliability indicator'
- Out_Of_Tol: float or bool: Out of tolerance result, i.e. the percentage of measurement intervals of the statistic count ([CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Scount#PowerVsTime CMDLINKRESOLVED]) exceeding the specified limits, see [CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Limit.LowEnergy.Le2M.PowerVsTime#set CMDLINKRESOLVED] and [CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Limit.LowEnergy.Lrange.Power' CMDLINKRESOLVED].
- Nominal_Power: float: Average power during the carrier-on state
- Peak_Power: float: Peak power during the carrier-on state
- Leakage_Power: float: Average power during the carrier-off state
- Peak_Min_Avg_Pow: float: Peak power minus average power

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:PVTime:LENergy:LE2M:AVERage
value: CalculateStruct = driver.bluetooth.measurement.multiEval.powerVsTime.
↪lowEnergy.le2M.average.calculate()
```

Returns the power results for LE 2M PHY (...:LE2M...) and LE coded PHY (...:LRANge...) . The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → FetchStruct

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:MEValuation:PVTime:LENergy:LE2M:AVERage
value: FetchStruct = driver.bluetooth.measurement.multiEval.powerVsTime.
↪lowEnergy.le2M.average.fetch()
```

Returns the power results for LE 2M PHY (...:LE2M...) and LE coded PHY (...:LRANge...) . The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for FetchStruct structure arguments.

read() → ReadStruct

```
# SCPI: READ:BLUetooth:MEASurement<Instance>
↪:MEValuation:PVTime:LENergy:LE2M:AVERage
value: ReadStruct = driver.bluetooth.measurement.multiEval.powerVsTime.
↪lowEnergy.le2M.average.read()
```

Returns the power results for LE 2M PHY (...:LE2M...) and LE coded PHY (...:LRANge...) . The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for ReadStruct structure arguments.

6.1.1.6.5.19 Current

SCPI Commands :

```
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:LENergy:LE2M:CURRent
READ:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:LENergy:LE2M:CURRent
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:LENergy:LE2M:CURRent
```

`class CurrentCls`

Current commands group definition. 3 total commands, 0 Subgroups, 3 group commands

`class CalculateStruct`

Response structure. Fields:

- Reliability: int: 'Reliability indicator'
- Out_Of_Tol: float or bool: Out of tolerance result, i.e. the percentage of measurement intervals of the statistic count ([CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Scount#PowerVsTime CMDLINKRESOLVED]) exceeding the specified limits, see [CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Limit.LowEnergy.Le2M.PowerVsTime#set CMDLINKRESOLVED] and [CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Limit.LowEnergy.Lrange.Power CMDLINKRESOLVED].
- Nominal_Power: float or bool: Average power during the carrier-on state
- Peak_Power: float or bool: Peak power during the carrier-on state
- Leakage_Power: float or bool: Average power during the carrier-off state
- Peak_Min_Avg_Pow: float or bool: Peak power minus average power

`class FetchStruct`

Response structure. Fields:

- Reliability: int: 'Reliability indicator'
- Out_Of_Tol: float: Out of tolerance result, i.e. the percentage of measurement intervals of the statistic count ([CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Scount#PowerVsTime CMDLINKRESOLVED]) exceeding the specified limits, see [CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Limit.LowEnergy.Le2M.PowerVsTime#set CMDLINKRESOLVED] and [CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Limit.LowEnergy.Lrange.Power CMDLINKRESOLVED].
- Nominal_Power: float: Average power during the carrier-on state

- Peak_Power: float: Peak power during the carrier-on state
- Leakage_Power: float: Average power during the carrier-off state
- Peak_Min_Avg_Pow: float: Peak power minus average power

class ReadStruct

Response structure. Fields:

- Reliability: int: 'Reliability indicator'
- Out_Of_Tol: float or bool: Out of tolerance result, i.e. the percentage of measurement intervals of the statistic count ([CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Scount#PowerVsTime CMDLINKRESOLVED]) exceeding the specified limits, see [CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Limit.LowEnergy.Le2M.PowerVsTime#set CMDLINKRESOLVED] and [CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Limit.LowEnergy.Lrange.Power' CMDLINKRESOLVED].
- Nominal_Power: float: Average power during the carrier-on state
- Peak_Power: float: Peak power during the carrier-on state
- Leakage_Power: float: Average power during the carrier-off state
- Peak_Min_Avg_Pow: float: Peak power minus average power

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:PVTime:LENergy:LE2M:CURRent
value: CalculateStruct = driver.bluetooth.measurement.multiEval.powerVsTime.
↪lowEnergy.le2M.current.calculate()
```

Returns the power results for LE 2M PHY (...:LE2M...) and LE coded PHY (...:LRANge...) . The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → FetchStruct

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:MEValuation:PVTime:LENergy:LE2M:CURRent
value: FetchStruct = driver.bluetooth.measurement.multiEval.powerVsTime.
↪lowEnergy.le2M.current.fetch()
```

Returns the power results for LE 2M PHY (...:LE2M...) and LE coded PHY (...:LRANge...) . The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for FetchStruct structure arguments.

read() → ReadStruct

```
# SCPI: READ:BLUetooth:MEASurement<Instance>
↪:MEValuation:PVTime:LENergy:LE2M:CURRent
```

(continues on next page)

(continued from previous page)

```
value: ReadStruct = driver.bluetooth.measurement.multiEval.powerVsTime.
↪lowEnergy.le2M.current.read()
```

Returns the power results for LE 2M PHY (...:LE2M...) and LE coded PHY (...:LRANge...) . The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for ReadStruct structure arguments.

6.1.1.6.5.20 Maximum

SCPI Commands :

```
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:LENergy:LE2M:MAXimum
READ:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:LENergy:LE2M:MAXimum
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:LENergy:LE2M:MAXimum
```

class MaximumCls

Maximum commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: 'Reliability indicator'
- Out_Of_Tol: float or bool: Out of tolerance result, i.e. the percentage of measurement intervals of the statistic count ([CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Scount#PowerVsTime CMDLINKRESOLVED]) exceeding the specified limits, see [CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Limit.LowEnergy.Le2M.PowerVsTime#set CMDLINKRESOLVED] and [CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Limit.LowEnergy.Lrange.Power CMDLINKRESOLVED].
- Nominal_Power: float or bool: Average power during the carrier-on state
- Peak_Power: float or bool: Peak power during the carrier-on state
- Leakage_Power: float or bool: Average power during the carrier-off state
- Peak_Min_Avg_Pow: float or bool: Peak power minus average power

class FetchStruct

Response structure. Fields:

- Reliability: int: 'Reliability indicator'
- Out_Of_Tol: float: Out of tolerance result, i.e. the percentage of measurement intervals of the statistic count ([CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Scount#PowerVsTime CMDLINKRESOLVED]) exceeding the specified limits, see [CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Limit.LowEnergy.Le2M.PowerVsTime#set CMDLINKRESOLVED] and [CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Limit.LowEnergy.Lrange.Power CMDLINKRESOLVED].
- Nominal_Power: float: Average power during the carrier-on state
- Peak_Power: float: Peak power during the carrier-on state

- Leakage_Power: float: Average power during the carrier-off state
- Peak_Min_Avg_Pow: float: Peak power minus average power

class ReadStruct

Response structure. Fields:

- Reliability: int: 'Reliability indicator'
- Out_Of_Tol: float or bool: Out of tolerance result, i.e. the percentage of measurement intervals of the statistic count ([CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Scount#PowerVsTime CMDLINKRESOLVED]) exceeding the specified limits, see [CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Limit.LowEnergy.Le2M.PowerVsTime#set CMDLINKRESOLVED] and [CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Limit.LowEnergy.Lrange.Power' CMDLINKRESOLVED].
- Nominal_Power: float: Average power during the carrier-on state
- Peak_Power: float: Peak power during the carrier-on state
- Leakage_Power: float: Average power during the carrier-off state
- Peak_Min_Avg_Pow: float: Peak power minus average power

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:PVTime:LENergy:LE2M:MAXimum
value: CalculateStruct = driver.bluetooth.measurement.multiEval.powerVsTime.
↪lowEnergy.le2M.maximum.calculate()
```

Returns the power results for LE 2M PHY (...:LE2M...) and LE coded PHY (...:LRANge...) . The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → FetchStruct

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:MEValuation:PVTime:LENergy:LE2M:MAXimum
value: FetchStruct = driver.bluetooth.measurement.multiEval.powerVsTime.
↪lowEnergy.le2M.maximum.fetch()
```

Returns the power results for LE 2M PHY (...:LE2M...) and LE coded PHY (...:LRANge...) . The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for FetchStruct structure arguments.

read() → ReadStruct

```
# SCPI: READ:BLUetooth:MEASurement<Instance>
↪:MEValuation:PVTime:LENergy:LE2M:MAXimum
value: ReadStruct = driver.bluetooth.measurement.multiEval.powerVsTime.
↪lowEnergy.le2M.maximum.read()
```

Returns the power results for LE 2M PHY (...:LE2M...) and LE coded PHY (...:LRANge...) . The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for ReadStruct structure arguments.

6.1.1.6.5.21 Minimum

SCPI Commands :

```
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:LENergy:LE2M:MINimum
READ:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:LENergy:LE2M:MINimum
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:LENergy:LE2M:MINimum
```

class MinimumCls

Minimum commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: 'Reliability indicator'
- Out_Of_Tol: float or bool: Out of tolerance result, i.e. the percentage of measurement intervals of the statistic count ([CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Scount#PowerVsTime CMDLINKRESOLVED]) exceeding the specified limits, see [CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Limit.LowEnergy.Le2M.PowerVsTime#set CMDLINKRESOLVED] and [CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Limit.LowEnergy.Lrange.PowerV CMDLINKRESOLVED].
- Nominal_Power: float or bool: Average power during the carrier-on state
- Peak_Power: float or bool: Peak power during the carrier-on state
- Leakage_Power: float or bool: Average power during the carrier-off state
- Peak_Min_Avg_Pow: float or bool: Peak power minus average power

class FetchStruct

Response structure. Fields:

- Reliability: int: 'Reliability indicator'
- Out_Of_Tol: float: Out of tolerance result, i.e. the percentage of measurement intervals of the statistic count ([CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Scount#PowerVsTime CMDLINKRESOLVED]) exceeding the specified limits, see [CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Limit.LowEnergy.Le2M.PowerVsTime#set CMDLINKRESOLVED] and [CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Limit.LowEnergy.Lrange.PowerV CMDLINKRESOLVED].
- Nominal_Power: float: Average power during the carrier-on state
- Peak_Power: float: Peak power during the carrier-on state
- Leakage_Power: float: Average power during the carrier-off state
- Peak_Min_Avg_Pow: float: Peak power minus average power

class ReadStruct

Response structure. Fields:

- Reliability: int: 'Reliability indicator'
- Out_Of_Tol: float or bool: Out of tolerance result, i.e. the percentage of measurement intervals of the statistic count ([CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Scount#PowerVsTime CMDLINKRESOLVED]) exceeding the specified limits, see [CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Limit.LowEnergy.Le2M.PowerVsTime#set CMDLINKRESOLVED] and [CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Limit.LowEnergy.Lrange.Power CMDLINKRESOLVED].
- Nominal_Power: float: Average power during the carrier-on state
- Peak_Power: float: Peak power during the carrier-on state
- Leakage_Power: float: Average power during the carrier-off state
- Peak_Min_Avg_Pow: float: Peak power minus average power

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:PVTime:LENergy:LE2M:MINimum
value: CalculateStruct = driver.bluetooth.measurement.multiEval.powerVsTime.
↪lowEnergy.le2M.minimum.calculate()
```

Returns the power results for LE 2M PHY (...:LE2M...) and LE coded PHY (...:LRANge...) . The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → FetchStruct

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:MEValuation:PVTime:LENergy:LE2M:MINimum
value: FetchStruct = driver.bluetooth.measurement.multiEval.powerVsTime.
↪lowEnergy.le2M.minimum.fetch()
```

Returns the power results for LE 2M PHY (...:LE2M...) and LE coded PHY (...:LRANge...) . The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for FetchStruct structure arguments.

read() → ReadStruct

```
# SCPI: READ:BLUetooth:MEASurement<Instance>
↪:MEValuation:PVTime:LENergy:LE2M:MINimum
value: ReadStruct = driver.bluetooth.measurement.multiEval.powerVsTime.
↪lowEnergy.le2M.minimum.read()
```

Returns the power results for LE 2M PHY (...:LE2M...) and LE coded PHY (...:LRANge...) . The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for ReadStruct structure arguments.

6.1.1.6.5.22 Lrange

class LrangeCls

Lrange commands group definition. 12 total commands, 4 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.bluetooth.measurement.multiEval.powerVsTime.lowEnergy.lrange.clone()
```

Subgroups

6.1.1.6.5.23 Average

SCPI Commands :

```
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:LENergy:LRANge:AVERage
READ:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:LENergy:LRANge:AVERage
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:LENergy:LRANge:AVERage
```

class AverageCls

Average commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: 'Reliability indicator'
- Out_Of_Tol: float or bool: Out of tolerance result, i.e. the percentage of measurement intervals of the statistic count ([CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Scount#PowerVsTime CMDLINKRESOLVED]) exceeding the specified limits, see [CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Limit.LowEnergy.Le2M.PowerVsTime#set CMDLINKRESOLVED] and [CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Limit.LowEnergy.Lrange.Power' CMDLINKRESOLVED].
- Nominal_Power: float or bool: Average power during the carrier-on state
- Peak_Power: float or bool: Peak power during the carrier-on state
- Leakage_Power: float or bool: Average power during the carrier-off state
- Peak_Min_Avg_Pow: float or bool: Peak power minus average power

class FetchStruct

Response structure. Fields:

- Reliability: int: 'Reliability indicator'

- Out_Of_Tol: float: Out of tolerance result, i.e. the percentage of measurement intervals of the statistic count ([CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Scount#PowerVsTime CMDLINKRESOLVED]) exceeding the specified limits, see [CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Limit.LowEnergy.Le2M.PowerVsTime#set CMDLINKRESOLVED] and [CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Limit.LowEnergy.Lrange.PowerV CMDLINKRESOLVED].
- Nominal_Power: float: Average power during the carrier-on state
- Peak_Power: float: Peak power during the carrier-on state
- Leakage_Power: float: Average power during the carrier-off state
- Peak_Min_Avg_Pow: float: Peak power minus average power

class ReadStruct

Response structure. Fields:

- Reliability: int: 'Reliability indicator'
- Out_Of_Tol: float or bool: Out of tolerance result, i.e. the percentage of measurement intervals of the statistic count ([CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Scount#PowerVsTime CMDLINKRESOLVED]) exceeding the specified limits, see [CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Limit.LowEnergy.Le2M.PowerVsTime#set CMDLINKRESOLVED] and [CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Limit.LowEnergy.Lrange.PowerV CMDLINKRESOLVED].
- Nominal_Power: float: Average power during the carrier-on state
- Peak_Power: float: Peak power during the carrier-on state
- Leakage_Power: float: Average power during the carrier-off state
- Peak_Min_Avg_Pow: float: Peak power minus average power

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:PVTime:LENergy:LRANge:AVERage
value: CalculateStruct = driver.bluetooth.measurement.multiEval.powerVsTime.
↪lowEnergy.lrange.average.calculate()
```

Returns the power results for LE 2M PHY (...:LE2M...) and LE coded PHY (...:LRANge...) . The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → FetchStruct

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:MEValuation:PVTime:LENergy:LRANge:AVERage
value: FetchStruct = driver.bluetooth.measurement.multiEval.powerVsTime.
↪lowEnergy.lrange.average.fetch()
```

Returns the power results for LE 2M PHY (...:LE2M...) and LE coded PHY (...:LRANge...) . The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for FetchStruct structure arguments.

read() → ReadStruct

```
# SCPI: READ:BLUetooth:MEASurement<Instance>
↪:MEValuation:PVTime:LENergy:LRANge:AVERage
value: ReadStruct = driver.bluetooth.measurement.multiEval.powerVsTime.
↪lowEnergy.lrange.average.read()
```

Returns the power results for LE 2M PHY (...:LE2M...) and LE coded PHY (...:LRANge...) . The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for ReadStruct structure arguments.

6.1.1.6.5.24 Current

SCPI Commands :

```
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:LENergy:LRANge:CURRent
READ:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:LENergy:LRANge:CURRent
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:LENergy:LRANge:CURRent
```

class CurrentCls

Current commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: 'Reliability indicator'
- Out_Of_Tol: float or bool: Out of tolerance result, i.e. the percentage of measurement intervals of the statistic count ([CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Scount#PowerVsTime CMDLINKRESOLVED]) exceeding the specified limits, see [CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Limit.LowEnergy.Le2M.PowerVsTime#set CMDLINKRESOLVED] and [CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Limit.LowEnergy.Lrange.Power' CMDLINKRESOLVED].
- Nominal_Power: float or bool: Average power during the carrier-on state
- Peak_Power: float or bool: Peak power during the carrier-on state
- Leakage_Power: float or bool: Average power during the carrier-off state
- Peak_Min_Avg_Pow: float or bool: Peak power minus average power

class FetchStruct

Response structure. Fields:

- Reliability: int: 'Reliability indicator'
- Out_Of_Tol: float: Out of tolerance result, i.e. the percentage of measurement intervals of the statistic count ([CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Scount#PowerVsTime

CMDLINKRESOLVED]) exceeding the specified limits, see [CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Limit.LowEnergy.Le2M.PowerVsTime#set CMDLINKRESOLVED] and [CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Limit.LowEnergy.Lrange.Power CMDLINKRESOLVED].

- Nominal_Power: float: Average power during the carrier-on state
- Peak_Power: float: Peak power during the carrier-on state
- Leakage_Power: float: Average power during the carrier-off state
- Peak_Min_Avg_Pow: float: Peak power minus average power

class ReadStruct

Response structure. Fields:

- Reliability: int: 'Reliability indicator'
- Out_Of_Tol: float or bool: Out of tolerance result, i.e. the percentage of measurement intervals of the statistic count ([CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Scount#PowerVsTime CMDLINKRESOLVED]) exceeding the specified limits, see [CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Limit.LowEnergy.Le2M.PowerVsTime#set CMDLINKRESOLVED] and [CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Limit.LowEnergy.Lrange.Power CMDLINKRESOLVED].
- Nominal_Power: float: Average power during the carrier-on state
- Peak_Power: float: Peak power during the carrier-on state
- Leakage_Power: float: Average power during the carrier-off state
- Peak_Min_Avg_Pow: float: Peak power minus average power

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:PVTime:LENergy:LRANge:CURRent
value: CalculateStruct = driver.bluetooth.measurement.multiEval.powerVsTime.
↪lowEnergy.lrange.current.calculate()
```

Returns the power results for LE 2M PHY (...:LE2M...) and LE coded PHY (...:LRANge...) . The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → FetchStruct

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:MEValuation:PVTime:LENergy:LRANge:CURRent
value: FetchStruct = driver.bluetooth.measurement.multiEval.powerVsTime.
↪lowEnergy.lrange.current.fetch()
```

Returns the power results for LE 2M PHY (...:LE2M...) and LE coded PHY (...:LRANge...) . The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for FetchStruct structure arguments.

read() → ReadStruct

```
# SCPI: READ:BLUetooth:MEASurement<Instance>
↪:MEValuation:PVTime:LENergy:LRANge:CURRent
value: ReadStruct = driver.bluetooth.measurement.multiEval.powerVsTime.
↪lowEnergy.lrange.current.read()
```

Returns the power results for LE 2M PHY (...:LE2M...) and LE coded PHY (...:LRANge...) . The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for ReadStruct structure arguments.

6.1.1.6.5.25 Maximum

SCPI Commands :

```
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:LENergy:LRANge:MAXimum
READ:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:LENergy:LRANge:MAXimum
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:LENergy:LRANge:MAXimum
```

class MaximumCls

Maximum commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: 'Reliability indicator'
- Out_Of_Tol: float or bool: Out of tolerance result, i.e. the percentage of measurement intervals of the statistic count ([CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Scount#PowerVsTime CMDLINKRESOLVED]) exceeding the specified limits, see [CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Limit.LowEnergy.Le2M.PowerVsTime#set CMDLINKRESOLVED] and [CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Limit.LowEnergy.Lrange.PowerV CMDLINKRESOLVED].
- Nominal_Power: float or bool: Average power during the carrier-on state
- Peak_Power: float or bool: Peak power during the carrier-on state
- Leakage_Power: float or bool: Average power during the carrier-off state
- Peak_Min_Avg_Pow: float or bool: Peak power minus average power

class FetchStruct

Response structure. Fields:

- Reliability: int: 'Reliability indicator'
- Out_Of_Tol: float: Out of tolerance result, i.e. the percentage of measurement intervals of the statistic count ([CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Scount#PowerVsTime CMDLINKRESOLVED]) exceeding the specified limits, see [CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Limit.LowEnergy.Le2M.PowerVsTime#set CMDLINKRESOLVED] and [CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Limit.LowEnergy.Lrange.PowerV CMDLINKRESOLVED].

- Nominal_Power: float: Average power during the carrier-on state
- Peak_Power: float: Peak power during the carrier-on state
- Leakage_Power: float: Average power during the carrier-off state
- Peak_Min_Avg_Pow: float: Peak power minus average power

class ReadStruct

Response structure. Fields:

- Reliability: int: 'Reliability indicator'
- Out_Of_Tol: float or bool: Out of tolerance result, i.e. the percentage of measurement intervals of the statistic count ([CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Scount#PowerVsTime CMDLINKRESOLVED]) exceeding the specified limits, see [CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Limit.LowEnergy.Le2M.PowerVsTime#set CMDLINKRESOLVED] and [CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Limit.LowEnergy.Lrange.Power' CMDLINKRESOLVED].
- Nominal_Power: float: Average power during the carrier-on state
- Peak_Power: float: Peak power during the carrier-on state
- Leakage_Power: float: Average power during the carrier-off state
- Peak_Min_Avg_Pow: float: Peak power minus average power

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:PVTime:LENergy:LRANge:MAXimum
value: CalculateStruct = driver.bluetooth.measurement.multiEval.powerVsTime.
↪lowEnergy.lrange.maximum.calculate()
```

Returns the power results for LE 2M PHY (...:LE2M...) and LE coded PHY (...:LRANge...) . The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → FetchStruct

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:MEValuation:PVTime:LENergy:LRANge:MAXimum
value: FetchStruct = driver.bluetooth.measurement.multiEval.powerVsTime.
↪lowEnergy.lrange.maximum.fetch()
```

Returns the power results for LE 2M PHY (...:LE2M...) and LE coded PHY (...:LRANge...) . The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for FetchStruct structure arguments.

read() → ReadStruct

```
# SCPI: READ:BLUetooth:MEASurement<Instance>
↪:MEValuation:PVTime:LENergy:LRANge:MAXimum
value: ReadStruct = driver.bluetooth.measurement.multiEval.powerVsTime.
↪lowEnergy.lrange.maximum.read()
```

Returns the power results for LE 2M PHY (...:LE2M...) and LE coded PHY (...:LRANge...) . The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for ReadStruct structure arguments.

6.1.1.6.5.26 Minimum

SCPI Commands :

```
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:LENergy:LRANge:MINimum
READ:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:LENergy:LRANge:MINimum
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:LENergy:LRANge:MINimum
```

class MinimumCls

Minimum commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: 'Reliability indicator'
- Out_Of_Tol: float or bool: Out of tolerance result, i.e. the percentage of measurement intervals of the statistic count ([CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Scount#PowerVsTime CMDLINKRESOLVED]) exceeding the specified limits, see [CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Limit.LowEnergy.Le2M.PowerVsTime#set CMDLINKRESOLVED] and [CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Limit.LowEnergy.Lrange.Power CMDLINKRESOLVED].
- Nominal_Power: float or bool: Average power during the carrier-on state
- Peak_Power: float or bool: Peak power during the carrier-on state
- Leakage_Power: float or bool: Average power during the carrier-off state
- Peak_Min_Avg_Pow: float or bool: Peak power minus average power

class FetchStruct

Response structure. Fields:

- Reliability: int: 'Reliability indicator'
- Out_Of_Tol: float: Out of tolerance result, i.e. the percentage of measurement intervals of the statistic count ([CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Scount#PowerVsTime CMDLINKRESOLVED]) exceeding the specified limits, see [CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Limit.LowEnergy.Le2M.PowerVsTime#set CMDLINKRESOLVED] and [CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Limit.LowEnergy.Lrange.Power CMDLINKRESOLVED].
- Nominal_Power: float: Average power during the carrier-on state

- Peak_Power: float: Peak power during the carrier-on state
- Leakage_Power: float: Average power during the carrier-off state
- Peak_Min_Avg_Pow: float: Peak power minus average power

class ReadStruct

Response structure. Fields:

- Reliability: int: 'Reliability indicator'
- Out_Of_Tol: float or bool: Out of tolerance result, i.e. the percentage of measurement intervals of the statistic count ([CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Scount#PowerVsTime CMDLINKRESOLVED]) exceeding the specified limits, see [CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Limit.LowEnergy.Le2M.PowerVsTime#set CMDLINKRESOLVED] and [CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Limit.LowEnergy.Lrange.Power' CMDLINKRESOLVED].
- Nominal_Power: float: Average power during the carrier-on state
- Peak_Power: float: Peak power during the carrier-on state
- Leakage_Power: float: Average power during the carrier-off state
- Peak_Min_Avg_Pow: float: Peak power minus average power

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:PVTime:LENergy:LRANge:MINimum
value: CalculateStruct = driver.bluetooth.measurement.multiEval.powerVsTime.
↪lowEnergy.lrange.minimum.calculate()
```

Returns the power results for LE 2M PHY (...:LE2M...) and LE coded PHY (...:LRANge...) . The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → FetchStruct

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:MEValuation:PVTime:LENergy:LRANge:MINimum
value: FetchStruct = driver.bluetooth.measurement.multiEval.powerVsTime.
↪lowEnergy.lrange.minimum.fetch()
```

Returns the power results for LE 2M PHY (...:LE2M...) and LE coded PHY (...:LRANge...) . The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for FetchStruct structure arguments.

read() → ReadStruct

```
# SCPI: READ:BLUetooth:MEASurement<Instance>
↪:MEValuation:PVTime:LENergy:LRANge:MINimum
```

(continues on next page)

(continued from previous page)

```
value: ReadStruct = driver.bluetooth.measurement.multiEval.powerVsTime.
↪lowEnergy.lrange.minimum.read()
```

Returns the power results for LE 2M PHY (...:LE2M...) and LE coded PHY (...:LRANge...) . The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for ReadStruct structure arguments.

6.1.1.6.5.27 Nmode

class NmodeCls

Nmode commands group definition. 48 total commands, 2 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.bluetooth.measurement.multiEval.powerVsTime.nmode.clone()
```

Subgroups

6.1.1.6.5.28 Classic

class ClassicCls

Classic commands group definition. 12 total commands, 4 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.bluetooth.measurement.multiEval.powerVsTime.nmode.classic.clone()
```

Subgroups

6.1.1.6.5.29 Average

SCPI Commands :

```
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:NMODe:CLASsic:AVERage
READ:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:NMODe:CLASsic:AVERage
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:NMODe:CLASsic:AVERage
```

class AverageCls

Average commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float or bool: No parameter help available
- Nominal_Power: float: No parameter help available

class FetchStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float: No parameter help available
- Nominal_Power: float: No parameter help available

class ReadStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float or bool: No parameter help available
- Nominal_Power: float: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:PVTime:NMODe:CLASsic:AVERage
value: CalculateStruct = driver.bluetooth.measurement.multiEval.powerVsTime.
↪nmode.classic.average.calculate()
```

No command help available

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → FetchStruct

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:MEValuation:PVTime:NMODe:CLASsic:AVERage
value: FetchStruct = driver.bluetooth.measurement.multiEval.powerVsTime.nmode.
↪classic.average.fetch()
```

No command help available

return
structure: for return value, see the help for FetchStruct structure arguments.

read() → ReadStruct

```
# SCPI: READ:BLUetooth:MEASurement<Instance>
↪:MEValuation:PVTime:NMODe:CLASsic:AVERage
value: ReadStruct = driver.bluetooth.measurement.multiEval.powerVsTime.nmode.
↪classic.average.read()
```

No command help available

return
structure: for return value, see the help for ReadStruct structure arguments.

6.1.1.6.5.30 Current

SCPI Commands :

```
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:NMODe:CLASsic:CURRent
READ:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:NMODe:CLASsic:CURRent
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:NMODe:CLASsic:CURRent
```

class CurrentCls

Current commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float or bool: No parameter help available
- Nominal_Power: float: No parameter help available

class FetchStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float: No parameter help available
- Nominal_Power: float: No parameter help available

class ReadStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float or bool: No parameter help available
- Nominal_Power: float: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:PVTime:NMODe:CLASsic:CURRent
value: CalculateStruct = driver.bluetooth.measurement.multiEval.powerVsTime.
↪nmode.classic.current.calculate()
```

No command help available

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → FetchStruct

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:MEValuation:PVTime:NMODe:CLASsic:CURRent
value: FetchStruct = driver.bluetooth.measurement.multiEval.powerVsTime.nmode.
↪classic.current.fetch()
```

No command help available

return
structure: for return value, see the help for FetchStruct structure arguments.

read() → ReadStruct

```
# SCPI: READ:BLUetooth:MEASurement<Instance>
↪:MEValuation:PVTime:NMODe:CLASsic:CURRent
value: ReadStruct = driver.bluetooth.measurement.multiEval.powerVsTime.nmode.
↪classic.current.read()
```

No command help available

return
structure: for return value, see the help for ReadStruct structure arguments.

6.1.1.6.5.31 Maximum

SCPI Commands :

```
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:NMODe:CLASsic:MAXimum
READ:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:NMODe:CLASsic:MAXimum
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:NMODe:CLASsic:MAXimum
```

class MaximumCls

Maximum commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float or bool: No parameter help available
- Nominal_Power: float: No parameter help available

class FetchStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float: No parameter help available
- Nominal_Power: float: No parameter help available

class ReadStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float or bool: No parameter help available
- Nominal_Power: float: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:PVTime:NMODe:CLASsic:MAXimum
value: CalculateStruct = driver.bluetooth.measurement.multiEval.powerVsTime.
↪nmode.classic.maximum.calculate()
```

No command help available

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → FetchStruct

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:MEValuation:PVTime:NMODe:CLASsic:MAXimum
value: FetchStruct = driver.bluetooth.measurement.multiEval.powerVsTime.nmode.
↪classic.maximum.fetch()
```

No command help available

return
structure: for return value, see the help for FetchStruct structure arguments.

read() → ReadStruct

```
# SCPI: READ:BLUetooth:MEASurement<Instance>
↪:MEValuation:PVTime:NMODe:CLASsic:MAXimum
value: ReadStruct = driver.bluetooth.measurement.multiEval.powerVsTime.nmode.
↪classic.maximum.read()
```

No command help available

return
structure: for return value, see the help for ReadStruct structure arguments.

6.1.1.6.5.32 Minimum

SCPI Commands :

```
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:NMODe:CLASsic:MINimum
READ:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:NMODe:CLASsic:MINimum
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:NMODe:CLASsic:MINimum
```

class MinimumCls

Minimum commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float or bool: No parameter help available
- Nominal_Power: float: No parameter help available

class FetchStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float: No parameter help available
- Nominal_Power: float: No parameter help available

class ReadStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float or bool: No parameter help available
- Nominal_Power: float: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:PVTime:NMODe:CLASsic:MINimum
value: CalculateStruct = driver.bluetooth.measurement.multiEval.powerVsTime.
↪nmode.classic.minimum.calculate()
```

No command help available

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → FetchStruct

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:MEValuation:PVTime:NMODe:CLASsic:MINimum
value: FetchStruct = driver.bluetooth.measurement.multiEval.powerVsTime.nmode.
↪classic.minimum.fetch()
```

No command help available

return
structure: for return value, see the help for FetchStruct structure arguments.

read() → ReadStruct

```
# SCPI: READ:BLUetooth:MEASurement<Instance>
↪:MEValuation:PVTime:NMODe:CLASsic:MINimum
value: ReadStruct = driver.bluetooth.measurement.multiEval.powerVsTime.nmode.
↪classic.minimum.read()
```

No command help available

return
structure: for return value, see the help for ReadStruct structure arguments.

6.1.1.6.5.33 LowEnergy

class LowEnergyCls

LowEnergy commands group definition. 36 total commands, 3 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.bluetooth.measurement.multiEval.powerVsTime.nmode.lowEnergy.clone()
```

Subgroups

6.1.1.6.5.34 Le1M

class Le1MCls

Le1M commands group definition. 12 total commands, 4 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.bluetooth.measurement.multiEval.powerVsTime.nmode.lowEnergy.le1M.clone()
```

Subgroups

6.1.1.6.5.35 Average

SCPI Commands :

```
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:NMODe:LENergy[:LE1M]:AVERage
READ:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:NMODe:LENergy[:LE1M]:AVERage
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:NMODe:LENergy[:LE1M]:AVERage
```

class AverageCls

Average commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float or bool: No parameter help available
- Nominal_Power: float or bool: No parameter help available
- Peak_Power: float or bool: No parameter help available
- Leakage_Power: float or bool: No parameter help available
- Peak_Min_Avg_Pow: float or bool: No parameter help available

class FetchStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float: No parameter help available
- Nominal_Power: float: No parameter help available

- Peak_Power: float: No parameter help available
- Leakage_Power: float: No parameter help available
- Peak_Min_Avg_Pow: float: No parameter help available

class ReadStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float or bool: No parameter help available
- Nominal_Power: float: No parameter help available
- Peak_Power: float: No parameter help available
- Leakage_Power: float: No parameter help available
- Peak_Min_Avg_Pow: float: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:PVTime:NMODe:LENergy[:LE1M]:AVERage
value: CalculateStruct = driver.bluetooth.measurement.multiEval.powerVsTime.
↪nmode.lowEnergy.le1M.average.calculate()
```

No command help available

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → FetchStruct

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:MEValuation:PVTime:NMODe:LENergy[:LE1M]:AVERage
value: FetchStruct = driver.bluetooth.measurement.multiEval.powerVsTime.nmode.
↪lowEnergy.le1M.average.fetch()
```

No command help available

return
structure: for return value, see the help for FetchStruct structure arguments.

read() → ReadStruct

```
# SCPI: READ:BLUetooth:MEASurement<Instance>
↪:MEValuation:PVTime:NMODe:LENergy[:LE1M]:AVERage
value: ReadStruct = driver.bluetooth.measurement.multiEval.powerVsTime.nmode.
↪lowEnergy.le1M.average.read()
```

No command help available

return
structure: for return value, see the help for ReadStruct structure arguments.

6.1.1.6.5.36 Current

SCPI Commands :

```
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:NMODe:LENergy[:LE1M]:CURRent
READ:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:NMODe:LENergy[:LE1M]:CURRent
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:NMODe:LENergy[:LE1M]:CURRent
```

class CurrentCls

Current commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float or bool: No parameter help available
- Nominal_Power: float or bool: No parameter help available
- Peak_Power: float or bool: No parameter help available
- Leakage_Power: float or bool: No parameter help available
- Peak_Min_Avg_Pow: float or bool: No parameter help available

class FetchStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float: No parameter help available
- Nominal_Power: float: No parameter help available
- Peak_Power: float: No parameter help available
- Leakage_Power: float: No parameter help available
- Peak_Min_Avg_Pow: float: No parameter help available

class ReadStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float or bool: No parameter help available
- Nominal_Power: float: No parameter help available
- Peak_Power: float: No parameter help available
- Leakage_Power: float: No parameter help available
- Peak_Min_Avg_Pow: float: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:PVTime:NMODe:LENergy[:LE1M]:CURRent
value: CalculateStruct = driver.bluetooth.measurement.multiEval.powerVsTime.
↪nmode.lowEnergy.le1M.current.calculate()
```

No command help available

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → FetchStruct

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:MEValuation:PVTime:NMODe:LENergy[:LE1M]:CURRent
value: FetchStruct = driver.bluetooth.measurement.multiEval.powerVsTime.nmode.
↪lowEnergy.le1M.current.fetch()
```

No command help available

return
structure: for return value, see the help for FetchStruct structure arguments.

read() → ReadStruct

```
# SCPI: READ:BLUetooth:MEASurement<Instance>
↪:MEValuation:PVTime:NMODe:LENergy[:LE1M]:CURRent
value: ReadStruct = driver.bluetooth.measurement.multiEval.powerVsTime.nmode.
↪lowEnergy.le1M.current.read()
```

No command help available

return
structure: for return value, see the help for ReadStruct structure arguments.

6.1.1.6.5.37 Maximum

SCPI Commands :

```
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:NMODe:LENergy[:LE1M]:MAXimum
READ:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:NMODe:LENergy[:LE1M]:MAXimum
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:NMODe:LENergy[:LE1M]:MAXimum
```

class MaximumCls

Maximum commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float or bool: No parameter help available
- Nominal_Power: float or bool: No parameter help available
- Peak_Power: float or bool: No parameter help available
- Leakage_Power: float or bool: No parameter help available
- Peak_Min_Avg_Pow: float or bool: No parameter help available

class FetchStruct

Response structure. Fields:

- Reliability: int: No parameter help available

- Out_Of_Tol: float: No parameter help available
- Nominal_Power: float: No parameter help available
- Peak_Power: float: No parameter help available
- Leakage_Power: float: No parameter help available
- Peak_Min_Avg_Pow: float: No parameter help available

class ReadStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float or bool: No parameter help available
- Nominal_Power: float: No parameter help available
- Peak_Power: float: No parameter help available
- Leakage_Power: float: No parameter help available
- Peak_Min_Avg_Pow: float: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:PVTime:NMODe:LENergy[:LE1M]:MAXimum
value: CalculateStruct = driver.bluetooth.measurement.multiEval.powerVsTime.
↪nmode.lowEnergy.le1M.maximum.calculate()
```

No command help available

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → FetchStruct

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:MEValuation:PVTime:NMODe:LENergy[:LE1M]:MAXimum
value: FetchStruct = driver.bluetooth.measurement.multiEval.powerVsTime.nmode.
↪lowEnergy.le1M.maximum.fetch()
```

No command help available

return
structure: for return value, see the help for FetchStruct structure arguments.

read() → ReadStruct

```
# SCPI: READ:BLUetooth:MEASurement<Instance>
↪:MEValuation:PVTime:NMODe:LENergy[:LE1M]:MAXimum
value: ReadStruct = driver.bluetooth.measurement.multiEval.powerVsTime.nmode.
↪lowEnergy.le1M.maximum.read()
```

No command help available

return
structure: for return value, see the help for ReadStruct structure arguments.

6.1.1.6.5.38 Minimum

SCPI Commands :

```
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:NMODe:LENergy[:LE1M]:MINimum
READ:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:NMODe:LENergy[:LE1M]:MINimum
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:NMODe:LENergy[:LE1M]:MINimum
```

class MinimumCls

Minimum commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float or bool: No parameter help available
- Nominal_Power: float or bool: No parameter help available
- Peak_Power: float or bool: No parameter help available
- Leakage_Power: float or bool: No parameter help available
- Peak_Min_Avg_Pow: float or bool: No parameter help available

class FetchStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float: No parameter help available
- Nominal_Power: float: No parameter help available
- Peak_Power: float: No parameter help available
- Leakage_Power: float: No parameter help available
- Peak_Min_Avg_Pow: float: No parameter help available

class ReadStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float or bool: No parameter help available
- Nominal_Power: float: No parameter help available
- Peak_Power: float: No parameter help available
- Leakage_Power: float: No parameter help available
- Peak_Min_Avg_Pow: float: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:PVTime:NMODe:LENergy[:LE1M]:MINimum
value: CalculateStruct = driver.bluetooth.measurement.multiEval.powerVsTime.
↪nmode.lowEnergy.le1M.minimum.calculate()
```

No command help available

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → FetchStruct

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:MEValuation:PVTime:NMODe:LENergy[:LE1M]:MINimum
value: FetchStruct = driver.bluetooth.measurement.multiEval.powerVsTime.nmode.
↪lowEnergy.le1M.minimum.fetch()
```

No command help available

return
structure: for return value, see the help for FetchStruct structure arguments.

read() → ReadStruct

```
# SCPI: READ:BLUetooth:MEASurement<Instance>
↪:MEValuation:PVTime:NMODe:LENergy[:LE1M]:MINimum
value: ReadStruct = driver.bluetooth.measurement.multiEval.powerVsTime.nmode.
↪lowEnergy.le1M.minimum.read()
```

No command help available

return
structure: for return value, see the help for ReadStruct structure arguments.

6.1.1.6.5.39 Le2M

class Le2MCls

Le2M commands group definition. 12 total commands, 4 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.bluetooth.measurement.multiEval.powerVsTime.nmode.lowEnergy.le2M.clone()
```

Subgroups

6.1.1.6.5.40 Average

SCPI Commands :

```
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:NMODe:LENergy:LE2M:AVERage
READ:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:NMODe:LENergy:LE2M:AVERage
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:NMODe:LENergy:LE2M:AVERage
```

class AverageCls

Average commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float or bool: No parameter help available
- Nominal_Power: float or bool: No parameter help available
- Peak_Power: float or bool: No parameter help available
- Leakage_Power: float or bool: No parameter help available
- Peak_Min_Avg_Pow: float or bool: No parameter help available

class FetchStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float: No parameter help available
- Nominal_Power: float: No parameter help available
- Peak_Power: float: No parameter help available
- Leakage_Power: float: No parameter help available
- Peak_Min_Avg_Pow: float: No parameter help available

class ReadStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float or bool: No parameter help available
- Nominal_Power: float: No parameter help available
- Peak_Power: float: No parameter help available
- Leakage_Power: float: No parameter help available
- Peak_Min_Avg_Pow: float: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:PVTime:NMODe:LENergy:LE2M:AVERage
value: CalculateStruct = driver.bluetooth.measurement.multiEval.powerVsTime.
↪nmode.lowEnergy.le2M.average.calculate()
```

No command help available

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → FetchStruct

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:MEValuation:PVTime:NMODe:LENergy:LE2M:AVERage
value: FetchStruct = driver.bluetooth.measurement.multiEval.powerVsTime.nmode.
↪lowEnergy.le2M.average.fetch()
```

No command help available

return
structure: for return value, see the help for FetchStruct structure arguments.

read() → ReadStruct

```
# SCPI: READ:BLUetooth:MEASurement<Instance>
↪:MEValuation:PVTime:NMODe:LENergy:LE2M:AVERage
value: ReadStruct = driver.bluetooth.measurement.multiEval.powerVsTime.nmode.
↪lowEnergy.le2M.average.read()
```

No command help available

return
structure: for return value, see the help for ReadStruct structure arguments.

6.1.1.6.5.41 Current

SCPI Commands :

```
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:NMODe:LENergy:LE2M:CURRent
READ:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:NMODe:LENergy:LE2M:CURRent
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:NMODe:LENergy:LE2M:CURRent
```

class CurrentCls

Current commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float or bool: No parameter help available
- Nominal_Power: float or bool: No parameter help available
- Peak_Power: float or bool: No parameter help available
- Leakage_Power: float or bool: No parameter help available
- Peak_Min_Avg_Pow: float or bool: No parameter help available

class FetchStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float: No parameter help available
- Nominal_Power: float: No parameter help available
- Peak_Power: float: No parameter help available
- Leakage_Power: float: No parameter help available
- Peak_Min_Avg_Pow: float: No parameter help available

class ReadStruct

Response structure. Fields:

- Reliability: int: No parameter help available

- Out_Of_Tol: float or bool: No parameter help available
- Nominal_Power: float: No parameter help available
- Peak_Power: float: No parameter help available
- Leakage_Power: float: No parameter help available
- Peak_Min_Avg_Pow: float: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:PVTime:NMODe:LENergy:LE2M:CURRent
value: CalculateStruct = driver.bluetooth.measurement.multiEval.powerVsTime.
↪nmode.lowEnergy.le2M.current.calculate()
```

No command help available

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → FetchStruct

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:MEValuation:PVTime:NMODe:LENergy:LE2M:CURRent
value: FetchStruct = driver.bluetooth.measurement.multiEval.powerVsTime.nmode.
↪lowEnergy.le2M.current.fetch()
```

No command help available

return
structure: for return value, see the help for FetchStruct structure arguments.

read() → ReadStruct

```
# SCPI: READ:BLUetooth:MEASurement<Instance>
↪:MEValuation:PVTime:NMODe:LENergy:LE2M:CURRent
value: ReadStruct = driver.bluetooth.measurement.multiEval.powerVsTime.nmode.
↪lowEnergy.le2M.current.read()
```

No command help available

return
structure: for return value, see the help for ReadStruct structure arguments.

6.1.1.6.5.42 Maximum

SCPI Commands :

```
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:NMODe:LENergy:LE2M:MAXimum
READ:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:NMODe:LENergy:LE2M:MAXimum
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:NMODe:LENergy:LE2M:MAXimum
```

class MaximumCls

Maximum commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float or bool: No parameter help available
- Nominal_Power: float or bool: No parameter help available
- Peak_Power: float or bool: No parameter help available
- Leakage_Power: float or bool: No parameter help available
- Peak_Min_Avg_Pow: float or bool: No parameter help available

class FetchStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float: No parameter help available
- Nominal_Power: float: No parameter help available
- Peak_Power: float: No parameter help available
- Leakage_Power: float: No parameter help available
- Peak_Min_Avg_Pow: float: No parameter help available

class ReadStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float or bool: No parameter help available
- Nominal_Power: float: No parameter help available
- Peak_Power: float: No parameter help available
- Leakage_Power: float: No parameter help available
- Peak_Min_Avg_Pow: float: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:PVTime:NMODe:LENergy:LE2M:MAXimum
value: CalculateStruct = driver.bluetooth.measurement.multiEval.powerVsTime.
↪nmode.lowEnergy.le2M.maximum.calculate()
```

No command help available

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → FetchStruct

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:MEValuation:PVTime:NMODe:LENergy:LE2M:MAXimum
value: FetchStruct = driver.bluetooth.measurement.multiEval.powerVsTime.nmode.
↪lowEnergy.le2M.maximum.fetch()
```

No command help available

return
structure: for return value, see the help for FetchStruct structure arguments.

read() → ReadStruct

```
# SCPI: READ:BLUetooth:MEASurement<Instance>
↪:MEValuation:PVTime:NMODe:LENergy:LE2M:MAXimum
value: ReadStruct = driver.bluetooth.measurement.multiEval.powerVsTime.nmode.
↪lowEnergy.le2M.maximum.read()
```

No command help available

return
structure: for return value, see the help for ReadStruct structure arguments.

6.1.1.6.5.43 Minimum

SCPI Commands :

```
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:NMODe:LENergy:LE2M:MINimum
READ:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:NMODe:LENergy:LE2M:MINimum
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:NMODe:LENergy:LE2M:MINimum
```

class MinimumCls

Minimum commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float or bool: No parameter help available
- Nominal_Power: float or bool: No parameter help available
- Peak_Power: float or bool: No parameter help available
- Leakage_Power: float or bool: No parameter help available
- Peak_Min_Avg_Pow: float or bool: No parameter help available

class FetchStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float: No parameter help available
- Nominal_Power: float: No parameter help available
- Peak_Power: float: No parameter help available
- Leakage_Power: float: No parameter help available
- Peak_Min_Avg_Pow: float: No parameter help available

class ReadStruct

Response structure. Fields:

- Reliability: int: No parameter help available

- Out_Of_Tol: float or bool: No parameter help available
- Nominal_Power: float: No parameter help available
- Peak_Power: float: No parameter help available
- Leakage_Power: float: No parameter help available
- Peak_Min_Avg_Pow: float: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:PVTime:NMODe:LENergy:LE2M:MINimum
value: CalculateStruct = driver.bluetooth.measurement.multiEval.powerVsTime.
↪nmode.lowEnergy.le2M.minimum.calculate()
```

No command help available

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → FetchStruct

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:MEValuation:PVTime:NMODe:LENergy:LE2M:MINimum
value: FetchStruct = driver.bluetooth.measurement.multiEval.powerVsTime.nmode.
↪lowEnergy.le2M.minimum.fetch()
```

No command help available

return
structure: for return value, see the help for FetchStruct structure arguments.

read() → ReadStruct

```
# SCPI: READ:BLUetooth:MEASurement<Instance>
↪:MEValuation:PVTime:NMODe:LENergy:LE2M:MINimum
value: ReadStruct = driver.bluetooth.measurement.multiEval.powerVsTime.nmode.
↪lowEnergy.le2M.minimum.read()
```

No command help available

return
structure: for return value, see the help for ReadStruct structure arguments.

6.1.1.6.5.44 Lrange

class LrangeCls

Lrange commands group definition. 12 total commands, 4 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.bluetooth.measurement.multiEval.powerVsTime.nmode.lowEnergy.lrange.
↪clone()
```

Subgroups

6.1.1.6.5.45 Average

SCPI Commands :

```
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:NMODe:LENergy:LRANge:AVERage
READ:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:NMODe:LENergy:LRANge:AVERage
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:NMODe:LENergy:LRANge:AVERage
```

class AverageCls

Average commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float or bool: No parameter help available
- Nominal_Power: float or bool: No parameter help available
- Peak_Power: float or bool: No parameter help available
- Leakage_Power: float or bool: No parameter help available
- Peak_Min_Avg_Pow: float or bool: No parameter help available

class FetchStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float: No parameter help available
- Nominal_Power: float: No parameter help available
- Peak_Power: float: No parameter help available
- Leakage_Power: float: No parameter help available
- Peak_Min_Avg_Pow: float: No parameter help available

class ReadStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float or bool: No parameter help available
- Nominal_Power: float: No parameter help available
- Peak_Power: float: No parameter help available
- Leakage_Power: float: No parameter help available

- Peak_Min_Avg_Pow: float: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:PVTime:NMODe:LENergy:LRANge:AVERage
value: CalculateStruct = driver.bluetooth.measurement.multiEval.powerVsTime.
↪nmode.lowEnergy.lrange.average.calculate()
```

No command help available

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → FetchStruct

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:MEValuation:PVTime:NMODe:LENergy:LRANge:AVERage
value: FetchStruct = driver.bluetooth.measurement.multiEval.powerVsTime.nmode.
↪lowEnergy.lrange.average.fetch()
```

No command help available

return
structure: for return value, see the help for FetchStruct structure arguments.

read() → ReadStruct

```
# SCPI: READ:BLUetooth:MEASurement<Instance>
↪:MEValuation:PVTime:NMODe:LENergy:LRANge:AVERage
value: ReadStruct = driver.bluetooth.measurement.multiEval.powerVsTime.nmode.
↪lowEnergy.lrange.average.read()
```

No command help available

return
structure: for return value, see the help for ReadStruct structure arguments.

6.1.1.6.5.46 Current

SCPI Commands :

```
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:NMODe:LENergy:LRANge:CURRent
READ:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:NMODe:LENergy:LRANge:CURRent
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:NMODe:LENergy:LRANge:CURRent
```

class CurrentCls

Current commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float or bool: No parameter help available
- Nominal_Power: float or bool: No parameter help available

- Peak_Power: float or bool: No parameter help available
- Leakage_Power: float or bool: No parameter help available
- Peak_Min_Avg_Pow: float or bool: No parameter help available

class FetchStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float: No parameter help available
- Nominal_Power: float: No parameter help available
- Peak_Power: float: No parameter help available
- Leakage_Power: float: No parameter help available
- Peak_Min_Avg_Pow: float: No parameter help available

class ReadStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float or bool: No parameter help available
- Nominal_Power: float: No parameter help available
- Peak_Power: float: No parameter help available
- Leakage_Power: float: No parameter help available
- Peak_Min_Avg_Pow: float: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:PVTime:NMODe:LENergy:LRANge:CURRent
value: CalculateStruct = driver.bluetooth.measurement.multiEval.powerVsTime.
↪nmode.lowEnergy.lrange.current.calculate()
```

No command help available

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → FetchStruct

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:MEValuation:PVTime:NMODe:LENergy:LRANge:CURRent
value: FetchStruct = driver.bluetooth.measurement.multiEval.powerVsTime.nmode.
↪lowEnergy.lrange.current.fetch()
```

No command help available

return
structure: for return value, see the help for FetchStruct structure arguments.

read() → ReadStruct

```
# SCPI: READ:BLUetooth:MEASurement<Instance>
↪:MEValuation:PVTime:NMODe:LENergy:LRANge:CURRent
value: ReadStruct = driver.bluetooth.measurement.multiEval.powerVsTime.nmode.
↪lowEnergy.lrange.current.read()
```

No command help available

return
structure: for return value, see the help for ReadStruct structure arguments.

6.1.1.6.5.47 Maximum

SCPI Commands :

```
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:NMODe:LENergy:LRANge:MAXimum
READ:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:NMODe:LENergy:LRANge:MAXimum
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:NMODe:LENergy:LRANge:MAXimum
```

class MaximumCls

Maximum commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float or bool: No parameter help available
- Nominal_Power: float or bool: No parameter help available
- Peak_Power: float or bool: No parameter help available
- Leakage_Power: float or bool: No parameter help available
- Peak_Min_Avg_Pow: float or bool: No parameter help available

class FetchStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float: No parameter help available
- Nominal_Power: float: No parameter help available
- Peak_Power: float: No parameter help available
- Leakage_Power: float: No parameter help available
- Peak_Min_Avg_Pow: float: No parameter help available

class ReadStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float or bool: No parameter help available
- Nominal_Power: float: No parameter help available
- Peak_Power: float: No parameter help available

- Leakage_Power: float: No parameter help available
- Peak_Min_Avg_Pow: float: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:PVTime:NMODe:LENergy:LRANge:MAXimum
value: CalculateStruct = driver.bluetooth.measurement.multiEval.powerVsTime.
↪nmode.lowEnergy.lrange.maximum.calculate()
```

No command help available

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → FetchStruct

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:MEValuation:PVTime:NMODe:LENergy:LRANge:MAXimum
value: FetchStruct = driver.bluetooth.measurement.multiEval.powerVsTime.nmode.
↪lowEnergy.lrange.maximum.fetch()
```

No command help available

return
structure: for return value, see the help for FetchStruct structure arguments.

read() → ReadStruct

```
# SCPI: READ:BLUetooth:MEASurement<Instance>
↪:MEValuation:PVTime:NMODe:LENergy:LRANge:MAXimum
value: ReadStruct = driver.bluetooth.measurement.multiEval.powerVsTime.nmode.
↪lowEnergy.lrange.maximum.read()
```

No command help available

return
structure: for return value, see the help for ReadStruct structure arguments.

6.1.1.6.5.48 Minimum

SCPI Commands :

```
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:NMODe:LENergy:LRANge:MINimum
READ:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:NMODe:LENergy:LRANge:MINimum
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:NMODe:LENergy:LRANge:MINimum
```

class MinimumCls

Minimum commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float or bool: No parameter help available

- Nominal_Power: float or bool: No parameter help available
- Peak_Power: float or bool: No parameter help available
- Leakage_Power: float or bool: No parameter help available
- Peak_Min_Avg_Pow: float or bool: No parameter help available

class FetchStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float: No parameter help available
- Nominal_Power: float: No parameter help available
- Peak_Power: float: No parameter help available
- Leakage_Power: float: No parameter help available
- Peak_Min_Avg_Pow: float: No parameter help available

class ReadStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float or bool: No parameter help available
- Nominal_Power: float: No parameter help available
- Peak_Power: float: No parameter help available
- Leakage_Power: float: No parameter help available
- Peak_Min_Avg_Pow: float: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:PVTime:NMODe:LENergy:LRANge:MINimum
value: CalculateStruct = driver.bluetooth.measurement.multiEval.powerVsTime.
↪nmode.lowEnergy.lrange.minimum.calculate()
```

No command help available

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → FetchStruct

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:MEValuation:PVTime:NMODe:LENergy:LRANge:MINimum
value: FetchStruct = driver.bluetooth.measurement.multiEval.powerVsTime.nmode.
↪lowEnergy.lrange.minimum.fetch()
```

No command help available

return
structure: for return value, see the help for FetchStruct structure arguments.

read() → ReadStruct

```
# SCPI: READ:BLUetooth:MEASurement<Instance>
↪:MEValuation:PVTime:NMODe:LENergy:LRANge:MINimum
value: ReadStruct = driver.bluetooth.measurement.multiEval.powerVsTime.nmode.
↪lowEnergy.lrange.minimum.read()
```

No command help available

return
structure: for return value, see the help for ReadStruct structure arguments.

6.1.1.6.5.49 Qhsl

class QhslCls

Qhsl commands group definition. 60 total commands, 5 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.bluetooth.measurement.multiEval.powerVsTime.qhsl.clone()
```

Subgroups

6.1.1.6.5.50 P2Q

class P2QCls

P2Q commands group definition. 12 total commands, 4 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.bluetooth.measurement.multiEval.powerVsTime.qhsl.p2Q.clone()
```

Subgroups

6.1.1.6.5.51 Average

SCPI Commands :

```
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:QHSL:P2Q:AVERage
READ:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:QHSL:P2Q:AVERage
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:QHSL:P2Q:AVERage
```

class AverageCls

Average commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float or bool: No parameter help available
- Nominal_Power: float or bool: No parameter help available
- Peak_Power: float or bool: No parameter help available
- Leakage_Power: float or bool: No parameter help available
- Peak_Min_Avg_Pow: float or bool: No parameter help available

class FetchStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float: No parameter help available
- Nominal_Power: float: No parameter help available
- Peak_Power: float: No parameter help available
- Leakage_Power: float: No parameter help available
- Peak_Min_Avg_Pow: float: No parameter help available

class ReadStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float or bool: No parameter help available
- Nominal_Power: float: No parameter help available
- Peak_Power: float: No parameter help available
- Leakage_Power: float: No parameter help available
- Peak_Min_Avg_Pow: float: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:PVTime:QHSL:P2Q:AVERage
value: CalculateStruct = driver.bluetooth.measurement.multiEval.powerVsTime.
↪qhsl.p2Q.average.calculate()
```

No command help available

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → FetchStruct

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:MEValuation:PVTime:QHSL:P2Q:AVERage
value: FetchStruct = driver.bluetooth.measurement.multiEval.powerVsTime.qhsl.
↪p2Q.average.fetch()
```

No command help available

return
structure: for return value, see the help for FetchStruct structure arguments.

read() → ReadStruct

```
# SCPI: READ:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:QHSL:P2Q:AVERage
value: ReadStruct = driver.bluetooth.measurement.multiEval.powerVsTime.qhsl.p2Q.
↪average.read()
```

No command help available

return
structure: for return value, see the help for ReadStruct structure arguments.

6.1.1.6.5.52 Current

SCPI Commands :

```
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:QHSL:P2Q:CURRent
READ:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:QHSL:P2Q:CURRent
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:QHSL:P2Q:CURRent
```

class CurrentCls

Current commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float or bool: No parameter help available
- Nominal_Power: float or bool: No parameter help available
- Peak_Power: float or bool: No parameter help available
- Leakage_Power: float or bool: No parameter help available
- Peak_Min_Avg_Pow: float or bool: No parameter help available

class FetchStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float: No parameter help available
- Nominal_Power: float: No parameter help available
- Peak_Power: float: No parameter help available
- Leakage_Power: float: No parameter help available
- Peak_Min_Avg_Pow: float: No parameter help available

class ReadStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float or bool: No parameter help available

- Nominal_Power: float: No parameter help available
- Peak_Power: float: No parameter help available
- Leakage_Power: float: No parameter help available
- Peak_Min_Avg_Pow: float: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:PVTime:QHSL:P2Q:CURRent
value: CalculateStruct = driver.bluetooth.measurement.multiEval.powerVsTime.
↪qhsl.p2Q.current.calculate()
```

No command help available

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → FetchStruct

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:MEValuation:PVTime:QHSL:P2Q:CURRent
value: FetchStruct = driver.bluetooth.measurement.multiEval.powerVsTime.qhsl.
↪p2Q.current.fetch()
```

No command help available

return
structure: for return value, see the help for FetchStruct structure arguments.

read() → ReadStruct

```
# SCPI: READ:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:QHSL:P2Q:CURRent
value: ReadStruct = driver.bluetooth.measurement.multiEval.powerVsTime.qhsl.p2Q.
↪current.read()
```

No command help available

return
structure: for return value, see the help for ReadStruct structure arguments.

6.1.1.6.5.53 Maximum

SCPI Commands :

```
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:QHSL:P2Q:MAXimum
READ:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:QHSL:P2Q:MAXimum
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:QHSL:P2Q:MAXimum
```

class MaximumCls

Maximum commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: No parameter help available

- Out_Of_Tol: float or bool: No parameter help available
- Nominal_Power: float or bool: No parameter help available
- Peak_Power: float or bool: No parameter help available
- Leakage_Power: float or bool: No parameter help available
- Peak_Min_Avg_Pow: float or bool: No parameter help available

class FetchStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float: No parameter help available
- Nominal_Power: float: No parameter help available
- Peak_Power: float: No parameter help available
- Leakage_Power: float: No parameter help available
- Peak_Min_Avg_Pow: float: No parameter help available

class ReadStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float or bool: No parameter help available
- Nominal_Power: float: No parameter help available
- Peak_Power: float: No parameter help available
- Leakage_Power: float: No parameter help available
- Peak_Min_Avg_Pow: float: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:PVTime:QHSL:P2Q:MAXimum
value: CalculateStruct = driver.bluetooth.measurement.multiEval.powerVsTime.
↪qhsl.p2Q.maximum.calculate()
```

No command help available

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → FetchStruct

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:MEValuation:PVTime:QHSL:P2Q:MAXimum
value: FetchStruct = driver.bluetooth.measurement.multiEval.powerVsTime.qhsl.
↪p2Q.maximum.fetch()
```

No command help available

return
structure: for return value, see the help for FetchStruct structure arguments.

read() → ReadStruct

```
# SCPI: READ:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:QHSL:P2Q:MAXimum
value: ReadStruct = driver.bluetooth.measurement.multiEval.powerVsTime.qhsl.p2Q.
↪maximum.read()
```

No command help available

return
structure: for return value, see the help for ReadStruct structure arguments.

6.1.1.6.5.54 Minimum

SCPI Commands :

```
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:QHSL:P2Q:MINimum
READ:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:QHSL:P2Q:MINimum
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:QHSL:P2Q:MINimum
```

class MinimumCls

Minimum commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float or bool: No parameter help available
- Nominal_Power: float or bool: No parameter help available
- Peak_Power: float or bool: No parameter help available
- Leakage_Power: float or bool: No parameter help available
- Peak_Min_Avg_Pow: float or bool: No parameter help available

class FetchStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float: No parameter help available
- Nominal_Power: float: No parameter help available
- Peak_Power: float: No parameter help available
- Leakage_Power: float: No parameter help available
- Peak_Min_Avg_Pow: float: No parameter help available

class ReadStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float or bool: No parameter help available
- Nominal_Power: float: No parameter help available
- Peak_Power: float: No parameter help available

- Leakage_Power: float: No parameter help available
- Peak_Min_Avg_Pow: float: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:PVTime:QHSL:P2Q:MINimum
value: CalculateStruct = driver.bluetooth.measurement.multiEval.powerVsTime.
↪qhsl.p2Q.minimum.calculate()
```

No command help available

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → FetchStruct

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:MEValuation:PVTime:QHSL:P2Q:MINimum
value: FetchStruct = driver.bluetooth.measurement.multiEval.powerVsTime.qhsl.
↪p2Q.minimum.fetch()
```

No command help available

return
structure: for return value, see the help for FetchStruct structure arguments.

read() → ReadStruct

```
# SCPI: READ:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:QHSL:P2Q:MINimum
value: ReadStruct = driver.bluetooth.measurement.multiEval.powerVsTime.qhsl.p2Q.
↪minimum.read()
```

No command help available

return
structure: for return value, see the help for ReadStruct structure arguments.

6.1.1.6.5.55 P3Q

class P3QCls

P3Q commands group definition. 12 total commands, 4 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.bluetooth.measurement.multiEval.powerVsTime.qhsl.p3Q.clone()
```

Subgroups

6.1.1.6.5.56 Average

SCPI Commands :

```
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:QHSL:P3Q:AVERage
READ:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:QHSL:P3Q:AVERage
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:QHSL:P3Q:AVERage
```

class AverageCls

Average commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float or bool: No parameter help available
- Nominal_Power: float or bool: No parameter help available
- Peak_Power: float or bool: No parameter help available
- Leakage_Power: float or bool: No parameter help available
- Peak_Min_Avg_Pow: float or bool: No parameter help available

class FetchStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float: No parameter help available
- Nominal_Power: float: No parameter help available
- Peak_Power: float: No parameter help available
- Leakage_Power: float: No parameter help available
- Peak_Min_Avg_Pow: float: No parameter help available

class ReadStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float or bool: No parameter help available
- Nominal_Power: float: No parameter help available
- Peak_Power: float: No parameter help available
- Leakage_Power: float: No parameter help available
- Peak_Min_Avg_Pow: float: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:PVTime:QHSL:P3Q:AVERage
value: CalculateStruct = driver.bluetooth.measurement.multiEval.powerVsTime.
↪qhsl.p3Q.average.calculate()
```

No command help available

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → FetchStruct

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:MEValuation:PVTime:QHSL:P3Q:AVERage
value: FetchStruct = driver.bluetooth.measurement.multiEval.powerVsTime.qhsl.
↪p3Q.average.fetch()
```

No command help available

return
structure: for return value, see the help for FetchStruct structure arguments.

read() → ReadStruct

```
# SCPI: READ:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:QHSL:P3Q:AVERage
value: ReadStruct = driver.bluetooth.measurement.multiEval.powerVsTime.qhsl.p3Q.
↪average.read()
```

No command help available

return
structure: for return value, see the help for ReadStruct structure arguments.

6.1.1.6.5.57 Current

SCPI Commands :

```
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:QHSL:P3Q:CURRent
READ:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:QHSL:P3Q:CURRent
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:QHSL:P3Q:CURRent
```

class CurrentCls

Current commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float or bool: No parameter help available
- Nominal_Power: float or bool: No parameter help available
- Peak_Power: float or bool: No parameter help available
- Leakage_Power: float or bool: No parameter help available
- Peak_Min_Avg_Pow: float or bool: No parameter help available

class FetchStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float: No parameter help available
- Nominal_Power: float: No parameter help available
- Peak_Power: float: No parameter help available
- Leakage_Power: float: No parameter help available
- Peak_Min_Avg_Pow: float: No parameter help available

class ReadStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float or bool: No parameter help available
- Nominal_Power: float: No parameter help available
- Peak_Power: float: No parameter help available
- Leakage_Power: float: No parameter help available
- Peak_Min_Avg_Pow: float: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:PVTime:QHSL:P3Q:CURRent
value: CalculateStruct = driver.bluetooth.measurement.multiEval.powerVsTime.
↪qhsl.p3Q.current.calculate()
```

No command help available

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → FetchStruct

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:MEValuation:PVTime:QHSL:P3Q:CURRent
value: FetchStruct = driver.bluetooth.measurement.multiEval.powerVsTime.qhsl.
↪p3Q.current.fetch()
```

No command help available

return
structure: for return value, see the help for FetchStruct structure arguments.

read() → ReadStruct

```
# SCPI: READ:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:QHSL:P3Q:CURRent
value: ReadStruct = driver.bluetooth.measurement.multiEval.powerVsTime.qhsl.p3Q.
↪current.read()
```

No command help available

return
structure: for return value, see the help for ReadStruct structure arguments.

6.1.1.6.5.58 Maximum

SCPI Commands :

```
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:QHSL:P3Q:MAXimum
READ:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:QHSL:P3Q:MAXimum
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:QHSL:P3Q:MAXimum
```

class MaximumCls

Maximum commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float or bool: No parameter help available
- Nominal_Power: float or bool: No parameter help available
- Peak_Power: float or bool: No parameter help available
- Leakage_Power: float or bool: No parameter help available
- Peak_Min_Avg_Pow: float or bool: No parameter help available

class FetchStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float: No parameter help available
- Nominal_Power: float: No parameter help available
- Peak_Power: float: No parameter help available
- Leakage_Power: float: No parameter help available
- Peak_Min_Avg_Pow: float: No parameter help available

class ReadStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float or bool: No parameter help available
- Nominal_Power: float: No parameter help available
- Peak_Power: float: No parameter help available
- Leakage_Power: float: No parameter help available
- Peak_Min_Avg_Pow: float: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:PVTime:QHSL:P3Q:MAXimum
value: CalculateStruct = driver.bluetooth.measurement.multiEval.powerVsTime.
↪qhsl.p3Q.maximum.calculate()
```

No command help available

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → FetchStruct

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:MEValuation:PVTime:QHSL:P3Q:MAXimum
value: FetchStruct = driver.bluetooth.measurement.multiEval.powerVsTime.qhsl.
↪p3Q.maximum.fetch()
```

No command help available

return
structure: for return value, see the help for FetchStruct structure arguments.

read() → ReadStruct

```
# SCPI: READ:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:QHSL:P3Q:MAXimum
value: ReadStruct = driver.bluetooth.measurement.multiEval.powerVsTime.qhsl.p3Q.
↪maximum.read()
```

No command help available

return
structure: for return value, see the help for ReadStruct structure arguments.

6.1.1.6.5.59 Minimum

SCPI Commands :

```
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:QHSL:P3Q:MINimum
READ:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:QHSL:P3Q:MINimum
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:QHSL:P3Q:MINimum
```

class MinimumCls

Minimum commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float or bool: No parameter help available
- Nominal_Power: float or bool: No parameter help available
- Peak_Power: float or bool: No parameter help available
- Leakage_Power: float or bool: No parameter help available
- Peak_Min_Avg_Pow: float or bool: No parameter help available

class FetchStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float: No parameter help available
- Nominal_Power: float: No parameter help available
- Peak_Power: float: No parameter help available
- Leakage_Power: float: No parameter help available
- Peak_Min_Avg_Pow: float: No parameter help available

class ReadStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float or bool: No parameter help available
- Nominal_Power: float: No parameter help available
- Peak_Power: float: No parameter help available
- Leakage_Power: float: No parameter help available
- Peak_Min_Avg_Pow: float: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:PVTime:QHSL:P3Q:MINimum
value: CalculateStruct = driver.bluetooth.measurement.multiEval.powerVsTime.
↪qhsl.p3Q.minimum.calculate()
```

No command help available

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → FetchStruct

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:MEValuation:PVTime:QHSL:P3Q:MINimum
value: FetchStruct = driver.bluetooth.measurement.multiEval.powerVsTime.qhsl.
↪p3Q.minimum.fetch()
```

No command help available

return
structure: for return value, see the help for FetchStruct structure arguments.

read() → ReadStruct

```
# SCPI: READ:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:QHSL:P3Q:MINimum
value: ReadStruct = driver.bluetooth.measurement.multiEval.powerVsTime.qhsl.p3Q.
↪minimum.read()
```

No command help available

return
structure: for return value, see the help for ReadStruct structure arguments.

6.1.1.6.5.60 P4Q

class P4QCls
P4Q commands group definition. 12 total commands, 4 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.bluetooth.measurement.multiEval.powerVsTime.qhsl.p4Q.clone()
```

Subgroups

6.1.1.6.5.61 Average

SCPI Commands :

```
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:QHSL:P4Q:AVERage
READ:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:QHSL:P4Q:AVERage
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:QHSL:P4Q:AVERage
```

class AverageCls
Average commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct
Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float or bool: No parameter help available
- Nominal_Power: float or bool: No parameter help available
- Peak_Power: float or bool: No parameter help available
- Leakage_Power: float or bool: No parameter help available
- Peak_Min_Avg_Pow: float or bool: No parameter help available

class FetchStruct
Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float: No parameter help available
- Nominal_Power: float: No parameter help available
- Peak_Power: float: No parameter help available
- Leakage_Power: float: No parameter help available
- Peak_Min_Avg_Pow: float: No parameter help available

class ReadStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float or bool: No parameter help available
- Nominal_Power: float: No parameter help available
- Peak_Power: float: No parameter help available
- Leakage_Power: float: No parameter help available
- Peak_Min_Avg_Pow: float: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:PVTime:QHSL:P4Q:AVERage
value: CalculateStruct = driver.bluetooth.measurement.multiEval.powerVsTime.
↪qhsl.p4Q.average.calculate()
```

No command help available

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → FetchStruct

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:MEValuation:PVTime:QHSL:P4Q:AVERage
value: FetchStruct = driver.bluetooth.measurement.multiEval.powerVsTime.qhsl.
↪p4Q.average.fetch()
```

No command help available

return
structure: for return value, see the help for FetchStruct structure arguments.

read() → ReadStruct

```
# SCPI: READ:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:QHSL:P4Q:AVERage
value: ReadStruct = driver.bluetooth.measurement.multiEval.powerVsTime.qhsl.p4Q.
↪average.read()
```

No command help available

return
structure: for return value, see the help for ReadStruct structure arguments.

6.1.1.6.5.62 Current

SCPI Commands :

```
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:QHSL:P4Q:CURRent
READ:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:QHSL:P4Q:CURRent
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:QHSL:P4Q:CURRent
```

class CurrentCls

Current commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float or bool: No parameter help available
- Nominal_Power: float or bool: No parameter help available
- Peak_Power: float or bool: No parameter help available
- Leakage_Power: float or bool: No parameter help available
- Peak_Min_Avg_Pow: float or bool: No parameter help available

class FetchStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float: No parameter help available
- Nominal_Power: float: No parameter help available
- Peak_Power: float: No parameter help available
- Leakage_Power: float: No parameter help available
- Peak_Min_Avg_Pow: float: No parameter help available

class ReadStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float or bool: No parameter help available
- Nominal_Power: float: No parameter help available
- Peak_Power: float: No parameter help available
- Leakage_Power: float: No parameter help available
- Peak_Min_Avg_Pow: float: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:PVTime:QHSL:P4Q:CURRent
value: CalculateStruct = driver.bluetooth.measurement.multiEval.powerVsTime.
↪qhsl.p4Q.current.calculate()
```

No command help available

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → FetchStruct

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:MEValuation:PVTime:QHSL:P4Q:CURRent
value: FetchStruct = driver.bluetooth.measurement.multiEval.powerVsTime.qhsl.
↪p4Q.current.fetch()
```

No command help available

return
structure: for return value, see the help for FetchStruct structure arguments.

read() → ReadStruct

```
# SCPI: READ:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:QHSL:P4Q:CURRent
value: ReadStruct = driver.bluetooth.measurement.multiEval.powerVsTime.qhsl.p4Q.
↪current.read()
```

No command help available

return
structure: for return value, see the help for ReadStruct structure arguments.

6.1.1.6.5.63 Maximum

SCPI Commands :

```
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:QHSL:P4Q:MAXimum
READ:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:QHSL:P4Q:MAXimum
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:QHSL:P4Q:MAXimum
```

class MaximumCls

Maximum commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float or bool: No parameter help available
- Nominal_Power: float or bool: No parameter help available
- Peak_Power: float or bool: No parameter help available
- Leakage_Power: float or bool: No parameter help available
- Peak_Min_Avg_Pow: float or bool: No parameter help available

class FetchStruct

Response structure. Fields:

- Reliability: int: No parameter help available

- Out_Of_Tol: float: No parameter help available
- Nominal_Power: float: No parameter help available
- Peak_Power: float: No parameter help available
- Leakage_Power: float: No parameter help available
- Peak_Min_Avg_Pow: float: No parameter help available

class ReadStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float or bool: No parameter help available
- Nominal_Power: float: No parameter help available
- Peak_Power: float: No parameter help available
- Leakage_Power: float: No parameter help available
- Peak_Min_Avg_Pow: float: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:PVTime:QHSL:P4Q:MAXimum
value: CalculateStruct = driver.bluetooth.measurement.multiEval.powerVsTime.
↪qhsl.p4Q.maximum.calculate()
```

No command help available

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → FetchStruct

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:MEValuation:PVTime:QHSL:P4Q:MAXimum
value: FetchStruct = driver.bluetooth.measurement.multiEval.powerVsTime.qhsl.
↪p4Q.maximum.fetch()
```

No command help available

return
structure: for return value, see the help for FetchStruct structure arguments.

read() → ReadStruct

```
# SCPI: READ:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:QHSL:P4Q:MAXimum
value: ReadStruct = driver.bluetooth.measurement.multiEval.powerVsTime.qhsl.p4Q.
↪maximum.read()
```

No command help available

return
structure: for return value, see the help for ReadStruct structure arguments.

6.1.1.6.5.64 Minimum

SCPI Commands :

```
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:QHSL:P4Q:MINimum
READ:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:QHSL:P4Q:MINimum
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:QHSL:P4Q:MINimum
```

class MinimumCls

Minimum commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float or bool: No parameter help available
- Nominal_Power: float or bool: No parameter help available
- Peak_Power: float or bool: No parameter help available
- Leakage_Power: float or bool: No parameter help available
- Peak_Min_Avg_Pow: float or bool: No parameter help available

class FetchStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float: No parameter help available
- Nominal_Power: float: No parameter help available
- Peak_Power: float: No parameter help available
- Leakage_Power: float: No parameter help available
- Peak_Min_Avg_Pow: float: No parameter help available

class ReadStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float or bool: No parameter help available
- Nominal_Power: float: No parameter help available
- Peak_Power: float: No parameter help available
- Leakage_Power: float: No parameter help available
- Peak_Min_Avg_Pow: float: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:PVTime:QHSL:P4Q:MINimum
value: CalculateStruct = driver.bluetooth.measurement.multiEval.powerVsTime.
↪qhsl.p4Q.minimum.calculate()
```

No command help available

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → FetchStruct

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:MEValuation:PVTime:QHSL:P4Q:MINimum
value: FetchStruct = driver.bluetooth.measurement.multiEval.powerVsTime.qhsl.
↪p4Q.minimum.fetch()
```

No command help available

return
structure: for return value, see the help for FetchStruct structure arguments.

read() → ReadStruct

```
# SCPI: READ:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:QHSL:P4Q:MINimum
value: ReadStruct = driver.bluetooth.measurement.multiEval.powerVsTime.qhsl.p4Q.
↪minimum.read()
```

No command help available

return
structure: for return value, see the help for ReadStruct structure arguments.

6.1.1.6.5.65 P5Q

class P5QCls

P5Q commands group definition. 12 total commands, 4 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.bluetooth.measurement.multiEval.powerVsTime.qhsl.p5Q.clone()
```

Subgroups

6.1.1.6.5.66 Average

SCPI Commands :

```
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:QHSL:P5Q:AVERage
READ:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:QHSL:P5Q:AVERage
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:QHSL:P5Q:AVERage
```

class AverageCls

Average commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float or bool: No parameter help available
- Nominal_Power: float or bool: No parameter help available
- Peak_Power: float or bool: No parameter help available
- Leakage_Power: float or bool: No parameter help available
- Peak_Min_Avg_Pow: float or bool: No parameter help available

class FetchStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float: No parameter help available
- Nominal_Power: float: No parameter help available
- Peak_Power: float: No parameter help available
- Leakage_Power: float: No parameter help available
- Peak_Min_Avg_Pow: float: No parameter help available

class ReadStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float or bool: No parameter help available
- Nominal_Power: float: No parameter help available
- Peak_Power: float: No parameter help available
- Leakage_Power: float: No parameter help available
- Peak_Min_Avg_Pow: float: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:PVTime:QHSL:P5Q:AVERage
value: CalculateStruct = driver.bluetooth.measurement.multiEval.powerVsTime.
↪qhsl.p5Q.average.calculate()
```

No command help available

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → FetchStruct

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:MEValuation:PVTime:QHSL:P5Q:AVERage
value: FetchStruct = driver.bluetooth.measurement.multiEval.powerVsTime.qhsl.
↪p5Q.average.fetch()
```

No command help available

return
structure: for return value, see the help for FetchStruct structure arguments.

read() → ReadStruct

```
# SCPI: READ:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:QHSL:P5Q:AVERage
value: ReadStruct = driver.bluetooth.measurement.multiEval.powerVsTime.qhsl.p5Q.
↪average.read()
```

No command help available

return
structure: for return value, see the help for ReadStruct structure arguments.

6.1.1.6.5.67 Current

SCPI Commands :

```
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:QHSL:P5Q:CURRent
READ:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:QHSL:P5Q:CURRent
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:QHSL:P5Q:CURRent
```

class CurrentCls

Current commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float or bool: No parameter help available
- Nominal_Power: float or bool: No parameter help available
- Peak_Power: float or bool: No parameter help available
- Leakage_Power: float or bool: No parameter help available
- Peak_Min_Avg_Pow: float or bool: No parameter help available

class FetchStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float: No parameter help available
- Nominal_Power: float: No parameter help available
- Peak_Power: float: No parameter help available
- Leakage_Power: float: No parameter help available
- Peak_Min_Avg_Pow: float: No parameter help available

class ReadStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float or bool: No parameter help available

- Nominal_Power: float: No parameter help available
- Peak_Power: float: No parameter help available
- Leakage_Power: float: No parameter help available
- Peak_Min_Avg_Pow: float: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:PVTime:QHSL:P5Q:CURRent
value: CalculateStruct = driver.bluetooth.measurement.multiEval.powerVsTime.
↪qhsl.p5Q.current.calculate()
```

No command help available

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → FetchStruct

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:MEValuation:PVTime:QHSL:P5Q:CURRent
value: FetchStruct = driver.bluetooth.measurement.multiEval.powerVsTime.qhsl.
↪p5Q.current.fetch()
```

No command help available

return
structure: for return value, see the help for FetchStruct structure arguments.

read() → ReadStruct

```
# SCPI: READ:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:QHSL:P5Q:CURRent
value: ReadStruct = driver.bluetooth.measurement.multiEval.powerVsTime.qhsl.p5Q.
↪current.read()
```

No command help available

return
structure: for return value, see the help for ReadStruct structure arguments.

6.1.1.6.5.68 Maximum

SCPI Commands :

```
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:QHSL:P5Q:MAXimum
READ:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:QHSL:P5Q:MAXimum
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:QHSL:P5Q:MAXimum
```

class MaximumCls

Maximum commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: No parameter help available

- Out_Of_Tol: float or bool: No parameter help available
- Nominal_Power: float or bool: No parameter help available
- Peak_Power: float or bool: No parameter help available
- Leakage_Power: float or bool: No parameter help available
- Peak_Min_Avg_Pow: float or bool: No parameter help available

class FetchStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float: No parameter help available
- Nominal_Power: float: No parameter help available
- Peak_Power: float: No parameter help available
- Leakage_Power: float: No parameter help available
- Peak_Min_Avg_Pow: float: No parameter help available

class ReadStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float or bool: No parameter help available
- Nominal_Power: float: No parameter help available
- Peak_Power: float: No parameter help available
- Leakage_Power: float: No parameter help available
- Peak_Min_Avg_Pow: float: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:PVTime:QHSL:P5Q:MAXimum
value: CalculateStruct = driver.bluetooth.measurement.multiEval.powerVsTime.
↪qhsl.p5Q.maximum.calculate()
```

No command help available

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → FetchStruct

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:MEValuation:PVTime:QHSL:P5Q:MAXimum
value: FetchStruct = driver.bluetooth.measurement.multiEval.powerVsTime.qhsl.
↪p5Q.maximum.fetch()
```

No command help available

return
structure: for return value, see the help for FetchStruct structure arguments.

read() → ReadStruct

```
# SCPI: READ:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:QHSL:P5Q:MAXimum
value: ReadStruct = driver.bluetooth.measurement.multiEval.powerVsTime.qhsl.p5Q.
↪maximum.read()
```

No command help available

return
structure: for return value, see the help for ReadStruct structure arguments.

6.1.1.6.5.69 Minimum

SCPI Commands :

```
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:QHSL:P5Q:MINimum
READ:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:QHSL:P5Q:MINimum
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:QHSL:P5Q:MINimum
```

class MinimumCls

Minimum commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float or bool: No parameter help available
- Nominal_Power: float or bool: No parameter help available
- Peak_Power: float or bool: No parameter help available
- Leakage_Power: float or bool: No parameter help available
- Peak_Min_Avg_Pow: float or bool: No parameter help available

class FetchStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float: No parameter help available
- Nominal_Power: float: No parameter help available
- Peak_Power: float: No parameter help available
- Leakage_Power: float: No parameter help available
- Peak_Min_Avg_Pow: float: No parameter help available

class ReadStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float or bool: No parameter help available
- Nominal_Power: float: No parameter help available
- Peak_Power: float: No parameter help available

- Leakage_Power: float: No parameter help available
- Peak_Min_Avg_Pow: float: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:PVTime:QHSL:P5Q:MINimum
value: CalculateStruct = driver.bluetooth.measurement.multiEval.powerVsTime.
↪qhsl.p5Q.minimum.calculate()
```

No command help available

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → FetchStruct

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:MEValuation:PVTime:QHSL:P5Q:MINimum
value: FetchStruct = driver.bluetooth.measurement.multiEval.powerVsTime.qhsl.
↪p5Q.minimum.fetch()
```

No command help available

return
structure: for return value, see the help for FetchStruct structure arguments.

read() → ReadStruct

```
# SCPI: READ:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:QHSL:P5Q:MINimum
value: ReadStruct = driver.bluetooth.measurement.multiEval.powerVsTime.qhsl.p5Q.
↪minimum.read()
```

No command help available

return
structure: for return value, see the help for ReadStruct structure arguments.

6.1.1.6.5.70 P6Q

class P6QCls

P6Q commands group definition. 12 total commands, 4 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.bluetooth.measurement.multiEval.powerVsTime.qhsl.p6Q.clone()
```

Subgroups

6.1.1.6.5.71 Average

SCPI Commands :

```
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:QHSL:P6Q:AVERage
READ:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:QHSL:P6Q:AVERage
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:QHSL:P6Q:AVERage
```

class AverageCls

Average commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float or bool: No parameter help available
- Nominal_Power: float or bool: No parameter help available
- Peak_Power: float or bool: No parameter help available
- Leakage_Power: float or bool: No parameter help available
- Peak_Min_Avg_Pow: float or bool: No parameter help available

class FetchStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float: No parameter help available
- Nominal_Power: float: No parameter help available
- Peak_Power: float: No parameter help available
- Leakage_Power: float: No parameter help available
- Peak_Min_Avg_Pow: float: No parameter help available

class ReadStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float or bool: No parameter help available
- Nominal_Power: float: No parameter help available
- Peak_Power: float: No parameter help available
- Leakage_Power: float: No parameter help available
- Peak_Min_Avg_Pow: float: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:PVTime:QHSL:P6Q:AVERage
value: CalculateStruct = driver.bluetooth.measurement.multiEval.powerVsTime.
↪qhsl.p6Q.average.calculate()
```

No command help available

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → FetchStruct

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:MEValuation:PVTime:QHSL:P6Q:AVERage
value: FetchStruct = driver.bluetooth.measurement.multiEval.powerVsTime.qhsl.
↪p6Q.average.fetch()
```

No command help available

return
structure: for return value, see the help for FetchStruct structure arguments.

read() → ReadStruct

```
# SCPI: READ:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:QHSL:P6Q:AVERage
value: ReadStruct = driver.bluetooth.measurement.multiEval.powerVsTime.qhsl.p6Q.
↪average.read()
```

No command help available

return
structure: for return value, see the help for ReadStruct structure arguments.

6.1.1.6.5.72 Current

SCPI Commands :

```
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:QHSL:P6Q:CURRent
READ:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:QHSL:P6Q:CURRent
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:QHSL:P6Q:CURRent
```

class CurrentCls

Current commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float or bool: No parameter help available
- Nominal_Power: float or bool: No parameter help available
- Peak_Power: float or bool: No parameter help available
- Leakage_Power: float or bool: No parameter help available
- Peak_Min_Avg_Pow: float or bool: No parameter help available

class FetchStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float: No parameter help available
- Nominal_Power: float: No parameter help available
- Peak_Power: float: No parameter help available
- Leakage_Power: float: No parameter help available
- Peak_Min_Avg_Pow: float: No parameter help available

class ReadStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float or bool: No parameter help available
- Nominal_Power: float: No parameter help available
- Peak_Power: float: No parameter help available
- Leakage_Power: float: No parameter help available
- Peak_Min_Avg_Pow: float: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:PVTime:QHSL:P6Q:CURRent
value: CalculateStruct = driver.bluetooth.measurement.multiEval.powerVsTime.
↪qhsl.p6Q.current.calculate()
```

No command help available

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → FetchStruct

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:MEValuation:PVTime:QHSL:P6Q:CURRent
value: FetchStruct = driver.bluetooth.measurement.multiEval.powerVsTime.qhsl.
↪p6Q.current.fetch()
```

No command help available

return
structure: for return value, see the help for FetchStruct structure arguments.

read() → ReadStruct

```
# SCPI: READ:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:QHSL:P6Q:CURRent
value: ReadStruct = driver.bluetooth.measurement.multiEval.powerVsTime.qhsl.p6Q.
↪current.read()
```

No command help available

return
structure: for return value, see the help for ReadStruct structure arguments.

6.1.1.6.5.73 Maximum

SCPI Commands :

```
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:QHSL:P6Q:MAXimum
READ:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:QHSL:P6Q:MAXimum
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:QHSL:P6Q:MAXimum
```

class MaximumCls

Maximum commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float or bool: No parameter help available
- Nominal_Power: float or bool: No parameter help available
- Peak_Power: float or bool: No parameter help available
- Leakage_Power: float or bool: No parameter help available
- Peak_Min_Avg_Pow: float or bool: No parameter help available

class FetchStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float: No parameter help available
- Nominal_Power: float: No parameter help available
- Peak_Power: float: No parameter help available
- Leakage_Power: float: No parameter help available
- Peak_Min_Avg_Pow: float: No parameter help available

class ReadStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float or bool: No parameter help available
- Nominal_Power: float: No parameter help available
- Peak_Power: float: No parameter help available
- Leakage_Power: float: No parameter help available
- Peak_Min_Avg_Pow: float: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:PVTime:QHSL:P6Q:MAXimum
value: CalculateStruct = driver.bluetooth.measurement.multiEval.powerVsTime.
↪qhsl.p6Q.maximum.calculate()
```

No command help available

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → FetchStruct

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:MEValuation:PVTime:QHSL:P6Q:MAXimum
value: FetchStruct = driver.bluetooth.measurement.multiEval.powerVsTime.qhsl.
↪p6Q.maximum.fetch()
```

No command help available

return
structure: for return value, see the help for FetchStruct structure arguments.

read() → ReadStruct

```
# SCPI: READ:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:QHSL:P6Q:MAXimum
value: ReadStruct = driver.bluetooth.measurement.multiEval.powerVsTime.qhsl.p6Q.
↪maximum.read()
```

No command help available

return
structure: for return value, see the help for ReadStruct structure arguments.

6.1.1.6.5.74 Minimum

SCPI Commands :

```
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:QHSL:P6Q:MINimum
READ:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:QHSL:P6Q:MINimum
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:QHSL:P6Q:MINimum
```

class MinimumCls

Minimum commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float or bool: No parameter help available
- Nominal_Power: float or bool: No parameter help available
- Peak_Power: float or bool: No parameter help available
- Leakage_Power: float or bool: No parameter help available
- Peak_Min_Avg_Pow: float or bool: No parameter help available

class FetchStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float: No parameter help available
- Nominal_Power: float: No parameter help available
- Peak_Power: float: No parameter help available
- Leakage_Power: float: No parameter help available
- Peak_Min_Avg_Pow: float: No parameter help available

class ReadStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float or bool: No parameter help available
- Nominal_Power: float: No parameter help available
- Peak_Power: float: No parameter help available
- Leakage_Power: float: No parameter help available
- Peak_Min_Avg_Pow: float: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:PVTime:QHSL:P6Q:MINimum
value: CalculateStruct = driver.bluetooth.measurement.multiEval.powerVsTime.
↪qhsl.p6Q.minimum.calculate()
```

No command help available

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → FetchStruct

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:MEValuation:PVTime:QHSL:P6Q:MINimum
value: FetchStruct = driver.bluetooth.measurement.multiEval.powerVsTime.qhsl.
↪p6Q.minimum.fetch()
```

No command help available

return
structure: for return value, see the help for FetchStruct structure arguments.

read() → ReadStruct

```
# SCPI: READ:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:QHSL:P6Q:MINimum
value: ReadStruct = driver.bluetooth.measurement.multiEval.powerVsTime.qhsl.p6Q.
↪minimum.read()
```

No command help available

return
structure: for return value, see the help for ReadStruct structure arguments.

6.1.1.6.6 Sacp

class SacpCls

Sacp commands group definition. 39 total commands, 4 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.bluetooth.measurement.multiEval.sacp.clone()
```

Subgroups

6.1.1.6.6.1 Brate

class BrateCls

Brate commands group definition. 3 total commands, 1 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.bluetooth.measurement.multiEval.sacp.brate.clone()
```

Subgroups

6.1.1.6.6.2 Ptx

SCPI Commands :

```
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:SACP:BRATe[:PTX]
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:SACP:BRATe[:PTX]
READ:BLUetooth:MEASurement<Instance>:MEValuation:SACP:BRATe[:PTX]
```

class PtxCls

Ptx commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: 'Reliability indicator'
- Out_Of_Tol: float or bool: Out of tolerance result, i.e. the percentage of measurement intervals of the statistic count ([CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Scount#Sacp CMDLINKRESOLVED]) exceeding the specified limits.
- Nominal_Power: float or bool: Average power during the carrier-on state

- No_Of_Exceptions: float or bool: Number of exceptions (channels ±3, ±4 ... with an ACP above the Exception PTx threshold)
- Acp: List[float or bool]: 79 ACP results

class ResultData

Response structure. Fields:

- Reliability: int: 'Reliability indicator'
- Out_Of_Tol: float: Out of tolerance result, i.e. the percentage of measurement intervals of the statistic count ([CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Scount#Sacp CMDLINKRESOLVED]) exceeding the specified limits.
- Nominal_Power: float: Average power during the carrier-on state
- No_Of_Exceptions: int: Number of exceptions (channels ±3, ±4 ... with an ACP above the Exception PTx threshold)
- Acp: List[float]: 79 ACP results

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:SACP:BRATe[:PTX]
value: CalculateStruct = driver.bluetooth.measurement.multiEval.sacp.brate.ptx.
↪calculate()
```

Returns the Spectrum ACP results for BR packets. The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

INTRO_CMD_HELP: The number of valid ACP results depends on the ACP measurement mode (method RsCMPX_BluetoothMeas.Configure.Bluetooth.Measurement.MultiEval.Sacp.Brate.Measurement.mode CH21 | CH79) :

- If ACP +/- 10 Channels is selected, the first 21 ACP values contain the results for the relative channels –10, ..., 0, ..., +10; the remaining 58 values are not displayed.
- If ACP 79 Channels is selected, valid ACP values are available for all channels in the Bluetooth regulatory range.

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → ResultData

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:MEValuation:SACP:BRATe[:PTX]
value: ResultData = driver.bluetooth.measurement.multiEval.sacp.brate.ptx.
↪fetch()
```

Returns the Spectrum ACP results for BR packets. The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

INTRO_CMD_HELP: The number of valid ACP results depends on the ACP measurement mode (method RsCMPX_BluetoothMeas.Configure.Bluetooth.Measurement.MultiEval.Sacp.Brate.Measurement.mode CH21 | CH79) :

- If ACP +/- 10 Channels is selected, the first 21 ACP values contain the results for the relative channels –10, ..., 0, ..., +10; the remaining 58 values are not displayed.

- If ACP 79 Channels is selected, valid ACP values are available for all channels in the Bluetooth regulatory range.

return
structure: for return value, see the help for ResultData structure arguments.

read() → ResultData

```
# SCPI: READ:BLUetooth:MEASurement<Instance>:MEValuation:SACP:BRATe[:PTX]
value: ResultData = driver.bluetooth.measurement.multiEval.sacp.brate.ptx.read()
```

Returns the Spectrum ACP results for BR packets. The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

INTRO_CMD_HELP: The number of valid ACP results depends on the ACP measurement mode (method RsCMPX_BluetoothMeas.Configure.Bluetooth.Measurement.MultiEval.Sacp.Brate.Measurement.mode CH21 | CH79) :

- If ACP +/- 10 Channels is selected, the first 21 ACP values contain the results for the relative channels –10, …, 0, …, +10; the remaining 58 values are not displayed.
- If ACP 79 Channels is selected, valid ACP values are available for all channels in the Bluetooth regulatory range.

return
structure: for return value, see the help for ResultData structure arguments.

6.1.1.6.6.3 LowEnergy

class LowEnergyCls

LowEnergy commands group definition. 9 total commands, 3 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.bluetooth.measurement.multiEval.sacp.lowEnergy.clone()
```

Subgroups

6.1.1.6.6.4 Le1M

SCPI Commands :

```
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:SACP:LENergy[:LE1M]
READ:BLUetooth:MEASurement<Instance>:MEValuation:SACP:LENergy[:LE1M]
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:SACP:LENergy[:LE1M]
```

class Le1MCls

Le1M commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: 'Reliability indicator'
- Nominal_Power: float or bool: Average power during the carrier-on state
- No_Of_Exceptions: float or bool: Number of exceptions (channels ±3, ±4 ... with an ACP above the Exception PTx threshold)

class ResultData

Response structure. Fields:

- Reliability: int: 'Reliability indicator'
- Nominal_Power: float: Average power during the carrier-on state
- No_Of_Exceptions: int: Number of exceptions (channels ±3, ±4 ... with an ACP above the Exception PTx threshold)

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:SACP:LENergy[:LE1M]
value: CalculateStruct = driver.bluetooth.measurement.multiEval.sacp.lowEnergy.
↪le1M.calculate()
```

Returns the Spectrum ACP results for LE1M. See 'Square Spectrum ACP'. The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → ResultData

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:MEValuation:SACP:LENergy[:LE1M]
value: ResultData = driver.bluetooth.measurement.multiEval.sacp.lowEnergy.le1M.
↪fetch()
```

Returns the Spectrum ACP results for LE1M. See 'Square Spectrum ACP'. The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for ResultData structure arguments.

read() → ResultData

```
# SCPI: READ:BLUetooth:MEASurement<Instance>:MEValuation:SACP:LENergy[:LE1M]
value: ResultData = driver.bluetooth.measurement.multiEval.sacp.lowEnergy.le1M.
↪read()
```

Returns the Spectrum ACP results for LE1M. See 'Square Spectrum ACP'. The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for ResultData structure arguments.

6.1.1.6.6.5 Le2M

SCPI Commands :

```
READ:BLUetooth:MEASurement<Instance>:MEValuation:SACP:LENergy:LE2M
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:SACP:LENergy:LE2M
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:SACP:LENergy:LE2M
```

class Le2MCls

Le2M commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: 'Reliability indicator'
- Nominal_Power: float or bool: Average power during the carrier-on state
- No_Of_Exceptions: float or bool: Number of exceptions (channels ±3, ±4 ... with an ACP above the Exception PTx threshold)

class ResultData

Response structure. Fields:

- Reliability: int: 'Reliability indicator'
- Nominal_Power: float: Average power during the carrier-on state
- No_Of_Exceptions: int: Number of exceptions (channels ±3, ±4 ... with an ACP above the Exception PTx threshold)

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:SACP:LENergy:LE2M
value: CalculateStruct = driver.bluetooth.measurement.multiEval.sacp.lowEnergy.
↪le2M.calculate()
```

Returns the Spectrum ACP results for LE 2M PHY (...:LE2M...) and LE coded PHY (...:LRANge...) . See 'Square Spectrum ACP'. The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → ResultData

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:MEValuation:SACP:LENergy:LE2M
value: ResultData = driver.bluetooth.measurement.multiEval.sacp.lowEnergy.le2M.
↪fetch()
```

Returns the Spectrum ACP results for LE 2M PHY (...:LE2M...) and LE coded PHY (...:LRANge...) . See 'Square Spectrum ACP'. The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for ResultData structure arguments.

read() → ResultData

```
# SCPI: READ:BLUetooth:MEASurement<Instance>:MEValuation:SACP:LENergy:LE2M
value: ResultData = driver.bluetooth.measurement.multiEval.sacp.lowEnergy.le2M.
↪read()
```

Returns the Spectrum ACP results for LE 2M PHY (...:LE2M...) and LE coded PHY (...:LRANge...) . See 'Square Spectrum ACP'. The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for ResultData structure arguments.

6.1.1.6.6.6 Lrange

SCPI Commands :

```
READ:BLUetooth:MEASurement<Instance>:MEValuation:SACP:LENergy:LRANge
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:SACP:LENergy:LRANge
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:SACP:LENergy:LRANge
```

class LrangeCls

Lrange commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: 'Reliability indicator'
- Nominal_Power: float or bool: Average power during the carrier-on state
- No_Of_Exceptions: float or bool: Number of exceptions (channels ±3, ±4 ... with an ACP above the Exception PTx threshold)

class ResultData

Response structure. Fields:

- Reliability: int: 'Reliability indicator'
- Nominal_Power: float: Average power during the carrier-on state
- No_Of_Exceptions: int: Number of exceptions (channels ±3, ±4 ... with an ACP above the Exception PTx threshold)

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:SACP:LENergy:LRANge
value: CalculateStruct = driver.bluetooth.measurement.multiEval.sacp.lowEnergy.
↪lrange.calculate()
```

Returns the Spectrum ACP results for LE 2M PHY (...:LE2M...) and LE coded PHY (...:LRANge...) . See 'Square Spectrum ACP'. The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → ResultData

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:MEValuation:SACP:LENergy:LRANge
value: ResultData = driver.bluetooth.measurement.multiEval.sacp.lowEnergy.
↪lrange.fetch()
```

Returns the Spectrum ACP results for LE 2M PHY (...:LE2M...) and LE coded PHY (...:LRANge...) . See 'Square Spectrum ACP'. The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for ResultData structure arguments.

read() → ResultData

```
# SCPI: READ:BLUetooth:MEASurement<Instance>:MEValuation:SACP:LENergy:LRANge
value: ResultData = driver.bluetooth.measurement.multiEval.sacp.lowEnergy.
↪lrange.read()
```

Returns the Spectrum ACP results for LE 2M PHY (...:LE2M...) and LE coded PHY (...:LRANge...) . See 'Square Spectrum ACP'. The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for ResultData structure arguments.

6.1.1.6.6.7 Nmode

class NmodeCls

Nmode commands group definition. 12 total commands, 2 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.bluetooth.measurement.multiEval.sacp.nmode.clone()
```

Subgroups

6.1.1.6.6.8 Classic

SCPI Commands :

```
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:SACP:NMODe:CLASsic
READ:BLUetooth:MEASurement<Instance>:MEValuation:SACP:NMODe:CLASsic
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:SACP:NMODe:CLASsic
```

class ClassicCls

Classic commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float or bool: No parameter help available
- Nominal_Power: float or bool: No parameter help available
- No_Of_Exceptions: float or bool: No parameter help available
- Acp: List[float or bool]: No parameter help available

class ResultData

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float: No parameter help available
- Nominal_Power: float: No parameter help available
- No_Of_Exceptions: int: No parameter help available
- Acp: List[float]: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:SACP:NMODe:CLASsic
value: CalculateStruct = driver.bluetooth.measurement.multiEval.sacp.nmode.
↪classic.calculate()
```

No command help available

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → ResultData

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:MEValuation:SACP:NMODe:CLASsic
value: ResultData = driver.bluetooth.measurement.multiEval.sacp.nmode.classic.
↪fetch()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

read() → ResultData

```
# SCPI: READ:BLUetooth:MEASurement<Instance>:MEValuation:SACP:NMODe:CLASsic
value: ResultData = driver.bluetooth.measurement.multiEval.sacp.nmode.classic.
↪read()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

6.1.1.6.6.9 LowEnergy

class LowEnergyCls

LowEnergy commands group definition. 9 total commands, 3 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.bluetooth.measurement.multiEval.sacp.nmode.lowEnergy.clone()
```

Subgroups

6.1.1.6.6.10 Le1M

SCPI Commands :

```
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:SACP:NMODe:LENergy[:LE1M]
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:SACP:NMODe:LENergy[:LE1M]
READ:BLUetooth:MEASurement<Instance>:MEValuation:SACP:NMODe:LENergy[:LE1M]
```

class Le1MCls

Le1M commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Nominal_Power: float or bool: No parameter help available
- No_Of_Exceptions: float or bool: No parameter help available

class ResultData

Response structure. Fields:

- Reliability: int: No parameter help available
- Nominal_Power: float: No parameter help available
- No_Of_Exceptions: int: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:SACP:NMODe:LENergy[:LE1M]
value: CalculateStruct = driver.bluetooth.measurement.multiEval.sacp.nmode.
↪lowEnergy.le1M.calculate()
```

No command help available

return

structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → ResultData

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:MEValuation:SACP:NMODe:LENergy[:LE1M]
value: ResultData = driver.bluetooth.measurement.multiEval.sacp.nmode.lowEnergy.
↪le1M.fetch()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

read() → ResultData

```
# SCPI: READ:BLUetooth:MEASurement<Instance>
↪:MEValuation:SACP:NMODe:LENergy[:LE1M]
value: ResultData = driver.bluetooth.measurement.multiEval.sacp.nmode.lowEnergy.
↪le1M.read()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

6.1.1.6.6.11 Le2M

SCPI Commands :

```
READ:BLUetooth:MEASurement<Instance>:MEValuation:SACP:NMODe:LENergy:LE2M
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:SACP:NMODe:LENergy:LE2M
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:SACP:NMODe:LENergy:LE2M
```

class Le2MCls

Le2M commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Nominal_Power: float or bool: No parameter help available
- No_Of_Exceptions: float or bool: No parameter help available

class ResultData

Response structure. Fields:

- Reliability: int: No parameter help available
- Nominal_Power: float: No parameter help available
- No_Of_Exceptions: int: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:SACP:NMODe:LENergy:LE2M
value: CalculateStruct = driver.bluetooth.measurement.multiEval.sacp.nmode.
↪lowEnergy.le2M.calculate()
```

No command help available

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → ResultData

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:MEValuation:SACP:NMODe:LENergy:LE2M
value: ResultData = driver.bluetooth.measurement.multiEval.sacp.nmode.lowEnergy.
↪le2M.fetch()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

read() → ResultData

```
# SCPI: READ:BLUetooth:MEASurement<Instance>:MEValuation:SACP:NMODe:LENergy:LE2M
value: ResultData = driver.bluetooth.measurement.multiEval.sacp.nmode.lowEnergy.
↪le2M.read()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

6.1.1.6.6.12 Lrange

SCPI Commands :

```
READ:BLUetooth:MEASurement<Instance>:MEValuation:SACP:NMODe:LENergy:LRANge
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:SACP:NMODe:LENergy:LRANge
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:SACP:NMODe:LENergy:LRANge
```

class LrangeCls

Lrange commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Nominal_Power: float or bool: No parameter help available
- No_Of_Exceptions: float or bool: No parameter help available

class ResultData

Response structure. Fields:

- Reliability: int: No parameter help available

- Nominal_Power: float: No parameter help available
- No_Of_Exceptions: int: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:SACP:NMODe:LENergy:LRANge
value: CalculateStruct = driver.bluetooth.measurement.multiEval.sacp.nmode.
↪lowEnergy.lrange.calculate()
```

No command help available

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → ResultData

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:MEValuation:SACP:NMODe:LENergy:LRANge
value: ResultData = driver.bluetooth.measurement.multiEval.sacp.nmode.lowEnergy.
↪lrange.fetch()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

read() → ResultData

```
# SCPI: READ:BLUetooth:MEASurement<Instance>
↪:MEValuation:SACP:NMODe:LENergy:LRANge
value: ResultData = driver.bluetooth.measurement.multiEval.sacp.nmode.lowEnergy.
↪lrange.read()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

6.1.1.6.6.13 Qhsl

class QhslCls

Qhsl commands group definition. 15 total commands, 5 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.bluetooth.measurement.multiEval.sacp.qhsl.clone()
```

Subgroups

6.1.1.6.6.14 P2Q

SCPI Commands :

```
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:SACP:QHSL:P2Q
READ:BLUetooth:MEASurement<Instance>:MEValuation:SACP:QHSL:P2Q
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:SACP:QHSL:P2Q
```

class P2QCls

P2Q commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Nominal_Power: float or bool: No parameter help available
- No_Of_Exceptions: float or bool: No parameter help available

class ResultData

Response structure. Fields:

- Reliability: int: No parameter help available
- Nominal_Power: float: No parameter help available
- No_Of_Exceptions: int: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:SACP:QHSL:P2Q
value: CalculateStruct = driver.bluetooth.measurement.multiEval.sacp.qhsl.p2Q.
↪calculate()
```

No command help available

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → ResultData

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:MEValuation:SACP:QHSL:P2Q
value: ResultData = driver.bluetooth.measurement.multiEval.sacp.qhsl.p2Q.fetch()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

read() → ResultData

```
# SCPI: READ:BLUetooth:MEASurement<Instance>:MEValuation:SACP:QHSL:P2Q
value: ResultData = driver.bluetooth.measurement.multiEval.sacp.qhsl.p2Q.read()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

6.1.1.6.6.15 P3Q

SCPI Commands :

```
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:SACP:QHSL:P3Q
READ:BLUetooth:MEASurement<Instance>:MEValuation:SACP:QHSL:P3Q
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:SACP:QHSL:P3Q
```

class P3QCls

P3Q commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Nominal_Power: float or bool: No parameter help available
- No_Of_Exceptions: float or bool: No parameter help available

class ResultData

Response structure. Fields:

- Reliability: int: No parameter help available
- Nominal_Power: float: No parameter help available
- No_Of_Exceptions: int: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:SACP:QHSL:P3Q
value: CalculateStruct = driver.bluetooth.measurement.multiEval.sacp.qhsl.p3Q.
↪calculate()
```

No command help available

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → ResultData

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:MEValuation:SACP:QHSL:P3Q
value: ResultData = driver.bluetooth.measurement.multiEval.sacp.qhsl.p3Q.fetch()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

read() → ResultData

```
# SCPI: READ:BLUetooth:MEASurement<Instance>:MEValuation:SACP:QHSL:P3Q
value: ResultData = driver.bluetooth.measurement.multiEval.sacp.qhsl.p3Q.read()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

6.1.1.6.6.16 P4Q

SCPI Commands :

```
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:SACP:QHSL:P4Q
READ:BLUetooth:MEASurement<Instance>:MEValuation:SACP:QHSL:P4Q
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:SACP:QHSL:P4Q
```

class P4QCls

P4Q commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Nominal_Power: float or bool: No parameter help available
- No_Of_Exceptions: float or bool: No parameter help available

class ResultData

Response structure. Fields:

- Reliability: int: No parameter help available
- Nominal_Power: float: No parameter help available
- No_Of_Exceptions: int: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:SACP:QHSL:P4Q
value: CalculateStruct = driver.bluetooth.measurement.multiEval.sacp.qhsl.p4Q.
↪calculate()
```

No command help available

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → ResultData

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:MEValuation:SACP:QHSL:P4Q
value: ResultData = driver.bluetooth.measurement.multiEval.sacp.qhsl.p4Q.fetch()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

read() → ResultData

```
# SCPI: READ:BLUetooth:MEASurement<Instance>:MEValuation:SACP:QHSL:P4Q
value: ResultData = driver.bluetooth.measurement.multiEval.sacp.qhsl.p4Q.read()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

6.1.1.6.6.17 P5Q

SCPI Commands :

```
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:SACP:QHSL:P5Q
READ:BLUetooth:MEASurement<Instance>:MEValuation:SACP:QHSL:P5Q
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:SACP:QHSL:P5Q
```

class P5QCls

P5Q commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Nominal_Power: float or bool: No parameter help available
- No_Of_Exceptions: float or bool: No parameter help available

class ResultData

Response structure. Fields:

- Reliability: int: No parameter help available
- Nominal_Power: float: No parameter help available
- No_Of_Exceptions: int: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:SACP:QHSL:P5Q
value: CalculateStruct = driver.bluetooth.measurement.multiEval.sacp.qhsl.p5Q.
↪calculate()
```

No command help available

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → ResultData

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:MEValuation:SACP:QHSL:P5Q
value: ResultData = driver.bluetooth.measurement.multiEval.sacp.qhsl.p5Q.fetch()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

read() → ResultData

```
# SCPI: READ:BLUetooth:MEASurement<Instance>:MEValuation:SACP:QHSL:P5Q
value: ResultData = driver.bluetooth.measurement.multiEval.sacp.qhsl.p5Q.read()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

6.1.1.6.6.18 P6Q

SCPI Commands :

```
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:SACP:QHSL:P6Q
READ:BLUetooth:MEASurement<Instance>:MEValuation:SACP:QHSL:P6Q
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:SACP:QHSL:P6Q
```

class P6QCls

P6Q commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Nominal_Power: float or bool: No parameter help available
- No_Of_Exceptions: float or bool: No parameter help available

class ResultData

Response structure. Fields:

- Reliability: int: No parameter help available
- Nominal_Power: float: No parameter help available
- No_Of_Exceptions: int: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:SACP:QHSL:P6Q
value: CalculateStruct = driver.bluetooth.measurement.multiEval.sacp.qhsl.p6Q.
↪calculate()
```

No command help available

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → ResultData

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:MEValuation:SACP:QHSL:P6Q
value: ResultData = driver.bluetooth.measurement.multiEval.sacp.qhsl.p6Q.fetch()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

read() → ResultData

```
# SCPI: READ:BLUetooth:MEASurement<Instance>:MEValuation:SACP:QHSL:P6Q
value: ResultData = driver.bluetooth.measurement.multiEval.sacp.qhsl.p6Q.read()
```

No command help available

return
structure: for return value, see the help for ResultData structure arguments.

6.1.1.6.7 Sgacp

class SgacpCls

Sgacp commands group definition. 3 total commands, 1 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.bluetooth.measurement.multiEval.sgacp.clone()
```

Subgroups

6.1.1.6.7.1 Edrate

class EdrateCls

Edrate commands group definition. 3 total commands, 1 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.bluetooth.measurement.multiEval.sgacp.edrate.clone()
```

Subgroups

6.1.1.6.7.2 Ptx

SCPI Commands :

```
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:SGACp:EDRate[:PTX]
READ:BLUetooth:MEASurement<Instance>:MEValuation:SGACp:EDRate[:PTX]
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:SGACp:EDRate[:PTX]
```

class PtxCls

Ptx commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: 'Reliability indicator'
- Out_Of_Tol: float or bool: Out of tolerance result, i.e. the percentage of measurement intervals of the statistic count ([CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Scount#Sgacp CMDLINKRESOLVED]) exceeding the specified limits.

- Nominal_Power: float or bool: Average power during the carrier-on state
- No_Of_Exceptions: float or bool: Number of exceptions (channels ±3, ±4 ... with an ACP above the Exception PTx threshold)
- Ptx_Ref: float or bool: Reference power PTXref, measured in the center channel
- Ptx_N_26_Ch_N_1_Abs: float or bool: No parameter help available
- Ptx_N_26_Ch_P_1_Abs: float or bool: No parameter help available
- Ptx_N_26_Ch_N_1_Rel: float or bool: No parameter help available
- Ptx_N_26_Ch_P_1_Rel: float or bool: No parameter help available

class ResultData

Response structure. Fields:

- Reliability: int: 'Reliability indicator'
- Out_Of_Tol: float: Out of tolerance result, i.e. the percentage of measurement intervals of the statistic count ([CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Scount#Sgacp CMDLINKRESOLVED]) exceeding the specified limits.
- Nominal_Power: float: Average power during the carrier-on state
- No_Of_Exceptions: int: Number of exceptions (channels ±3, ±4 ... with an ACP above the Exception PTx threshold)
- Ptx_Ref: float: Reference power PTXref, measured in the center channel
- Ptx_N_26_Ch_N_1_Abs: float: No parameter help available
- Ptx_N_26_Ch_P_1_Abs: float: No parameter help available
- Ptx_N_26_Ch_N_1_Rel: float: No parameter help available
- Ptx_N_26_Ch_P_1_Rel: float: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:SGACp:EDRate[:PTX]
value: CalculateStruct = driver.bluetooth.measurement.multiEval.sgacp.edrate.
↪ptx.calculate()
```

Returns the Spectrum Gated ACP results for EDR packets (single values) . The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → ResultData

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:MEValuation:SGACp:EDRate[:PTX]
value: ResultData = driver.bluetooth.measurement.multiEval.sgacp.edrate.ptx.
↪fetch()
```

Returns the Spectrum Gated ACP results for EDR packets (single values) . The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for ResultData structure arguments.

read() → ResultData

```
# SCPI: READ:BLUetooth:MEASurement<Instance>:MEValuation:SGACp:EDRate[:PTX]
value: ResultData = driver.bluetooth.measurement.multiEval.sgacp.edrate.ptx.
↪read()
```

Returns the Spectrum Gated ACP results for EDR packets (single values) . The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for ResultData structure arguments.

6.1.1.6.8 SoBw

class SoBwCls

SoBw commands group definition. 3 total commands, 1 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.bluetooth.measurement.multiEval.soBw.clone()
```

Subgroups

6.1.1.6.8.1 Brate

class BrateCls

Brate commands group definition. 3 total commands, 1 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.bluetooth.measurement.multiEval.soBw.brate.clone()
```

Subgroups

6.1.1.6.8.2 Maximum

SCPI Commands :

```
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:SOBW:BRATe:MAXimum
READ:BLUetooth:MEASurement<Instance>:MEValuation:SOBW:BRATe:MAXimum
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:SOBW:BRATe:MAXimum
```

class MaximumCls

Maximum commands group definition. 3 total commands, 0 Subgroups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: 'Reliability indicator'
- Out_Of_Tol: float or bool: Out of tolerance result, i.e. the percentage of measurement intervals of the statistic count ([CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Scount#SoBw CMDLINKRESOLVED]) exceeding the specified limit.
- Nominal_Power: float or bool: Average power during the carrier-on state.
- Peak_Emission: float or bool: Peak power in the measured spectral range.
- Fl: float or bool: Lower frequency where the transmit power drops 20 dB below the peak emission.
- Fh: float or bool: Higher frequency where the transmit power drops 20 dB below the peak emission.
- Fh_Min_Fl: float or bool: 20 dB bandwidth; difference between fH – fL.

class FetchStruct

Response structure. Fields:

- Reliability: int: 'Reliability indicator'
- Out_Of_Tol: float: Out of tolerance result, i.e. the percentage of measurement intervals of the statistic count ([CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Scount#SoBw CMDLINKRESOLVED]) exceeding the specified limit.
- Nominal_Power: float: Average power during the carrier-on state.
- Peak_Emission: float: Peak power in the measured spectral range.
- Fl: float: Lower frequency where the transmit power drops 20 dB below the peak emission.
- Fh: float: Higher frequency where the transmit power drops 20 dB below the peak emission.
- Fh_Min_Fl: float: 20 dB bandwidth; difference between fH – fL.

class ReadStruct

Response structure. Fields:

- Reliability: int: 'Reliability indicator'
- Out_Of_Tol: float or bool: Out of tolerance result, i.e. the percentage of measurement intervals of the statistic count ([CMDLINKRESOLVED Configure.Bluetooth.Measurement.MultiEval.Scount#SoBw CMDLINKRESOLVED]) exceeding the specified limit.
- Nominal_Power: float: Average power during the carrier-on state.
- Peak_Emission: float: Peak power in the measured spectral range.
- Fl: float: Lower frequency where the transmit power drops 20 dB below the peak emission.
- Fh: float: Higher frequency where the transmit power drops 20 dB below the peak emission.
- Fh_Min_Fl: float: 20 dB bandwidth; difference between fH – fL.

calculate() → CalculateStruct

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:SOBW:BRATe:MAXimum
value: CalculateStruct = driver.bluetooth.measurement.multiEval.soBw.brate.
↪maximum.calculate()
```

Returns the Spectrum 20 dB Bandwidth results. The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for CalculateStruct structure arguments.

fetch() → FetchStruct

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:MEValuation:SOBW:BRATe:MAXimum
value: FetchStruct = driver.bluetooth.measurement.multiEval.soBw.brate.maximum.
↪fetch()
```

Returns the Spectrum 20 dB Bandwidth results. The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for FetchStruct structure arguments.

read() → ReadStruct

```
# SCPI: READ:BLUetooth:MEASurement<Instance>:MEValuation:SOBW:BRATe:MAXimum
value: ReadStruct = driver.bluetooth.measurement.multiEval.soBw.brate.maximum.
↪read()
```

Returns the Spectrum 20 dB Bandwidth results. The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

return
structure: for return value, see the help for ReadStruct structure arguments.

6.1.1.6.9 State

SCPI Command :

```
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:STATe
```

class StateCls

State commands group definition. 2 total commands, 1 Subgroups, 1 group commands

fetch(*timeout: float = None*, *target_main_state: ResourceState = None*, *target_sync_state: SyncState = None*) → ResourceState

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:MEValuation:STATe
value: enums.ResourceState = driver.bluetooth.measurement.multiEval.state.
↪fetch(timeout = 1.0, target_main_state = enums.ResourceState.ACTive, target_
↪sync_state = enums.SyncState.ADJusted)
```

Queries the main measurement state. Without query parameters, the state is returned immediately. With query parameters, the state is returned when the <TargetMainState> and the <TargetSyncState> are reached or when the <Timeout> expires.

param timeout
No help available

param target_main_state
Target MainState for the query Default is RUN.

param target_sync_state
Target SyncState for the query Default is ADJ.

return
meas_state: Current state or target state of ongoing state transition OFF: measurement off RUN: measurement running RDY: measurement completed

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.bluetooth.measurement.multiEval.state.clone()
```

Subgroups

6.1.1.6.9.1 All

SCPI Command :

```
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:STATe:ALL
```

class AllCls

All commands group definition. 1 total commands, 0 Subgroups, 1 group commands

class FetchStruct

Response structure. Fields:

- Main_State: enums.ResourceState: Current state or target state of ongoing state transition OFF: measurement off RUN: measurement running RDY: measurement completed
- Sync_State: enums.ResourceState: PEND: transition to MainState ongoing ADJ: MainState reached
- Resource_State: enums.ResourceState: QUE: waiting for resource allocation ACT: resources allocated INV: no resources allocated

fetch(*timeout: float = None*, *target_main_state: ResourceState = None*, *target_sync_state: SyncState = None*) → FetchStruct

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:MEValuation:STATe:ALL
value: FetchStruct = driver.bluetooth.measurement.multiEval.state.all.
↪fetch(timeout = 1.0, target_main_state = enums.ResourceState.ACTive, target_
↪sync_state = enums.SyncState.ADJusted)
```

Queries the main measurement state and the measurement substates. Without query parameters, the states are returned immediately. With query parameters, the states are returned when the <TargetMainState> and the <TargetSyncState> are reached or when the <Timeout> expires.

param timeout
No help available

param target_main_state
Target MainState for the query Default is RUN.

param target_sync_state
Target SyncState for the query Default is ADJ.

return
structure: for return value, see the help for FetchStruct structure arguments.

6.1.1.6.10 Trace

class TraceCls

Trace commands group definition. 72 total commands, 13 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.bluetooth.measurement.multiEval.trace.clone()
```

Subgroups

6.1.1.6.10.1 DevMagnitude

class DevMagnitudeCls

DevMagnitude commands group definition. 6 total commands, 3 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.bluetooth.measurement.multiEval.trace.devMagnitude.clone()
```

Subgroups

6.1.1.6.10.2 Average

SCPI Commands :

```
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:TRACe:DEVMagnitude:AVERage
READ:BLUetooth:MEASurement<Instance>:MEValuation:TRACe:DEVMagnitude:AVERage
```

class AverageCls

Average commands group definition. 2 total commands, 0 Subgroups, 2 group commands

fetch() → List[float]

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:MEValuation:TRACe:DEVMagnitude:AVERage
value: List[float] = driver.bluetooth.measurement.multiEval.trace.devMagnitude.
↪average.fetch()
```

Returns the values of the DEVM traces. The results of the current, average minimum and maximum traces can be retrieved. The DEVM traces are available for EDR bursts (method RsCMPX_BluetoothMeas.Configure.Bluetooth.Measurement.InputSignal. btype EDR) .

Suppressed linked return values: reliability

return
devm: N DEVM results, depending on the packet type and payload length.

read() → List[float]

```
# SCPI: READ:BLUetooth:MEASurement<Instance>
↪:MEValuation:TRACe:DEVMagnitude:AVERage
value: List[float] = driver.bluetooth.measurement.multiEval.trace.devMagnitude.
↪average.read()
```

Returns the values of the DEVM traces. The results of the current, average minimum and maximum traces can be retrieved. The DEVM traces are available for EDR bursts (method RsCMPX_BluetoothMeas.Configure.Bluetooth.Measurement.InputSignal. btype EDR) .

Suppressed linked return values: reliability

return
devm: N DEVM results, depending on the packet type and payload length.

6.1.1.6.10.3 Current

SCPI Commands :

```
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:TRACe:DEVMagnitude:CURRent
READ:BLUetooth:MEASurement<Instance>:MEValuation:TRACe:DEVMagnitude:CURRent
```

class CurrentCls

Current commands group definition. 2 total commands, 0 Subgroups, 2 group commands

fetch() → List[float]

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:MEValuation:TRACe:DEVMagnitude:CURRent
value: List[float] = driver.bluetooth.measurement.multiEval.trace.devMagnitude.
↪current.fetch()
```

Returns the values of the DEVM traces. The results of the current, average minimum and maximum traces can be retrieved. The DEVM traces are available for EDR bursts (method RsCMPX_BluetoothMeas.Configure.Bluetooth.Measurement.InputSignal. btype EDR) .

Suppressed linked return values: reliability

return
devm: N DEVM results, depending on the packet type and payload length.

read() → List[float]

```
# SCPI: READ:BLUetooth:MEASurement<Instance>
↪:MEValuation:TRACe:DEVMagnitude:CURRent
value: List[float] = driver.bluetooth.measurement.multiEval.trace.devMagnitude.
↪current.read()
```

Returns the values of the DEVM traces. The results of the current, average minimum and maximum traces can be retrieved. The DEVM traces are available for EDR bursts (method RsCMPX_BluetoothMeas.Configure.Bluetooth.Measurement.InputSignal. btype EDR) .

Suppressed linked return values: reliability

return
devm: N DEVM results, depending on the packet type and payload length.

6.1.1.6.10.4 Maximum

SCPI Commands :

```
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:TRACe:DEVMagnitude:MAXimum
READ:BLUetooth:MEASurement<Instance>:MEValuation:TRACe:DEVMagnitude:MAXimum
```

class MaximumCls

Maximum commands group definition. 2 total commands, 0 Subgroups, 2 group commands

fetch() → List[float]

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:MEValuation:TRACe:DEVMagnitude:MAXimum
value: List[float] = driver.bluetooth.measurement.multiEval.trace.devMagnitude.
↪maximum.fetch()
```

Returns the values of the DEVM traces. The results of the current, average minimum and maximum traces can be retrieved. The DEVM traces are available for EDR bursts (method RsCMPX_BluetoothMeas.Configure.Bluetooth.Measurement.InputSignal. btype EDR) .

Suppressed linked return values: reliability

return
devm: N DEVM results, depending on the packet type and payload length.

read() → List[float]

```
# SCPI: READ:BLUetooth:MEASurement<Instance>
↪:MEValuation:TRACe:DEVMagnitude:MAXimum
value: List[float] = driver.bluetooth.measurement.multiEval.trace.devMagnitude.
↪maximum.read()
```

Returns the values of the DEVM traces. The results of the current, average minimum and maximum traces can be retrieved. The DEVM traces are available for EDR bursts (method RsCMPX_BluetoothMeas.Configure.Bluetooth.Measurement.InputSignal. btype EDR) .

Suppressed linked return values: reliability

return
devm: N DEVM results, depending on the packet type and payload length.

6.1.1.6.10.5 Fdeviation

class FdeviationCls

Fdeviation commands group definition. 8 total commands, 4 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.bluetooth.measurement.multiEval.trace.fdeviation.clone()
```

Subgroups

6.1.1.6.10.6 Average

SCPI Commands :

```
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:TRACe:FDEViation:AVERage
READ:BLUetooth:MEASurement<Instance>:MEValuation:TRACe:FDEViation:AVERage
```

class AverageCls

Average commands group definition. 2 total commands, 0 Subgroups, 2 group commands

fetch() → List[float]

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:MEValuation:TRACe:FDEViation:AVERage
value: List[float] = driver.bluetooth.measurement.multiEval.trace.fdeviation.
↪average.fetch()
```

Returns the values of the frequency deviation traces. The results of the current, average minimum and maximum traces can be retrieved. See also 'PvT and modulation trace points (LE) '

Suppressed linked return values: reliability

return
freq_deviation: m frequency deviation results, depending on the packet type and payload length

read() → List[float]

```
# SCPI: READ:BLUetooth:MEASurement<Instance>
↪:MEValuation:TRACe:FDEViation:AVERage
value: List[float] = driver.bluetooth.measurement.multiEval.trace.fdeviation.
↪average.read()
```

Returns the values of the frequency deviation traces. The results of the current, average minimum and maximum traces can be retrieved. See also 'PvT and modulation trace points (LE) '

Suppressed linked return values: reliability

return
freq_deviation: m frequency deviation results, depending on the packet type and payload length

6.1.1.6.10.7 Current

SCPI Commands :

```
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:TRACe:FDEViation:CURRent
READ:BLUetooth:MEASurement<Instance>:MEValuation:TRACe:FDEViation:CURRent
```

class CurrentCls

Current commands group definition. 2 total commands, 0 Subgroups, 2 group commands

fetch() → List[float]

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:MEValuation:TRACe:FDEViation:CURRent
value: List[float] = driver.bluetooth.measurement.multiEval.trace.fdeviation.
↪current.fetch()
```

Returns the values of the frequency deviation traces. The results of the current, average minimum and maximum traces can be retrieved. See also 'PvT and modulation trace points (LE) '

Suppressed linked return values: reliability

return
freq_deviation: m frequency deviation results, depending on the packet type and payload length

read() → List[float]

```
# SCPI: READ:BLUetooth:MEASurement<Instance>
↪:MEValuation:TRACe:FDEViation:CURRent
value: List[float] = driver.bluetooth.measurement.multiEval.trace.fdeviation.
↪current.read()
```

Returns the values of the frequency deviation traces. The results of the current, average minimum and maximum traces can be retrieved. See also 'PvT and modulation trace points (LE) '

Suppressed linked return values: reliability

return
freq_deviation: m frequency deviation results, depending on the packet type and payload length

6.1.1.6.10.8 Maximum

SCPI Commands :

```
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:TRACe:FDEViation:MAXimum
READ:BLUetooth:MEASurement<Instance>:MEValuation:TRACe:FDEViation:MAXimum
```

class MaximumCls

Maximum commands group definition. 2 total commands, 0 Subgroups, 2 group commands

fetch() → List[float]

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:MEValuation:TRACe:FDEViation:MAXimum
value: List[float] = driver.bluetooth.measurement.multiEval.trace.fdeviation.
↪maximum.fetch()
```

Returns the values of the frequency deviation traces. The results of the current, average minimum and maximum traces can be retrieved. See also 'PvT and modulation trace points (LE) '

Suppressed linked return values: reliability

return
freq_deviation: m frequency deviation results, depending on the packet type and payload length

read() → List[float]

```
# SCPI: READ:BLUetooth:MEASurement<Instance>
↪:MEValuation:TRACe:FDEViation:MAXimum
value: List[float] = driver.bluetooth.measurement.multiEval.trace.fdeviation.
↪maximum.read()
```

Returns the values of the frequency deviation traces. The results of the current, average minimum and maximum traces can be retrieved. See also 'PvT and modulation trace points (LE) '

Suppressed linked return values: reliability

return
freq_deviation: m frequency deviation results, depending on the packet type and payload length

6.1.1.6.10.9 Minimum

SCPI Commands :

```
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:TRACe:FDEViation:MINimum
READ:BLUetooth:MEASurement<Instance>:MEValuation:TRACe:FDEViation:MINimum
```

class MinimumCls

Minimum commands group definition. 2 total commands, 0 Subgroups, 2 group commands

fetch() → List[float]

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:MEValuation:TRACe:FDEViation:MINimum
value: List[float] = driver.bluetooth.measurement.multiEval.trace.fdeviation.
↪minimum.fetch()
```

Returns the values of the frequency deviation traces. The results of the current, average minimum and maximum traces can be retrieved. See also 'PvT and modulation trace points (LE) '

Suppressed linked return values: reliability

return
freq_deviation: m frequency deviation results, depending on the packet type and payload length

read() → List[float]

```
# SCPI: READ:BLUetooth:MEASurement<Instance>
↪:MEValuation:TRACe:FDEViation:MINimum
value: List[float] = driver.bluetooth.measurement.multiEval.trace.fdeviation.
↪minimum.read()
```

Returns the values of the frequency deviation traces. The results of the current, average minimum and maximum traces can be retrieved. See also 'PvT and modulation trace points (LE) '

Suppressed linked return values: reliability

return
freq_deviation: m frequency deviation results, depending on the packet type and payload length

6.1.1.6.10.10 Frange

class FrangeCls

Frange commands group definition. 2 total commands, 1 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.bluetooth.measurement.multiEval.trace.frange.clone()
```

Subgroups

6.1.1.6.10.11 Average

SCPI Commands :

```
READ:BLUetooth:MEASurement<Instance>:MEValuation:TRACe:FRANge:AVERage
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:TRACe:FRANge:AVERage
```

class AverageCls

Average commands group definition. 2 total commands, 0 Subgroups, 2 group commands

fetch() → List[float]

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:MEValuation:TRACe:FRANge:AVERage
value: List[float] = driver.bluetooth.measurement.multiEval.trace.frange.
↪average.fetch()
```

Returns the average values of the Frequency Range trace. The values are available for BR packets.

Suppressed linked return values: reliability

return
frequency_range: RX signal level measured at the frequencies between 501 pixels (502 values)

read() → List[float]

```
# SCPI: READ:BLUetooth:MEASurement<Instance>:MEValuation:TRACe:FRANge:AVERage
value: List[float] = driver.bluetooth.measurement.multiEval.trace.frange.
↪average.read()
```

Returns the average values of the Frequency Range trace. The values are available for BR packets.

Suppressed linked return values: reliability

return
frequency_range: RX signal level measured at the frequencies between 501 pixels (502 values)

6.1.1.6.10.12 IqAbs

SCPI Commands :

```
READ:BLUetooth:MEASurement<Instance>:MEValuation:TRACe:IQABs
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:TRACe:IQABs
```

class IqAbsCls

IqAbs commands group definition. 2 total commands, 0 Subgroups, 2 group commands

class ResultData

Response structure. Fields:

- Reliability: int: 'Reliability indicator'
- Iphase: List[float]: N in-phase amplitudes (IPhase) and N quadrature-phase (QPhase) amplitudes, where N is equal to the number of processed 50-symbol blocks; see 'I/Q constellation trace points (EDR) '.
- Qphase: List[float]: No parameter help available

fetch() → ResultData

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:MEValuation:TRACe:IQABs
value: ResultData = driver.bluetooth.measurement.multiEval.trace.iqAbs.fetch()
```

Returns the values of the traces in the I/Q constellation diagrams. The mnemonics IQABs, IQDiff, and IQERr denote the absolute, differential and I/Q constellation error results. The I/Q traces are available for EDR packets (method RsCMPX_BluetoothMeas.Configure.Bluetooth.Measurement.InputSignal.btype EDR) .

return
structure: for return value, see the help for ResultData structure arguments.

read() → ResultData

```
# SCPI: READ:BLUetooth:MEASurement<Instance>:MEValuation:TRACe:IQABs
value: ResultData = driver.bluetooth.measurement.multiEval.trace.iqAbs.read()
```

Returns the values of the traces in the I/Q constellation diagrams. The mnemonics IQABs, IQDiff, and IQERr denote the absolute, differential and I/Q constellation error results. The I/Q traces are available for EDR packets (method RsCMPX_BluetoothMeas.Configure.Bluetooth.Measurement.InputSignal.btype EDR) .

return
structure: for return value, see the help for ResultData structure arguments.

6.1.1.6.10.13 IqDifference

SCPI Commands :

```
READ:BLUetooth:MEASurement<Instance>:MEValuation:TRACe:IQDiff
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:TRACe:IQDiff
```

class IqDifferenceCls

IqDifference commands group definition. 2 total commands, 0 Subgroups, 2 group commands

class ResultData

Response structure. Fields:

- Reliability: int: 'Reliability indicator'
- Iphase: List[float]: N in-phase amplitudes (IPhase) and N quadrature-phase (QPhase) amplitudes, where N is equal to the number of processed 50-symbol blocks; see 'I/Q constellation trace points (EDR) '.
- Qphase: List[float]: No parameter help available

fetch() → ResultData

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:MEValuation:TRACe:IQDiff
value: ResultData = driver.bluetooth.measurement.multiEval.trace.iqDifference.
↪fetch()
```

Returns the values of the traces in the I/Q constellation diagrams. The mnemonics IQABs, IQDiff, and IQERr denote the absolute, differential and I/Q constellation error results. The I/Q traces are available for EDR packets (method RsCMPX_BluetoothMeas.Configure.Bluetooth.Measurement.InputSignal.btype EDR) .

return
structure: for return value, see the help for ResultData structure arguments.

read() → ResultData

```
# SCPI: READ:BLUetooth:MEASurement<Instance>:MEValuation:TRACe:IQDiff
value: ResultData = driver.bluetooth.measurement.multiEval.trace.iqDifference.
↪read()
```

Returns the values of the traces in the I/Q constellation diagrams. The mnemonics IQABs, IQDiff, and IQERr denote the absolute, differential and I/Q constellation error results. The I/Q traces are available for EDR packets (method RsCMPX_BluetoothMeas.Configure.Bluetooth.Measurement.InputSignal.btype EDR) .

return
structure: for return value, see the help for ResultData structure arguments.

6.1.1.6.10.14 IqError

SCPI Commands :

```
READ:BLUetooth:MEASurement<Instance>:MEValuation:TRACe:IQERr
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:TRACe:IQERr
```

class IqErrorCls

IqError commands group definition. 2 total commands, 0 Subgroups, 2 group commands

class ResultData

Response structure. Fields:

- Reliability: int: 'Reliability indicator'
- Iphase: List[float]: N in-phase amplitudes (IPhase) and N quadrature-phase (QPhase) amplitudes, where N is equal to the number of processed 50-symbol blocks; see 'I/Q constellation trace points (EDR) '.
- Qphase: List[float]: No parameter help available

fetch() → ResultData

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:MEValuation:TRACe:IQERr
value: ResultData = driver.bluetooth.measurement.multiEval.trace.iqError.fetch()
```

Returns the values of the traces in the I/Q constellation diagrams. The mnemonics IQABs, IQDiff, and IQERr denote the absolute, differential and I/Q constellation error results. The I/Q traces are available for EDR packets (method RsCMPX_BluetoothMeas.Configure.Bluetooth.Measurement.InputSignal.btype EDR) .

return
structure: for return value, see the help for ResultData structure arguments.

read() → ResultData

```
# SCPI: READ:BLUetooth:MEASurement<Instance>:MEValuation:TRACe:IQERr
value: ResultData = driver.bluetooth.measurement.multiEval.trace.iqError.read()
```

Returns the values of the traces in the I/Q constellation diagrams. The mnemonics IQABs, IQDiff, and IQERr denote the absolute, differential and I/Q constellation error results. The I/Q traces are available for EDR packets (method RsCMPX_BluetoothMeas.Configure.Bluetooth.Measurement.InputSignal.btype EDR) .

return
structure: for return value, see the help for ResultData structure arguments.

6.1.1.6.10.15 Pdeviation

class PdeviationCls

Pdeviation commands group definition. 6 total commands, 2 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.bluetooth.measurement.multiEval.trace.pdeviation.clone()
```

Subgroups

6.1.1.6.10.16 Maximum

SCPI Commands :

```
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:TRACe:PDEViation:MAXimum
READ:BLUetooth:MEASurement<Instance>:MEValuation:TRACe:PDEViation:MAXimum
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:TRACe:PDEViation:MAXimum
```

class MaximumCls

Maximum commands group definition. 3 total commands, 0 Subgroups, 3 group commands

calculate() → List[float]

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:TRACe:PDEViation:MAXimum
value: List[float or bool] = driver.bluetooth.measurement.multiEval.trace.
↪pdeviation.maximum.calculate()
```

Returns the results of power deviation per slot for LE CTE traces. Deviation value is calculated as the peak-to-average power ratio. The results of the minimum and maximum traces can be retrieved. The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

Suppressed linked return values: reliability

return
power_vs_slot: (float or boolean items) No help available

fetch() → List[float]

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:MEValuation:TRACe:PDEViation:MAXimum
value: List[float] = driver.bluetooth.measurement.multiEval.trace.pdeviation.
↪maximum.fetch()
```

Returns the results of power deviation per slot for LE CTE traces. Deviation value is calculated as the peak-to-average power ratio. The results of the minimum and maximum traces can be retrieved. The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

Suppressed linked return values: reliability

return
power_vs_slot: No help available

read() → List[float]

```
# SCPI: READ:BLUetooth:MEASurement<Instance>
↪:MEValuation:TRACe:PDEViation:MAXimum
value: List[float] = driver.bluetooth.measurement.multiEval.trace.pdeviation.
↪maximum.read()
```

Returns the results of power deviation per slot for LE CTE traces. Deviation value is calculated as the peak-to-average power ratio. The results of the minimum and maximum traces can be retrieved. The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

Suppressed linked return values: reliability

return
power_vs_slot: No help available

6.1.1.6.10.17 Minimum

SCPI Commands :

```
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:TRACe:PDEViation:MINimum
READ:BLUetooth:MEASurement<Instance>:MEValuation:TRACe:PDEViation:MINimum
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:TRACe:PDEViation:MINimum
```

class MinimumCls

Minimum commands group definition. 3 total commands, 0 Subgroups, 3 group commands

calculate() → List[float]

```
# SCPI: CALCulate:BLUetooth:MEASurement<Instance>
↪:MEValuation:TRACe:PDEViation:MINimum
value: List[float or bool] = driver.bluetooth.measurement.multiEval.trace.
↪pdeviation.minimum.calculate()
```

Returns the results of power deviation per slot for LE CTE traces. Deviation value is calculated as the peak-to-average power ratio. The results of the minimum and maximum traces can be retrieved. The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

Suppressed linked return values: reliability

return
power_vs_slot: (float or boolean items) No help available

fetch() → List[float]

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:MEValuation:TRACe:PDEViation:MINimum
value: List[float] = driver.bluetooth.measurement.multiEval.trace.pdeviation.
↪minimum.fetch()
```

Returns the results of power deviation per slot for LE CTE traces. Deviation value is calculated as the peak-to-average power ratio. The results of the minimum and maximum traces can be retrieved. The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

Suppressed linked return values: reliability

return
power_vs_slot: No help available

read() → List[float]

```
# SCPI: READ:BLUetooth:MEASurement<Instance>
↪:MEValuation:TRACe:PDEViation:MINimum
value: List[float] = driver.bluetooth.measurement.multiEval.trace.pdeviation.
↪minimum.read()
```

Returns the results of power deviation per slot for LE CTE traces. Deviation value is calculated as the peak-to-average power ratio. The results of the minimum and maximum traces can be retrieved. The values described below are returned by FETCh and READ commands. CALCulate commands return limit check results instead, one value for each result listed below.

Suppressed linked return values: reliability

return
power_vs_slot: No help available

6.1.1.6.10.18 Pdifference

class PdifferenceCls

Pdifference commands group definition. 6 total commands, 3 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.bluetooth.measurement.multiEval.trace.pdifference.clone()
```

Subgroups

6.1.1.6.10.19 Average

SCPI Commands :

```
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:TRACe:PDIFference:AVERage
READ:BLUetooth:MEASurement<Instance>:MEValuation:TRACe:PDIFference:AVERage
```

class AverageCls

Average commands group definition. 2 total commands, 0 Subgroups, 2 group commands

fetch() → List[float]

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:MEValuation:TRACe:PDIFference:AVERage
value: List[float] = driver.bluetooth.measurement.multiEval.trace.pdifference.
↪average.fetch()
```

Returns the values of the phase difference traces. The results of the current, average minimum and maximum traces can be retrieved. The phase difference traces are available for EDR bursts (method RsCMPX_BluetoothMeas.Configure.Bluetooth. Measurement.InputSignal.btype EDR) .

Suppressed linked return values: reliability

return
phase_difference: N phase difference results, depending on the packet type and payload length; see 'Phase difference trace points (EDR) '.

read() → List[float]

```
# SCPI: READ:BLUetooth:MEASurement<Instance>
↪:MEValuation:TRACe:PDIFference:AVERage
value: List[float] = driver.bluetooth.measurement.multiEval.trace.pdifference.
↪average.read()
```

Returns the values of the phase difference traces. The results of the current, average minimum and maximum traces can be retrieved. The phase difference traces are available for EDR bursts (method RsCMPX_BluetoothMeas.Configure.Bluetooth. Measurement.InputSignal.btype EDR) .

Suppressed linked return values: reliability

return
phase_difference: N phase difference results, depending on the packet type and payload length; see 'Phase difference trace points (EDR) '.

6.1.1.6.10.20 Current

SCPI Commands :

```
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:TRACe:PDIFference:CURRent
READ:BLUetooth:MEASurement<Instance>:MEValuation:TRACe:PDIFference:CURRent
```

class CurrentCls

Current commands group definition. 2 total commands, 0 Subgroups, 2 group commands

fetch() → List[float]

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:MEValuation:TRACe:PDIFference:CURRent
value: List[float] = driver.bluetooth.measurement.multiEval.trace.pdifference.
↪current.fetch()
```

Returns the values of the phase difference traces. The results of the current, average minimum and maximum traces can be retrieved. The phase difference traces are available for EDR bursts (method RsCMPX_BluetoothMeas.Configure.Bluetooth. Measurement.InputSignal.btype EDR) .

Suppressed linked return values: reliability

return
phase_difference: N phase difference results, depending on the packet type and payload length; see 'Phase difference trace points (EDR) '.

read() → List[float]

```
# SCPI: READ:BLUetooth:MEASurement<Instance>
↪:MEValuation:TRACe:PDIFference:CURRent
value: List[float] = driver.bluetooth.measurement.multiEval.trace.pdifference.
↪current.read()
```

Returns the values of the phase difference traces. The results of the current, average minimum and maximum traces can be retrieved. The phase difference traces are available for EDR bursts (method RsCMPX_BluetoothMeas.Configure.Bluetooth. Measurement.InputSignal.btype EDR) .

Suppressed linked return values: reliability

return
phase_difference: N phase difference results, depending on the packet type and payload length; see 'Phase difference trace points (EDR) '.

6.1.1.6.10.21 Maximum

SCPI Commands :

```
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:TRACe:PDIFference:MAXimum
READ:BLUetooth:MEASurement<Instance>:MEValuation:TRACe:PDIFference:MAXimum
```

class MaximumCls

Maximum commands group definition. 2 total commands, 0 Subgroups, 2 group commands

fetch() → List[float]

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>
↪:MEValuation:TRACe:PDIFference:MAXimum
value: List[float] = driver.bluetooth.measurement.multiEval.trace.pdifference.
↪maximum.fetch()
```

Returns the values of the phase difference traces. The results of the current, average minimum and maximum traces can be retrieved. The phase difference traces are available for EDR bursts (method RsCMPX_BluetoothMeas.Configure.Bluetooth. Measurement.InputSignal.btype EDR) .

Suppressed linked return values: reliability

return
phase_difference: N phase difference results, depending on the packet type and payload length; see 'Phase difference trace points (EDR) '.

read() → List[float]

```
# SCPI: READ:BLUetooth:MEASurement<Instance>
↪:MEValuation:TRACe:PDIFference:MAXimum
value: List[float] = driver.bluetooth.measurement.multiEval.trace.pdifference.
↪maximum.read()
```

Returns the values of the phase difference traces. The results of the current, average minimum and maximum traces can be retrieved. The phase difference traces are available for EDR bursts (method RsCMPX_BluetoothMeas.Configure.Bluetooth. Measurement.InputSignal.btype EDR) .

Suppressed linked return values: reliability

return
phase_difference: N phase difference results, depending on the packet type and payload length; see 'Phase difference trace points (EDR) '.

6.1.1.6.10.22 PowerVsTime

class PowerVsTimeCls

PowerVsTime commands group definition. 8 total commands, 4 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.bluetooth.measurement.multiEval.trace.powerVsTime.clone()
```

Subgroups

6.1.1.6.10.23 Average

SCPI Commands :

```
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:TRACe:PVTime:AVERage
READ:BLUetooth:MEASurement<Instance>:MEValuation:TRACe:PVTime:AVERage
```

class AverageCls

Average commands group definition. 2 total commands, 0 Subgroups, 2 group commands

fetch() → List[float]

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:MEValuation:TRACe:PVTime:AVERage
value: List[float] = driver.bluetooth.measurement.multiEval.trace.powerVsTime.
↪average.fetch()
```

Returns the values of the power vs time traces. The results of the current, average minimum and maximum traces can be retrieved.

Suppressed linked return values: reliability

return
power_vs_time: N power results, depending on the packet type and payload length; see 'PvT and modulation trace points (LE) '.

read() → List[float]

```
# SCPI: READ:BLUetooth:MEASurement<Instance>:MEValuation:TRACe:PVTime:AVERage
value: List[float] = driver.bluetooth.measurement.multiEval.trace.powerVsTime.
↪average.read()
```

Returns the values of the power vs time traces. The results of the current, average minimum and maximum traces can be retrieved.

Suppressed linked return values: reliability

return
power_vs_time: N power results, depending on the packet type and payload length; see 'PvT and modulation trace points (LE) '.

6.1.1.6.10.24 Current

SCPI Commands :

```
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:TRACe:PVTime:CURRent
READ:BLUetooth:MEASurement<Instance>:MEValuation:TRACe:PVTime:CURRent
```

class CurrentCls

Current commands group definition. 2 total commands, 0 Subgroups, 2 group commands

fetch() → List[float]

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:MEValuation:TRACe:PVTime:CURRent
value: List[float] = driver.bluetooth.measurement.multiEval.trace.powerVsTime.
↪current.fetch()
```

Returns the values of the power vs time traces. The results of the current, average minimum and maximum traces can be retrieved.

Suppressed linked return values: reliability

return
power_vs_time: N power results, depending on the packet type and payload length; see 'PvT and modulation trace points (LE) '.

read() → List[float]

```
# SCPI: READ:BLUetooth:MEASurement<Instance>:MEValuation:TRACe:PVTime:CURRent
value: List[float] = driver.bluetooth.measurement.multiEval.trace.powerVsTime.
↪current.read()
```

Returns the values of the power vs time traces. The results of the current, average minimum and maximum traces can be retrieved.

Suppressed linked return values: reliability

return
power_vs_time: N power results, depending on the packet type and payload length; see 'PvT and modulation trace points (LE) '.

6.1.1.6.10.25 Maximum

SCPI Commands :

```
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:TRACe:PVTime:MAXimum
READ:BLUetooth:MEASurement<Instance>:MEValuation:TRACe:PVTime:MAXimum
```

class MaximumCls

Maximum commands group definition. 2 total commands, 0 Subgroups, 2 group commands

fetch() → List[float]

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:MEValuation:TRACe:PVTime:MAXimum
value: List[float] = driver.bluetooth.measurement.multiEval.trace.powerVsTime.
↪maximum.fetch()
```

Returns the values of the power vs time traces. The results of the current, average minimum and maximum traces can be retrieved.

Suppressed linked return values: reliability

return
power_vs_time: N power results, depending on the packet type and payload length; see 'PvT and modulation trace points (LE) '.

read() → List[float]

```
# SCPI: READ:BLUetooth:MEASurement<Instance>:MEValuation:TRACe:PVTime:MAXimum
value: List[float] = driver.bluetooth.measurement.multiEval.trace.powerVsTime.
↪maximum.read()
```

Returns the values of the power vs time traces. The results of the current, average minimum and maximum traces can be retrieved.

Suppressed linked return values: reliability

return
power_vs_time: N power results, depending on the packet type and payload length; see 'PvT and modulation trace points (LE) '.

6.1.1.6.10.26 Minimum

SCPI Commands :

```
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:TRACe:PVTime:MINimum
READ:BLUetooth:MEASurement<Instance>:MEValuation:TRACe:PVTime:MINimum
```

class MinimumCls

Minimum commands group definition. 2 total commands, 0 Subgroups, 2 group commands

fetch() → List[float]

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:MEValuation:TRACe:PVTime:MINimum
value: List[float] = driver.bluetooth.measurement.multiEval.trace.powerVsTime.
↪minimum.fetch()
```

Returns the values of the power vs time traces. The results of the current, average minimum and maximum traces can be retrieved.

Suppressed linked return values: reliability

return
power_vs_time: N power results, depending on the packet type and payload length; see 'PvT and modulation trace points (LE) '.

read() → List[float]

```
# SCPI: READ:BLUetooth:MEASurement<Instance>:MEValuation:TRACe:PVTime:MINimum
value: List[float] = driver.bluetooth.measurement.multiEval.trace.powerVsTime.
↪minimum.read()
```

Returns the values of the power vs time traces. The results of the current, average minimum and maximum traces can be retrieved.

Suppressed linked return values: reliability

return
power_vs_time: N power results, depending on the packet type and payload length; see 'PvT and modulation trace points (LE) '.

6.1.1.6.10.27 Sacp

class SacpCls

Sacp commands group definition. 8 total commands, 4 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.bluetooth.measurement.multiEval.trace.sacp.clone()
```

Subgroups

6.1.1.6.10.28 Average

SCPI Commands :

```
READ:BLUetooth:MEASurement<Instance>:MEValuation:TRACe:SACP:AVERage
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:TRACe:SACP:AVERage
```

class AverageCls

Average commands group definition. 2 total commands, 0 Subgroups, 2 group commands

fetch() → List[float]

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:MEValuation:TRACe:SACP:AVERage
value: List[float] = driver.bluetooth.measurement.multiEval.trace.sacp.average.
↪fetch()
```

Returns 81 values of the Spectrum ACP results in line with the Bluetooth test specification.

INTRO_CMD_HELP: The number of valid ACP results depends on the ACP measurement mode (method RsCMPX_BluetoothMeas.Configure.Bluetooth.Measurement.MultiEval.Sacp.LowEnergy.Le2M.Measurement:

- For LE bursts, the trace returns 81 values.

INTRO_CMD_HELP: The number of valid ACP results depends on the ACP measurement mode (method RsCMPX_BluetoothMeas.Configure.Bluetooth.Measurement.MultiEval.Sacp.LowEnergy.Le2M.Measurement:

- In CH10 mode (ACP +/- 5 Channels) , the first 21 ACP values contain results for the 1 MHz channels centered at fTX – 10 MHz, fTX – 9 MHz, …, fTX + 10 MHz. The remaining 58 values are invalid (NAV) . This mode is applicable to all types of LE bursts.
- In CH40 mode (LE All Channels) , ACP values 1 to 81 contain results for the 1 MHz channels centered at 2401 MHz, 2402 MHz, …, 2481 MHz This mode is only applicable to test packets using LE 1M PHY or LE 2M PHY.

Suppressed linked return values: reliability

return
acp: 81 spectrum ACP results

read() → List[float]

```
# SCPI: READ:BLUetooth:MEASurement<Instance>:MEValuation:TRACe:SACP:AVERage
value: List[float] = driver.bluetooth.measurement.multiEval.trace.sacp.average.
↪read()
```

Returns 81 values of the Spectrum ACP results in line with the Bluetooth test specification.

INTRO_CMD_HELP: The number of valid ACP results depends on the ACP measurement mode (method RsCMPX_BluetoothMeas.Configure.Bluetooth.Measurement.MultiEval.Sacp.LowEnergy.Le2M.Measurement:

- For LE bursts, the trace returns 81 values.

INTRO_CMD_HELP: The number of valid ACP results depends on the ACP measurement mode (method RsCMPX_BluetoothMeas.Configure.Bluetooth.Measurement.MultiEval.Sacp.LowEnergy.Le2M.Measurement:

- In CH10 mode (ACP +/- 5 Channels) , the first 21 ACP values contain results for the 1 MHz channels centered at fTX – 10 MHz, fTX – 9 MHz, …, fTX + 10 MHz. The remaining 58 values are invalid (NAV) . This mode is applicable to all types of LE bursts.
- In CH40 mode (LE All Channels) , ACP values 1 to 81 contain results for the 1 MHz channels centered at 2401 MHz, 2402 MHz, …, 2481 MHz This mode is only applicable to test packets using LE 1M PHY or LE 2M PHY.

Suppressed linked return values: reliability

return
acp: 81 spectrum ACP results

6.1.1.6.10.29 Current

SCPI Commands :

```
READ:BLUetooth:MEASurement<Instance>:MEValuation:TRACe:SACP:CURRent
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:TRACe:SACP:CURRent
```

class CurrentCls

Current commands group definition. 2 total commands, 0 Subgroups, 2 group commands

fetch() → List[float]

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:MEValuation:TRACe:SACP:CURRent
value: List[float] = driver.bluetooth.measurement.multiEval.trace.sacp.current.
↪fetch()
```

Returns 81 values of the Spectrum ACP results in line with the Bluetooth test specification.

INTRO_CMD_HELP: The number of valid ACP results depends on the ACP measurement mode (method RsCMPX_BluetoothMeas.Configure.Bluetooth.Measurement.MultiEval.Sacp.LowEnergy.Le2M.Measurement :

- For LE bursts, the trace returns 81 values.

INTRO_CMD_HELP: The number of valid ACP results depends on the ACP measurement mode (method RsCMPX_BluetoothMeas.Configure.Bluetooth.Measurement.MultiEval.Sacp.LowEnergy.Le2M.Measurement :

- In CH10 mode (ACP +/- 5 Channels) , the first 21 ACP values contain results for the 1 MHz channels centered at fTX – 10 MHz, fTX – 9 MHz, …, fTX + 10 MHz. The remaining 58 values are invalid (NAV) . This mode is applicable to all types of LE bursts.
- In CH40 mode (LE All Channels) , ACP values 1 to 81 contain results for the 1 MHz channels centered at 2401 MHz, 2402 MHz, …, 2481 MHz This mode is only applicable to test packets using LE 1M PHY or LE 2M PHY.

Suppressed linked return values: reliability

return
acp: 81 spectrum ACP results

read() → List[float]

```
# SCPI: READ:BLUetooth:MEASurement<Instance>:MEValuation:TRACe:SACP:CURRent
value: List[float] = driver.bluetooth.measurement.multiEval.trace.sacp.current.
↪read()
```

Returns 81 values of the Spectrum ACP results in line with the Bluetooth test specification.

INTRO_CMD_HELP: The number of valid ACP results depends on the ACP measurement mode (method RsCMPX_BluetoothMeas.Configure.Bluetooth.Measurement.MultiEval.Sacp.LowEnergy.Le2M.Measurement :

- For LE bursts, the trace returns 81 values.

INTRO_CMD_HELP: The number of valid ACP results depends on the ACP measurement mode (method RsCMPX_BluetoothMeas.Configure.Bluetooth.Measurement.MultiEval.Sacp.LowEnergy.Le2M.Measurement :

- In CH10 mode (ACP +/- 5 Channels) , the first 21 ACP values contain results for the 1 MHz channels centered at fTX – 10 MHz, fTX – 9 MHz, …, fTX + 10 MHz. The remaining 58 values are invalid (NAV) . This mode is applicable to all types of LE bursts.
- In CH40 mode (LE All Channels) , ACP values 1 to 81 contain results for the 1 MHz channels centered at 2401 MHz, 2402 MHz, …, 2481 MHz This mode is only applicable to test packets using LE 1M PHY or LE 2M PHY.

Suppressed linked return values: reliability

return
acp: 81 spectrum ACP results

6.1.1.6.10.30 Maximum

SCPI Commands :

```
READ:BLUetooth:MEASurement<Instance>:MEValuation:TRACe:SACP:MAXimum
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:TRACe:SACP:MAXimum
```

class MaximumCls

Maximum commands group definition. 2 total commands, 0 Subgroups, 2 group commands

fetch() → List[float]

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:MEValuation:TRACe:SACP:MAXimum
value: List[float] = driver.bluetooth.measurement.multiEval.trace.sacp.maximum.
↪fetch()
```

Returns 81 values of the Spectrum ACP results in line with the Bluetooth test specification.

INTRO_CMD_HELP: The number of valid ACP results depends on the ACP measurement mode (method RsCMPX_BluetoothMeas.Configure.Bluetooth.Measurement.MultiEval.Sacp.LowEnergy.Le2M.Measurement
:

- For LE bursts, the trace returns 81 values.

INTRO_CMD_HELP: The number of valid ACP results depends on the ACP measurement mode (method RsCMPX_BluetoothMeas.Configure.Bluetooth.Measurement.MultiEval.Sacp.LowEnergy.Le2M.Measurement
:

- In CH10 mode (ACP +/- 5 Channels) , the first 21 ACP values contain results for the 1 MHz channels centered at fTX – 10 MHz, fTX – 9 MHz, …, fTX + 10 MHz. The remaining 58 values are invalid (NAV) . This mode is applicable to all types of LE bursts.
- In CH40 mode (LE All Channels) , ACP values 1 to 81 contain results for the 1 MHz channels centered at 2401 MHz, 2402 MHz, …, 2481 MHz This mode is only applicable to test packets using LE 1M PHY or LE 2M PHY.

Suppressed linked return values: reliability

return
acp: 81 spectrum ACP results

read() → List[float]

```
# SCPI: READ:BLUetooth:MEASurement<Instance>:MEValuation:TRACe:SACP:MAXimum
value: List[float] = driver.bluetooth.measurement.multiEval.trace.sacp.maximum.
↪read()
```

Returns 81 values of the Spectrum ACP results in line with the Bluetooth test specification.

INTRO_CMD_HELP: The number of valid ACP results depends on the ACP measurement mode (method RsCMPX_BluetoothMeas.Configure.Bluetooth.Measurement.MultiEval.Sacp.LowEnergy.Le2M.Measurement :

- For LE bursts, the trace returns 81 values.

INTRO_CMD_HELP: The number of valid ACP results depends on the ACP measurement mode (method RsCMPX_BluetoothMeas.Configure.Bluetooth.Measurement.MultiEval.Sacp.LowEnergy.Le2M.Measurement :

- In CH10 mode (ACP +/- 5 Channels) , the first 21 ACP values contain results for the 1 MHz channels centered at fTX – 10 MHz, fTX – 9 MHz, …, fTX + 10 MHz. The remaining 58 values are invalid (NAV) . This mode is applicable to all types of LE bursts.
- In CH40 mode (LE All Channels) , ACP values 1 to 81 contain results for the 1 MHz channels centered at 2401 MHz, 2402 MHz, …, 2481 MHz This mode is only applicable to test packets using LE 1M PHY or LE 2M PHY.

Suppressed linked return values: reliability

return
acp: 81 spectrum ACP results

6.1.1.6.10.31 Ptx

SCPI Commands :

```
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:TRACe:SACP[:PTX]
READ:BLUetooth:MEASurement<Instance>:MEValuation:TRACe:SACP[:PTX]
```

class PtxCls

Ptx commands group definition. 2 total commands, 0 Subgroups, 2 group commands

fetch() → List[float]

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:MEValuation:TRACe:SACP[:PTX]
value: List[float] = driver.bluetooth.measurement.multiEval.trace.sacp.ptx.
↪fetch()


    INTRO_CMD_HELP: Returns the results of the Spectrum ACP table for BR and LE␣
↪packets in line with the Bluetooth test specification. Note that the number␣
↪of returned values depends on the current burst type.

    - For BR bursts, the trace returns 79 values.
    INTRO_CMD_HELP: The number of valid ACP results depends on the ACP␣
↪measurement mode (method RsCMPX_BluetoothMeas.Configure.Bluetooth.Measurement.
↪MultiEval.Sacp.Brate.Measurement.mode) :
```

(continues on next page)

(continued from previous page)

```
    - In CH21 mode, the first 21 ACP values contain results for the relative␣
↪channels -10, ..., 0, ..., +10; the remaining 58 values are not displayed.
    - In CH79 mode, valid ACP values are available for all 79 Bluetooth␣
↪channels (2402 MHz, 2403 MHz, ..., 2480 MHz)
    - For LE bursts, the trace returns 81 values.
    INTRO_CMD_HELP: The number of valid ACP results depends on the ACP␣
↪measurement mode (method RsCMPX_BluetoothMeas.Configure.Bluetooth.Measurement.
↪MultiEval.Sacp.LowEnergy.Le2M.Measurement.mode) :

    - In CH10 (ACP +/- 5 Channels) mode, the first 21 ACP values contain␣
↪results for the 1 MHz channels centered at fTX - 10 MHz, fTX - 9 MHz, ...,␣
↪fTX + 10 MHz. The remaining 58 values are invalid (NAV) . This mode is␣
↪applicable to all types of LE bursts.
    - In CH40 mode (LE All Channels) , ACP values 1 to 81 contain results for␣
↪the 1 MHz channels centered at 2401 MHz, 2402 MHz, ..., 2481 MHz This mode is␣
↪only applicable to test packets using LE 1M PHY or LE 2M PHY.
```

Suppressed linked return values: reliability

return
acp: As explained above, for BR bursts the trace returns 79, and for LE bursts it returns 81 values.

read() → List[float]

```
# SCPI: READ:BLUetooth:MEASurement<Instance>:MEValuation:TRACe:SACP[:PTX]
value: List[float] = driver.bluetooth.measurement.multiEval.trace.sacp.ptx.
↪read()


    INTRO_CMD_HELP: Returns the results of the Spectrum ACP table for BR and LE␣
↪packets in line with the Bluetooth test specification. Note that the number␣
↪of returned values depends on the current burst type.

    - For BR bursts, the trace returns 79 values.
    INTRO_CMD_HELP: The number of valid ACP results depends on the ACP␣
↪measurement mode (method RsCMPX_BluetoothMeas.Configure.Bluetooth.Measurement.
↪MultiEval.Sacp.Brate.Measurement.mode) :

    - In CH21 mode, the first 21 ACP values contain results for the relative␣
↪channels -10, ..., 0, ..., +10; the remaining 58 values are not displayed.
    - In CH79 mode, valid ACP values are available for all 79 Bluetooth␣
↪channels (2402 MHz, 2403 MHz, ..., 2480 MHz)
    - For LE bursts, the trace returns 81 values.
    INTRO_CMD_HELP: The number of valid ACP results depends on the ACP␣
↪measurement mode (method RsCMPX_BluetoothMeas.Configure.Bluetooth.Measurement.
↪MultiEval.Sacp.LowEnergy.Le2M.Measurement.mode) :

    - In CH10 (ACP +/- 5 Channels) mode, the first 21 ACP values contain␣
↪results for the 1 MHz channels centered at fTX - 10 MHz, fTX - 9 MHz, ...,␣
↪fTX + 10 MHz. The remaining 58 values are invalid (NAV) . This mode is␣
↪applicable to all types of LE bursts.
```

(continues on next page)

(continued from previous page)

```
    - In CH40 mode (LE All Channels) , ACP values 1 to 81 contain results for␣
↪the 1 MHz channels centered at 2401 MHz, 2402 MHz, ..., 2481 MHz This mode is␣
↪only applicable to test packets using LE 1M PHY or LE 2M PHY.
```

Suppressed linked return values: reliability

return
acp: As explained above, for BR bursts the trace returns 79, and for LE bursts it returns 81 values.

6.1.1.6.10.32 Sgacp

class SgacpCls

Sgacp commands group definition. 8 total commands, 4 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.bluetooth.measurement.multiEval.trace.sgacp.clone()
```

Subgroups

6.1.1.6.10.33 Average

SCPI Commands :

```
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:TRACe:SGACp:AVERage
READ:BLUetooth:MEASurement<Instance>:MEValuation:TRACe:SGACp:AVERage
```

class AverageCls

Average commands group definition. 2 total commands, 0 Subgroups, 2 group commands

fetch() → List[float]

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:MEValuation:TRACe:SGACp:AVERage
value: List[float] = driver.bluetooth.measurement.multiEval.trace.sgacp.average.
↪fetch()
```

Returns the results of the Spectrum Gated ACP traces for EDR packets. The CMP180 measures the current, average and maximum adjacent channel power values.

INTRO_CMD_HELP: The number of valid results depends on the ACP measurement mode (method RsCMPX_BluetoothMeas.Configure.Bluetooth.Measurement.MultiEval.Sgacp.Edrate.Measurement.modeCH | CH79) :

- If CH21 mode (ACP +/- 10 Channels) is selected, the first 21 values contain the results for the relative channels –10, …, 0, …, +10; the remaining 58 values are not displayed.
- If CH79 mode (ACP 79 Channels) is selected, valid ACP values are available for all channels in the Bluetooth regulatory range.

Suppressed linked return values: reliability

return
acp: 79 ACP results

read() → List[float]

```
# SCPI: READ:BLUetooth:MEASurement<Instance>:MEValuation:TRACe:SGACp:AVERage
value: List[float] = driver.bluetooth.measurement.multiEval.trace.sgacp.average.
↪read()
```

Returns the results of the Spectrum Gated ACP traces for EDR packets. The CMP180 measures the current, average and maximum adjacent channel power values.

INTRO_CMD_HELP: The number of valid results depends on the ACP measurement mode (method RsCMPX_BluetoothMeas.Configure.Bluetooth.Measurement.MultiEval.Sgacp.Edrate.Measurement.modeCH | CH79) :

- If CH21 mode (ACP +/- 10 Channels) is selected, the first 21 values contain the results for the relative channels –10, …, 0, …, +10; the remaining 58 values are not displayed.
- If CH79 mode (ACP 79 Channels) is selected, valid ACP values are available for all channels in the Bluetooth regulatory range.

Suppressed linked return values: reliability

return
acp: 79 ACP results

6.1.1.6.10.34 Current

SCPI Commands :

```
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:TRACe:SGACp:CURRent
READ:BLUetooth:MEASurement<Instance>:MEValuation:TRACe:SGACp:CURRent
```

class CurrentCls

Current commands group definition. 2 total commands, 0 Subgroups, 2 group commands

fetch() → List[float]

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:MEValuation:TRACe:SGACp:CURRent
value: List[float] = driver.bluetooth.measurement.multiEval.trace.sgacp.current.
↪fetch()
```

Returns the results of the Spectrum Gated ACP traces for EDR packets. The CMP180 measures the current, average and maximum adjacent channel power values.

INTRO_CMD_HELP: The number of valid results depends on the ACP measurement mode (method RsCMPX_BluetoothMeas.Configure.Bluetooth.Measurement.MultiEval.Sgacp.Edrate.Measurement.modeCH | CH79) :

- If CH21 mode (ACP +/- 10 Channels) is selected, the first 21 values contain the results for the relative channels –10, …, 0, …, +10; the remaining 58 values are not displayed.
- If CH79 mode (ACP 79 Channels) is selected, valid ACP values are available for all channels in the Bluetooth regulatory range.

Suppressed linked return values: reliability

return
acp: 79 ACP results

read() → List[float]

```
# SCPI: READ:BLUetooth:MEASurement<Instance>:MEValuation:TRACe:SGACp:CURRent
value: List[float] = driver.bluetooth.measurement.multiEval.trace.sgacp.current.
↪read()
```

Returns the results of the Spectrum Gated ACP traces for EDR packets. The CMP180 measures the current, average and maximum adjacent channel power values.

INTRO_CMD_HELP: The number of valid results depends on the ACP measurement mode (method RsCMPX_BluetoothMeas.Configure.Bluetooth.Measurement.MultiEval.Sgacp.Edrate.Measurement.modeCH | CH79) :

- If CH21 mode (ACP +/- 10 Channels) is selected, the first 21 values contain the results for the relative channels –10, …, 0, …, +10; the remaining 58 values are not displayed.
- If CH79 mode (ACP 79 Channels) is selected, valid ACP values are available for all channels in the Bluetooth regulatory range.

Suppressed linked return values: reliability

return
acp: 79 ACP results

6.1.1.6.10.35 Maximum

SCPI Commands :

```
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:TRACe:SGACp:MAXimum
READ:BLUetooth:MEASurement<Instance>:MEValuation:TRACe:SGACp:MAXimum
```

class MaximumCls

Maximum commands group definition. 2 total commands, 0 Subgroups, 2 group commands

fetch() → List[float]

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:MEValuation:TRACe:SGACp:MAXimum
value: List[float] = driver.bluetooth.measurement.multiEval.trace.sgacp.maximum.
↪fetch()
```

Returns the results of the Spectrum Gated ACP traces for EDR packets. The CMP180 measures the current, average and maximum adjacent channel power values.

INTRO_CMD_HELP: The number of valid results depends on the ACP measurement mode (method RsCMPX_BluetoothMeas.Configure.Bluetooth.Measurement.MultiEval.Sgacp.Edrate.Measurement.modeCH | CH79) :

- If CH21 mode (ACP +/- 10 Channels) is selected, the first 21 values contain the results for the relative channels –10, …, 0, …, +10; the remaining 58 values are not displayed.
- If CH79 mode (ACP 79 Channels) is selected, valid ACP values are available for all channels in the Bluetooth regulatory range.

Suppressed linked return values: reliability

return
acp: 79 ACP results

read() → List[float]

```
# SCPI: READ:BLUetooth:MEASurement<Instance>:MEValuation:TRACe:SGACp:MAXimum
value: List[float] = driver.bluetooth.measurement.multiEval.trace.sgacp.maximum.
↪read()
```

Returns the results of the Spectrum Gated ACP traces for EDR packets. The CMP180 measures the current, average and maximum adjacent channel power values.

INTRO_CMD_HELP: The number of valid results depends on the ACP measurement mode (method RsCMPX_BluetoothMeas.Configure.Bluetooth.Measurement.MultiEval.Sgacp.Edrate.Measurement.modeCH
| CH79) :

- If CH21 mode (ACP +/- 10 Channels) is selected, the first 21 values contain the results for the relative channels –10, …, 0, …, +10; the remaining 58 values are not displayed.
- If CH79 mode (ACP 79 Channels) is selected, valid ACP values are available for all channels in the Bluetooth regulatory range.

Suppressed linked return values: reliability

return
acp: 79 ACP results

6.1.1.6.10.36 Ptx

SCPI Commands :

```
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:TRACe:SGACp[:PTX]
READ:BLUetooth:MEASurement<Instance>:MEValuation:TRACe:SGACp[:PTX]
```

class PtxCls

Ptx commands group definition. 2 total commands, 0 Subgroups, 2 group commands

fetch() → List[float]

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:MEValuation:TRACe:SGACp[:PTX]
value: List[float] = driver.bluetooth.measurement.multiEval.trace.sgacp.ptx.
↪fetch()
```

Returns the values of the Spectrum Gated ACP tables for EDR packets. The CMP180 measures the adjacent channel power values PTX(f) in line with Bluetooth test specification.

INTRO_CMD_HELP: The number of valid results depends on the ACP measurement mode (method RsCMPX_BluetoothMeas.Configure.Bluetooth.Measurement.MultiEval.Sgacp.Edrate.Measurement.mode CH21 | CH79) :

- If ACP +/- 10 Channels is selected, the first 21 values contain the results for the relative channels –10, …, 0, …, +10; the remaining 58 values are not displayed.
- If ACP 79 Channels is selected, valid ACP values are available for all channels in the Bluetooth regulatory range.

Suppressed linked return values: reliability

return
acp: 79 ACP results

read() → List[float]

```
# SCPI: READ:BLUetooth:MEASurement<Instance>:MEValuation:TRACe:SGACp[:PTX]
value: List[float] = driver.bluetooth.measurement.multiEval.trace.sgacp.ptx.
↪read()
```

Returns the values of the Spectrum Gated ACP tables for EDR packets. The CMP180 measures the adjacent channel power values PTX(f) in line with Bluetooth test specification.

INTRO_CMD_HELP: The number of valid results depends on the ACP measurement mode (method RsCMPX_BluetoothMeas.Configure.Bluetooth.Measurement.MultiEval.Sgacp.Edrate.Measurement.mode CH21 | CH79) :

- If ACP +/- 10 Channels is selected, the first 21 values contain the results for the relative channels –10, …, 0, …, +10; the remaining 58 values are not displayed.
- If ACP 79 Channels is selected, valid ACP values are available for all channels in the Bluetooth regulatory range.

Suppressed linked return values: reliability

return
acp: 79 ACP results

6.1.1.6.10.37 SoBw

class SoBwCls

SoBw commands group definition. 6 total commands, 3 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.bluetooth.measurement.multiEval.trace.soBw.clone()
```

Subgroups

6.1.1.6.10.38 Average

SCPI Commands :

```
READ:BLUetooth:MEASurement<Instance>:MEValuation:TRACe:SOBW:AVERage
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:TRACe:SOBW:AVERage
```

class AverageCls

Average commands group definition. 2 total commands, 0 Subgroups, 2 group commands

fetch() → List[float]

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:MEValuation:TRACe:SOBW:AVERage
value: List[float] = driver.bluetooth.measurement.multiEval.trace.soBw.average.
↪fetch()
```

Returns current, average and maximum results of the Spectrum 20 dB Bandwidth trace. The 20 dB bandwidth values are available for BR bursts (method RsCMPX_BluetoothMeas.Configure.Bluetooth.Measurement.InputSignal.btype) .

Suppressed linked return values: reliability

return
obw: 769 bandwidth results, covering a frequency range [–1.5 MHz, +1.5 MHz], relative to the peak emission within the measured Bluetooth channel. The spacing between adjacent trace points is 3.906 kHz (4 MHz/1024) .

read() → List[float]

```
# SCPI: READ:BLUetooth:MEASurement<Instance>:MEValuation:TRACe:SOBW:AVERage
value: List[float] = driver.bluetooth.measurement.multiEval.trace.soBw.average.
↪read()
```

Returns current, average and maximum results of the Spectrum 20 dB Bandwidth trace. The 20 dB bandwidth values are available for BR bursts (method RsCMPX_BluetoothMeas.Configure.Bluetooth.Measurement.InputSignal.btype) .

Suppressed linked return values: reliability

return
obw: 769 bandwidth results, covering a frequency range [–1.5 MHz, +1.5 MHz], relative to the peak emission within the measured Bluetooth channel. The spacing between adjacent trace points is 3.906 kHz (4 MHz/1024) .

6.1.1.6.10.39 Current

SCPI Commands :

```
READ:BLUetooth:MEASurement<Instance>:MEValuation:TRACe:SOBW:CURRent
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:TRACe:SOBW:CURRent
```

class CurrentCls

Current commands group definition. 2 total commands, 0 Subgroups, 2 group commands

fetch() → List[float]

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:MEValuation:TRACe:SOBW:CURRent
value: List[float] = driver.bluetooth.measurement.multiEval.trace.soBw.current.
↪fetch()
```

Returns current, average and maximum results of the Spectrum 20 dB Bandwidth trace. The 20 dB bandwidth values are available for BR bursts (method RsCMPX_BluetoothMeas.Configure.Bluetooth.Measurement.InputSignal.btype) .

Suppressed linked return values: reliability

return
obw: 769 bandwidth results, covering a frequency range [–1.5 MHz, +1.5 MHz], relative to the peak emission within the measured Bluetooth channel. The spacing between adjacent trace points is 3.906 kHz (4 MHz/1024) .

read() → List[float]

```
# SCPI: READ:BLUetooth:MEASurement<Instance>:MEValuation:TRACe:SOBW:CURRent
value: List[float] = driver.bluetooth.measurement.multiEval.trace.soBw.current.
↪read()
```

Returns current, average and maximum results of the Spectrum 20 dB Bandwidth trace. The 20 dB bandwidth values are available for BR bursts (method RsCMPX_BluetoothMeas.Configure.Bluetooth.Measurement.InputSignal.btype) .

Suppressed linked return values: reliability

return
obw: 769 bandwidth results, covering a frequency range [–1.5 MHz, +1.5 MHz], relative to the peak emission within the measured Bluetooth channel. The spacing between adjacent trace points is 3.906 kHz (4 MHz/1024) .

6.1.1.6.10.40 Maximum

SCPI Commands :

```
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:TRACe:SOBW:MAXimum
READ:BLUetooth:MEASurement<Instance>:MEValuation:TRACe:SOBW:MAXimum
```

class MaximumCls

Maximum commands group definition. 2 total commands, 0 Subgroups, 2 group commands

fetch() → List[float]

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:MEValuation:TRACe:SOBW:MAXimum
value: List[float] = driver.bluetooth.measurement.multiEval.trace.soBw.maximum.
↪fetch()
```

Returns current, average and maximum results of the Spectrum 20 dB Bandwidth trace. The 20 dB bandwidth values are available for BR bursts (method RsCMPX_BluetoothMeas.Configure.Bluetooth.Measurement.InputSignal.btype) .

Suppressed linked return values: reliability

return
obw: 769 bandwidth results, covering a frequency range [–1.5 MHz, +1.5 MHz], relative to the peak emission within the measured Bluetooth channel. The spacing between adjacent trace points is 3.906 kHz (4 MHz/1024) .

read() → List[float]

```
# SCPI: READ:BLUetooth:MEASurement<Instance>:MEValuation:TRACe:SOBW:MAXimum
value: List[float] = driver.bluetooth.measurement.multiEval.trace.soBw.maximum.
↪read()
```

Returns current, average and maximum results of the Spectrum 20 dB Bandwidth trace. The 20 dB bandwidth values are available for BR bursts (method RsCMPX_BluetoothMeas.Configure.Bluetooth.Measurement.InputSignal.btype) .

Suppressed linked return values: reliability

return
obw: 769 bandwidth results, covering a frequency range [–1.5 MHz, +1.5 MHz], relative to the peak emission within the measured Bluetooth channel. The spacing between adjacent trace points is 3.906 kHz (4 MHz/1024) .

6.1.1.6.10.41 Spower

class SpowerCls

Spower commands group definition. 8 total commands, 4 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.bluetooth.measurement.multiEval.trace.spower.clone()
```

Subgroups

6.1.1.6.10.42 Average

SCPI Commands :

```
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:TRACe:SPOWer:AVERage
READ:BLUetooth:MEASurement<Instance>:MEValuation:TRACe:SPOWer:AVERage
```

class AverageCls

Average commands group definition. 2 total commands, 0 Subgroups, 2 group commands

fetch() → List[float]

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:MEValuation:TRACe:SPOWer:AVERage
value: List[float] = driver.bluetooth.measurement.multiEval.trace.spower.
↪average.fetch()
```

Returns the results of the power per CTE slot for LE CTE traces. The results of the current, average, minimum and maximum traces can be retrieved.

Suppressed linked return values: reliability

return
power_vs_slot: No help available

read() → List[float]

```
# SCPI: READ:BLUetooth:MEASurement<Instance>:MEValuation:TRACe:SPOWer:AVERage
value: List[float] = driver.bluetooth.measurement.multiEval.trace.spower.
↪average.read()
```

Returns the results of the power per CTE slot for LE CTE traces. The results of the current, average, minimum and maximum traces can be retrieved.

Suppressed linked return values: reliability

return
power_vs_slot: No help available

6.1.1.6.10.43 Current

SCPI Commands :

```
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:TRACe:SPOWer:CURRent
READ:BLUetooth:MEASurement<Instance>:MEValuation:TRACe:SPOWer:CURRent
```

class CurrentCls

Current commands group definition. 2 total commands, 0 Subgroups, 2 group commands

fetch() → List[float]

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:MEValuation:TRACe:SPOWer:CURRent
value: List[float] = driver.bluetooth.measurement.multiEval.trace.spower.
↪current.fetch()
```

Returns the results of the power per CTE slot for LE CTE traces. The results of the current, average, minimum and maximum traces can be retrieved.

Suppressed linked return values: reliability

return
power_vs_slot: No help available

read() → List[float]

```
# SCPI: READ:BLUetooth:MEASurement<Instance>:MEValuation:TRACe:SPOWer:CURRent
value: List[float] = driver.bluetooth.measurement.multiEval.trace.spower.
↪current.read()
```

Returns the results of the power per CTE slot for LE CTE traces. The results of the current, average, minimum and maximum traces can be retrieved.

Suppressed linked return values: reliability

return
power_vs_slot: No help available

6.1.1.6.10.44 Maximum

SCPI Commands :

```
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:TRACe:SPOWer:MAXimum
READ:BLUetooth:MEASurement<Instance>:MEValuation:TRACe:SPOWer:MAXimum
```

class MaximumCls

Maximum commands group definition. 2 total commands, 0 Subgroups, 2 group commands

fetch() → List[float]

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:MEValuation:TRACe:SPOWer:MAXimum
value: List[float] = driver.bluetooth.measurement.multiEval.trace.spower.
↪maximum.fetch()
```

Returns the results of the power per CTE slot for LE CTE traces. The results of the current, average, minimum and maximum traces can be retrieved.

Suppressed linked return values: reliability

return
power_vs_slot: No help available

read() → List[float]

```
# SCPI: READ:BLUetooth:MEASurement<Instance>:MEValuation:TRACe:SPOWer:MAXimum
value: List[float] = driver.bluetooth.measurement.multiEval.trace.spower.
↪maximum.read()
```

Returns the results of the power per CTE slot for LE CTE traces. The results of the current, average, minimum and maximum traces can be retrieved.

Suppressed linked return values: reliability

return
power_vs_slot: No help available

6.1.1.6.10.45 Minimum

SCPI Commands :

```
FETCh:BLUetooth:MEASurement<Instance>:MEValuation:TRACe:SPOWer:MINimum
READ:BLUetooth:MEASurement<Instance>:MEValuation:TRACe:SPOWer:MINimum
```

class MinimumCls

Minimum commands group definition. 2 total commands, 0 Subgroups, 2 group commands

fetch() → List[float]

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:MEValuation:TRACe:SPOWer:MINimum
value: List[float] = driver.bluetooth.measurement.multiEval.trace.spower.
↪minimum.fetch()
```

Returns the results of the power per CTE slot for LE CTE traces. The results of the current, average, minimum and maximum traces can be retrieved.

Suppressed linked return values: reliability

return
power_vs_slot: No help available

read() → List[float]

```
# SCPI: READ:BLUetooth:MEASurement<Instance>:MEValuation:TRACe:SPOWer:MINimum
value: List[float] = driver.bluetooth.measurement.multiEval.trace.spower.
↪minimum.read()
```

Returns the results of the power per CTE slot for LE CTE traces. The results of the current, average, minimum and maximum traces can be retrieved.

Suppressed linked return values: reliability

return
power_vs_slot: No help available

6.1.1.7 RxQuality

SCPI Commands :

```
INITiate:BLUetooth:MEASurement<Instance>:RXQuality
STOP:BLUetooth:MEASurement<Instance>:RXQuality
ABORt:BLUetooth:MEASurement<Instance>:RXQuality
```

class RxQualityCls

RxQuality commands group definition. 10 total commands, 5 Subgroups, 3 group commands

abort(*opc_timeout_ms: int = -1*) → None

```
# SCPI: ABORt:BLUetooth:MEASurement<Instance>:RXQuality
driver.bluetooth.measurement.rxQuality.abort()
```

No command help available

param opc_timeout_ms
Maximum time to wait in milliseconds, valid only for this call.

initiate(*opc_timeout_ms: int = -1*) → None

```
# SCPI: INITiate:BLUetooth:MEASurement<Instance>:RXQuality
driver.bluetooth.measurement.rxQuality.initiate()
```

No command help available

param opc_timeout_ms
Maximum time to wait in milliseconds, valid only for this call.

stop() → None

```
# SCPI: STOP:BLUetooth:MEASurement<Instance>:RXQuality
driver.bluetooth.measurement.rxQuality.stop()
```

No command help available

stop_with_opc(*opc_timeout_ms: int = -1*) → None

```
# SCPI: STOP:BLUetooth:MEASurement<Instance>:RXQuality
driver.bluetooth.measurement.rxQuality.stop_with_opc()
```

No command help available

Same as stop, but waits for the operation to complete before continuing further. Use the RsCMPX_BluetoothMeas.utilities.opc_timeout_set() to set the timeout value.

param opc_timeout_ms
Maximum time to wait in milliseconds, valid only for this call.

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.bluetooth.measurement.rxQuality.clone()
```

Subgroups

6.1.1.7.1 Adetected

class AdetectedCls

Adetected commands group definition. 1 total commands, 1 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.bluetooth.measurement.rxQuality.adetected.clone()
```

Subgroups

6.1.1.7.1.1 Aaddress

SCPI Command :

```
FETCh:BLUetooth:MEASurement<Instance>:RXQuality:ADETected:AADDress
```

class AaddressCls

Aaddress commands group definition. 1 total commands, 0 Subgroups, 1 group commands

fetch() → str

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:RXQuality:ADETected:AADDress
value: str = driver.bluetooth.measurement.rxQuality.adetected.aaddress.fetch()
```

No command help available

Suppressed linked return values: reliability

return
adv_address: No help available

6.1.1.7.2 Per

SCPI Command :

```
FETCh:BLUetooth:MEASurement<Instance>:RXQuality:PER
```

class PerCls

Per commands group definition. 2 total commands, 1 Subgroups, 1 group commands

fetch() → float

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:RXQuality:PER
value: float = driver.bluetooth.measurement.rxQuality.per.fetch()
```

No command help available

Suppressed linked return values: reliability

return
result: No help available

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.bluetooth.measurement.rxQuality.per.clone()
```

Subgroups

6.1.1.7.2.1 RxPackets

SCPI Command :

```
FETCh:BLUetooth:MEASurement<Instance>:RXQuality:PER:RXPackets
```

class RxPacketsCls

RxPackets commands group definition. 1 total commands, 0 Subgroups, 1 group commands

fetch() → int

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:RXQuality:PER:RXPackets
value: int = driver.bluetooth.measurement.rxQuality.per.rxPackets.fetch()
```

No command help available

Suppressed linked return values: reliability

return
packets_received: No help available

6.1.1.7.3 Sensitivity

SCPI Command :

```
FETCh:BLUetooth:MEASurement<Instance>:RXQuality:SENSitivity
```

class SensitivityCls

Sensitivity commands group definition. 1 total commands, 0 Subgroups, 1 group commands

fetch() → float

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:RXQuality:SENSitivity
value: float = driver.bluetooth.measurement.rxQuality.sensitivity.fetch()
```

No command help available

Suppressed linked return values: reliability

return
sensitivity_search: No help available

6.1.1.7.4 SpotCheck

SCPI Command :

```
FETCh:BLUetooth:MEASurement<Instance>:RXQuality:SPOTcheck
```

class SpotCheckCls

SpotCheck commands group definition. 1 total commands, 0 Subgroups, 1 group commands

fetch() → Result

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:RXQuality:SPOTcheck
value: enums.Result = driver.bluetooth.measurement.rxQuality.spotCheck.fetch()
```

No command help available

Suppressed linked return values: reliability

return
spot_check_result: No help available

6.1.1.7.5 State

SCPI Command :

```
FETCh:BLUetooth:MEASurement<Instance>:RXQuality:STATe
```

class StateCls

State commands group definition. 2 total commands, 1 Subgroups, 1 group commands

fetch(*timeout: float = None*, *target_main_state: ResourceState = None*, *target_sync_state: SyncState = None*) → ResourceState

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:RXQuality:STATe
value: enums.ResourceState = driver.bluetooth.measurement.rxQuality.state.
↪fetch(timeout = 1.0, target_main_state = enums.ResourceState.ACTive, target_
↪sync_state = enums.SyncState.ADJusted)
```

No command help available

param timeout
No help available

param target_main_state
No help available

param target_sync_state
No help available

return
meas_state: No help available

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.bluetooth.measurement.rxQuality.state.clone()
```

Subgroups

6.1.1.7.5.1 All

SCPI Command :

```
FETCh:BLUetooth:MEASurement<Instance>:RXQuality:STATe:ALL
```

class AllCls

All commands group definition. 1 total commands, 0 Subgroups, 1 group commands

class FetchStruct

Response structure. Fields:

- Main_State: enums.ResourceState: No parameter help available
- Sync_State: enums.ResourceState: No parameter help available
- Resource_State: enums.ResourceState: No parameter help available

fetch(*timeout: float = None*, *target_main_state: ResourceState = None*, *target_sync_state: SyncState = None*) → FetchStruct

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:RXQuality:STATe:ALL
value: FetchStruct = driver.bluetooth.measurement.rxQuality.state.all.
↪fetch(timeout = 1.0, target_main_state = enums.ResourceState.ACTive, target_
↪sync_state = enums.SyncState.ADJusted)
```

No command help available

param timeout
No help available

param target_main_state
No help available

param target_sync_state
No help available

return
structure: for return value, see the help for FetchStruct structure arguments.

6.1.1.8 Trx

SCPI Commands :

```
INITiate:BLUetooth:MEASurement<Instance>:TRX
STOP:BLUetooth:MEASurement<Instance>:TRX
ABORt:BLUetooth:MEASurement<Instance>:TRX
```

class TrxCls

Trx commands group definition. 9 total commands, 5 Subgroups, 3 group commands

abort(*opc_timeout_ms: int = -1*) → None

```
# SCPI: ABORt:BLUetooth:MEASurement<Instance>:TRX
driver.bluetooth.measurement.trx.abort()
```

No command help available

param opc_timeout_ms
Maximum time to wait in milliseconds, valid only for this call.

initiate(*opc_timeout_ms: int = -1*) → None

```
# SCPI: INITiate:BLUetooth:MEASurement<Instance>:TRX
driver.bluetooth.measurement.trx.initiate()
```

No command help available

param opc_timeout_ms
Maximum time to wait in milliseconds, valid only for this call.

stop() → None

```
# SCPI: STOP:BLUetooth:MEASurement<Instance>:TRX
driver.bluetooth.measurement.trx.stop()
```

No command help available

stop_with_opc(*opc_timeout_ms: int = -1*) → None

```
# SCPI: STOP:BLUetooth:MEASurement<Instance>:TRX
driver.bluetooth.measurement.trx.stop_with_opc()
```

No command help available

Same as stop, but waits for the operation to complete before continuing further. Use the RsCMPX_BluetoothMeas.utilities.opc_timeout_set() to set the timeout value.

param opc_timeout_ms
Maximum time to wait in milliseconds, valid only for this call.

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.bluetooth.measurement.trx.clone()
```

Subgroups

6.1.1.8.1 Acp

SCPI Command :

```
FETCh:BLUetooth:MEASurement<Instance>:TRX:ACP
```

class AcpCls

Acp commands group definition. 1 total commands, 0 Subgroups, 1 group commands

class FetchStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Nominal_Power: float: No parameter help available
- No_Of_Exceptions: int: No parameter help available

fetch() → FetchStruct

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:TRX:ACP
value: FetchStruct = driver.bluetooth.measurement.trx.acp.fetch()
```

No command help available

return
structure: for return value, see the help for FetchStruct structure arguments.

6.1.1.8.2 Modulation

SCPI Command :

```
FETCh:BLUetooth:MEASurement<Instance>:TRX:MODulation
```

class ModulationCls

Modulation commands group definition. 1 total commands, 0 Subgroups, 1 group commands

class FetchStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float: No parameter help available
- Delta_F_299_P_9: float: No parameter help available
- Freq_Accuracy: float: No parameter help available
- Freq_Drift: float: No parameter help available
- Max_Drift: float: No parameter help available
- Freq_Dev_Avg_F_1: float: No parameter help available
- Freq_Dev_Min_F_1: float: No parameter help available
- Freq_Dev_Max_F_1: float: No parameter help available
- Freq_Dev_Avg_F_2: float: No parameter help available
- Freq_Dev_Min_F_2: float: No parameter help available
- Freq_Dev_Max_F_2: float: No parameter help available
- Nominal_Power: float: No parameter help available
- Mod_Ratio: float: No parameter help available
- Freq_Offset: float: No parameter help available
- Init_Freq_Drift: float: No parameter help available

fetch() → FetchStruct

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:TRX:MODulation
value: FetchStruct = driver.bluetooth.measurement.trx.modulation.fetch()
```

No command help available

return
structure: for return value, see the help for FetchStruct structure arguments.

6.1.1.8.3 Power

SCPI Command :

```
FETCh:BLUetooth:MEASurement<Instance>:TRX:POWer
```

class PowerCls

Power commands group definition. 1 total commands, 0 Subgroups, 1 group commands

class FetchStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Out_Of_Tol: float: No parameter help available
- Nominal_Power: float: No parameter help available

- Peak_Power: float: No parameter help available
- Leakage_Power: float: No parameter help available
- Peak_Min_Avg_Pow: float: No parameter help available

fetch() → FetchStruct

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:TRX:POWer
value: FetchStruct = driver.bluetooth.measurement.trx.power.fetch()
```

No command help available

return
structure: for return value, see the help for FetchStruct structure arguments.

6.1.1.8.4 Spot

SCPI Command :

```
FETCh:BLUetooth:MEASurement<Instance>:TRX:SPOT
```

class SpotCls

Spot commands group definition. 1 total commands, 0 Subgroups, 1 group commands

fetch() → Result

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:TRX:SPOT
value: enums.Result = driver.bluetooth.measurement.trx.spot.fetch()
```

No command help available

Suppressed linked return values: reliability

return
spot_check_result: No help available

6.1.1.8.5 State

SCPI Command :

```
FETCh:BLUetooth:MEASurement<Instance>:TRX:STATe
```

class StateCls

State commands group definition. 2 total commands, 1 Subgroups, 1 group commands

fetch(*timeout: float = None*, *target_main_state: ResourceState = None*, *target_sync_state: SyncState = None*) → ResourceState

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:TRX:STATe
value: enums.ResourceState = driver.bluetooth.measurement.trx.state.
↪fetch(timeout = 1.0, target_main_state = enums.ResourceState.ACTive, target_
↪sync_state = enums.SyncState.ADJusted)
```

No command help available

param timeout
No help available

param target_main_state
No help available

param target_sync_state
No help available

return
meas_state: No help available

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.bluetooth.measurement.trx.state.clone()
```

Subgroups

6.1.1.8.5.1 All

SCPI Command :

```
FETCh:BLUetooth:MEASurement<Instance>:TRX:STATe:ALL
```

class AllCls

All commands group definition. 1 total commands, 0 Subgroups, 1 group commands

class FetchStruct

Response structure. Fields:

- Main_State: enums.ResourceState: No parameter help available
- Sync_State: enums.ResourceState: No parameter help available
- Resource_State: enums.ResourceState: No parameter help available

fetch(*timeout: float = None*, *target_main_state: ResourceState = None*, *target_sync_state: SyncState = None*) → FetchStruct

```
# SCPI: FETCh:BLUetooth:MEASurement<Instance>:TRX:STATe:ALL
value: FetchStruct = driver.bluetooth.measurement.trx.state.all.fetch(timeout =␣
↪1.0, target_main_state = enums.ResourceState.ACTive, target_sync_state =␣
↪enums.SyncState.ADJusted)
```

No command help available

param timeout
No help available

param target_main_state
No help available

param target_sync_state
No help available

return
structure: for return value, see the help for FetchStruct structure arguments.

6.2 Call

class CallCls

Call commands group definition. 3 total commands, 1 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.call.clone()
```

Subgroups

6.2.1 Bluetooth

class BluetoothCls

Bluetooth commands group definition. 3 total commands, 1 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.call.bluetooth.clone()
```

Subgroups

6.2.1.1 Measurement

class MeasurementCls

Measurement commands group definition. 3 total commands, 1 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.call.bluetooth.measurement.clone()
```

Subgroups

6.2.1.1.1 DtMode

class DtModeCls

DtMode commands group definition. 3 total commands, 1 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.call.bluetooth.measurement.dtMode.clone()
```

Subgroups

6.2.1.1.1.1 LowEnergy

SCPI Commands :

```
CALL:BLUetooth:MEASurement<Instance>:DTMode:LENergy:RRESult
CALL:BLUetooth:MEASurement<Instance>:DTMode:LENergy:RESet
```

class LowEnergyCls

LowEnergy commands group definition. 3 total commands, 1 Subgroups, 2 group commands

get_rresult() → Result

```
# SCPI: CALL:BLUetooth:MEASurement<Instance>:DTMode:LENergy:RRESult
value: enums.Result = driver.call.bluetooth.measurement.dtMode.lowEnergy.get_
↪rresult()
```

No command help available

return
result: No help available

reset(*opc_timeout_ms: int = -1*) → None

```
# SCPI: CALL:BLUetooth:MEASurement<Instance>:DTMode:LENergy:RESet
driver.call.bluetooth.measurement.dtMode.lowEnergy.reset()
```

No command help available

param opc_timeout_ms
Maximum time to wait in milliseconds, valid only for this call.

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.call.bluetooth.measurement.dtMode.lowEnergy.clone()
```

Subgroups

6.2.1.1.1.2 Rdevices

SCPI Command :

```
CALL:BLUetooth:MEASurement<Instance>:DTMode:LENergy:RDEVices
```

class RdevicesCls

Rdevices commands group definition. 1 total commands, 0 Subgroups, 1 group commands

set(*opc_timeout_ms: int = -1*) → None

```
# SCPI: CALL:BLUetooth:MEASurement<Instance>:DTMode:LENergy:RDEVices
driver.call.bluetooth.measurement.dtMode.lowEnergy.rdevices.set()
```

No command help available

param opc_timeout_ms
Maximum time to wait in milliseconds, valid only for this call.

6.3 Clean

class CleanCls

Clean commands group definition. 1 total commands, 1 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.clean.clone()
```

Subgroups

6.3.1 Bluetooth

class BluetoothCls

Bluetooth commands group definition. 1 total commands, 1 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.clean.bluetooth.clone()
```

Subgroups

6.3.1.1 Measurement

class MeasurementCls

Measurement commands group definition. 1 total commands, 1 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.clean.bluetooth.measurement.clone()
```

Subgroups

6.3.1.1.1 Elogging

SCPI Command :

```
CLEan:BLUetooth:MEASurement<Instance>:ELOGging
```

class EloggingCls

Elogging commands group definition. 1 total commands, 0 Subgroups, 1 group commands

set() → None

```
# SCPI: CLEan:BLUetooth:MEASurement<Instance>:ELOGging
driver.clean.bluetooth.measurement.elogging.set()
```

No command help available

set_with_opc(*opc_timeout_ms: int = -1*) → None

```
# SCPI: CLEan:BLUetooth:MEASurement<Instance>:ELOGging
driver.clean.bluetooth.measurement.elogging.set_with_opc()
```

No command help available

Same as set, but waits for the operation to complete before continuing further. Use the RsCMPX_BluetoothMeas.utilities.opc_timeout_set() to set the timeout value.

param opc_timeout_ms
Maximum time to wait in milliseconds, valid only for this call.

6.4 Configure

class ConfigureCls

Configure commands group definition. 389 total commands, 1 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.configure.clone()
```

Subgroups

6.4.1 Bluetooth

class BluetoothCls

Bluetooth commands group definition. 389 total commands, 1 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.configure.bluetooth.clone()
```

Subgroups

6.4.1.1 Measurement

SCPI Commands :

```
CONFigure:BLUetooth:MEASurement<Instance>:HWINterface
CONFigure:BLUetooth:MEASurement<Instance>:CPRotocol
CONFigure:BLUetooth:MEASurement<Instance>:GDELay
CONFigure:BLUetooth:MEASurement<Instance>:CFILter
CONFigure:BLUetooth:MEASurement<Instance>:OTHReshold
```

class MeasurementCls

Measurement commands group definition. 389 total commands, 11 Subgroups, 5 group commands

get_cfilter() → FilterWidth

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:CFILter
value: enums.FilterWidth = driver.configure.bluetooth.measurement.get_cfilter()
```

No command help available

return
capture_filter: No help available

get_cprotocol() → CommProtocol

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:CPRotocol
value: enums.CommProtocol = driver.configure.bluetooth.measurement.get_
↪cprotocol()
```

No command help available

return
comm_protocol: No help available

get_gdelay() → float

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:GDELay
value: float = driver.configure.bluetooth.measurement.get_gdelay()
```

No command help available

return
delay: No help available

get_hw_interface() → HwInterface

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:HWINterface
value: enums.HwInterface = driver.configure.bluetooth.measurement.get_hw_
↪interface()
```

No command help available

return
hw_interface: No help available

get_othreshold() → float

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:OTHReshold
value: float = driver.configure.bluetooth.measurement.get_othreshold()
```

No command help available

return
overdriven_threshold: No help available

set_cfilter(*capture_filter: FilterWidth*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:CFILter
driver.configure.bluetooth.measurement.set_cfilter(capture_filter = enums.
↪FilterWidth.NARRow)
```

No command help available

param capture_filter
No help available

set_cprotocol(*comm_protocol: CommProtocol*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:CPRotocol
driver.configure.bluetooth.measurement.set_cprotocol(comm_protocol = enums.
↪CommProtocol.HCI)
```

No command help available

param comm_protocol
No help available

set_gdelay(*delay: float*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:GDELay
driver.configure.bluetooth.measurement.set_gdelay(delay = 1.0)
```

No command help available

param delay
No help available

set_hw_interface(*hw_interface: HwInterface*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:HWINterface
driver.configure.bluetooth.measurement.set_hw_interface(hw_interface = enums.
↪HwInterface.NONE)
```

No command help available

param hw_interface
No help available

set_othreshold(*overdriven_threshold: float*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:OTHReshold
driver.configure.bluetooth.measurement.set_othreshold(overdriven_threshold = 1.
↪0)
```

No command help available

param overdriven_threshold
No help available

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.configure.bluetooth.measurement.clone()
```

Subgroups

6.4.1.1.1 BhRate

SCPI Commands :

```
CONFigure:BLUetooth:MEASurement<Instance>:BHRate:MOEXception
CONFigure:BLUetooth:MEASurement<Instance>:BHRate:SCONdition
CONFigure:BLUetooth:MEASurement<Instance>:BHRate:REPetition
```

class BhRateCls

BhRate commands group definition. 35 total commands, 5 Subgroups, 3 group commands

get_mo_exception() → bool

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:BHRate:MOEXception
value: bool = driver.configure.bluetooth.measurement.bhRate.get_mo_exception()
```

No command help available

return
meas_on_exception: No help available

get_repetition() → Repeat

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:BHRate:REPetition
value: enums.Repeat = driver.configure.bluetooth.measurement.bhRate.get_
↪repetition()
```

No command help available

return
repetition: No help available

get_scondition() → StopCondition

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:BHRate:SCONdition
value: enums.StopCondition = driver.configure.bluetooth.measurement.bhRate.get_
↪scondition()
```

No command help available

return
stop_condition: No help available

set_mo_exception(*meas_on_exception: bool*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:BHRate:MOEXception
driver.configure.bluetooth.measurement.bhRate.set_mo_exception(meas_on_
↪exception = False)
```

No command help available

param meas_on_exception
No help available

set_repetition(*repetition: Repeat*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:BHRate:REPetition
driver.configure.bluetooth.measurement.bhRate.set_repetition(repetition = enums.
↪Repeat.CONTinuous)
```

No command help available

param repetition
No help available

set_scondition(*stop_condition: StopCondition*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:BHRate:SCONdition
driver.configure.bluetooth.measurement.bhRate.set_scondition(stop_condition =␣
↪enums.StopCondition.NONE)
```

No command help available

param stop_condition
No help available

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.configure.bluetooth.measurement.bhRate.clone()
```

Subgroups

6.4.1.1.1.1 InputSignal

SCPI Commands :

```
CONFigure:BLUetooth:MEASurement<Instance>:BHRate:ISIGnal:PLENgth
CONFigure:BLUetooth:MEASurement<Instance>:BHRate:ISIGnal:PTYPe
CONFigure:BLUetooth:MEASurement<Instance>:BHRate:ISIGnal:NAP
CONFigure:BLUetooth:MEASurement<Instance>:BHRate:ISIGnal:UAP
CONFigure:BLUetooth:MEASurement<Instance>:BHRate:ISIGnal:LAP
CONFigure:BLUetooth:MEASurement<Instance>:BHRate:ISIGnal:BDADdress
CONFigure:BLUetooth:MEASurement<Instance>:BHRate:ISIGnal:ASYNchronize
CONFigure:BLUetooth:MEASurement<Instance>:BHRate:ISIGnal:DMODe
```

class InputSignalCls

InputSignal commands group definition. 8 total commands, 0 Subgroups, 8 group commands

get_asynchronize() → bool

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:BHRate:ISIGnal:ASYNchronize
value: bool = driver.configure.bluetooth.measurement.bhRate.inputSignal.get_
↪asynchronize()
```

No command help available

return
auto_sync: No help available

get_bd_address() → str

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:BHRate:ISIGnal:BDADdress
value: str = driver.configure.bluetooth.measurement.bhRate.inputSignal.get_bd_
↪address()
```

No command help available

return
bd_address: No help available

get_dmode() → AutoManualMode

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:BHRate:ISIGnal:DMODe
value: enums.AutoManualMode = driver.configure.bluetooth.measurement.bhRate.
↪inputSignal.get_dmode()
```

No command help available

return
detection_mode: No help available

get_lap() → str

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:BHRate:ISIGnal:LAP
value: str = driver.configure.bluetooth.measurement.bhRate.inputSignal.get_lap()
```

No command help available

return
lap_address: No help available

get_nap() → str

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:BHRate:ISIGnal:NAP
value: str = driver.configure.bluetooth.measurement.bhRate.inputSignal.get_nap()
```

No command help available

return
nap_address: No help available

get_plength() → List[int]

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:BHRate:ISIGnal:PLENgth
value: List[int] = driver.configure.bluetooth.measurement.bhRate.inputSignal.
↪get_plength()
```

No command help available

return
payload_length: No help available

get_ptype() → PacketTypeA

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:BHRate:ISIGnal:PTYPe
value: enums.PacketTypeA = driver.configure.bluetooth.measurement.bhRate.
↪inputSignal.get_ptype()
```

No command help available

return
packet_type: No help available

get_uap() → str

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:BHRate:ISIGnal:UAP
value: str = driver.configure.bluetooth.measurement.bhRate.inputSignal.get_uap()
```

No command help available

return
uap_address: No help available

set_asynchronize(*auto_sync: bool*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:BHRate:ISIGnal:ASYNchronize
driver.configure.bluetooth.measurement.bhRate.inputSignal.set_asynchronize(auto_
↪sync = False)
```

No command help available

param auto_sync
No help available

set_bd_address(*bd_address: str*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:BHRate:ISIGnal:BDADdress
driver.configure.bluetooth.measurement.bhRate.inputSignal.set_bd_address(bd_
↪address = rawAbc)
```

No command help available

param bd_address
No help available

set_dmode(*detection_mode: AutoManualMode*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:BHRate:ISIGnal:DMODe
driver.configure.bluetooth.measurement.bhRate.inputSignal.set_dmode(detection_
↪mode = enums.AutoManualMode.AUTO)
```

No command help available

param detection_mode
No help available

set_lap(*lap_address: str*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:BHRate:ISIGnal:LAP
driver.configure.bluetooth.measurement.bhRate.inputSignal.set_lap(lap_address =␣
↪rawAbc)
```

No command help available

param lap_address
No help available

set_nap(*nap_address: str*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:BHRate:ISIGnal:NAP
driver.configure.bluetooth.measurement.bhRate.inputSignal.set_nap(nap_address =␣
↪rawAbc)
```

No command help available

param nap_address
No help available

set_plength(*payload_length: List[int]*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:BHRate:ISIGnal:PLENgth
driver.configure.bluetooth.measurement.bhRate.inputSignal.set_plength(payload_
↪length = [1, 2, 3])
```

No command help available

param payload_length
No help available

set_ptype(*packet_type: PacketTypeA*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:BHRate:ISIGnal:PTYPe
driver.configure.bluetooth.measurement.bhRate.inputSignal.set_ptype(packet_type␣
↪= enums.PacketTypeA.E21P)
```

No command help available

param packet_type
No help available

set_uap(*uap_address: str*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:BHRate:ISIGnal:UAP
driver.configure.bluetooth.measurement.bhRate.inputSignal.set_uap(uap_address =␣
↪rawAbc)
```

No command help available

param uap_address
No help available

6.4.1.1.1.2 Limit

SCPI Command :

```
CONFigure:BLUetooth:MEASurement<Instance>:BHRate:LIMit:FSTability
```

class LimitCls

Limit commands group definition. 9 total commands, 5 Subgroups, 1 group commands

class FstabilityStruct

Structure for setting input parameters. Fields:

- Wi: float: No parameter help available
- Wi_W_0: float: No parameter help available
- W_0_Max: float: No parameter help available
- Wi_Enabled: List[bool]: No parameter help available
- Wi_Wo_Enabled: List[bool]: No parameter help available
- W_0_Max_Enabled: List[bool]: No parameter help available

get_fstability() → FstabilityStruct

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:BHRate:LIMit:FSTability
value: FstabilityStruct = driver.configure.bluetooth.measurement.bhRate.limit.
↪get_fstability()
```

No command help available

return
structure: for return value, see the help for FstabilityStruct structure arguments.

set_fstability(*value: FstabilityStruct*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:BHRate:LIMit:FSTability
structure = driver.configure.bluetooth.measurement.bhRate.limit.
↪FstabilityStruct()
structure.Wi: float = 1.0
structure.Wi_W_0: float = 1.0
structure.W_0_Max: float = 1.0
structure.Wi_Enabled: List[bool] = [True, False, True]
structure.Wi_Wo_Enabled: List[bool] = [True, False, True]
structure.W_0_Max_Enabled: List[bool] = [True, False, True]
driver.configure.bluetooth.measurement.bhRate.limit.set_fstability(value =␣
↪structure)
```

No command help available

param value
see the help for FstabilityStruct structure arguments.

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.configure.bluetooth.measurement.bhRate.limit.clone()
```

Subgroups

6.4.1.1.1.3 P2H

SCPI Commands :

```
CONFigure:BLUetooth:MEASurement<Instance>:BHRate:LIMit:P2H:DEVM
CONFigure:BLUetooth:MEASurement<Instance>:BHRate:LIMit:P2H:SGACp
```

class P2HCls

P2H commands group definition. 2 total commands, 0 Subgroups, 2 group commands

class DevmStruct

Structure for setting input parameters. Fields:

- Rms: float: No parameter help available
- Peak: float: No parameter help available

- P_99: float: No parameter help available
- Rms_Enabled: List[bool]: No parameter help available
- Peak_Enabled: List[bool]: No parameter help available
- P_99_Enabled: bool: No parameter help available

class SgacpStruct

Structure for setting input parameters. Fields:

- Ptx_Rel_Limit: float: No parameter help available
- Ptx_Abs_Limit_1: float: No parameter help available
- Ptx_Abs_Limit_2: float: No parameter help available
- Ptx_Rel_Enabled: bool: No parameter help available
- Ptx_Abs_1_Enable: bool: No parameter help available
- Ptx_Abs_2_Enable: bool: No parameter help available

get_devm() → DevmStruct

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:BHRate:LIMit:P2H:DEVM
value: DevmStruct = driver.configure.bluetooth.measurement.bhRate.limit.p2H.get_
↪devm()
```

No command help available

return
structure: for return value, see the help for DevmStruct structure arguments.

get_sgacp() → SgacpStruct

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:BHRate:LIMit:P2H:SGACp
value: SgacpStruct = driver.configure.bluetooth.measurement.bhRate.limit.p2H.
↪get_sgacp()
```

No command help available

return
structure: for return value, see the help for SgacpStruct structure arguments.

set_devm(*value: DevmStruct*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:BHRate:LIMit:P2H:DEVM
structure = driver.configure.bluetooth.measurement.bhRate.limit.p2H.DevmStruct()
structure.Rms: float = 1.0
structure.Peak: float = 1.0
structure.P_99: float = 1.0
structure.Rms_Enabled: List[bool] = [True, False, True]
structure.Peak_Enabled: List[bool] = [True, False, True]
structure.P_99_Enabled: bool = False
driver.configure.bluetooth.measurement.bhRate.limit.p2H.set_devm(value =␣
↪structure)
```

No command help available

param value
see the help for DevmStruct structure arguments.

set_sgacp(*value: SgacpStruct*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:BHRate:LIMit:P2H:SGACp
structure = driver.configure.bluetooth.measurement.bhRate.limit.p2H.
↪SgacpStruct()
structure.Ptx_Rel_Limit: float = 1.0
structure.Ptx_Abs_Limit_1: float = 1.0
structure.Ptx_Abs_Limit_2: float = 1.0
structure.Ptx_Rel_Enabled: bool = False
structure.Ptx_Abs_1_Enable: bool = False
structure.Ptx_Abs_2_Enable: bool = False
driver.configure.bluetooth.measurement.bhRate.limit.p2H.set_sgacp(value =␣
↪structure)
```

No command help available

param value
see the help for SgacpStruct structure arguments.

6.4.1.1.1.4 P4H

SCPI Commands :

```
CONFigure:BLUetooth:MEASurement<Instance>:BHRate:LIMit:P4H:DEVM
CONFigure:BLUetooth:MEASurement<Instance>:BHRate:LIMit:P4H:SGACp
```

class P4HCls

P4H commands group definition. 2 total commands, 0 Subgroups, 2 group commands

class DevmStruct

Structure for setting input parameters. Fields:

- Rms: float: No parameter help available
- Peak: float: No parameter help available
- P_99: float: No parameter help available
- Rms_Enabled: List[bool]: No parameter help available
- Peak_Enabled: List[bool]: No parameter help available
- P_99_Enabled: bool: No parameter help available

class SgacpStruct

Structure for setting input parameters. Fields:

- Ptx_Rel_Limit: float: No parameter help available
- Ptx_Abs_Limit_1: float: No parameter help available
- Ptx_Abs_Limit_2: float: No parameter help available
- Ptx_Rel_Enable: bool: No parameter help available
- Ptx_Abs_1_Enable: bool: No parameter help available
- Ptx_Abs_2_Enable: bool: No parameter help available

get_devm() → DevmStruct

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:BHRate:LIMit:P4H:DEVM
value: DevmStruct = driver.configure.bluetooth.measurement.bhRate.limit.p4H.get_
↪devm()
```

No command help available

return
structure: for return value, see the help for DevmStruct structure arguments.

get_sgacp() → SgacpStruct

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:BHRate:LIMit:P4H:SGACp
value: SgacpStruct = driver.configure.bluetooth.measurement.bhRate.limit.p4H.
↪get_sgacp()
```

No command help available

return
structure: for return value, see the help for SgacpStruct structure arguments.

set_devm(*value: DevmStruct*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:BHRate:LIMit:P4H:DEVM
structure = driver.configure.bluetooth.measurement.bhRate.limit.p4H.DevmStruct()
structure.Rms: float = 1.0
structure.Peak: float = 1.0
structure.P_99: float = 1.0
structure.Rms_Enabled: List[bool] = [True, False, True]
structure.Peak_Enabled: List[bool] = [True, False, True]
structure.P_99_Enabled: bool = False
driver.configure.bluetooth.measurement.bhRate.limit.p4H.set_devm(value =␣
↪structure)
```

No command help available

param value
see the help for DevmStruct structure arguments.

set_sgacp(*value: SgacpStruct*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:BHRate:LIMit:P4H:SGACp
structure = driver.configure.bluetooth.measurement.bhRate.limit.p4H.
↪SgacpStruct()
structure.Ptx_Rel_Limit: float = 1.0
structure.Ptx_Abs_Limit_1: float = 1.0
structure.Ptx_Abs_Limit_2: float = 1.0
structure.Ptx_Rel_Enable: bool = False
structure.Ptx_Abs_1_Enable: bool = False
structure.Ptx_Abs_2_Enable: bool = False
driver.configure.bluetooth.measurement.bhRate.limit.p4H.set_sgacp(value =␣
↪structure)
```

No command help available

param value
see the help for SgacpStruct structure arguments.

6.4.1.1.1.5 P8H

SCPI Commands :

```
CONFigure:BLUetooth:MEASurement<Instance>:BHRate:LIMit:P8H:DEVM
CONFigure:BLUetooth:MEASurement<Instance>:BHRate:LIMit:P8H:SGACp
```

class P8HCls

P8H commands group definition. 2 total commands, 0 Subgroups, 2 group commands

class DevmStruct

Structure for setting input parameters. Fields:

- Rms: float: No parameter help available
- Peak: float: No parameter help available
- P_99: float: No parameter help available
- Rms_Enabled: List[bool]: No parameter help available
- Peak_Enabled: List[bool]: No parameter help available
- P_99_Enabled: bool: No parameter help available

class SgacpStruct

Structure for setting input parameters. Fields:

- Ptx_Rel_Limit: float: No parameter help available
- Ptx_Abs_Limit_1: float: No parameter help available
- Ptx_Abs_Limit_2: float: No parameter help available
- Ptx_Abs_Limit_3: float: No parameter help available
- Ptx_Rel_Enabled: bool: No parameter help available
- Ptx_Abs_1_Enable: bool: No parameter help available
- Ptx_Abs_2_Enable: bool: No parameter help available
- Ptx_Abs_3_Enable: bool: No parameter help available

get_devm() → DevmStruct

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:BHRate:LIMit:P8H:DEVM
value: DevmStruct = driver.configure.bluetooth.measurement.bhRate.limit.p8H.get_
↪devm()
```

No command help available

return
structure: for return value, see the help for DevmStruct structure arguments.

get_sgacp() → SgacpStruct

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:BHRate:LIMit:P8H:SGACp
value: SgacpStruct = driver.configure.bluetooth.measurement.bhRate.limit.p8H.
↪get_sgacp()
```

No command help available

return
structure: for return value, see the help for SgacpStruct structure arguments.

set_devm(*value: DevmStruct*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:BHRate:LIMit:P8H:DEVM
structure = driver.configure.bluetooth.measurement.bhRate.limit.p8H.DevmStruct()
structure.Rms: float = 1.0
structure.Peak: float = 1.0
structure.P_99: float = 1.0
structure.Rms_Enabled: List[bool] = [True, False, True]
structure.Peak_Enabled: List[bool] = [True, False, True]
structure.P_99_Enabled: bool = False
driver.configure.bluetooth.measurement.bhRate.limit.p8H.set_devm(value =␣
↪structure)
```

No command help available

param value
see the help for DevmStruct structure arguments.

set_sgacp(*value: SgacpStruct*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:BHRate:LIMit:P8H:SGACp
structure = driver.configure.bluetooth.measurement.bhRate.limit.p8H.
↪SgacpStruct()
structure.Ptx_Rel_Limit: float = 1.0
structure.Ptx_Abs_Limit_1: float = 1.0
structure.Ptx_Abs_Limit_2: float = 1.0
structure.Ptx_Abs_Limit_3: float = 1.0
structure.Ptx_Rel_Enabled: bool = False
structure.Ptx_Abs_1_Enable: bool = False
structure.Ptx_Abs_2_Enable: bool = False
structure.Ptx_Abs_3_Enable: bool = False
driver.configure.bluetooth.measurement.bhRate.limit.p8H.set_sgacp(value =␣
↪structure)
```

No command help available

param value
see the help for SgacpStruct structure arguments.

6.4.1.1.1.6 Pencoding

SCPI Command :

```
CONFigure:BLUetooth:MEASurement<Instance>:BHRate:LIMit:PENCoding
```

class PencodingCls

Pencoding commands group definition. 1 total commands, 0 Subgroups, 1 group commands

class PencodingStruct

Response structure. Fields:

- Sync_Bit_Upper: int: No parameter help available

- Trailer_Bit_Upper: int: No parameter help available
- Sync_Bit_Enable: bool: No parameter help available
- Trailer_Bit_Enable: bool: No parameter help available

get() → PencodingStruct

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:BHRate:LIMit:PENCoding
value: PencodingStruct = driver.configure.bluetooth.measurement.bhRate.limit.
↪pencoding.get()
```

No command help available

return
structure: for return value, see the help for PencodingStruct structure arguments.

set(*sync_bit_upper: int*, *trailer_bit_upper: int*, *sync_bit_enable: bool*, *trailer_bit_enable: bool*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:BHRate:LIMit:PENCoding
driver.configure.bluetooth.measurement.bhRate.limit.pencoding.set(sync_bit_
↪upper = 1, trailer_bit_upper = 1, sync_bit_enable = False, trailer_bit_enable␣
↪= False)
```

No command help available

param sync_bit_upper
No help available

param trailer_bit_upper
No help available

param sync_bit_enable
No help available

param trailer_bit_enable
No help available

6.4.1.1.1.7 PowerVsTime

SCPI Command :

```
CONFigure:BLUetooth:MEASurement<Instance>:BHRate:LIMit:PVTime
```

class PowerVsTimeCls

PowerVsTime commands group definition. 1 total commands, 0 Subgroups, 1 group commands

class PowerVsTimeStruct

Response structure. Fields:

- Psk_Min_Gfsk_Low: float: No parameter help available
- Psk_Min_Gfsk_Upp: float: No parameter help available
- Psk_Min_Gfsk_Enable: List[bool]: No parameter help available

get() → PowerVsTimeStruct

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:BHRate:LIMit:PVTime
value: PowerVsTimeStruct = driver.configure.bluetooth.measurement.bhRate.limit.
↪powerVsTime.get()
```

No command help available

return
structure: for return value, see the help for PowerVsTimeStruct structure arguments.

set(*psk_min_gfsk_low: float*, *psk_min_gfsk_upp: float*, *psk_min_gfsk_enable: List[bool]*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:BHRate:LIMit:PVTime
driver.configure.bluetooth.measurement.bhRate.limit.powerVsTime.set(psk_min_
↪gfsk_low = 1.0, psk_min_gfsk_upp = 1.0, psk_min_gfsk_enable = [True, False,␣
↪True])
```

No command help available

param psk_min_gfsk_low
No help available

param psk_min_gfsk_upp
No help available

param psk_min_gfsk_enable
No help available

6.4.1.1.1.8 Result

SCPI Commands :

```
CONFigure:BLUetooth:MEASurement<Instance>:BHRate:RESult[:ALL]
CONFigure:BLUetooth:MEASurement<Instance>:BHRate:RESult:PVTime
CONFigure:BLUetooth:MEASurement<Instance>:BHRate:RESult:SGACp
CONFigure:BLUetooth:MEASurement<Instance>:BHRate:RESult:PENCoding
CONFigure:BLUetooth:MEASurement<Instance>:BHRate:RESult:IQERr
CONFigure:BLUetooth:MEASurement<Instance>:BHRate:RESult:IQDiff
CONFigure:BLUetooth:MEASurement<Instance>:BHRate:RESult:IQABsolute
CONFigure:BLUetooth:MEASurement<Instance>:BHRate:RESult:PDIFference
CONFigure:BLUetooth:MEASurement<Instance>:BHRate:RESult:DEVMagnitude
CONFigure:BLUetooth:MEASurement<Instance>:BHRate:RESult:TXSCalar
```

class ResultCls

Result commands group definition. 10 total commands, 0 Subgroups, 10 group commands

class AllStruct

Structure for setting input parameters. Fields:

- Devm: bool: No parameter help available
- Phase_Diff: bool: No parameter help available
- Tx_Scalars: bool: No parameter help available
- Iq_Absolute: bool: No parameter help available

- Iq_Differential: bool: No parameter help available
- Iq_Error: bool: No parameter help available
- Phase_Encoding: bool: No parameter help available
- Power_Vs_Time: bool: No parameter help available
- Spectrum_Gat_Acp: bool: No parameter help available

get_all() → AllStruct

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:BHRate:RESult[:ALL]
value: AllStruct = driver.configure.bluetooth.measurement.bhRate.result.get_
↪all()
```

No command help available

return
structure: for return value, see the help for AllStruct structure arguments.

get_dev_magnitude() → bool

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:BHRate:RESult:DEVMagnitude
value: bool = driver.configure.bluetooth.measurement.bhRate.result.get_dev_
↪magnitude()
```

No command help available

return
state: No help available

get_iq_absolute() → bool

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:BHRate:RESult:IQABsolute
value: bool = driver.configure.bluetooth.measurement.bhRate.result.get_iq_
↪absolute()
```

No command help available

return
state: No help available

get_iq_difference() → bool

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:BHRate:RESult:IQDiff
value: bool = driver.configure.bluetooth.measurement.bhRate.result.get_iq_
↪difference()
```

No command help available

return
state: No help available

get_iq_error() → bool

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:BHRate:RESult:IQERr
value: bool = driver.configure.bluetooth.measurement.bhRate.result.get_iq_
↪error()
```

No command help available

return
state: No help available

get_pdifference() → bool

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:BHRate:RESult:PDIFference
value: bool = driver.configure.bluetooth.measurement.bhRate.result.get_
↪pdifference()
```

No command help available

return
state: No help available

get_pencoding() → bool

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:BHRate:RESult:PENCoding
value: bool = driver.configure.bluetooth.measurement.bhRate.result.get_
↪pencoding()
```

No command help available

return
state: No help available

get_power_vs_time() → bool

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:BHRate:RESult:PVTime
value: bool = driver.configure.bluetooth.measurement.bhRate.result.get_power_vs_
↪time()
```

No command help available

return
state: No help available

get_sgacp() → bool

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:BHRate:RESult:SGACp
value: bool = driver.configure.bluetooth.measurement.bhRate.result.get_sgacp()
```

No command help available

return
state: No help available

get_tx_scalar() → bool

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:BHRate:RESult:TXSCalar
value: bool = driver.configure.bluetooth.measurement.bhRate.result.get_tx_
↪scalar()
```

No command help available

return
state: No help available

set_all(*value: AllStruct*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:BHRate:RESult[:ALL]
structure = driver.configure.bluetooth.measurement.bhRate.result.AllStruct()
structure.Devm: bool = False
structure.Phase_Diff: bool = False
structure.Tx_Scalars: bool = False
structure.Iq_Absolute: bool = False
structure.Iq_Differential: bool = False
structure.Iq_Error: bool = False
structure.Phase_Encoding: bool = False
structure.Power_Vs_Time: bool = False
structure.Spectrum_Gat_Acp: bool = False
driver.configure.bluetooth.measurement.bhRate.result.set_all(value = structure)
```

No command help available

param value
see the help for AllStruct structure arguments.

set_dev_magnitude(*state: bool*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:BHRate:RESult:DEVMagnitude
driver.configure.bluetooth.measurement.bhRate.result.set_dev_magnitude(state =␣
↪False)
```

No command help available

param state
No help available

set_iq_absolute(*state: bool*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:BHRate:RESult:IQABsolute
driver.configure.bluetooth.measurement.bhRate.result.set_iq_absolute(state =␣
↪False)
```

No command help available

param state
No help available

set_iq_difference(*state: bool*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:BHRate:RESult:IQDiff
driver.configure.bluetooth.measurement.bhRate.result.set_iq_difference(state =␣
↪False)
```

No command help available

param state
No help available

set_iq_error(*state: bool*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:BHRate:RESult:IQERr
driver.configure.bluetooth.measurement.bhRate.result.set_iq_error(state = False)
```

No command help available

param state
No help available

set_pdifference(*state: bool*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:BHRate:RESult:PDIFference
driver.configure.bluetooth.measurement.bhRate.result.set_pdifference(state =␣
↪False)
```

No command help available

param state
No help available

set_pencoding(*state: bool*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:BHRate:RESult:PENCoding
driver.configure.bluetooth.measurement.bhRate.result.set_pencoding(state =␣
↪False)
```

No command help available

param state
No help available

set_power_vs_time(*state: bool*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:BHRate:RESult:PVTime
driver.configure.bluetooth.measurement.bhRate.result.set_power_vs_time(state =␣
↪False)
```

No command help available

param state
No help available

set_sgacp(*state: bool*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:BHRate:RESult:SGACp
driver.configure.bluetooth.measurement.bhRate.result.set_sgacp(state = False)
```

No command help available

param state
No help available

set_tx_scalar(*state: bool*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:BHRate:RESult:TXSCalar
driver.configure.bluetooth.measurement.bhRate.result.set_tx_scalar(state =␣
↪False)
```

No command help available

param state
No help available

6.4.1.1.1.9 Scount

SCPI Commands :

```
CONFigure:BLUetooth:MEASurement<Instance>:BHRate:SCOunt:PVTime
CONFigure:BLUetooth:MEASurement<Instance>:BHRate:SCOunt:SGACp
CONFigure:BLUetooth:MEASurement<Instance>:BHRate:SCOunt:MODulation
CONFigure:BLUetooth:MEASurement<Instance>:BHRate:SCOunt:PENCoding
```

class ScountCls

Scount commands group definition. 4 total commands, 0 Subgroups, 4 group commands

get_modulation() → int

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:BHRate:SCOunt:MODulation
value: int = driver.configure.bluetooth.measurement.bhRate.scount.get_
↪modulation()
```

No command help available

return
stat_count: No help available

get_pencoding() → int

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:BHRate:SCOunt:PENCoding
value: int = driver.configure.bluetooth.measurement.bhRate.scount.get_
↪pencoding()
```

No command help available

return
stat_count: No help available

get_power_vs_time() → int

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:BHRate:SCOunt:PVTime
value: int = driver.configure.bluetooth.measurement.bhRate.scount.get_power_vs_
↪time()
```

No command help available

return
stat_count: No help available

get_sgacp() → int

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:BHRate:SCOunt:SGACp
value: int = driver.configure.bluetooth.measurement.bhRate.scount.get_sgacp()
```

No command help available

return
stat_count: No help available

set_modulation(*stat_count: int*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:BHRate:SCOunt:MODulation
driver.configure.bluetooth.measurement.bhRate.scount.set_modulation(stat_count␣
↪= 1)
```

No command help available

param stat_count
No help available

set_pencoding(*stat_count: int*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:BHRate:SCOunt:PENCoding
driver.configure.bluetooth.measurement.bhRate.scount.set_pencoding(stat_count =␣
↪1)
```

No command help available

param stat_count
No help available

set_power_vs_time(*stat_count: int*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:BHRate:SCOunt:PVTime
driver.configure.bluetooth.measurement.bhRate.scount.set_power_vs_time(stat_
↪count = 1)
```

No command help available

param stat_count
No help available

set_sgacp(*stat_count: int*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:BHRate:SCOunt:SGACp
driver.configure.bluetooth.measurement.bhRate.scount.set_sgacp(stat_count = 1)
```

No command help available

param stat_count
No help available

6.4.1.1.1.10 Sgacp

class SgacpCls

Sgacp commands group definition. 1 total commands, 1 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.configure.bluetooth.measurement.bhRate.sgacp.clone()
```

Subgroups

6.4.1.1.1.11 Measurement

SCPI Command :

```
CONFigure:BLUetooth:MEASurement<Instance>:BHRate:SGACp:MEASurement:MODE
```

class MeasurementCls

Measurement commands group definition. 1 total commands, 0 Subgroups, 1 group commands

get_mode() → BrEdrChannelsRange

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:BHRate:SGACp:MEASurement:MODE
value: enums.BrEdrChannelsRange = driver.configure.bluetooth.measurement.bhRate.
↪sgacp.measurement.get_mode()
```

No command help available

return
meas_mode: No help available

set_mode(*meas_mode: BrEdrChannelsRange*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:BHRate:SGACp:MEASurement:MODE
driver.configure.bluetooth.measurement.bhRate.sgacp.measurement.set_mode(meas_
↪mode = enums.BrEdrChannelsRange.CH21)
```

No command help available

param meas_mode
No help available

6.4.1.1.2 ComSettings

SCPI Commands :

```
CONFigure:BLUetooth:MEASurement<Instance>:COMSettings:COMPort
CONFigure:BLUetooth:MEASurement<Instance>:COMSettings:BAUDrate
CONFigure:BLUetooth:MEASurement<Instance>:COMSettings:STOPbits
CONFigure:BLUetooth:MEASurement<Instance>:COMSettings:PARity
CONFigure:BLUetooth:MEASurement<Instance>:COMSettings:PROTocol
CONFigure:BLUetooth:MEASurement<Instance>:COMSettings:DBITs
```

class ComSettingsCls

ComSettings commands group definition. 7 total commands, 1 Subgroups, 6 group commands

get_baud_rate() → BaudRate

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:COMSettings:BAUDrate
value: enums.BaudRate = driver.configure.bluetooth.measurement.comSettings.get_
↪baud_rate()
```

No command help available

return
baud_rate: No help available

get_com_port() → int

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:COMSettings:COMPort
value: int = driver.configure.bluetooth.measurement.comSettings.get_com_port()
```

No command help available

return
no: No help available

get_dbits() → DataBits

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:COMSettings:DBITs
value: enums.DataBits = driver.configure.bluetooth.measurement.comSettings.get_
↪dbits()
```

No command help available

return
data_bits: No help available

get_parity() → Parity

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:COMSettings:PARity
value: enums.Parity = driver.configure.bluetooth.measurement.comSettings.get_
↪parity()
```

No command help available

return
parity: No help available

get_protocol() → Protocol

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:COMSettings:PROTocol
value: enums.Protocol = driver.configure.bluetooth.measurement.comSettings.get_
↪protocol()
```

No command help available

return
protocol: No help available

get_stop_bits() → StopBits

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:COMSettings:STOPbits
value: enums.StopBits = driver.configure.bluetooth.measurement.comSettings.get_
↪stop_bits()
```

No command help available

return
stop_bits: No help available

set_baud_rate(*baud_rate: BaudRate*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:COMSettings:BAUDrate
driver.configure.bluetooth.measurement.comSettings.set_baud_rate(baud_rate =␣
↪enums.BaudRate.B110)
```

No command help available

param baud_rate
No help available

set_com_port(*no: int*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:COMSettings:COMPort
driver.configure.bluetooth.measurement.comSettings.set_com_port(no = 1)
```

No command help available

param no
No help available

set_dbits(*data_bits: DataBits*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:COMSettings:DBITs
driver.configure.bluetooth.measurement.comSettings.set_dbits(data_bits = enums.
↪DataBits.D7)
```

No command help available

param data_bits
No help available

set_parity(*parity: Parity*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:COMSettings:PARity
driver.configure.bluetooth.measurement.comSettings.set_parity(parity = enums.
↪Parity.EVEN)
```

No command help available

param parity
No help available

set_protocol(*protocol: Protocol*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:COMSettings:PROTocol
driver.configure.bluetooth.measurement.comSettings.set_protocol(protocol =␣
↪enums.Protocol.CTSRts)
```

No command help available

param protocol
No help available

set_stop_bits(*stop_bits: StopBits*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:COMSettings:STOPbits
driver.configure.bluetooth.measurement.comSettings.set_stop_bits(stop_bits =␣
↪enums.StopBits.S1)
```

No command help available

param stop_bits
No help available

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.configure.bluetooth.measurement.comSettings.clone()
```

Subgroups

6.4.1.1.2.1 Ports

SCPI Command :

```
CONFigure:BLUetooth:MEASurement<Instance>:COMSettings:PORTs:CATalog
```

class PortsCls

Ports commands group definition. 1 total commands, 0 Subgroups, 1 group commands

class CatalogStruct

Structure for reading output parameters. Fields:

- No_Devices: int: No parameter help available
- Item_Number: List[int]: No parameter help available
- Discovered_Port: List[str]: No parameter help available

get_catalog() → CatalogStruct

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:COMSettings:PORTs:CATalog
value: CatalogStruct = driver.configure.bluetooth.measurement.comSettings.ports.
↪get_catalog()
```

No command help available

return
structure: for return value, see the help for CatalogStruct structure arguments.

6.4.1.1.3 Display

SCPI Command :

```
CONFigure:BLUetooth:MEASurement<Instance>:DISPlay
```

class DisplayCls

Display commands group definition. 1 total commands, 0 Subgroups, 1 group commands

class DisplayStruct

Response structure. Fields:

- Measurement: enums.DisplayMeasurement: No parameter help available
- View: enums.DisplayView: No parameter help available

get() → DisplayStruct

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:DISPlay
value: DisplayStruct = driver.configure.bluetooth.measurement.display.get()
```

No command help available

return
structure: for return value, see the help for DisplayStruct structure arguments.

set(*measurement: DisplayMeasurement*, *view: DisplayView = None*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:DISPlay
driver.configure.bluetooth.measurement.display.set(measurement = enums.
↪DisplayMeasurement.MEV, view = enums.DisplayView.DEVM)
```

No command help available

param measurement
No help available

param view
No help available

6.4.1.1.4 DtMode

class DtModeCls

DtMode commands group definition. 23 total commands, 1 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.configure.bluetooth.measurement.dtMode.clone()
```

Subgroups

6.4.1.1.4.1 RxQuality

class RxQualityCls

RxQuality commands group definition. 23 total commands, 6 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.configure.bluetooth.measurement.dtMode.rxQuality.clone()
```

Subgroups

6.4.1.1.4.2 Eattenuation

SCPI Command :

```
CONFigure:BLUetooth:MEASurement<Instance>:DTMode:RXQuality:EATTenuation:OUTPut
```

class EattenuationCls

Eattenuation commands group definition. 1 total commands, 0 Subgroups, 1 group commands

get_output() → float

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:DTMode:RXQuality:EATTenuation:OUTPut
value: float = driver.configure.bluetooth.measurement.dtMode.rxQuality.
↪eattenuation.get_output()
```

No command help available

return
atten: No help available

set_output(*atten: float*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:DTMode:RXQuality:EATTenuation:OUTPut
driver.configure.bluetooth.measurement.dtMode.rxQuality.eattenuation.set_
↪output(atten = 1.0)
```

No command help available

param atten
No help available

6.4.1.1.4.3 Limit

class LimitCls

Limit commands group definition. 3 total commands, 1 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.configure.bluetooth.measurement.dtMode.rxQuality.limit.clone()
```

Subgroups

6.4.1.1.4.4 Mper

class MperCls

Mper commands group definition. 3 total commands, 1 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.configure.bluetooth.measurement.dtMode.rxQuality.limit.mper.clone()
```

Subgroups

6.4.1.1.4.5 LowEnergy

SCPI Commands :

```
CONFigure:BLUetooth:MEASurement<Instance>:DTMode:RXQuality:LIMit:MPER:LENergy:LE1M
CONFigure:BLUetooth:MEASurement<Instance>:DTMode:RXQuality:LIMit:MPER:LENergy:LE2M
CONFigure:BLUetooth:MEASurement<Instance>:DTMode:RXQuality:LIMit:MPER:LENergy:LRANge
```

class LowEnergyCls

LowEnergy commands group definition. 3 total commands, 0 Subgroups, 3 group commands

get_le_1_m() → float

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:DTMode:RXQuality:LIMit:MPER:LENergy:LE1M
value: float or bool = driver.configure.bluetooth.measurement.dtMode.rxQuality.
↪limit.mper.lowEnergy.get_le_1_m()
```

No command help available

return
limit: (float or boolean) No help available

get_le_2_m() → float

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:DTMode:RXQuality:LIMit:MPER:LENergy:LE2M
value: float or bool = driver.configure.bluetooth.measurement.dtMode.rxQuality.
↪limit.mper.lowEnergy.get_le_2_m()
```

No command help available

return
limit: (float or boolean) No help available

get_lrange() → float

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:DTMode:RXQuality:LIMit:MPER:LENergy:LRANge
value: float or bool = driver.configure.bluetooth.measurement.dtMode.rxQuality.
↪limit.mper.lowEnergy.get_lrange()
```

No command help available

return
limit: (float or boolean) No help available

set_le_1_m(*limit: float*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:DTMode:RXQuality:LIMit:MPER:LENergy:LE1M
driver.configure.bluetooth.measurement.dtMode.rxQuality.limit.mper.lowEnergy.
↪set_le_1_m(limit = 1.0)
```

No command help available

param limit
(float or boolean) No help available

set_le_2_m(*limit: float*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:DTMode:RXQuality:LIMit:MPER:LENergy:LE2M
driver.configure.bluetooth.measurement.dtMode.rxQuality.limit.mper.lowEnergy.
↪set_le_2_m(limit = 1.0)
```

No command help available

param limit
(float or boolean) No help available

set_lrange(*limit: float*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:DTMode:RXQuality:LIMit:MPER:LENergy:LRANge
driver.configure.bluetooth.measurement.dtMode.rxQuality.limit.mper.lowEnergy.
↪set_lrange(limit = 1.0)
```

No command help available

param limit
(float or boolean) No help available

6.4.1.1.4.6 Per

SCPI Command :

```
CONFigure:BLUetooth:MEASurement<Instance>:DTMode:RXQuality:PER:LEVel
```

class PerCls

Per commands group definition. 4 total commands, 1 Subgroups, 1 group commands

get_level() → float

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:DTMode:RXQuality:PER:LEVel
value: float = driver.configure.bluetooth.measurement.dtMode.rxQuality.per.get_
↪level()
```

No command help available

return
level: No help available

set_level(*level: float*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:DTMode:RXQuality:PER:LEVel
driver.configure.bluetooth.measurement.dtMode.rxQuality.per.set_level(level = 1.
↪0)
```

No command help available

param level
No help available

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.configure.bluetooth.measurement.dtMode.rxQuality.per.clone()
```

Subgroups

6.4.1.1.4.7 Packets

class PacketsCls

Packets commands group definition. 3 total commands, 1 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.configure.bluetooth.measurement.dtMode.rxQuality.per.packets.clone()
```

Subgroups

6.4.1.1.4.8 LowEnergy

SCPI Commands :

```
CONFigure:BLUetooth:MEASurement<Instance>:DTMode:RXQuality:PER:PACKets:LENergy:LE1M
CONFigure:BLUetooth:MEASurement<Instance>:DTMode:RXQuality:PER:PACKets:LENergy:LE2M
CONFigure:BLUetooth:MEASurement<Instance>:DTMode:RXQuality:PER:PACKets:LENergy:LRANge
```

class LowEnergyCls

LowEnergy commands group definition. 3 total commands, 0 Subgroups, 3 group commands

get_le_1_m() → int

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:DTMode:RXQuality:PER:PACKets:LENergy:LE1M
value: int = driver.configure.bluetooth.measurement.dtMode.rxQuality.per.
↪packets.lowEnergy.get_le_1_m()
```

No command help available

return
number_of_packets: No help available

get_le_2_m() → int

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:DTMode:RXQuality:PER:PACKets:LENergy:LE2M
value: int = driver.configure.bluetooth.measurement.dtMode.rxQuality.per.
↪packets.lowEnergy.get_le_2_m()
```

No command help available

return
number_of_packets: No help available

get_lrange() → int

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:DTMode:RXQuality:PER:PACKets:LENergy:LRANge
value: int = driver.configure.bluetooth.measurement.dtMode.rxQuality.per.
↪packets.lowEnergy.get_lrange()
```

No command help available

return
number_of_packets: No help available

set_le_1_m(*number_of_packets: int*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:DTMode:RXQuality:PER:PACKets:LENergy:LE1M
driver.configure.bluetooth.measurement.dtMode.rxQuality.per.packets.lowEnergy.
↪set_le_1_m(number_of_packets = 1)
```

No command help available

param number_of_packets
No help available

set_le_2_m(*number_of_packets: int*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:DTMode:RXQuality:PER:PACKets:LENergy:LE2M
driver.configure.bluetooth.measurement.dtMode.rxQuality.per.packets.lowEnergy.
↪set_le_2_m(number_of_packets = 1)
```

No command help available

param number_of_packets
No help available

set_lrange(*number_of_packets: int*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:DTMode:RXQuality:PER:PACKets:LENergy:LRANge
driver.configure.bluetooth.measurement.dtMode.rxQuality.per.packets.lowEnergy.
↪set_lrange(number_of_packets = 1)
```

No command help available

param number_of_packets
No help available

6.4.1.1.4.9 Rintegrity

class RintegrityCls

Rintegrity commands group definition. 3 total commands, 1 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.configure.bluetooth.measurement.dtMode.rxQuality.rintegrity.clone()
```

Subgroups

6.4.1.1.4.10 LowEnergy

SCPI Commands :

```
CONFigure:BLUetooth:MEASurement<Instance>:DTMode:RXQuality:RINTegrity:LENergy:LE1M
CONFigure:BLUetooth:MEASurement<Instance>:DTMode:RXQuality:RINTegrity:LENergy:LE2M
CONFigure:BLUetooth:MEASurement<Instance>:DTMode:RXQuality:RINTegrity:LENergy:LRANge
```

class LowEnergyCls

LowEnergy commands group definition. 3 total commands, 0 Subgroups, 3 group commands

get_le_1_m() → bool

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:DTMode:RXQuality:RINTegrity:LENergy:LE1M
value: bool = driver.configure.bluetooth.measurement.dtMode.rxQuality.
↪rintegrity.lowEnergy.get_le_1_m()
```

No command help available

return
report_integrity: No help available

get_le_2_m() → bool

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:DTMode:RXQuality:RINTegrity:LENergy:LE2M
value: bool = driver.configure.bluetooth.measurement.dtMode.rxQuality.
↪rintegrity.lowEnergy.get_le_2_m()
```

No command help available

return
report_integrity: No help available

get_lrange() → bool

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:DTMode:RXQuality:RINTegrity:LENergy:LRANge
value: bool = driver.configure.bluetooth.measurement.dtMode.rxQuality.
↪rintegrity.lowEnergy.get_lrange()
```

No command help available

return
report_integrity: No help available

set_le_1_m(*report_integrity: bool*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:DTMode:RXQuality:RINTegrity:LENergy:LE1M
driver.configure.bluetooth.measurement.dtMode.rxQuality.rintegrity.lowEnergy.
↪set_le_1_m(report_integrity = False)
```

No command help available

param report_integrity
No help available

set_le_2_m(*report_integrity: bool*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:DTMode:RXQuality:RINTegrity:LENergy:LE2M
driver.configure.bluetooth.measurement.dtMode.rxQuality.rintegrity.lowEnergy.
↪set_le_2_m(report_integrity = False)
```

No command help available

param report_integrity
No help available

set_lrange(*report_integrity: bool*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:DTMode:RXQuality:RINTegrity:LENergy:LRANge
driver.configure.bluetooth.measurement.dtMode.rxQuality.rintegrity.lowEnergy.
↪set_lrange(report_integrity = False)
```

No command help available

param report_integrity
No help available

6.4.1.1.4.11 Search

SCPI Commands :

```
CONFigure:BLUetooth:MEASurement<Instance>:DTMode:RXQuality:SEARch:STARtlevel
CONFigure:BLUetooth:MEASurement<Instance>:DTMode:RXQuality:SEARch:STEP
```

class SearchCls

Search commands group definition. 11 total commands, 3 Subgroups, 2 group commands

get_start_level() → float

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:DTMode:RXQuality:SEARch:STARtlevel
value: float or bool = driver.configure.bluetooth.measurement.dtMode.rxQuality.
↪search.get_start_level()
```

No command help available

return
start_level: (float or boolean) No help available

get_step() → float

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:DTMode:RXQuality:SEARch:STEP
value: float or bool = driver.configure.bluetooth.measurement.dtMode.rxQuality.
↪search.get_step()
```

No command help available

return
level_step: (float or boolean) No help available

set_start_level(*start_level: float*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:DTMode:RXQuality:SEARch:STARtlevel
driver.configure.bluetooth.measurement.dtMode.rxQuality.search.set_start_
↪level(start_level = 1.0)
```

No command help available

param start_level
(float or boolean) No help available

set_step(*level_step: float*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:DTMode:RXQuality:SEARch:STEP
driver.configure.bluetooth.measurement.dtMode.rxQuality.search.set_step(level_
↪step = 1.0)
```

No command help available

param level_step
(float or boolean) No help available

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.configure.bluetooth.measurement.dtMode.rxQuality.search.clone()
```

Subgroups

6.4.1.1.4.12 Limit

class LimitCls

Limit commands group definition. 3 total commands, 1 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.configure.bluetooth.measurement.dtMode.rxQuality.search.limit.clone()
```

Subgroups

6.4.1.1.4.13 Mper

class MperCls

Mper commands group definition. 3 total commands, 1 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.configure.bluetooth.measurement.dtMode.rxQuality.search.limit.mper.
↪clone()
```

Subgroups

6.4.1.1.4.14 LowEnergy

SCPI Commands :

```
CONFigure:BLUetooth:MEASurement<Instance>:DTMode:RXQuality:SEARch:LIMit:MPER:LENergy:LE1M
CONFigure:BLUetooth:MEASurement<Instance>:DTMode:RXQuality:SEARch:LIMit:MPER:LENergy:LE2M
CONFigure:BLUetooth:MEASurement<Instance>
↪:DTMode:RXQuality:SEARch:LIMit:MPER:LENergy:LRANge
```

class LowEnergyCls

LowEnergy commands group definition. 3 total commands, 0 Subgroups, 3 group commands

get_le_1_m() → float

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:DTMode:RXQuality:SEARch:LIMit:MPER:LENergy:LE1M
value: float or bool = driver.configure.bluetooth.measurement.dtMode.rxQuality.
↪search.limit.mper.lowEnergy.get_le_1_m()
```

No command help available

return
limit: (float or boolean) No help available

get_le_2_m() → float

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:DTMode:RXQuality:SEARch:LIMit:MPER:LENergy:LE2M
value: float or bool = driver.configure.bluetooth.measurement.dtMode.rxQuality.
↪search.limit.mper.lowEnergy.get_le_2_m()
```

No command help available

return
limit: (float or boolean) No help available

get_lrange() → float

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:DTMode:RXQuality:SEARch:LIMit:MPER:LENergy:LRANge
value: float or bool = driver.configure.bluetooth.measurement.dtMode.rxQuality.
↪search.limit.mper.lowEnergy.get_lrange()
```

No command help available

return
limit: (float or boolean) No help available

set_le_1_m(*limit: float*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:DTMode:RXQuality:SEARch:LIMit:MPER:LENergy:LE1M
driver.configure.bluetooth.measurement.dtMode.rxQuality.search.limit.mper.
↪lowEnergy.set_le_1_m(limit = 1.0)
```

No command help available

param limit
(float or boolean) No help available

set_le_2_m(*limit: float*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:DTMode:RXQuality:SEARch:LIMit:MPER:LENergy:LE2M
driver.configure.bluetooth.measurement.dtMode.rxQuality.search.limit.mper.
↪lowEnergy.set_le_2_m(limit = 1.0)
```

No command help available

param limit
(float or boolean) No help available

set_lrange(*limit: float*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:DTMode:RXQuality:SEARch:LIMit:MPER:LENergy:LRANge
driver.configure.bluetooth.measurement.dtMode.rxQuality.search.limit.mper.
↪lowEnergy.set_lrange(limit = 1.0)
```

No command help available

param limit
(float or boolean) No help available

6.4.1.1.4.15 Packets

class PacketsCls

Packets commands group definition. 3 total commands, 1 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.configure.bluetooth.measurement.dtMode.rxQuality.search.packets.clone()
```

Subgroups

6.4.1.1.4.16 LowEnergy

SCPI Commands :

```
CONFigure:BLUetooth:MEASurement<Instance>:DTMode:RXQuality:SEARch:PACKets:LENergy:LE1M
CONFigure:BLUetooth:MEASurement<Instance>:DTMode:RXQuality:SEARch:PACKets:LENergy:LE2M
CONFigure:BLUetooth:MEASurement<Instance>:DTMode:RXQuality:SEARch:PACKets:LENergy:LRANge
```

class LowEnergyCls

LowEnergy commands group definition. 3 total commands, 0 Subgroups, 3 group commands

get_le_1_m() → int

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:DTMode:RXQuality:SEARch:PACKets:LENergy:LE1M
value: int = driver.configure.bluetooth.measurement.dtMode.rxQuality.search.
↪packets.lowEnergy.get_le_1_m()
```

No command help available

return
number_of_packets: No help available

get_le_2_m() → int

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:DTMode:RXQuality:SEARch:PACKets:LENergy:LE2M
value: int = driver.configure.bluetooth.measurement.dtMode.rxQuality.search.
↪packets.lowEnergy.get_le_2_m()
```

No command help available

return
number_of_packets: No help available

get_lrange() → int

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:DTMode:RXQuality:SEARch:PACKets:LENergy:LRANge
value: int = driver.configure.bluetooth.measurement.dtMode.rxQuality.search.
↪packets.lowEnergy.get_lrange()
```

No command help available

return
number_of_packets: No help available

set_le_1_m(*number_of_packets: int*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:DTMode:RXQuality:SEARch:PACKets:LENergy:LE1M
driver.configure.bluetooth.measurement.dtMode.rxQuality.search.packets.
↪lowEnergy.set_le_1_m(number_of_packets = 1)
```

No command help available

param number_of_packets
No help available

set_le_2_m(*number_of_packets: int*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:DTMode:RXQuality:SEARch:PACKets:LENergy:LE2M
driver.configure.bluetooth.measurement.dtMode.rxQuality.search.packets.
↪lowEnergy.set_le_2_m(number_of_packets = 1)
```

No command help available

param number_of_packets
No help available

set_lrange(*number_of_packets: int*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:DTMode:RXQuality:SEARch:PACKets:LENergy:LRANge
driver.configure.bluetooth.measurement.dtMode.rxQuality.search.packets.
↪lowEnergy.set_lrange(number_of_packets = 1)
```

No command help available

param number_of_packets
No help available

6.4.1.1.4.17 Rintegrity

class RintegrityCls

Rintegrity commands group definition. 3 total commands, 1 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.configure.bluetooth.measurement.dtMode.rxQuality.search.rintegrity.
↪clone()
```

Subgroups

6.4.1.1.4.18 LowEnergy

SCPI Commands :

```
CONFigure:BLUetooth:MEASurement<Instance>:DTMode:RXQuality:SEARch:RINTegrity:LENergy:LE1M
CONFigure:BLUetooth:MEASurement<Instance>:DTMode:RXQuality:SEARch:RINTegrity:LENergy:LE2M
CONFigure:BLUetooth:MEASurement<Instance>
↪:DTMode:RXQuality:SEARch:RINTegrity:LENergy:LRANge
```

class LowEnergyCls

LowEnergy commands group definition. 3 total commands, 0 Subgroups, 3 group commands

get_le_1_m() → bool

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:DTMode:RXQuality:SEARch:RINTegrity:LENergy:LE1M
value: bool = driver.configure.bluetooth.measurement.dtMode.rxQuality.search.
↪rintegrity.lowEnergy.get_le_1_m()
```

No command help available

return
report_integrity: No help available

get_le_2_m() → bool

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:DTMode:RXQuality:SEARch:RINTegrity:LENergy:LE2M
value: bool = driver.configure.bluetooth.measurement.dtMode.rxQuality.search.
↪rintegrity.lowEnergy.get_le_2_m()
```

No command help available

return
report_integrity: No help available

get_lrange() → bool

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:DTMode:RXQuality:SEARch:RINTegrity:LENergy:LRANge
value: bool = driver.configure.bluetooth.measurement.dtMode.rxQuality.search.
↪rintegrity.lowEnergy.get_lrange()
```

No command help available

return
report_integrity: No help available

set_le_1_m(*report_integrity: bool*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:DTMode:RXQuality:SEARch:RINTegrity:LENergy:LE1M
driver.configure.bluetooth.measurement.dtMode.rxQuality.search.rintegrity.
↪lowEnergy.set_le_1_m(report_integrity = False)
```

No command help available

param report_integrity
No help available

set_le_2_m(*report_integrity: bool*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:DTMode:RXQuality:SEARch:RINTegrity:LENergy:LE2M
driver.configure.bluetooth.measurement.dtMode.rxQuality.search.rintegrity.
↪lowEnergy.set_le_2_m(report_integrity = False)
```

No command help available

param report_integrity
No help available

set_lrange(*report_integrity: bool*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:DTMode:RXQuality:SEARch:RINTegrity:LENergy:LRANge
driver.configure.bluetooth.measurement.dtMode.rxQuality.search.rintegrity.
↪lowEnergy.set_lrange(report_integrity = False)
```

No command help available

param report_integrity
No help available

6.4.1.1.4.19 SmIndex

SCPI Command :

```
CONFigure:BLUetooth:MEASurement<Instance>:DTMode:RXQuality:SMINdex:LENergy
```

class SmIndexCls

SmIndex commands group definition. 1 total commands, 0 Subgroups, 1 group commands

get_low_energy() → bool

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:DTMode:RXQuality:SMINdex:LENergy
value: bool = driver.configure.bluetooth.measurement.dtMode.rxQuality.smIndex.
↪get_low_energy()
```

No command help available

return
mod_index_type: No help available

set_low_energy(*mod_index_type: bool*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:DTMode:RXQuality:SMINdex:LENergy
driver.configure.bluetooth.measurement.dtMode.rxQuality.smIndex.set_low_
↪energy(mod_index_type = False)
```

No command help available

param mod_index_type
No help available

6.4.1.1.5 Hdr

SCPI Commands :

```
CONFigure:BLUetooth:MEASurement<Instance>:HDR:MOEXception
CONFigure:BLUetooth:MEASurement<Instance>:HDR:SCONdition
CONFigure:BLUetooth:MEASurement<Instance>:HDR:REPetition
```

class HdrCls

Hdr commands group definition. 34 total commands, 5 Subgroups, 3 group commands

get_mo_exception() → bool

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:HDR:MOEXception
value: bool = driver.configure.bluetooth.measurement.hdr.get_mo_exception()
```

No command help available

return
meas_on_exception: No help available

get_repetition() → Repeat

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:HDR:REPetition
value: enums.Repeat = driver.configure.bluetooth.measurement.hdr.get_
↪repetition()
```

No command help available

return
repetition: No help available

get_scondition() → StopCondition

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:HDR:SCONdition
value: enums.StopCondition = driver.configure.bluetooth.measurement.hdr.get_
↪scondition()
```

No command help available

return
stop_condition: No help available

set_mo_exception(*meas_on_exception: bool*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:HDR:MOEXception
driver.configure.bluetooth.measurement.hdr.set_mo_exception(meas_on_exception =␣
↪False)
```

No command help available

param meas_on_exception
No help available

set_repetition(*repetition: Repeat*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:HDR:REPetition
driver.configure.bluetooth.measurement.hdr.set_repetition(repetition = enums.
↪Repeat.CONTinuous)
```

No command help available

param repetition
No help available

set_scondition(*stop_condition: StopCondition*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:HDR:SCONdition
driver.configure.bluetooth.measurement.hdr.set_scondition(stop_condition =␣
↪enums.StopCondition.NONE)
```

No command help available

param stop_condition
No help available

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.configure.bluetooth.measurement.hdr.clone()
```

Subgroups

6.4.1.1.5.1 InputSignal

SCPI Commands :

```
CONFigure:BLUetooth:MEASurement<Instance>:HDR:ISIGnal:PATTern
CONFigure:BLUetooth:MEASurement<Instance>:HDR:ISIGnal:PLENgth
CONFigure:BLUetooth:MEASurement<Instance>:HDR:ISIGnal:PTYPe
CONFigure:BLUetooth:MEASurement<Instance>:HDR:ISIGnal:NAP
CONFigure:BLUetooth:MEASurement<Instance>:HDR:ISIGnal:UAP
CONFigure:BLUetooth:MEASurement<Instance>:HDR:ISIGnal:LAP
CONFigure:BLUetooth:MEASurement<Instance>:HDR:ISIGnal:BDADdress
CONFigure:BLUetooth:MEASurement<Instance>:HDR:ISIGnal:ASYNchronize
CONFigure:BLUetooth:MEASurement<Instance>:HDR:ISIGnal:DMODe
```

class InputSignalCls

InputSignal commands group definition. 9 total commands, 0 Subgroups, 9 group commands

get_asynchronize() → bool

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:HDR:ISIGnal:ASYNchronize
value: bool = driver.configure.bluetooth.measurement.hdr.inputSignal.get_
↪asynchronize()
```

No command help available

return
auto_sync: No help available

get_bd_address() → str

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:HDR:ISIGnal:BDADdress
value: str = driver.configure.bluetooth.measurement.hdr.inputSignal.get_bd_
↪address()
```

No command help available

return
bd_address: No help available

get_dmode() → AutoManualMode

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:HDR:ISIGnal:DMODe
value: enums.AutoManualMode = driver.configure.bluetooth.measurement.hdr.
↪inputSignal.get_dmode()
```

No command help available

return
detection_mode: No help available

get_lap() → str

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:HDR:ISIGnal:LAP
value: str = driver.configure.bluetooth.measurement.hdr.inputSignal.get_lap()
```

No command help available

return
lap_address: No help available

get_nap() → str

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:HDR:ISIGnal:NAP
value: str = driver.configure.bluetooth.measurement.hdr.inputSignal.get_nap()
```

No command help available

return
nap_address: No help available

get_pattern() → PatternTypeHdr

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:HDR:ISIGnal:PATTern
value: enums.PatternTypeHdr = driver.configure.bluetooth.measurement.hdr.
↪inputSignal.get_pattern()
```

No command help available

return
pattern_type: No help available

get_plength() → List[int]

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:HDR:ISIGnal:PLENgth
value: List[int] = driver.configure.bluetooth.measurement.hdr.inputSignal.get_
↪plength()
```

No command help available

return
payload_length: No help available

get_ptype() → PacketTypeC

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:HDR:ISIGnal:PTYPe
value: enums.PacketTypeC = driver.configure.bluetooth.measurement.hdr.
↪inputSignal.get_ptype()
```

No command help available

return
packet_type: No help available

get_uap() → str

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:HDR:ISIGnal:UAP
value: str = driver.configure.bluetooth.measurement.hdr.inputSignal.get_uap()
```

No command help available

return
uap_address: No help available

set_asynchronize(*auto_sync: bool*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:HDR:ISIGnal:ASYNchronize
driver.configure.bluetooth.measurement.hdr.inputSignal.set_asynchronize(auto_
↪sync = False)
```

No command help available

param auto_sync
No help available

set_bd_address(*bd_address: str*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:HDR:ISIGnal:BDADdress
driver.configure.bluetooth.measurement.hdr.inputSignal.set_bd_address(bd_
↪address = rawAbc)
```

No command help available

param bd_address
No help available

set_dmode(*detection_mode: AutoManualMode*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:HDR:ISIGnal:DMODe
driver.configure.bluetooth.measurement.hdr.inputSignal.set_dmode(detection_mode␣
↪= enums.AutoManualMode.AUTO)
```

No command help available

param detection_mode
No help available

set_lap(*lap_address: str*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:HDR:ISIGnal:LAP
driver.configure.bluetooth.measurement.hdr.inputSignal.set_lap(lap_address =␣
↪rawAbc)
```

No command help available

param lap_address
No help available

set_nap(*nap_address: str*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:HDR:ISIGnal:NAP
driver.configure.bluetooth.measurement.hdr.inputSignal.set_nap(nap_address =␣
↪rawAbc)
```

No command help available

param nap_address
No help available

set_pattern(*pattern_type: PatternTypeHdr*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:HDR:ISIGnal:PATTern
driver.configure.bluetooth.measurement.hdr.inputSignal.set_pattern(pattern_type␣
↪= enums.PatternTypeHdr.OTHer)
```

No command help available

param pattern_type
No help available

set_plength(*payload_length: List[int]*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:HDR:ISIGnal:PLENgth
driver.configure.bluetooth.measurement.hdr.inputSignal.set_plength(payload_
↪length = [1, 2, 3])
```

No command help available

param payload_length
No help available

set_ptype(*packet_type: PacketTypeC*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:HDR:ISIGnal:PTYPe
driver.configure.bluetooth.measurement.hdr.inputSignal.set_ptype(packet_type =␣
↪enums.PacketTypeC.H41P)
```

No command help available

param packet_type
No help available

set_uap(*uap_address: str*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:HDR:ISIGnal:UAP
driver.configure.bluetooth.measurement.hdr.inputSignal.set_uap(uap_address =␣
↪rawAbc)
```

No command help available

param uap_address
No help available

6.4.1.1.5.2 Limit

SCPI Commands :

```
CONFigure:BLUetooth:MEASurement<Instance>:HDR:LIMit:PVTime
CONFigure:BLUetooth:MEASurement<Instance>:HDR:LIMit:FSTability
```

class LimitCls

Limit commands group definition. 7 total commands, 3 Subgroups, 2 group commands

class FstabilityStruct

Structure for setting input parameters. Fields:

- Wi: float: No parameter help available
- Wi_W_0: float: No parameter help available
- W_0_Max: float: No parameter help available
- Wi_Enabled: List[bool]: No parameter help available
- Wi_Wo_Enabled: List[bool]: No parameter help available
- W_0_Max_Enabled: List[bool]: No parameter help available

class PowerVsTimeStruct

Structure for setting input parameters. Fields:

- Dpsk_Minus_Gfsk_Low: float: No parameter help available
- Dpsk_Minus_Gfsk_Upp: float: No parameter help available
- Guard_Period_Low: float: No parameter help available
- Guard_Period_Upp: float: No parameter help available
- Dpsk_Minus_Gfsk_Enable: List[bool]: No parameter help available

- Guard_Period_Enable: List[bool]: No parameter help available

get_fstability() → FstabilityStruct

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:HDR:LIMit:FSTability
value: FstabilityStruct = driver.configure.bluetooth.measurement.hdr.limit.get_
↪fstability()
```

No command help available

return
structure: for return value, see the help for FstabilityStruct structure arguments.

get_power_vs_time() → PowerVsTimeStruct

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:HDR:LIMit:PVTime
value: PowerVsTimeStruct = driver.configure.bluetooth.measurement.hdr.limit.get_
↪power_vs_time()
```

No command help available

return
structure: for return value, see the help for PowerVsTimeStruct structure arguments.

set_fstability(*value: FstabilityStruct*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:HDR:LIMit:FSTability
structure = driver.configure.bluetooth.measurement.hdr.limit.FstabilityStruct()
structure.Wi: float = 1.0
structure.Wi_W_0: float = 1.0
structure.W_0_Max: float = 1.0
structure.Wi_Enabled: List[bool] = [True, False, True]
structure.Wi_Wo_Enabled: List[bool] = [True, False, True]
structure.W_0_Max_Enabled: List[bool] = [True, False, True]
driver.configure.bluetooth.measurement.hdr.limit.set_fstability(value =␣
↪structure)
```

No command help available

param value
see the help for FstabilityStruct structure arguments.

set_power_vs_time(*value: PowerVsTimeStruct*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:HDR:LIMit:PVTime
structure = driver.configure.bluetooth.measurement.hdr.limit.PowerVsTimeStruct()
structure.Dpsk_Minus_Gfsk_Low: float = 1.0
structure.Dpsk_Minus_Gfsk_Upp: float = 1.0
structure.Guard_Period_Low: float = 1.0
structure.Guard_Period_Upp: float = 1.0
structure.Dpsk_Minus_Gfsk_Enable: List[bool] = [True, False, True]
structure.Guard_Period_Enable: List[bool] = [True, False, True]
driver.configure.bluetooth.measurement.hdr.limit.set_power_vs_time(value =␣
↪structure)
```

No command help available

param value
see the help for PowerVsTimeStruct structure arguments.

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.configure.bluetooth.measurement.hdr.limit.clone()
```

Subgroups

6.4.1.1.5.3 P4H

SCPI Commands :

```
CONFigure:BLUetooth:MEASurement<Instance>:HDR:LIMit:P4H:DEVM
CONFigure:BLUetooth:MEASurement<Instance>:HDR:LIMit:P4H:SGACp
```

class P4HCls

P4H commands group definition. 2 total commands, 0 Subgroups, 2 group commands

class DevmStruct

Structure for setting input parameters. Fields:

- Rms: float: No parameter help available
- Peak: float: No parameter help available
- P_99: float: No parameter help available
- Rms_Enabled: List[bool]: No parameter help available
- Peak_Enabled: List[bool]: No parameter help available
- P_99_Enabled: bool: No parameter help available

class SgacpStruct

Structure for setting input parameters. Fields:

- Ptx_Rel_Limit: float: No parameter help available
- Ptx_Abs_Limit_1: float: No parameter help available
- Ptx_Abs_Limit_2: float: No parameter help available
- Ptx_Rel_Enable: bool: No parameter help available
- Ptx_Abs_1_Enable: bool: No parameter help available
- Ptx_Abs_2_Enable: bool: No parameter help available

get_devm() → DevmStruct

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:HDR:LIMit:P4H:DEVM
value: DevmStruct = driver.configure.bluetooth.measurement.hdr.limit.p4H.get_
↪devm()
```

No command help available

return
structure: for return value, see the help for DevmStruct structure arguments.

get_sgacp() → SgacpStruct

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:HDR:LIMit:P4H:SGACp
value: SgacpStruct = driver.configure.bluetooth.measurement.hdr.limit.p4H.get_
↪sgacp()
```

No command help available

return
structure: for return value, see the help for SgacpStruct structure arguments.

set_devm(*value: DevmStruct*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:HDR:LIMit:P4H:DEVM
structure = driver.configure.bluetooth.measurement.hdr.limit.p4H.DevmStruct()
structure.Rms: float = 1.0
structure.Peak: float = 1.0
structure.P_99: float = 1.0
structure.Rms_Enabled: List[bool] = [True, False, True]
structure.Peak_Enabled: List[bool] = [True, False, True]
structure.P_99_Enabled: bool = False
driver.configure.bluetooth.measurement.hdr.limit.p4H.set_devm(value = structure)
```

No command help available

param value
see the help for DevmStruct structure arguments.

set_sgacp(*value: SgacpStruct*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:HDR:LIMit:P4H:SGACp
structure = driver.configure.bluetooth.measurement.hdr.limit.p4H.SgacpStruct()
structure.Ptx_Rel_Limit: float = 1.0
structure.Ptx_Abs_Limit_1: float = 1.0
structure.Ptx_Abs_Limit_2: float = 1.0
structure.Ptx_Rel_Enable: bool = False
structure.Ptx_Abs_1_Enable: bool = False
structure.Ptx_Abs_2_Enable: bool = False
driver.configure.bluetooth.measurement.hdr.limit.p4H.set_sgacp(value =␣
↪structure)
```

No command help available

param value
see the help for SgacpStruct structure arguments.

6.4.1.1.5.4 P8H

SCPI Commands :

```
CONFigure:BLUetooth:MEASurement<Instance>:HDR:LIMit:P8H:DEVM
CONFigure:BLUetooth:MEASurement<Instance>:HDR:LIMit:P8H:SGACp
```

class P8HCls

P8H commands group definition. 2 total commands, 0 Subgroups, 2 group commands

class DevmStruct

Structure for setting input parameters. Fields:

- Rms: float: No parameter help available
- Peak: float: No parameter help available
- P_99: float: No parameter help available
- Rms_Enabled: List[bool]: No parameter help available
- Peak_Enabled: List[bool]: No parameter help available
- P_99_Enabled: bool: No parameter help available

class SgacpStruct

Structure for setting input parameters. Fields:

- Ptx_Rel_Limit: float: No parameter help available
- Ptx_Abs_Limit_1: float: No parameter help available
- Ptx_Abs_Limit_2: float: No parameter help available
- Ptx_Abs_Limit_3: float: No parameter help available
- Ptx_Rel_Enabled: bool: No parameter help available
- Ptx_Abs_1_Enable: bool: No parameter help available
- Ptx_Abs_2_Enable: bool: No parameter help available
- Ptx_Abs_3_Enable: bool: No parameter help available

get_devm() → DevmStruct

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:HDR:LIMit:P8H:DEVM
value: DevmStruct = driver.configure.bluetooth.measurement.hdr.limit.p8H.get_
↪devm()
```

No command help available

return
structure: for return value, see the help for DevmStruct structure arguments.

get_sgacp() → SgacpStruct

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:HDR:LIMit:P8H:SGACp
value: SgacpStruct = driver.configure.bluetooth.measurement.hdr.limit.p8H.get_
↪sgacp()
```

No command help available

return
structure: for return value, see the help for SgacpStruct structure arguments.

set_devm(*value: DevmStruct*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:HDR:LIMit:P8H:DEVM
structure = driver.configure.bluetooth.measurement.hdr.limit.p8H.DevmStruct()
structure.Rms: float = 1.0
structure.Peak: float = 1.0
structure.P_99: float = 1.0
structure.Rms_Enabled: List[bool] = [True, False, True]
structure.Peak_Enabled: List[bool] = [True, False, True]
structure.P_99_Enabled: bool = False
driver.configure.bluetooth.measurement.hdr.limit.p8H.set_devm(value = structure)
```

No command help available

param value
see the help for DevmStruct structure arguments.

set_sgacp(*value: SgacpStruct*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:HDR:LIMit:P8H:SGACp
structure = driver.configure.bluetooth.measurement.hdr.limit.p8H.SgacpStruct()
structure.Ptx_Rel_Limit: float = 1.0
structure.Ptx_Abs_Limit_1: float = 1.0
structure.Ptx_Abs_Limit_2: float = 1.0
structure.Ptx_Abs_Limit_3: float = 1.0
structure.Ptx_Rel_Enabled: bool = False
structure.Ptx_Abs_1_Enable: bool = False
structure.Ptx_Abs_2_Enable: bool = False
structure.Ptx_Abs_3_Enable: bool = False
driver.configure.bluetooth.measurement.hdr.limit.p8H.set_sgacp(value =␣
↪structure)
```

No command help available

param value
see the help for SgacpStruct structure arguments.

6.4.1.1.5.5 Pencoding

SCPI Command :

```
CONFigure:BLUetooth:MEASurement<Instance>:HDR:LIMit:PENCoding
```

class PencodingCls

Pencoding commands group definition. 1 total commands, 0 Subgroups, 1 group commands

class PencodingStruct

Response structure. Fields:

- Phase_Limit: float: No parameter help available
- Phase_Enable: bool: No parameter help available

get() → PencodingStruct

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:HDR:LIMit:PENCoding
value: PencodingStruct = driver.configure.bluetooth.measurement.hdr.limit.
↪pencoding.get()
```

No command help available

return
structure: for return value, see the help for PencodingStruct structure arguments.

set(*phase_limit: float*, *phase_enable: bool*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:HDR:LIMit:PENCoding
driver.configure.bluetooth.measurement.hdr.limit.pencoding.set(phase_limit = 1.
↪0, phase_enable = False)
```

No command help available

param phase_limit
No help available

param phase_enable
No help available

6.4.1.1.5.6 Result

SCPI Commands :

```
CONFigure:BLUetooth:MEASurement<Instance>:HDR:RESult[:ALL]
CONFigure:BLUetooth:MEASurement<Instance>:HDR:RESult:PVTime
CONFigure:BLUetooth:MEASurement<Instance>:HDR:RESult:SGACp
CONFigure:BLUetooth:MEASurement<Instance>:HDR:RESult:PENCoding
CONFigure:BLUetooth:MEASurement<Instance>:HDR:RESult:IQERr
CONFigure:BLUetooth:MEASurement<Instance>:HDR:RESult:IQDiff
CONFigure:BLUetooth:MEASurement<Instance>:HDR:RESult:IQABsolute
CONFigure:BLUetooth:MEASurement<Instance>:HDR:RESult:PDIFference
CONFigure:BLUetooth:MEASurement<Instance>:HDR:RESult:DEVMagnitude
CONFigure:BLUetooth:MEASurement<Instance>:HDR:RESult:TXSCalar
```

class ResultCls

Result commands group definition. 10 total commands, 0 Subgroups, 10 group commands

class AllStruct

Structure for setting input parameters. Fields:

- Devm: bool: No parameter help available
- Phase_Diff: bool: No parameter help available
- Tx_Scalars: bool: No parameter help available
- Iq_Absolute: bool: No parameter help available
- Iq_Differential: bool: No parameter help available
- Iq_Error: bool: No parameter help available

- Phase_Encoding: bool: No parameter help available
- Power_Vs_Time: bool: No parameter help available
- Spectrum_Gat_Acp: bool: No parameter help available

get_all() → AllStruct

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:HDR:RESult[:ALL]
value: AllStruct = driver.configure.bluetooth.measurement.hdr.result.get_all()
```

No command help available

return
structure: for return value, see the help for AllStruct structure arguments.

get_dev_magnitude() → bool

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:HDR:RESult:DEVMagnitude
value: bool = driver.configure.bluetooth.measurement.hdr.result.get_dev_
↪magnitude()
```

No command help available

return
state: No help available

get_iq_absolute() → bool

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:HDR:RESult:IQABsolute
value: bool = driver.configure.bluetooth.measurement.hdr.result.get_iq_
↪absolute()
```

No command help available

return
state: No help available

get_iq_difference() → bool

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:HDR:RESult:IQDiff
value: bool = driver.configure.bluetooth.measurement.hdr.result.get_iq_
↪difference()
```

No command help available

return
state: No help available

get_iq_error() → bool

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:HDR:RESult:IQERr
value: bool = driver.configure.bluetooth.measurement.hdr.result.get_iq_error()
```

No command help available

return
state: No help available

get_pdifference() → bool

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:HDR:RESult:PDIFference
value: bool = driver.configure.bluetooth.measurement.hdr.result.get_
↪pdifference()
```

No command help available

return
state: No help available

get_pencoding() → bool

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:HDR:RESult:PENCoding
value: bool = driver.configure.bluetooth.measurement.hdr.result.get_pencoding()
```

No command help available

return
state: No help available

get_power_vs_time() → bool

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:HDR:RESult:PVTime
value: bool = driver.configure.bluetooth.measurement.hdr.result.get_power_vs_
↪time()
```

No command help available

return
state: No help available

get_sgacp() → bool

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:HDR:RESult:SGACp
value: bool = driver.configure.bluetooth.measurement.hdr.result.get_sgacp()
```

No command help available

return
state: No help available

get_tx_scalar() → bool

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:HDR:RESult:TXSCalar
value: bool = driver.configure.bluetooth.measurement.hdr.result.get_tx_scalar()
```

No command help available

return
state: No help available

set_all(*value: AllStruct*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:HDR:RESult[:ALL]
structure = driver.configure.bluetooth.measurement.hdr.result.AllStruct()
structure.Devm: bool = False
structure.Phase_Diff: bool = False
```

(continues on next page)

(continued from previous page)

```
structure.Tx_Scalars: bool = False
structure.Iq_Absolute: bool = False
structure.Iq_Differential: bool = False
structure.Iq_Error: bool = False
structure.Phase_Encoding: bool = False
structure.Power_Vs_Time: bool = False
structure.Spectrum_Gat_Acp: bool = False
driver.configure.bluetooth.measurement.hdr.result.set_all(value = structure)
```

No command help available

param value
see the help for AllStruct structure arguments.

set_dev_magnitude(*state: bool*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:HDR:RESult:DEVMagnitude
driver.configure.bluetooth.measurement.hdr.result.set_dev_magnitude(state =␣
↪False)
```

No command help available

param state
No help available

set_iq_absolute(*state: bool*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:HDR:RESult:IQABsolute
driver.configure.bluetooth.measurement.hdr.result.set_iq_absolute(state = False)
```

No command help available

param state
No help available

set_iq_difference(*state: bool*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:HDR:RESult:IQDiff
driver.configure.bluetooth.measurement.hdr.result.set_iq_difference(state =␣
↪False)
```

No command help available

param state
No help available

set_iq_error(*state: bool*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:HDR:RESult:IQERr
driver.configure.bluetooth.measurement.hdr.result.set_iq_error(state = False)
```

No command help available

param state
No help available

set_pdifference(*state: bool*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:HDR:RESult:PDIFference
driver.configure.bluetooth.measurement.hdr.result.set_pdifference(state = False)
```

No command help available

param state
No help available

set_pencoding(*state: bool*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:HDR:RESult:PENCoding
driver.configure.bluetooth.measurement.hdr.result.set_pencoding(state = False)
```

No command help available

param state
No help available

set_power_vs_time(*state: bool*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:HDR:RESult:PVTime
driver.configure.bluetooth.measurement.hdr.result.set_power_vs_time(state =␣
↪False)
```

No command help available

param state
No help available

set_sgacp(*state: bool*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:HDR:RESult:SGACp
driver.configure.bluetooth.measurement.hdr.result.set_sgacp(state = False)
```

No command help available

param state
No help available

set_tx_scalar(*state: bool*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:HDR:RESult:TXSCalar
driver.configure.bluetooth.measurement.hdr.result.set_tx_scalar(state = False)
```

No command help available

param state
No help available

6.4.1.1.5.7 Scount

SCPI Commands :

```
CONFigure:BLUetooth:MEASurement<Instance>:HDR:SCOunt:PVTime
CONFigure:BLUetooth:MEASurement<Instance>:HDR:SCOunt:SGACp
CONFigure:BLUetooth:MEASurement<Instance>:HDR:SCOunt:MODulation
CONFigure:BLUetooth:MEASurement<Instance>:HDR:SCOunt:PENCoding
```

class ScountCls

Scount commands group definition. 4 total commands, 0 Subgroups, 4 group commands

get_modulation() → int

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:HDR:SCOunt:MODulation
value: int = driver.configure.bluetooth.measurement.hdr.scount.get_modulation()
```

No command help available

return
stat_count: No help available

get_pencoding() → int

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:HDR:SCOunt:PENCoding
value: int = driver.configure.bluetooth.measurement.hdr.scount.get_pencoding()
```

No command help available

return
stat_count: No help available

get_power_vs_time() → int

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:HDR:SCOunt:PVTime
value: int = driver.configure.bluetooth.measurement.hdr.scount.get_power_vs_
↪time()
```

No command help available

return
stat_count: No help available

get_sgacp() → int

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:HDR:SCOunt:SGACp
value: int = driver.configure.bluetooth.measurement.hdr.scount.get_sgacp()
```

No command help available

return
stat_count: No help available

set_modulation(*stat_count: int*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:HDR:SCOunt:MODulation
driver.configure.bluetooth.measurement.hdr.scount.set_modulation(stat_count = 1)
```

No command help available

param stat_count
No help available

set_pencoding(*stat_count: int*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:HDR:SCOunt:PENCoding
driver.configure.bluetooth.measurement.hdr.scount.set_pencoding(stat_count = 1)
```

No command help available

param stat_count
No help available

set_power_vs_time(*stat_count: int*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:HDR:SCOunt:PVTime
driver.configure.bluetooth.measurement.hdr.scount.set_power_vs_time(stat_count␣
↪= 1)
```

No command help available

param stat_count
No help available

set_sgacp(*stat_count: int*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:HDR:SCOunt:SGACp
driver.configure.bluetooth.measurement.hdr.scount.set_sgacp(stat_count = 1)
```

No command help available

param stat_count
No help available

6.4.1.1.5.8 Sgacp

class SgacpCls

Sgacp commands group definition. 1 total commands, 1 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.configure.bluetooth.measurement.hdr.sgacp.clone()
```

Subgroups

6.4.1.1.5.9 Measurement

SCPI Command :

```
CONFigure:BLUetooth:MEASurement<Instance>:HDR:SGACp:MEASurement:MODE
```

class MeasurementCls

Measurement commands group definition. 1 total commands, 0 Subgroups, 1 group commands

get_mode() → BrEdrChannelsRange

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:HDR:SGACp:MEASurement:MODE
value: enums.BrEdrChannelsRange = driver.configure.bluetooth.measurement.hdr.
↪sgacp.measurement.get_mode()
```

No command help available

return
meas_mode: No help available

set_mode(*meas_mode: BrEdrChannelsRange*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:HDR:SGACp:MEASurement:MODE
driver.configure.bluetooth.measurement.hdr.sgacp.measurement.set_mode(meas_mode␣
↪= enums.BrEdrChannelsRange.CH21)
```

No command help available

param meas_mode
No help available

6.4.1.1.6 Hdrp

SCPI Commands :

```
CONFigure:BLUetooth:MEASurement<Instance>:HDRP:MOEXception
CONFigure:BLUetooth:MEASurement<Instance>:HDRP:SCONdition
CONFigure:BLUetooth:MEASurement<Instance>:HDRP:REPetition
```

class HdrpCls

Hdrp commands group definition. 23 total commands, 5 Subgroups, 3 group commands

get_mo_exception() → bool

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:HDRP:MOEXception
value: bool = driver.configure.bluetooth.measurement.hdrp.get_mo_exception()
```

No command help available

return
meas_on_exception: No help available

get_repetition() → Repeat

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:HDRP:REPetition
value: enums.Repeat = driver.configure.bluetooth.measurement.hdrp.get_
↪repetition()
```

No command help available

return
repetition: No help available

get_scondition() → StopCondition

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:HDRP:SCONdition
value: enums.StopCondition = driver.configure.bluetooth.measurement.hdrp.get_
↪scondition()
```

No command help available

return
stop_condition: No help available

set_mo_exception(*meas_on_exception: bool*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:HDRP:MOEXception
driver.configure.bluetooth.measurement.hdrp.set_mo_exception(meas_on_exception␣
↪= False)
```

No command help available

param meas_on_exception
No help available

set_repetition(*repetition: Repeat*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:HDRP:REPetition
driver.configure.bluetooth.measurement.hdrp.set_repetition(repetition = enums.
↪Repeat.CONTinuous)
```

No command help available

param repetition
No help available

set_scondition(*stop_condition: StopCondition*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:HDRP:SCONdition
driver.configure.bluetooth.measurement.hdrp.set_scondition(stop_condition =␣
↪enums.StopCondition.NONE)
```

No command help available

param stop_condition
No help available

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.configure.bluetooth.measurement.hdrp.clone()
```

Subgroups

6.4.1.1.6.1 InputSignal

SCPI Commands :

```
CONFigure:BLUetooth:MEASurement<Instance>:HDRP:ISIGnal:PHY
CONFigure:BLUetooth:MEASurement<Instance>:HDRP:ISIGnal:PCODing
CONFigure:BLUetooth:MEASurement<Instance>:HDRP:ISIGnal:DMODe
```

class InputSignalCls

InputSignal commands group definition. 4 total commands, 1 Subgroups, 3 group commands

get_dmode() → AutoManualMode

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:HDRP:ISIGnal:DMODe
value: enums.AutoManualMode = driver.configure.bluetooth.measurement.hdrp.
↪inputSignal.get_dmode()
```

No command help available

return
detection_mode: No help available

get_pcoding() → PayloadCoding

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:HDRP:ISIGnal:PCODing
value: enums.PayloadCoding = driver.configure.bluetooth.measurement.hdrp.
↪inputSignal.get_pcoding()
```

No command help available

return
coding: No help available

get_phy() → PhysicalLayer

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:HDRP:ISIGnal:PHY
value: enums.PhysicalLayer = driver.configure.bluetooth.measurement.hdrp.
↪inputSignal.get_phy()
```

No command help available

return
phy: No help available

set_dmode(*detection_mode: AutoManualMode*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:HDRP:ISIGnal:DMODe
driver.configure.bluetooth.measurement.hdrp.inputSignal.set_dmode(detection_
↪mode = enums.AutoManualMode.AUTO)
```

No command help available

param detection_mode
No help available

set_pcoding(*coding: PayloadCoding*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:HDRP:ISIGnal:PCODing
driver.configure.bluetooth.measurement.hdrp.inputSignal.set_pcoding(coding =␣
↪enums.PayloadCoding.L12D)
```

No command help available

param coding
No help available

set_phy(*phy: PhysicalLayer*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:HDRP:ISIGnal:PHY
driver.configure.bluetooth.measurement.hdrp.inputSignal.set_phy(phy = enums.
↪PhysicalLayer.P4HP)
```

No command help available

param phy
No help available

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.configure.bluetooth.measurement.hdrp.inputSignal.clone()
```

Subgroups

6.4.1.1.6.2 Plength

SCPI Command :

```
CONFigure:BLUetooth:MEASurement<Instance>:HDRP:ISIGnal:PLENgth
```

class PlengthCls

Plength commands group definition. 1 total commands, 0 Subgroups, 1 group commands

class PlengthStruct

Response structure. Fields:

- Payload_Len_Four: List[int]: No parameter help available
- Payload_Len_Eight: List[int]: No parameter help available

get() → PlengthStruct

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:HDRP:ISIGnal:PLENgth
value: PlengthStruct = driver.configure.bluetooth.measurement.hdrp.inputSignal.
↪plength.get()
```

No command help available

return
structure: for return value, see the help for PlengthStruct structure arguments.

set(*payload_len_four: List[int]*, *payload_len_eight: List[int]*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:HDRP:ISIGnal:PLENgth
driver.configure.bluetooth.measurement.hdrp.inputSignal.plength.set(payload_len_
↪four = [1, 2, 3], payload_len_eight = [1, 2, 3])
```

No command help available

param payload_len_four
No help available

param payload_len_eight
No help available

6.4.1.1.6.3 Limit

SCPI Command :

```
CONFigure:BLUetooth:MEASurement<Instance>:HDRP:LIMit:FSTability
```

class LimitCls

Limit commands group definition. 6 total commands, 3 Subgroups, 1 group commands

class FstabilityStruct

Structure for setting input parameters. Fields:

- Wi: float: No parameter help available
- Wi_W_0: float: No parameter help available
- W_0_Max: float: No parameter help available
- Wi_Enabled: List[bool]: No parameter help available
- Wi_Wo_Enabled: List[bool]: No parameter help available
- W_0_Max_Enabled: List[bool]: No parameter help available

get_fstability() → FstabilityStruct

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:HDRP:LIMit:FSTability
value: FstabilityStruct = driver.configure.bluetooth.measurement.hdrp.limit.get_
↪fstability()
```

No command help available

return
structure: for return value, see the help for FstabilityStruct structure arguments.

set_fstability(*value: FstabilityStruct*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:HDRP:LIMit:FSTability
structure = driver.configure.bluetooth.measurement.hdrp.limit.FstabilityStruct()
structure.Wi: float = 1.0
structure.Wi_W_0: float = 1.0
structure.W_0_Max: float = 1.0
structure.Wi_Enabled: List[bool] = [True, False, True]
structure.Wi_Wo_Enabled: List[bool] = [True, False, True]
structure.W_0_Max_Enabled: List[bool] = [True, False, True]
driver.configure.bluetooth.measurement.hdrp.limit.set_fstability(value =␣
↪structure)
```

No command help available

param value
see the help for FstabilityStruct structure arguments.

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.configure.bluetooth.measurement.hdrp.limit.clone()
```

Subgroups

6.4.1.1.6.4 P4Hp

SCPI Command :

```
CONFigure:BLUetooth:MEASurement<Instance>:HDRP:LIMit:P4HP:SACP
```

class P4HpCls

P4Hp commands group definition. 2 total commands, 1 Subgroups, 1 group commands

class SacpStruct

Structure for setting input parameters. Fields:

- Ptx_Rel_Limit: float: No parameter help available
- Ptx_Abs_Limit_1: float: No parameter help available
- Ptx_Abs_Limit_2: float: No parameter help available
- Ptx_Rel_Enable: bool: No parameter help available
- Ptx_Abs_1_Enable: bool: No parameter help available
- Ptx_Abs_2_Enable: bool: No parameter help available

get_sacp() → SacpStruct

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:HDRP:LIMit:P4HP:SACP
value: SacpStruct = driver.configure.bluetooth.measurement.hdrp.limit.p4Hp.get_
↪sacp()
```

No command help available

return
structure: for return value, see the help for SacpStruct structure arguments.

set_sacp(*value: SacpStruct*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:HDRP:LIMit:P4HP:SACP
structure = driver.configure.bluetooth.measurement.hdrp.limit.p4Hp.SacpStruct()
structure.Ptx_Rel_Limit: float = 1.0
structure.Ptx_Abs_Limit_1: float = 1.0
structure.Ptx_Abs_Limit_2: float = 1.0
structure.Ptx_Rel_Enable: bool = False
structure.Ptx_Abs_1_Enable: bool = False
structure.Ptx_Abs_2_Enable: bool = False
driver.configure.bluetooth.measurement.hdrp.limit.p4Hp.set_sacp(value =␣
↪structure)
```

No command help available

param value
see the help for SacpStruct structure arguments.

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.configure.bluetooth.measurement.hdrp.limit.p4Hp.clone()
```

Subgroups

6.4.1.1.6.5 EvMagnitude

SCPI Command :

```
CONFigure:BLUetooth:MEASurement<Instance>:HDRP:LIMit:P4HP:EVMagnitude:OFFSet
```

class EvMagnitudeCls

EvMagnitude commands group definition. 1 total commands, 0 Subgroups, 1 group commands

class OffsetStruct

Structure for setting input parameters. Fields:

- Rms: float: No parameter help available
- Peak: float: No parameter help available
- P_99: float: No parameter help available
- Rms_Enabled: List[bool]: No parameter help available
- Peak_Enabled: List[bool]: No parameter help available
- P_99_Enabled: bool: No parameter help available

get_offset() → OffsetStruct

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:HDRP:LIMit:P4HP:EVMagnitude:OFFSet
value: OffsetStruct = driver.configure.bluetooth.measurement.hdrp.limit.p4Hp.
↪evMagnitude.get_offset()
```

No command help available

return
structure: for return value, see the help for OffsetStruct structure arguments.

set_offset(*value: OffsetStruct*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:HDRP:LIMit:P4HP:EVMagnitude:OFFSet
structure = driver.configure.bluetooth.measurement.hdrp.limit.p4Hp.evMagnitude.
↪OffsetStruct()
structure.Rms: float = 1.0
structure.Peak: float = 1.0
structure.P_99: float = 1.0
structure.Rms_Enabled: List[bool] = [True, False, True]
structure.Peak_Enabled: List[bool] = [True, False, True]
structure.P_99_Enabled: bool = False
driver.configure.bluetooth.measurement.hdrp.limit.p4Hp.evMagnitude.set_
↪offset(value = structure)
```

No command help available

param value
see the help for OffsetStruct structure arguments.

6.4.1.1.6.6 P8Hp

SCPI Command :

```
CONFigure:BLUetooth:MEASurement<Instance>:HDRP:LIMit:P8HP:SACP
```

class P8HpCls

P8Hp commands group definition. 2 total commands, 1 Subgroups, 1 group commands

class SacpStruct

Structure for setting input parameters. Fields:

- Ptx_Rel_Limit: float: No parameter help available
- Ptx_Abs_Limit_1: float: No parameter help available
- Ptx_Abs_Limit_2: float: No parameter help available
- Ptx_Abs_Limit_3: float: No parameter help available
- Ptx_Rel_Enabled: bool: No parameter help available
- Ptx_Abs_1_Enable: bool: No parameter help available
- Ptx_Abs_2_Enable: bool: No parameter help available

- Ptx_Abs_3_Enable: bool: No parameter help available

get_sacp() → SacpStruct

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:HDRP:LIMit:P8HP:SACP
value: SacpStruct = driver.configure.bluetooth.measurement.hdrp.limit.p8Hp.get_
↪sacp()
```

No command help available

return
structure: for return value, see the help for SacpStruct structure arguments.

set_sacp(*value: SacpStruct*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:HDRP:LIMit:P8HP:SACP
structure = driver.configure.bluetooth.measurement.hdrp.limit.p8Hp.SacpStruct()
structure.Ptx_Rel_Limit: float = 1.0
structure.Ptx_Abs_Limit_1: float = 1.0
structure.Ptx_Abs_Limit_2: float = 1.0
structure.Ptx_Abs_Limit_3: float = 1.0
structure.Ptx_Rel_Enabled: bool = False
structure.Ptx_Abs_1_Enable: bool = False
structure.Ptx_Abs_2_Enable: bool = False
structure.Ptx_Abs_3_Enable: bool = False
driver.configure.bluetooth.measurement.hdrp.limit.p8Hp.set_sacp(value =␣
↪structure)
```

No command help available

param value
see the help for SacpStruct structure arguments.

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.configure.bluetooth.measurement.hdrp.limit.p8Hp.clone()
```

Subgroups

6.4.1.1.6.7 EvMagnitude

SCPI Command :

```
CONFigure:BLUetooth:MEASurement<Instance>:HDRP:LIMit:P8HP:EVMagnitude:OFFSet
```

class EvMagnitudeCls

EvMagnitude commands group definition. 1 total commands, 0 Subgroups, 1 group commands

class OffsetStruct

Structure for setting input parameters. Fields:

- Rms: float: No parameter help available

- Peak: float: No parameter help available
- P_99: float: No parameter help available
- Rms_Enabled: List[bool]: No parameter help available
- Peak_Enabled: List[bool]: No parameter help available
- P_99_Enabled: bool: No parameter help available

get_offset() → OffsetStruct

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:HDRP:LIMit:P8HP:EVMagnitude:OFFSet
value: OffsetStruct = driver.configure.bluetooth.measurement.hdrp.limit.p8Hp.
↪evMagnitude.get_offset()
```

No command help available

return
structure: for return value, see the help for OffsetStruct structure arguments.

set_offset(*value: OffsetStruct*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:HDRP:LIMit:P8HP:EVMagnitude:OFFSet
structure = driver.configure.bluetooth.measurement.hdrp.limit.p8Hp.evMagnitude.
↪OffsetStruct()
structure.Rms: float = 1.0
structure.Peak: float = 1.0
structure.P_99: float = 1.0
structure.Rms_Enabled: List[bool] = [True, False, True]
structure.Peak_Enabled: List[bool] = [True, False, True]
structure.P_99_Enabled: bool = False
driver.configure.bluetooth.measurement.hdrp.limit.p8Hp.evMagnitude.set_
↪offset(value = structure)
```

No command help available

param value
see the help for OffsetStruct structure arguments.

6.4.1.1.6.8 PowerVsTime

SCPI Command :

```
CONFigure:BLUetooth:MEASurement<Instance>:HDRP:LIMit:PVTime
```

class PowerVsTimeCls

PowerVsTime commands group definition. 1 total commands, 0 Subgroups, 1 group commands

class PowerVsTimeStruct

Response structure. Fields:

- Average_Power_Low: float: No parameter help available
- Average_Power_Upp: float: No parameter help available

- Average_Pow_Enable: List[bool]: No parameter help available

get() → PowerVsTimeStruct

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:HDRP:LIMit:PVTime
value: PowerVsTimeStruct = driver.configure.bluetooth.measurement.hdrp.limit.
↪powerVsTime.get()
```

No command help available

return
structure: for return value, see the help for PowerVsTimeStruct structure arguments.

set(*average_power_low: float*, *average_power_upp: float*, *average_pow_enable: List[bool]*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:HDRP:LIMit:PVTime
driver.configure.bluetooth.measurement.hdrp.limit.powerVsTime.set(average_power_
↪low = 1.0, average_power_upp = 1.0, average_pow_enable = [True, False, True])
```

No command help available

param average_power_low
No help available

param average_power_upp
No help available

param average_pow_enable
No help available

6.4.1.1.6.9 Result

SCPI Commands :

```
CONFigure:BLUetooth:MEASurement<Instance>:HDRP:RESult:PVTime
CONFigure:BLUetooth:MEASurement<Instance>:HDRP:RESult:SACP
CONFigure:BLUetooth:MEASurement<Instance>:HDRP:RESult:IQ
CONFigure:BLUetooth:MEASurement<Instance>:HDRP:RESult:EVMagnitude
CONFigure:BLUetooth:MEASurement<Instance>:HDRP:RESult:TXSCalar
```

class ResultCls

Result commands group definition. 6 total commands, 1 Subgroups, 5 group commands

get_ev_magnitude() → bool

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:HDRP:RESult:EVMagnitude
value: bool = driver.configure.bluetooth.measurement.hdrp.result.get_ev_
↪magnitude()
```

No command help available

return
state: No help available

get_iq() → bool

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:HDRP:RESult:IQ
value: bool = driver.configure.bluetooth.measurement.hdrp.result.get_iq()
```

No command help available

return
state: No help available

get_power_vs_time() → bool

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:HDRP:RESult:PVTime
value: bool = driver.configure.bluetooth.measurement.hdrp.result.get_power_vs_
↪time()
```

No command help available

return
state: No help available

get_sacp() → bool

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:HDRP:RESult:SACP
value: bool = driver.configure.bluetooth.measurement.hdrp.result.get_sacp()
```

No command help available

return
state: No help available

get_tx_scalar() → bool

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:HDRP:RESult:TXSCalar
value: bool = driver.configure.bluetooth.measurement.hdrp.result.get_tx_scalar()
```

No command help available

return
state: No help available

set_ev_magnitude(*state: bool*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:HDRP:RESult:EVMagnitude
driver.configure.bluetooth.measurement.hdrp.result.set_ev_magnitude(state =␣
↪False)
```

No command help available

param state
No help available

set_iq(*state: bool*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:HDRP:RESult:IQ
driver.configure.bluetooth.measurement.hdrp.result.set_iq(state = False)
```

No command help available

param state
No help available

set_power_vs_time(*state: bool*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:HDRP:RESult:PVTime
driver.configure.bluetooth.measurement.hdrp.result.set_power_vs_time(state =␣
↪False)
```

No command help available

param state
No help available

set_sacp(*state: bool*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:HDRP:RESult:SACP
driver.configure.bluetooth.measurement.hdrp.result.set_sacp(state = False)
```

No command help available

param state
No help available

set_tx_scalar(*state: bool*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:HDRP:RESult:TXSCalar
driver.configure.bluetooth.measurement.hdrp.result.set_tx_scalar(state = False)
```

No command help available

param state
No help available

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.configure.bluetooth.measurement.hdrp.result.clone()
```

Subgroups

6.4.1.1.6.10 All

SCPI Command :

```
CONFigure:BLUetooth:MEASurement<Instance>:HDRP:RESult[:ALL]
```

class AllCls

All commands group definition. 1 total commands, 0 Subgroups, 1 group commands

class AllStruct

Response structure. Fields:

- Evm: bool: No parameter help available

- Tx_Scalars: bool: No parameter help available
- Iq_Constellation: bool: No parameter help available
- Power_Vs_Time: bool: No parameter help available
- Spectrum_Acp: bool: No parameter help available

get() → AllStruct

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:HDRP:RESult[:ALL]
value: AllStruct = driver.configure.bluetooth.measurement.hdrp.result.all.get()
```

No command help available

return
structure: for return value, see the help for AllStruct structure arguments.

set(*evm: bool*, *tx_scalars: bool*, *iq_constellation: bool*, *power_vs_time: bool*, *spectrum_acp: bool*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:HDRP:RESult[:ALL]
driver.configure.bluetooth.measurement.hdrp.result.all.set(evm = False, tx_
↪scalars = False, iq_constellation = False, power_vs_time = False, spectrum_
↪acp = False)
```

No command help available

param evm
No help available

param tx_scalars
No help available

param iq_constellation
No help available

param power_vs_time
No help available

param spectrum_acp
No help available

6.4.1.1.6.11 Sacp

class SacpCls

Sacp commands group definition. 1 total commands, 1 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.configure.bluetooth.measurement.hdrp.sacp.clone()
```

Subgroups

6.4.1.1.6.12 Measurement

SCPI Command :

```
CONFigure:BLUetooth:MEASurement<Instance>:HDRP:SACP:MEASurement:MODE
```

class MeasurementCls

Measurement commands group definition. 1 total commands, 0 Subgroups, 1 group commands

get_mode() → BrEdrChannelsRange

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:HDRP:SACP:MEASurement:MODE
value: enums.BrEdrChannelsRange = driver.configure.bluetooth.measurement.hdrp.
↪sacp.measurement.get_mode()
```

No command help available

return
meas_mode: No help available

set_mode(*meas_mode: BrEdrChannelsRange*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:HDRP:SACP:MEASurement:MODE
driver.configure.bluetooth.measurement.hdrp.sacp.measurement.set_mode(meas_mode␣
↪= enums.BrEdrChannelsRange.CH21)
```

No command help available

param meas_mode
No help available

6.4.1.1.6.13 Scount

SCPI Commands :

```
CONFigure:BLUetooth:MEASurement<Instance>:HDRP:SCOunt:PVTime
CONFigure:BLUetooth:MEASurement<Instance>:HDRP:SCOunt:SACP
CONFigure:BLUetooth:MEASurement<Instance>:HDRP:SCOunt:MODulation
```

class ScountCls

Scount commands group definition. 3 total commands, 0 Subgroups, 3 group commands

get_modulation() → int

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:HDRP:SCOunt:MODulation
value: int = driver.configure.bluetooth.measurement.hdrp.scount.get_modulation()
```

No command help available

return
stat_count: No help available

get_power_vs_time() → int

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:HDRP:SCOunt:PVTime
value: int = driver.configure.bluetooth.measurement.hdrp.scount.get_power_vs_
↪time()
```

No command help available

return
stat_count: No help available

get_sacp() → int

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:HDRP:SCOunt:SACP
value: int = driver.configure.bluetooth.measurement.hdrp.scount.get_sacp()
```

No command help available

return
stat_count: No help available

set_modulation(*stat_count: int*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:HDRP:SCOunt:MODulation
driver.configure.bluetooth.measurement.hdrp.scount.set_modulation(stat_count =␣
↪1)
```

No command help available

param stat_count
No help available

set_power_vs_time(*stat_count: int*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:HDRP:SCOunt:PVTime
driver.configure.bluetooth.measurement.hdrp.scount.set_power_vs_time(stat_count␣
↪= 1)
```

No command help available

param stat_count
No help available

set_sacp(*stat_count: int*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:HDRP:SCOunt:SACP
driver.configure.bluetooth.measurement.hdrp.scount.set_sacp(stat_count = 1)
```

No command help available

param stat_count
No help available

6.4.1.1.7 InputSignal

SCPI Commands :

```
CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:DMODe
CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:BTYPe
CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:ASYNchronize
```

class InputSignalCls

InputSignal commands group definition. 72 total commands, 18 Subgroups, 3 group commands

get_asynchronize() → bool

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:ASYNchronize
value: bool = driver.configure.bluetooth.measurement.inputSignal.get_
↪asynchronize()
```

Disables / enables automatic synchronization to the captured signal for an unspecified Bluetooth device address.

return
auto_synch: No help available

get_btype() → BurstType

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:BTYPe
value: enums.BurstType = driver.configure.bluetooth.measurement.inputSignal.get_
↪btype()
```

Specifies the measured burst / packet type.

return
burst_type: BR: Basic Rate EDR: Enhanced Data Rate LE: Low Energy

get_dmode() → AutoManualMode

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:DMODe
value: enums.AutoManualMode = driver.configure.bluetooth.measurement.
↪inputSignal.get_dmode()
```

Selects an algorithm which the CMP180 uses to detect the measured burst.

return
detection_mode: No help available

set_asynchronize(*auto_synch: bool*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:ASYNchronize
driver.configure.bluetooth.measurement.inputSignal.set_asynchronize(auto_synch␣
↪= False)
```

Disables / enables automatic synchronization to the captured signal for an unspecified Bluetooth device address.

param auto_synch
No help available

set_btype(*burst_type: BurstType*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:BTYPe
driver.configure.bluetooth.measurement.inputSignal.set_btype(burst_type = enums.
↪BurstType.BR)
```

Specifies the measured burst / packet type.

param burst_type
BR: Basic Rate EDR: Enhanced Data Rate LE: Low Energy

set_dmode(*detection_mode: AutoManualMode*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:DMODe
driver.configure.bluetooth.measurement.inputSignal.set_dmode(detection_mode =␣
↪enums.AutoManualMode.AUTO)
```

Selects an algorithm which the CMP180 uses to detect the measured burst.

param detection_mode
No help available

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.configure.bluetooth.measurement.inputSignal.clone()
```

Subgroups

6.4.1.1.7.1 Aacc

SCPI Command :

```
CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:AACC:QHSL
```

class AaccCls

Aacc commands group definition. 1 total commands, 0 Subgroups, 1 group commands

get_qhsl() → str

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:AACC:QHSL
value: str = driver.configure.bluetooth.measurement.inputSignal.aacc.get_qhsl()
```

No command help available

return
access_address: No help available

set_qhsl(*access_address: str*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:AACC:QHSL
driver.configure.bluetooth.measurement.inputSignal.aacc.set_qhsl(access_address␣
↪= rawAbc)
```

No command help available

param access_address
No help available

6.4.1.1.7.2 AccAddress

SCPI Command :

```
CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:ACCaddress:LENergy
```

class AccAddressCls

AccAddress commands group definition. 1 total commands, 0 Subgroups, 1 group commands

get_low_energy() → str

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:ACCaddress:LENergy
value: str = driver.configure.bluetooth.measurement.inputSignal.accAddress.get_
↪low_energy()
```

Specifies the access address of the advertiser for standalone LE measurements.

return
access_address: No help available

set_low_energy(*access_address: str*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:ACCaddress:LENergy
driver.configure.bluetooth.measurement.inputSignal.accAddress.set_low_
↪energy(access_address = rawAbc)
```

Specifies the access address of the advertiser for standalone LE measurements.

param access_address
No help available

6.4.1.1.7.3 BdAddress

SCPI Commands :

```
CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:BDADdress:QHSL
CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:BDADdress
```

class BdAddressCls

BdAddress commands group definition. 2 total commands, 0 Subgroups, 2 group commands

get_qhsl() → str

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:BDADdress:QHSL
value: str = driver.configure.bluetooth.measurement.inputSignal.bdAddress.get_
↪qhsl()
```

No command help available

return
bd_address: No help available

get_value() → str

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:BDADdress
value: str = driver.configure.bluetooth.measurement.inputSignal.bdAddress.get_
↪value()
```

Specifies the Bluetooth device address that the CMP180 expects the DUT to use to generate its access code.

return
bd_address: 12-digit hex number

set_qhsl(*bd_address: str*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:BDADdress:QHSL
driver.configure.bluetooth.measurement.inputSignal.bdAddress.set_qhsl(bd_
↪address = rawAbc)
```

No command help available

param bd_address
No help available

set_value(*bd_address: str*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:BDADdress
driver.configure.bluetooth.measurement.inputSignal.bdAddress.set_value(bd_
↪address = rawAbc)
```

Specifies the Bluetooth device address that the CMP180 expects the DUT to use to generate its access code.

param bd_address
12-digit hex number

6.4.1.1.7.4 Cscheme

class CschemeCls

Cscheme commands group definition. 1 total commands, 1 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.configure.bluetooth.measurement.inputSignal.cscheme.clone()
```

Subgroups

6.4.1.1.7.5 LowEnergy

SCPI Command :

```
CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:CSCHeme:LENergy:LRANge
```

class LowEnergyCls

LowEnergy commands group definition. 1 total commands, 0 Subgroups, 1 group commands

get_lrange() → CodingScheme

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:CSCHeme:LENergy:LRANge
value: enums.CodingScheme = driver.configure.bluetooth.measurement.inputSignal.
↪cscheme.lowEnergy.get_lrange()
```

No command help available

return
coding_scheme: No help available

set_lrange(*coding_scheme: CodingScheme*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:CSCHeme:LENergy:LRANge
driver.configure.bluetooth.measurement.inputSignal.cscheme.lowEnergy.set_
↪lrange(coding_scheme = enums.CodingScheme.S2)
```

No command help available

param coding_scheme
No help available

6.4.1.1.7.6 Cte

class CteCls

Cte commands group definition. 14 total commands, 2 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.configure.bluetooth.measurement.inputSignal.cte.clone()
```

Subgroups

6.4.1.1.7.7 LowEnergy

class LowEnergyCls

LowEnergy commands group definition. 4 total commands, 2 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.configure.bluetooth.measurement.inputSignal.cte.lowEnergy.clone()
```

Subgroups

6.4.1.1.7.8 Le1M

SCPI Commands :

```
CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:CTE:LENergy:LE1M:TYPE
CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:CTE:LENergy:LE1M:UNITs
```

class Le1MCls

Le1M commands group definition. 2 total commands, 0 Subgroups, 2 group commands

get_type_py() → CtePacketType

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:CTE:LENergy:LE1M:TYPE
value: enums.CtePacketType = driver.configure.bluetooth.measurement.inputSignal.
↪cte.lowEnergy.le1M.get_type_py()
```

Specifies the CTE slot type for LE with CTE. Commands for uncoded LE 1M PHY (..:LE1M..) and LE 2M PHY (..:LE2M..) are available.

return

cte_type: AOD1us, AOD2us: CTE type angle of departure, 1 µs or 2 µs slot AOAus, AOA2us: CTE type angle of arrival, 2 µs slot AOA1us: CTE type angle of arrival, 1 µs slot

get_units() → int

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:CTE:LENergy:LE1M:UNITs
value: int = driver.configure.bluetooth.measurement.inputSignal.cte.lowEnergy.
↪le1M.get_units()
```

Specifies the number of CTE units for LE with CTE. Commands for uncoded LE 1M PHY (..:LE1M..) and LE 2M PHY (..:LE2M..) are available.

return

cte_units: No help available

set_type_py(*cte_type: CtePacketType*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:CTE:LENergy:LE1M:TYPE
driver.configure.bluetooth.measurement.inputSignal.cte.lowEnergy.le1M.set_type_
↪py(cte_type = enums.CtePacketType.AOA1us)
```

Specifies the CTE slot type for LE with CTE. Commands for uncoded LE 1M PHY (..:LE1M..) and LE 2M PHY (..:LE2M..) are available.

param cte_type
AOD1us, AOD2us: CTE type angle of departure, 1 µs or 2 µs slot AOAus, AOA2us: CTE type angle of arrival, 2 µs slot AOA1us: CTE type angle of arrival, 1 µs slot

set_units(*cte_units: int*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:CTE:LENergy:LE1M:UNITs
driver.configure.bluetooth.measurement.inputSignal.cte.lowEnergy.le1M.set_
↪units(cte_units = 1)
```

Specifies the number of CTE units for LE with CTE. Commands for uncoded LE 1M PHY (..:LE1M..) and LE 2M PHY (..:LE2M..) are available.

param cte_units
No help available

6.4.1.1.7.9 Le2M

SCPI Commands :

```
CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:CTE:LENergy:LE2M:TYPE
CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:CTE:LENergy:LE2M:UNITs
```

class Le2MCls

Le2M commands group definition. 2 total commands, 0 Subgroups, 2 group commands

get_type_py() → CtePacketType

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:CTE:LENergy:LE2M:TYPE
value: enums.CtePacketType = driver.configure.bluetooth.measurement.inputSignal.
↪cte.lowEnergy.le2M.get_type_py()
```

Specifies the CTE slot type for LE with CTE. Commands for uncoded LE 1M PHY (..:LE1M..) and LE 2M PHY (..:LE2M..) are available.

return
cte_type: AOD1us, AOD2us: CTE type angle of departure, 1 µs or 2 µs slot AOAus, AOA2us: CTE type angle of arrival, 2 µs slot AOA1us: CTE type angle of arrival, 1 µs slot

get_units() → int

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:CTE:LENergy:LE2M:UNITs
value: int = driver.configure.bluetooth.measurement.inputSignal.cte.lowEnergy.
↪le2M.get_units()
```

Specifies the number of CTE units for LE with CTE. Commands for uncoded LE 1M PHY (..:LE1M..) and LE 2M PHY (..:LE2M..) are available.

return
cte_units: No help available

set_type_py(*cte_type: CtePacketType*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:CTE:LENergy:LE2M:TYPE
driver.configure.bluetooth.measurement.inputSignal.cte.lowEnergy.le2M.set_type_
↪py(cte_type = enums.CtePacketType.AOA1us)
```

Specifies the CTE slot type for LE with CTE. Commands for uncoded LE 1M PHY (..:LE1M..) and LE 2M PHY (..:LE2M..) are available.

param cte_type
AOD1us, AOD2us: CTE type angle of departure, 1 µs or 2 µs slot AOAus, AOA2us: CTE type angle of arrival, 2 µs slot AOA1us: CTE type angle of arrival, 1 µs slot

set_units(*cte_units: int*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:CTE:LENergy:LE2M:UNITs
driver.configure.bluetooth.measurement.inputSignal.cte.lowEnergy.le2M.set_
↪units(cte_units = 1)
```

Specifies the number of CTE units for LE with CTE. Commands for uncoded LE 1M PHY (..:LE1M..) and LE 2M PHY (..:LE2M..) are available.

param cte_units
No help available

6.4.1.1.7.10 Qhsl

class QhslCls

Qhsl commands group definition. 10 total commands, 5 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.configure.bluetooth.measurement.inputSignal.cte.qhsl.clone()
```

Subgroups

6.4.1.1.7.11 P2Q

SCPI Commands :

```
CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:CTE:QHSL:P2Q:TYPE
CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:CTE:QHSL:P2Q:UNITs
```

class P2QCls

P2Q commands group definition. 2 total commands, 0 Subgroups, 2 group commands

get_type_py() → CtePacketType

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:CTE:QHSL:P2Q:TYPE
value: enums.CtePacketType = driver.configure.bluetooth.measurement.inputSignal.
↪cte.qhsl.p2Q.get_type_py()
```

No command help available

return
cte_type: No help available

get_units() → int

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:CTE:QHSL:P2Q:UNITs
value: int = driver.configure.bluetooth.measurement.inputSignal.cte.qhsl.p2Q.
↪get_units()
```

No command help available

return
cte_units: No help available

set_type_py(*cte_type: CtePacketType*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:CTE:QHSL:P2Q:TYPE
driver.configure.bluetooth.measurement.inputSignal.cte.qhsl.p2Q.set_type_py(cte_
↪type = enums.CtePacketType.AOA1us)
```

No command help available

param cte_type
No help available

set_units(*cte_units: int*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:CTE:QHSL:P2Q:UNITs
driver.configure.bluetooth.measurement.inputSignal.cte.qhsl.p2Q.set_units(cte_
↪units = 1)
```

No command help available

param cte_units
No help available

6.4.1.1.7.12 P3Q

SCPI Commands :

```
CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:CTE:QHSL:P3Q:TYPE
CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:CTE:QHSL:P3Q:UNITs
```

class P3QCls

P3Q commands group definition. 2 total commands, 0 Subgroups, 2 group commands

get_type_py() → CtePacketType

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:CTE:QHSL:P3Q:TYPE
value: enums.CtePacketType = driver.configure.bluetooth.measurement.inputSignal.
↪cte.qhsl.p3Q.get_type_py()
```

No command help available

return
cte_type: No help available

get_units() → int

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:CTE:QHSL:P3Q:UNITs
value: int = driver.configure.bluetooth.measurement.inputSignal.cte.qhsl.p3Q.
↪get_units()
```

No command help available

return
cte_units: No help available

set_type_py(*cte_type: CtePacketType*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:CTE:QHSL:P3Q:TYPE
driver.configure.bluetooth.measurement.inputSignal.cte.qhsl.p3Q.set_type_py(cte_
↪type = enums.CtePacketType.AOA1us)
```

No command help available

param cte_type
No help available

set_units(*cte_units: int*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:CTE:QHSL:P3Q:UNITs
driver.configure.bluetooth.measurement.inputSignal.cte.qhsl.p3Q.set_units(cte_
↪units = 1)
```

No command help available

param cte_units
No help available

6.4.1.1.7.13 P4Q

SCPI Commands :

```
CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:CTE:QHSL:P4Q:TYPE
CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:CTE:QHSL:P4Q:UNITs
```

class P4QCls

P4Q commands group definition. 2 total commands, 0 Subgroups, 2 group commands

get_type_py() → CtePacketType

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:CTE:QHSL:P4Q:TYPE
value: enums.CtePacketType = driver.configure.bluetooth.measurement.inputSignal.
↪cte.qhsl.p4Q.get_type_py()
```

No command help available

return
cte_type: No help available

get_units() → int

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:CTE:QHSL:P4Q:UNITs
value: int = driver.configure.bluetooth.measurement.inputSignal.cte.qhsl.p4Q.
↪get_units()
```

No command help available

return
cte_units: No help available

set_type_py(*cte_type: CtePacketType*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:CTE:QHSL:P4Q:TYPE
driver.configure.bluetooth.measurement.inputSignal.cte.qhsl.p4Q.set_type_py(cte_
↪type = enums.CtePacketType.AOA1us)
```

No command help available

param cte_type
No help available

set_units(*cte_units: int*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:CTE:QHSL:P4Q:UNITs
driver.configure.bluetooth.measurement.inputSignal.cte.qhsl.p4Q.set_units(cte_
↪units = 1)
```

No command help available

param cte_units
No help available

6.4.1.1.7.14 P5Q

SCPI Commands :

```
CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:CTE:QHSL:P5Q:TYPE
CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:CTE:QHSL:P5Q:UNITs
```

class P5QCls

P5Q commands group definition. 2 total commands, 0 Subgroups, 2 group commands

get_type_py() → CtePacketType

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:CTE:QHSL:P5Q:TYPE
value: enums.CtePacketType = driver.configure.bluetooth.measurement.inputSignal.
↪cte.qhsl.p5Q.get_type_py()
```

No command help available

return
cte_type: No help available

get_units() → int

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:CTE:QHSL:P5Q:UNITs
value: int = driver.configure.bluetooth.measurement.inputSignal.cte.qhsl.p5Q.
↪get_units()
```

No command help available

return
cte_units: No help available

set_type_py(*cte_type: CtePacketType*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:CTE:QHSL:P5Q:TYPE
driver.configure.bluetooth.measurement.inputSignal.cte.qhsl.p5Q.set_type_py(cte_
↪type = enums.CtePacketType.AOA1us)
```

No command help available

param cte_type
No help available

set_units(*cte_units: int*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:CTE:QHSL:P5Q:UNITs
driver.configure.bluetooth.measurement.inputSignal.cte.qhsl.p5Q.set_units(cte_
↪units = 1)
```

No command help available

param cte_units
No help available

6.4.1.1.7.15 P6Q

SCPI Commands :

```
CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:CTE:QHSL:P6Q:TYPE
CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:CTE:QHSL:P6Q:UNITs
```

class P6QCls

P6Q commands group definition. 2 total commands, 0 Subgroups, 2 group commands

get_type_py() → CtePacketType

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:CTE:QHSL:P6Q:TYPE
value: enums.CtePacketType = driver.configure.bluetooth.measurement.inputSignal.
↪cte.qhsl.p6Q.get_type_py()
```

No command help available

return
cte_type: No help available

get_units() → int

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:CTE:QHSL:P6Q:UNITs
value: int = driver.configure.bluetooth.measurement.inputSignal.cte.qhsl.p6Q.
↪get_units()
```

No command help available

return
cte_units: No help available

set_type_py(*cte_type: CtePacketType*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:CTE:QHSL:P6Q:TYPE
driver.configure.bluetooth.measurement.inputSignal.cte.qhsl.p6Q.set_type_py(cte_
↪type = enums.CtePacketType.AOA1us)
```

No command help available

param cte_type
No help available

set_units(*cte_units: int*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:CTE:QHSL:P6Q:UNITs
driver.configure.bluetooth.measurement.inputSignal.cte.qhsl.p6Q.set_units(cte_
↪units = 1)
```

No command help available

param cte_units
No help available

6.4.1.1.7.16 Dacc

SCPI Command :

```
CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:DACC:QHSL
```

class DaccCls

Dacc commands group definition. 1 total commands, 0 Subgroups, 1 group commands

get_qhsl() → str

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:DACC:QHSL
value: str = driver.configure.bluetooth.measurement.inputSignal.dacc.get_qhsl()
```

No command help available

return
access_address: No help available

set_qhsl(*access_address: str*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:DACC:QHSL
driver.configure.bluetooth.measurement.inputSignal.dacc.set_qhsl(access_address␣
↪= rawAbc)
```

No command help available

param access_address
No help available

6.4.1.1.7.17 DtMode

class DtModeCls

DtMode commands group definition. 9 total commands, 3 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.configure.bluetooth.measurement.inputSignal.dtMode.clone()
```

Subgroups

6.4.1.1.7.18 Pattern

class PatternCls

Pattern commands group definition. 3 total commands, 1 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.configure.bluetooth.measurement.inputSignal.dtMode.pattern.clone()
```

Subgroups

6.4.1.1.7.19 LowEnergy

SCPI Commands :

```
CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:DTMode:PATTern:LENergy:LE1M
CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:DTMode:PATTern:LENergy:LE2M
CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:DTMode:PATTern:LENergy:LRANge
```

class LowEnergyCls

LowEnergy commands group definition. 3 total commands, 0 Subgroups, 3 group commands

get_le_1_m() → PatternType

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:ISIGnal:DTMode:PATTern:LENergy:LE1M
value: enums.PatternType = driver.configure.bluetooth.measurement.inputSignal.
↪dtMode.pattern.lowEnergy.get_le_1_m()
```

No command help available

return
pattern_type: No help available

get_le_2_m() → PatternType

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:ISIGnal:DTMode:PATTern:LENergy:LE2M
value: enums.PatternType = driver.configure.bluetooth.measurement.inputSignal.
↪dtMode.pattern.lowEnergy.get_le_2_m()
```

No command help available

return
pattern_type: No help available

get_lrange() → PatternType

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:ISIGnal:DTMode:PATTern:LENergy:LRANge
value: enums.PatternType = driver.configure.bluetooth.measurement.inputSignal.
↪dtMode.pattern.lowEnergy.get_lrange()
```

No command help available

return
pattern_type: No help available

set_le_1_m(*pattern_type: PatternType*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:ISIGnal:DTMode:PATTern:LENergy:LE1M
driver.configure.bluetooth.measurement.inputSignal.dtMode.pattern.lowEnergy.set_
↪le_1_m(pattern_type = enums.PatternType.ALL0)
```

No command help available

param pattern_type
No help available

set_le_2_m(*pattern_type: PatternType*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:ISIGnal:DTMode:PATTern:LENergy:LE2M
driver.configure.bluetooth.measurement.inputSignal.dtMode.pattern.lowEnergy.set_
↪le_2_m(pattern_type = enums.PatternType.ALL0)
```

No command help available

param pattern_type
No help available

set_lrange(*pattern_type: PatternType*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:ISIGnal:DTMode:PATTern:LENergy:LRANge
driver.configure.bluetooth.measurement.inputSignal.dtMode.pattern.lowEnergy.set_
↪lrange(pattern_type = enums.PatternType.ALL0)
```

No command help available

param pattern_type
No help available

6.4.1.1.7.20 Plength

class PlengthCls

Plength commands group definition. 3 total commands, 1 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.configure.bluetooth.measurement.inputSignal.dtMode.plength.clone()
```

Subgroups

6.4.1.1.7.21 LowEnergy

SCPI Commands :

```
CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:DTMode:PLENgth:LENergy:LE1M
CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:DTMode:PLENgth:LENergy:LE2M
CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:DTMode:PLENgth:LENergy:LRANge
```

class LowEnergyCls

LowEnergy commands group definition. 3 total commands, 0 Subgroups, 3 group commands

get_le_1_m() → int

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:ISIGnal:DTMode:PLENgth:LENergy:LE1M
value: int = driver.configure.bluetooth.measurement.inputSignal.dtMode.plength.
↪lowEnergy.get_le_1_m()
```

No command help available

return
payload_length: No help available

get_le_2_m() → int

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:ISIGnal:DTMode:PLENgth:LENergy:LE2M
value: int = driver.configure.bluetooth.measurement.inputSignal.dtMode.plength.
↪lowEnergy.get_le_2_m()
```

No command help available

return
payload_length: No help available

get_lrange() → int

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:ISIGnal:DTMode:PLENgth:LENergy:LRANge
value: int = driver.configure.bluetooth.measurement.inputSignal.dtMode.plength.
↪lowEnergy.get_lrange()
```

No command help available

return
payload_length: No help available

set_le_1_m(*payload_length: int*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:ISIGnal:DTMode:PLENgth:LENergy:LE1M
driver.configure.bluetooth.measurement.inputSignal.dtMode.plength.lowEnergy.set_
↪le_1_m(payload_length = 1)
```

No command help available

param payload_length
No help available

set_le_2_m(*payload_length: int*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:ISIGnal:DTMode:PLENgth:LENergy:LE2M
driver.configure.bluetooth.measurement.inputSignal.dtMode.plength.lowEnergy.set_
↪le_2_m(payload_length = 1)
```

No command help available

param payload_length
No help available

set_lrange(*payload_length: int*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:ISIGnal:DTMode:PLENgth:LENergy:LRANge
driver.configure.bluetooth.measurement.inputSignal.dtMode.plength.lowEnergy.set_
↪lrange(payload_length = 1)
```

No command help available

param payload_length
No help available

6.4.1.1.7.22 RxQuality

class RxQualityCls

RxQuality commands group definition. 3 total commands, 1 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.configure.bluetooth.measurement.inputSignal.dtMode.rxQuality.clone()
```

Subgroups

6.4.1.1.7.23 Plength

class PlengthCls

Plength commands group definition. 3 total commands, 1 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.configure.bluetooth.measurement.inputSignal.dtMode.rxQuality.plength.
↪clone()
```

Subgroups

6.4.1.1.7.24 LowEnergy

SCPI Commands :

```
CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:DTMode:RXQuality:PLENgth:LENergy:LE1M
CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:DTMode:RXQuality:PLENgth:LENergy:LE2M
CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:DTMode:RXQuality:PLENgth:LENergy:LRANge
```

class LowEnergyCls

LowEnergy commands group definition. 3 total commands, 0 Subgroups, 3 group commands

get_le_1_m() → PayloadLength

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:ISIGnal:DTMode:RXQuality:PLENgth:LENergy:LE1M
value: enums.PayloadLength = driver.configure.bluetooth.measurement.inputSignal.
↪dtMode.rxQuality.plength.lowEnergy.get_le_1_m()
```

No command help available

return
payload_length: No help available

get_le_2_m() → PayloadLength

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:ISIGnal:DTMode:RXQuality:PLENgth:LENergy:LE2M
value: enums.PayloadLength = driver.configure.bluetooth.measurement.inputSignal.
↪dtMode.rxQuality.plength.lowEnergy.get_le_2_m()
```

No command help available

return
payload_length: No help available

get_lrange() → PayloadLength

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:ISIGnal:DTMode:RXQuality:PLENgth:LENergy:LRANge
value: enums.PayloadLength = driver.configure.bluetooth.measurement.inputSignal.
↪dtMode.rxQuality.plength.lowEnergy.get_lrange()
```

No command help available

return
payload_length: No help available

set_le_1_m(*payload_length: PayloadLength*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:ISIGnal:DTMode:RXQuality:PLENgth:LENergy:LE1M
driver.configure.bluetooth.measurement.inputSignal.dtMode.rxQuality.plength.
↪lowEnergy.set_le_1_m(payload_length = enums.PayloadLength._255)
```

No command help available

param payload_length
No help available

set_le_2_m(*payload_length: PayloadLength*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:ISIGnal:DTMode:RXQuality:PLENgth:LENergy:LE2M
driver.configure.bluetooth.measurement.inputSignal.dtMode.rxQuality.plength.
↪lowEnergy.set_le_2_m(payload_length = enums.PayloadLength._255)
```

No command help available

param payload_length
No help available

set_lrange(*payload_length: PayloadLength*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:ISIGnal:DTMode:RXQuality:PLENgth:LENergy:LRANge
driver.configure.bluetooth.measurement.inputSignal.dtMode.rxQuality.plength.
↪lowEnergy.set_lrange(payload_length = enums.PayloadLength._255)
```

No command help available

param payload_length
No help available

6.4.1.1.7.25 Fec

class FecCls

Fec commands group definition. 1 total commands, 1 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.configure.bluetooth.measurement.inputSignal.fec.clone()
```

Subgroups

6.4.1.1.7.26 LowEnergy

SCPI Command :

```
CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:FEC:LENergy:LRANge
```

class LowEnergyCls

LowEnergy commands group definition. 1 total commands, 0 Subgroups, 1 group commands

get_lrange() → CodingScheme

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:FEC:LENergy:LRANge
value: enums.CodingScheme = driver.configure.bluetooth.measurement.inputSignal.
↪fec.lowEnergy.get_lrange()
```

Defines S coding for LE coded PHY according to the core specification version 5.0 for Bluetooth wireless technology.

return
coding_scheme: Coding S = 8 or S = 2

set_lrange(*coding_scheme: CodingScheme*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:FEC:LENergy:LRANge
driver.configure.bluetooth.measurement.inputSignal.fec.lowEnergy.set_
↪lrange(coding_scheme = enums.CodingScheme.S2)
```

Defines S coding for LE coded PHY according to the core specification version 5.0 for Bluetooth wireless technology.

param coding_scheme
Coding S = 8 or S = 2

6.4.1.1.7.27 Lap

SCPI Commands :

```
CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:LAP:QHSL
CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:LAP
```

class LapCls

Lap commands group definition. 2 total commands, 0 Subgroups, 2 group commands

get_qhsl() → str

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:LAP:QHSL
value: str = driver.configure.bluetooth.measurement.inputSignal.lap.get_qhsl()
```

No command help available

return
bd_address_lap: No help available

get_value() → str

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:LAP
value: str = driver.configure.bluetooth.measurement.inputSignal.lap.get_value()
```

Specifies the lower address part of the DUT's Bluetooth device address.

return
bd_address_lap: Six-digit hex number

set_qhsl(*bd_address_lap: str*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:LAP:QHSL
driver.configure.bluetooth.measurement.inputSignal.lap.set_qhsl(bd_address_lap␣
↪= rawAbc)
```

No command help available

param bd_address_lap
No help available

set_value(*bd_address_lap: str*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:LAP
driver.configure.bluetooth.measurement.inputSignal.lap.set_value(bd_address_lap␣
↪= rawAbc)
```

Specifies the lower address part of the DUT's Bluetooth device address.

param bd_address_lap
Six-digit hex number

6.4.1.1.7.28 LowEnergy

SCPI Commands :

```
CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:LENergy:SYNWord
CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:LENergy:PHY
```

class LowEnergyCls

LowEnergy commands group definition. 2 total commands, 0 Subgroups, 2 group commands

get_phy() → LePhysicalType

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:LENergy:PHY
value: enums.LePhysicalType = driver.configure.bluetooth.measurement.
↪inputSignal.lowEnergy.get_phy()
```

Selects the physical layer used for LE measurements.

return
phy: LE1M: LE 1 Msymbol/s uncoded PHY LE2M: LE 2 Msymbol/s uncoded PHY LELR: LE 1 Msymbol/s long range (LE coded PHY)

get_syn_word() → str

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:LENergy:SYNWord
value: str = driver.configure.bluetooth.measurement.inputSignal.lowEnergy.get_
↪syn_word()
```

Specifies the synchronization word used during direct test mode.

return
synch_word: No help available

set_phy(*phy: LePhysicalType*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:LENergy:PHY
driver.configure.bluetooth.measurement.inputSignal.lowEnergy.set_phy(phy =␣
↪enums.LePhysicalType.LE1M)
```

Selects the physical layer used for LE measurements.

param phy
LE1M: LE 1 Msymbol/s uncoded PHY LE2M: LE 2 Msymbol/s uncoded PHY LELR: LE 1 Msymbol/s long range (LE coded PHY)

set_syn_word(*synch_word: str*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:LENergy:SYNWord
driver.configure.bluetooth.measurement.inputSignal.lowEnergy.set_syn_word(synch_
↪word = rawAbc)
```

Specifies the synchronization word used during direct test mode.

param synch_word
No help available

6.4.1.1.7.29 Nap

SCPI Commands :

```
CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:NAP:QHSL
CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:NAP
```

class NapCls

Nap commands group definition. 2 total commands, 0 Subgroups, 2 group commands

get_qhsl() → str

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:NAP:QHSL
value: str = driver.configure.bluetooth.measurement.inputSignal.nap.get_qhsl()
```

No command help available

return
bd_address_nap: No help available

get_value() → str

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:NAP
value: str = driver.configure.bluetooth.measurement.inputSignal.nap.get_value()
```

Specifies the non-specific address part of the DUT's Bluetooth device address.

return
bd_address_nap: Four-digit hex number

set_qhsl(*bd_address_nap: str*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:NAP:QHSL
driver.configure.bluetooth.measurement.inputSignal.nap.set_qhsl(bd_address_nap␣
↪= rawAbc)
```

No command help available

param bd_address_nap
No help available

set_value(*bd_address_nap: str*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:NAP
driver.configure.bluetooth.measurement.inputSignal.nap.set_value(bd_address_nap␣
↪= rawAbc)
```

Specifies the non-specific address part of the DUT's Bluetooth device address.

param bd_address_nap
Four-digit hex number

6.4.1.1.7.30 Oslots

SCPI Commands :

```
CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:OSLots:EDRate
CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:OSLots:BRATe
```

class OslotsCls

Oslots commands group definition. 5 total commands, 1 Subgroups, 2 group commands

get_brate() → List[int]

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:OSLots:BRATe
value: List[int] = driver.configure.bluetooth.measurement.inputSignal.oslots.
↪get_brate()
```

No command help available

return
no_of_off_slots: No help available

get_edrate() → List[int]

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:OSLots:EDRate
value: List[int] = driver.configure.bluetooth.measurement.inputSignal.oslots.
↪get_edrate()
```

No command help available

return
no_of_off_slots: No help available

set_brate(*no_of_off_slots: List[int]*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:OSLots:BRATe
driver.configure.bluetooth.measurement.inputSignal.oslots.set_brate(no_of_off_
↪slots = [1, 2, 3])
```

No command help available

param no_of_off_slots
No help available

set_edrate(*no_of_off_slots: List[int]*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:OSLots:EDRate
driver.configure.bluetooth.measurement.inputSignal.oslots.set_edrate(no_of_off_
↪slots = [1, 2, 3])
```

No command help available

param no_of_off_slots
No help available

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.configure.bluetooth.measurement.inputSignal.oslots.clone()
```

Subgroups

6.4.1.1.7.31 LowEnergy

SCPI Commands :

```
CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:OSLots:LENergy[:LE1M]
CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:OSLots:LENergy:LRANge
CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:OSLots:LENergy:LE2M
```

class LowEnergyCls

LowEnergy commands group definition. 3 total commands, 0 Subgroups, 3 group commands

get_le_1_m() → int

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:OSLots:LENergy[:LE1M]
value: int = driver.configure.bluetooth.measurement.inputSignal.oslots.
↪lowEnergy.get_le_1_m()
```

No command help available

return
no_of_off_slots: No help available

get_le_2_m() → int

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:OSLots:LENergy:LE2M
value: int = driver.configure.bluetooth.measurement.inputSignal.oslots.
↪lowEnergy.get_le_2_m()
```

No command help available

return
no_of_off_slots: No help available

get_lrange() → int

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:OSLots:LENergy:LRANge
value: int = driver.configure.bluetooth.measurement.inputSignal.oslots.
↪lowEnergy.get_lrange()
```

No command help available

return
no_of_off_slots: No help available

set_le_1_m(*no_of_off_slots: int*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:OSLots:LENergy[:LE1M]
driver.configure.bluetooth.measurement.inputSignal.oslots.lowEnergy.set_le_1_
↪m(no_of_off_slots = 1)
```

No command help available

param no_of_off_slots
No help available

set_le_2_m(*no_of_off_slots: int*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:OSLots:LENergy:LE2M
driver.configure.bluetooth.measurement.inputSignal.oslots.lowEnergy.set_le_2_
↪m(no_of_off_slots = 1)
```

No command help available

param no_of_off_slots
No help available

set_lrange(*no_of_off_slots: int*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:OSLots:LENergy:LRANge
driver.configure.bluetooth.measurement.inputSignal.oslots.lowEnergy.set_
↪lrange(no_of_off_slots = 1)
```

No command help available

param no_of_off_slots
No help available

6.4.1.1.7.32 Pattern

SCPI Command :

```
CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:PATTern
```

class PatternCls

Pattern commands group definition. 4 total commands, 1 Subgroups, 1 group commands

get_value() → DetectedPatternType

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:PATTern
value: enums.DetectedPatternType = driver.configure.bluetooth.measurement.
↪inputSignal.pattern.get_value()
```

The command specifies the data pattern type that the DUT transmits as user payload data on its BR packets.

return
pattern_type: P11: 10101010 P44: 11110000 OTHer: any pattern except P11, P44
ALTernating: the periodical change between the pattern P11 and P44

set_value(*pattern_type: DetectedPatternType*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:PATTern
driver.configure.bluetooth.measurement.inputSignal.pattern.set_value(pattern_
↪type = enums.DetectedPatternType.ALTernating)
```

The command specifies the data pattern type that the DUT transmits as user payload data on its BR packets.

param pattern_type
P11: 10101010 P44: 11110000 OTHer: any pattern except P11, P44 ALTernating: the periodical change between the pattern P11 and P44

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.configure.bluetooth.measurement.inputSignal.pattern.clone()
```

Subgroups

6.4.1.1.7.33 LowEnergy

SCPI Commands :

```
CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:PATTern:LENergy[:LE1M]
CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:PATTern:LENergy:LRANge
CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:PATTern:LENergy:LE2M
```

class LowEnergyCls

LowEnergy commands group definition. 3 total commands, 0 Subgroups, 3 group commands

get_le_1_m() → LePatternType

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:PATTern:LENergy[:LE1M]
value: enums.LePatternType = driver.configure.bluetooth.measurement.inputSignal.
↪pattern.lowEnergy.get_le_1_m()
```

It specifies the data pattern type that the EUT transmits as user payload data in its LE packets. Commands for LE 1M PHY (...:LE1M...) and LE 2M PHY (...:LE2M...) are available.

return
pattern_type: P11: '10101010' in transmission order (LSB first) P44: '11110000' in transmission order (LSB first) OTHer: any pattern except P11, P44

get_le_2_m() → LePatternType

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:PATTern:LENergy:LE2M
value: enums.LePatternType = driver.configure.bluetooth.measurement.inputSignal.
↪pattern.lowEnergy.get_le_2_m()
```

It specifies the data pattern type that the EUT transmits as user payload data in its LE packets. Commands for LE 1M PHY (...:LE1M...) and LE 2M PHY (...:LE2M...) are available.

return
pattern_type: P11: '10101010' in transmission order (LSB first) P44: '11110000' in transmission order (LSB first) OTHer: any pattern except P11, P44

get_lrange() → TransmitPatternType

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:PATTern:LENergy:LRANge
value: enums.TransmitPatternType = driver.configure.bluetooth.measurement.
↪inputSignal.pattern.lowEnergy.get_lrange()
```

Specifies the data pattern type for LE coded PHY, that the EUT transmits as user payload data.

return
pattern_type: ALL1: '11111111' OTHer: any pattern except ALL1

set_le_1_m(*pattern_type: LePatternType*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:PATTern:LENergy[:LE1M]
driver.configure.bluetooth.measurement.inputSignal.pattern.lowEnergy.set_le_1_
↪m(pattern_type = enums.LePatternType.OTHer)
```

It specifies the data pattern type that the EUT transmits as user payload data in its LE packets. Commands for LE 1M PHY (...:LE1M...) and LE 2M PHY (...:LE2M...) are available.

param pattern_type
P11: '10101010' in transmission order (LSB first) P44: '11110000' in transmission order (LSB first) OTHer: any pattern except P11, P44

set_le_2_m(*pattern_type: LePatternType*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:PATTern:LENergy:LE2M
driver.configure.bluetooth.measurement.inputSignal.pattern.lowEnergy.set_le_2_
↪m(pattern_type = enums.LePatternType.OTHer)
```

It specifies the data pattern type that the EUT transmits as user payload data in its LE packets. Commands for LE 1M PHY (...:LE1M...) and LE 2M PHY (...:LE2M...) are available.

param pattern_type
P11: '10101010' in transmission order (LSB first) P44: '11110000' in transmission order (LSB first) OTHer: any pattern except P11, P44

set_lrange(*pattern_type: TransmitPatternType*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:PATTern:LENergy:LRANge
driver.configure.bluetooth.measurement.inputSignal.pattern.lowEnergy.set_
↪lrange(pattern_type = enums.TransmitPatternType.ALL1)
```

Specifies the data pattern type for LE coded PHY, that the EUT transmits as user payload data.

param pattern_type
ALL1: '11111111' OTHer: any pattern except ALL1

6.4.1.1.7.34 Plength

SCPI Commands :

```
CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:PLENgth:BRATe
CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:PLENgth:EDRate
```

class PlengthCls

Plength commands group definition. 10 total commands, 2 Subgroups, 2 group commands

get_brate() → List[int]

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:PLENgth:BRATe
value: List[int] = driver.configure.bluetooth.measurement.inputSignal.plength.
↪get_brate()
```

Specifies the number of bytes (octets) in the payload data of the measured BR signal. The range of values depends on the packet type (method RsCMPX_BluetoothMeas.Configure.Bluetooth.Measurement.InputSignal.Ptype.brate) . The command requires 3 comma-separated parameters, one for each BR packet type (order: DH1, DH3, DH5) .

return
payload_length: 3 payload lengths for BR packets

get_edrate() → List[int]

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:PLENgth:EDRate
value: List[int] = driver.configure.bluetooth.measurement.inputSignal.plength.
↪get_edrate()
```

Specifies the number of bytes (octets) in the payload data of the measured EDR signal. The range of values depends on the packet type (method RsCMPX_BluetoothMeas.Configure.Bluetooth.Measurement.InputSignal.Ptype.edrate) . The command requires 6 comma-separated parameters, one for each EDR packet type (order: 2-DH1, 2-DH3, 2-DH5, 3-DH1, 3-DH3, 3-DH5) .

return
payload_length: 6 payload lengths for EDR packets

set_brate(*payload_length: List[int]*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:PLENgth:BRATe
driver.configure.bluetooth.measurement.inputSignal.plength.set_brate(payload_
↪length = [1, 2, 3])
```

Specifies the number of bytes (octets) in the payload data of the measured BR signal. The range of values depends on the packet type (method RsCMPX_BluetoothMeas.Configure.Bluetooth.Measurement.InputSignal.Ptype.brate) . The command requires 3 comma-separated parameters, one for each BR packet type (order: DH1, DH3, DH5) .

param payload_length
3 payload lengths for BR packets

set_edrate(*payload_length: List[int]*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:PLENgth:EDRate
driver.configure.bluetooth.measurement.inputSignal.plength.set_edrate(payload_
↪length = [1, 2, 3])
```

Specifies the number of bytes (octets) in the payload data of the measured EDR signal. The range of values depends on the packet type (method RsCMPX_BluetoothMeas.Configure.Bluetooth.Measurement.InputSignal.Ptype.edrate) . The command requires 6 comma-separated parameters, one for each EDR packet type (order: 2-DH1, 2-DH3, 2-DH5, 3-DH1, 3-DH3, 3-DH5) .

param payload_length
6 payload lengths for EDR packets

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.configure.bluetooth.measurement.inputSignal.plength.clone()
```

Subgroups

6.4.1.1.7.35 LowEnergy

SCPI Commands :

```
CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:PLENgth:LENergy[:LE1M]
CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:PLENgth:LENergy:LRANge
CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:PLENgth:LENergy:LE2M
```

class LowEnergyCls

LowEnergy commands group definition. 3 total commands, 0 Subgroups, 3 group commands

get_le_1_m() → int

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:PLENgth:LENergy[:LE1M]
value: int = driver.configure.bluetooth.measurement.inputSignal.plength.
↪lowEnergy.get_le_1_m()
```

Specifies the number of bytes (octets) in the payload data of the measured LE packets. Commands for uncoded LE 1M PHY (.. :LE1M..) , LE 2M PHY (..:LE2M..) , and LE coded PHY (..:LRANge..) are available.

return
payload_length: Payload lengths for LE packets

get_le_2_m() → int

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:PLENgth:LENergy:LE2M
value: int = driver.configure.bluetooth.measurement.inputSignal.plength.
↪lowEnergy.get_le_2_m()
```

Specifies the number of bytes (octets) in the payload data of the measured LE packets. Commands for uncoded LE 1M PHY (.. :LE1M..) , LE 2M PHY (..:LE2M..) , and LE coded PHY (..:LRANge..) are available.

return
payload_length: Payload lengths for LE packets

get_lrange() → int

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:PLENgth:LENergy:LRANge
value: int = driver.configure.bluetooth.measurement.inputSignal.plength.
↪lowEnergy.get_lrange()
```

Specifies the number of bytes (octets) in the payload data of the measured LE packets. Commands for uncoded LE 1M PHY (.. :LE1M..) , LE 2M PHY (..:LE2M..) , and LE coded PHY (..:LRANge..) are available.

return
payload_length: Payload lengths for LE packets

set_le_1_m(*payload_length: int*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:PLENgth:LENergy[:LE1M]
driver.configure.bluetooth.measurement.inputSignal.plength.lowEnergy.set_le_1_
↪m(payload_length = 1)
```

Specifies the number of bytes (octets) in the payload data of the measured LE packets. Commands for uncoded LE 1M PHY (.. :LE1M..) , LE 2M PHY (..:LE2M..) , and LE coded PHY (..:LRANge..) are available.

param payload_length
Payload lengths for LE packets

set_le_2_m(*payload_length: int*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:PLENgth:LENergy:LE2M
driver.configure.bluetooth.measurement.inputSignal.plength.lowEnergy.set_le_2_
↪m(payload_length = 1)
```

Specifies the number of bytes (octets) in the payload data of the measured LE packets. Commands for uncoded LE 1M PHY (.. :LE1M..) , LE 2M PHY (..:LE2M..) , and LE coded PHY (..:LRANge..) are available.

param payload_length
Payload lengths for LE packets

set_lrange(*payload_length: int*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:PLENgth:LENergy:LRANge
driver.configure.bluetooth.measurement.inputSignal.plength.lowEnergy.set_
↪lrange(payload_length = 1)
```

Specifies the number of bytes (octets) in the payload data of the measured LE packets. Commands for uncoded LE 1M PHY (.. :LE1M..) , LE 2M PHY (..:LE2M..) , and LE coded PHY (..:LRANge..) are available.

param payload_length
Payload lengths for LE packets

6.4.1.1.7.36 Qhsl

SCPI Commands :

```
CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:PLENgth:QHSL:P2Q
CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:PLENgth:QHSL:P3Q
CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:PLENgth:QHSL:P4Q
CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:PLENgth:QHSL:P5Q
CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:PLENgth:QHSL:P6Q
```

class QhslCls

Qhsl commands group definition. 5 total commands, 0 Subgroups, 5 group commands

get_p_2_q() → int

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:PLENgth:QHSL:P2Q
value: int = driver.configure.bluetooth.measurement.inputSignal.plength.qhsl.
↪get_p_2_q()
```

No command help available

return
payload_length: No help available

get_p_3_q() → int

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:PLENgth:QHSL:P3Q
value: int = driver.configure.bluetooth.measurement.inputSignal.plength.qhsl.
↪get_p_3_q()
```

No command help available

return
payload_length: No help available

get_p_4_q() → int

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:PLENgth:QHSL:P4Q
value: int = driver.configure.bluetooth.measurement.inputSignal.plength.qhsl.
↪get_p_4_q()
```

No command help available

return
payload_length: No help available

get_p_5_q() → int

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:PLENgth:QHSL:P5Q
value: int = driver.configure.bluetooth.measurement.inputSignal.plength.qhsl.
↪get_p_5_q()
```

No command help available

return
payload_length: No help available

get_p_6_q() → int

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:PLENgth:QHSL:P6Q
value: int = driver.configure.bluetooth.measurement.inputSignal.plength.qhsl.
↪get_p_6_q()
```

No command help available

return
payload_length: No help available

set_p_2_q(*payload_length: int*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:PLENgth:QHSL:P2Q
driver.configure.bluetooth.measurement.inputSignal.plength.qhsl.set_p_2_
↪q(payload_length = 1)
```

No command help available

param payload_length
No help available

set_p_3_q(*payload_length: int*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:PLENgth:QHSL:P3Q
driver.configure.bluetooth.measurement.inputSignal.plength.qhsl.set_p_3_
↪q(payload_length = 1)
```

No command help available

param payload_length
No help available

set_p_4_q(*payload_length: int*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:PLENgth:QHSL:P4Q
driver.configure.bluetooth.measurement.inputSignal.plength.qhsl.set_p_4_
↪q(payload_length = 1)
```

No command help available

param payload_length
No help available

set_p_5_q(*payload_length: int*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:PLENgth:QHSL:P5Q
driver.configure.bluetooth.measurement.inputSignal.plength.qhsl.set_p_5_
↪q(payload_length = 1)
```

No command help available

param payload_length
No help available

set_p_6_q(*payload_length: int*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:PLENgth:QHSL:P6Q
driver.configure.bluetooth.measurement.inputSignal.plength.qhsl.set_p_6_
↪q(payload_length = 1)
```

No command help available

param payload_length
No help available

6.4.1.1.7.37 Ptype

SCPI Commands :

```
CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:PTYPe:EDRate
CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:PTYPe:BRATe
```

class PtypeCls

Ptype commands group definition. 10 total commands, 2 Subgroups, 2 group commands

get_brate() → BrPacketType

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:PTYPe:BRATe
value: enums.BrPacketType = driver.configure.bluetooth.measurement.inputSignal.
↪ptype.get_brate()
```

Specifies the BR packet type of the measured signal.

return
packet_type: No help available

get_edrate() → EdrPacketType

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:PTYPe:EDRate
value: enums.EdrPacketType = driver.configure.bluetooth.measurement.inputSignal.
↪ptype.get_edrate()
```

Specifies the EDR packet type of the measured signal.

return
packet_type: 2-DH1, 2-DH3, 2-DH5, 3-DH1, 3-DH3, or 3-DH5 packets

set_brate(*packet_type: BrPacketType*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:PTYPe:BRATe
driver.configure.bluetooth.measurement.inputSignal.ptype.set_brate(packet_type␣
↪= enums.BrPacketType.DH1)
```

Specifies the BR packet type of the measured signal.

param packet_type
No help available

set_edrate(*packet_type: EdrPacketType*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:PTYPe:EDRate
driver.configure.bluetooth.measurement.inputSignal.ptype.set_edrate(packet_type
↪= enums.EdrPacketType.E21P)
```

Specifies the EDR packet type of the measured signal.

param packet_type
2-DH1, 2-DH3, 2-DH5, 3-DH1, 3-DH3, or 3-DH5 packets

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.configure.bluetooth.measurement.inputSignal.ptype.clone()
```

Subgroups

6.4.1.1.7.38 LowEnergy

SCPI Commands :

```
CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:PTYPe:LENergy[:LE1M]
CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:PTYPe:LENergy:LRANge
CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:PTYPe:LENergy:LE2M
```

class LowEnergyCls

LowEnergy commands group definition. 3 total commands, 0 Subgroups, 3 group commands

get_le_1_m() → LePacketType

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:PTYPe:LENergy[:LE1M]
value: enums.LePacketType = driver.configure.bluetooth.measurement.inputSignal.
↪ptype.lowEnergy.get_le_1_m()
```

Specifies the packet type of the measured LE signal.

return
le_packet_type: RFPHytest: LE test packet ADVertiser: air interface packet with advertising channel PDU RFCTe: LE with CTE test packet

get_le_2_m() → LePacketType

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:PTYPe:LENergy:LE2M
value: enums.LePacketType = driver.configure.bluetooth.measurement.inputSignal.
↪ptype.lowEnergy.get_le_2_m()
```

Specifies the packet type of the measured LE signal.

return
ele_packet_type: No help available

get_lrange() → LePacketType

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:PTYPe:LENergy:LRANge
value: enums.LePacketType = driver.configure.bluetooth.measurement.inputSignal.
↪ptype.lowEnergy.get_lrange()
```

Specifies the packet type of the measured signal for LE coded PHY.

return
le_lr_packet_type: RFPHytest: LE test packet ADVertiser: air interface packet with advertising channel PDU

set_le_1_m(*le_packet_type: LePacketType*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:PTYPe:LENergy[:LE1M]
driver.configure.bluetooth.measurement.inputSignal.ptype.lowEnergy.set_le_1_
↪m(le_packet_type = enums.LePacketType.ADVertiser)
```

Specifies the packet type of the measured LE signal.

param le_packet_type
RFPHytest: LE test packet ADVertiser: air interface packet with advertising channel PDU RFCTe: LE with CTE test packet

set_le_2_m(*ele_packet_type: LePacketType*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:PTYPe:LENergy:LE2M
driver.configure.bluetooth.measurement.inputSignal.ptype.lowEnergy.set_le_2_
↪m(ele_packet_type = enums.LePacketType.ADVertiser)
```

Specifies the packet type of the measured LE signal.

param ele_packet_type
RFPHytest: LE test packet ADVertiser: air interface packet with advertising channel PDU RFCTe: LE with CTE test packet

set_lrange(*le_lr_packet_type: LePacketType*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:PTYPe:LENergy:LRANge
driver.configure.bluetooth.measurement.inputSignal.ptype.lowEnergy.set_
↪lrange(le_lr_packet_type = enums.LePacketType.ADVertiser)
```

Specifies the packet type of the measured signal for LE coded PHY.

param le_lr_packet_type
RFPHytest: LE test packet ADVertiser: air interface packet with advertising channel PDU

6.4.1.1.7.39 Qhsl

SCPI Commands :

```
CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:PTYPe:QHSL:P2Q
CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:PTYPe:QHSL:P3Q
CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:PTYPe:QHSL:P4Q
CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:PTYPe:QHSL:P5Q
CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:PTYPe:QHSL:P6Q
```

class QhslCls

Qhsl commands group definition. 5 total commands, 0 Subgroups, 5 group commands

get_p_2_q() → PacketTypeB

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:PTYPe:QHSL:P2Q
value: enums.PacketTypeB = driver.configure.bluetooth.measurement.inputSignal.
↪ptype.qhsl.get_p_2_q()
```

No command help available

return
packet_type: No help available

get_p_3_q() → PacketTypeB

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:PTYPe:QHSL:P3Q
value: enums.PacketTypeB = driver.configure.bluetooth.measurement.inputSignal.
↪ptype.qhsl.get_p_3_q()
```

No command help available

return
packet_type: No help available

get_p_4_q() → PacketTypeB

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:PTYPe:QHSL:P4Q
value: enums.PacketTypeB = driver.configure.bluetooth.measurement.inputSignal.
↪ptype.qhsl.get_p_4_q()
```

No command help available

return
packet_type: No help available

get_p_5_q() → PacketTypeB

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:PTYPe:QHSL:P5Q
value: enums.PacketTypeB = driver.configure.bluetooth.measurement.inputSignal.
↪ptype.qhsl.get_p_5_q()
```

No command help available

return
packet_type: No help available

get_p_6_q() → PacketTypeB

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:PTYPe:QHSL:P6Q
value: enums.PacketTypeB = driver.configure.bluetooth.measurement.inputSignal.
↪ptype.qhsl.get_p_6_q()
```

No command help available

return
packet_type: No help available

set_p_2_q(*packet_type: PacketTypeB*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:PTYPe:QHSL:P2Q
driver.configure.bluetooth.measurement.inputSignal.ptype.qhsl.set_p_2_q(packet_
↪type = enums.PacketTypeB.DATA)
```

No command help available

param packet_type
No help available

set_p_3_q(*packet_type: PacketTypeB*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:PTYPe:QHSL:P3Q
driver.configure.bluetooth.measurement.inputSignal.ptype.qhsl.set_p_3_q(packet_
↪type = enums.PacketTypeB.DATA)
```

No command help available

param packet_type
No help available

set_p_4_q(*packet_type: PacketTypeB*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:PTYPe:QHSL:P4Q
driver.configure.bluetooth.measurement.inputSignal.ptype.qhsl.set_p_4_q(packet_
↪type = enums.PacketTypeB.DATA)
```

No command help available

param packet_type
No help available

set_p_5_q(*packet_type: PacketTypeB*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:PTYPe:QHSL:P5Q
driver.configure.bluetooth.measurement.inputSignal.ptype.qhsl.set_p_5_q(packet_
↪type = enums.PacketTypeB.DATA)
```

No command help available

param packet_type
No help available

set_p_6_q(*packet_type: PacketTypeB*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:PTYPe:QHSL:P6Q
driver.configure.bluetooth.measurement.inputSignal.ptype.qhsl.set_p_6_q(packet_
↪type = enums.PacketTypeB.DATA)
```

No command help available

param packet_type
No help available

6.4.1.1.7.40 Qhsl

SCPI Command :

```
CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:QHSL:PHY
```

class QhslCls

Qhsl commands group definition. 1 total commands, 0 Subgroups, 1 group commands

get_phy() → DetectedPhyType

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:QHSL:PHY
value: enums.DetectedPhyType = driver.configure.bluetooth.measurement.
↪inputSignal.qhsl.get_phy()
```

No command help available

return
phy: No help available

set_phy(*phy: DetectedPhyType*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:QHSL:PHY
driver.configure.bluetooth.measurement.inputSignal.qhsl.set_phy(phy = enums.
↪DetectedPhyType.P2Q)
```

No command help available

param phy
No help available

6.4.1.1.7.41 SynWord

SCPI Command :

```
CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:SYNWord:LENergy
```

class SynWordCls

SynWord commands group definition. 1 total commands, 0 Subgroups, 1 group commands

get_low_energy() → str

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:SYNWord:LENergy
value: str = driver.configure.bluetooth.measurement.inputSignal.synWord.get_low_
↪energy()
```

No command help available

return
synch_word: No help available

set_low_energy(*synch_word: str*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:SYNWord:LENergy
driver.configure.bluetooth.measurement.inputSignal.synWord.set_low_energy(synch_
↪word = rawAbc)
```

No command help available

param synch_word
No help available

6.4.1.1.7.42 Uap

SCPI Commands :

```
CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:UAP:QHSL
CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:UAP
```

class UapCls

Uap commands group definition. 2 total commands, 0 Subgroups, 2 group commands

get_qhsl() → str

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:UAP:QHSL
value: str = driver.configure.bluetooth.measurement.inputSignal.uap.get_qhsl()
```

No command help available

return
bd_address_uap: No help available

get_value() → str

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:UAP
value: str = driver.configure.bluetooth.measurement.inputSignal.uap.get_value()
```

Specifies the upper address part of the DUT's Bluetooth device address.

return
bd_address_uap: Two-digit hex number

set_qhsl(*bd_address_uap: str*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:UAP:QHSL
driver.configure.bluetooth.measurement.inputSignal.uap.set_qhsl(bd_address_uap␣
↪= rawAbc)
```

No command help available

param bd_address_uap
No help available

set_value(*bd_address_uap: str*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:UAP
driver.configure.bluetooth.measurement.inputSignal.uap.set_value(bd_address_uap␣
↪= rawAbc)
```

Specifies the upper address part of the DUT's Bluetooth device address.

param bd_address_uap
Two-digit hex number

6.4.1.1.8 MultiEval

SCPI Commands :

```
CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:TOUT
CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:MOEXception
CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:SCONdition
CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:REPetition
```

class MultiEvalCls

MultiEval commands group definition. 154 total commands, 14 Subgroups, 4 group commands

get_mo_exception() → bool

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:MOEXception
value: bool = driver.configure.bluetooth.measurement.multiEval.get_mo_
↪exception()
```

Specifies whether measurement results that are identified as faulty or inaccurate are rejected.

return
meas_on_exception: ON: Results are never rejected. OFF: Faulty results are rejected.

get_repetition() → Repeat

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:REPetition
value: enums.Repeat = driver.configure.bluetooth.measurement.multiEval.get_
↪repetition()
```

Specifies the repetition mode of the measurement. The repetition mode specifies whether the measurement is stopped after a single shot or repeated continuously. Use CONFigure:..:MEAS<i>:...:SCOunt to determine the number of measurement intervals per single shot.

return
repetition: SINGleshot: Single-shot measurement CONTinuous: Continuous measurement

get_scondition() → StopCondition

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:SCONdition
value: enums.StopCondition = driver.configure.bluetooth.measurement.multiEval.
↪get_scondition()
```

Qualifies whether the measurement is stopped after a failed limit check or continued. SLFail means that the measurement is stopped and reaches the RDY state when one of the results exceeds the limits.

return
stop_condition: NONE: Continue measurement irrespective of the limit check. SLFail: Stop measurement on limit failure.

get_timeout() → float

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:TOUT
value: float = driver.configure.bluetooth.measurement.multiEval.get_timeout()
```

Defines a timeout for the measurement. The timer is started when the measurement is initiated via a READ or INIT command. It is not started if the measurement is initiated manually. When the measurement has completed the first measurement cycle (first single shot) , the statistical depth is reached and the timer is reset. If the first measurement cycle has not been completed when the timer expires, the measurement is stopped. The measurement state changes to RDY. The reliability indicator is set to 1, indicating that a measurement timeout occurred. Still running READ, FETCh or CALCulate commands are completed, returning the available results. At least for some results, there are no values at all or the statistical depth has not been reached. A timeout of 0 s corresponds to an infinite measurement timeout.

return
timeout: No help available

set_mo_exception(*meas_on_exception: bool*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:MOEXception
driver.configure.bluetooth.measurement.multiEval.set_mo_exception(meas_on_
↪exception = False)
```

Specifies whether measurement results that are identified as faulty or inaccurate are rejected.

param meas_on_exception
ON: Results are never rejected. OFF: Faulty results are rejected.

set_repetition(*repetition: Repeat*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:REPetition
driver.configure.bluetooth.measurement.multiEval.set_repetition(repetition =␣
↪enums.Repeat.CONTinuous)
```

Specifies the repetition mode of the measurement. The repetition mode specifies whether the measurement is stopped after a single shot or repeated continuously. Use CONFigure:..:MEAS<i>:...:SCOunt to determine the number of measurement intervals per single shot.

param repetition
SINGleshot: Single-shot measurement CONTinuous: Continuous measurement

set_scondition(*stop_condition: StopCondition*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:SCONdition
driver.configure.bluetooth.measurement.multiEval.set_scondition(stop_condition␣
↪= enums.StopCondition.NONE)
```

Qualifies whether the measurement is stopped after a failed limit check or continued. SLFail means that the measurement is stopped and reaches the RDY state when one of the results exceeds the limits.

param stop_condition
NONE: Continue measurement irrespective of the limit check. SLFail: Stop measurement on limit failure.

set_timeout(*timeout: float*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:TOUT
driver.configure.bluetooth.measurement.multiEval.set_timeout(timeout = 1.0)
```

Defines a timeout for the measurement. The timer is started when the measurement is initiated via a READ or INIT command. It is not started if the measurement is initiated manually. When the measurement has completed the first measurement cycle (first single shot) , the statistical depth is reached and the timer is reset. If the first measurement cycle has not been completed when the timer expires, the measurement is stopped. The measurement state changes to RDY. The reliability indicator is set to 1, indicating that a measurement timeout occurred. Still running READ, FETCh or CALCulate commands are completed, returning the available results. At least for some results, there are no values at all or the statistical depth has not been reached. A timeout of 0 s corresponds to an infinite measurement timeout.

param timeout
No help available

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.configure.bluetooth.measurement.multiEval.clone()
```

Subgroups

6.4.1.1.8.1 Brate

class BrateCls

Brate commands group definition. 1 total commands, 1 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.configure.bluetooth.measurement.multiEval.brate.clone()
```

Subgroups

6.4.1.1.8.2 FilterPy

SCPI Command :

```
CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:BRATe:FILTer:BWIDth
```

class FilterPyCls

FilterPy commands group definition. 1 total commands, 0 Subgroups, 1 group commands

get_bandwidth() → FilterWidth

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:MEValuation:BRATe:FILTer:BWIDth
value: enums.FilterWidth = driver.configure.bluetooth.measurement.multiEval.
↪brate.filterPy.get_bandwidth()
```

Selects the filter bandwidth for BR measurements.

return
filter_bandwidth: NARRow: Narrowband filter WIDE: Wideband filter

set_bandwidth(*filter_bandwidth: FilterWidth*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:MEValuation:BRATe:FILTer:BWIDth
driver.configure.bluetooth.measurement.multiEval.brate.filterPy.set_
↪bandwidth(filter_bandwidth = enums.FilterWidth.NARRow)
```

Selects the filter bandwidth for BR measurements.

param filter_bandwidth
NARRow: Narrowband filter WIDE: Wideband filter

6.4.1.1.8.3 Edrate

class EdrateCls

Edrate commands group definition. 1 total commands, 1 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.configure.bluetooth.measurement.multiEval.edrate.clone()
```

Subgroups

6.4.1.1.8.4 FilterPy

SCPI Command :

```
CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:EDRate:FILTer:BWIDth
```

class FilterPyCls

FilterPy commands group definition. 1 total commands, 0 Subgroups, 1 group commands

get_bandwidth() → FilterWidth

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:MEValuation:EDRate:FILTer:BWIDth
value: enums.FilterWidth = driver.configure.bluetooth.measurement.multiEval.
↪edrate.filterPy.get_bandwidth()
```

Selects the filter bandwidth for EDR measurements. The filter is applied only to the GFSK portion of the bursts.

return
filter_bandwidth: NARRow: Narrowband filter WIDE: Wideband filter

set_bandwidth(*filter_bandwidth: FilterWidth*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:MEValuation:EDRate:FILTer:BWIDth
driver.configure.bluetooth.measurement.multiEval.edrate.filterPy.set_
↪bandwidth(filter_bandwidth = enums.FilterWidth.NARRow)
```

Selects the filter bandwidth for EDR measurements. The filter is applied only to the GFSK portion of the bursts.

param filter_bandwidth
NARRow: Narrowband filter WIDE: Wideband filter

6.4.1.1.8.5 Frange

class FrangeCls

Frange commands group definition. 1 total commands, 1 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.configure.bluetooth.measurement.multiEval.frange.clone()
```

Subgroups

6.4.1.1.8.6 Brate

class BrateCls

Brate commands group definition. 1 total commands, 1 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.configure.bluetooth.measurement.multiEval.frange.brate.clone()
```

Subgroups

6.4.1.1.8.7 Measurement

SCPI Command :

```
CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:FRANge:BRATe:MEASurement
```

class MeasurementCls

Measurement commands group definition. 1 total commands, 0 Subgroups, 1 group commands

class MeasurementStruct

Response structure. Fields:

- Left_Channel: int: Left adjacent channel relative to the EUT center TX channel
- Right_Channel: int: Right adjacent channel relative to the EUT center TX channel
- Threshold: float: Threshold for the spectral power density drop to search the frequencies fL and fH Specification defines - 80 dBm/Hz for equivalent isotropically radiated power or - 30 dBm if measured in a 100 kHz bandwidth.

get() → MeasurementStruct

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:MEValuation:FRANge:BRATe:MEASurement
value: MeasurementStruct = driver.configure.bluetooth.measurement.multiEval.
↪frange.brate.measurement.get()
```

Specifies the number of 1 MHz channels to be measured below and above the current measured channel. The threshold is the level that needs to be crossed to search the frequencies fL and fH.

return
structure: for return value, see the help for MeasurementStruct structure arguments.

set(*left_channel: int*, *right_channel: int*, *threshold: float*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:MEValuation:FRANge:BRATe:MEASurement
driver.configure.bluetooth.measurement.multiEval.frange.brate.measurement.
↪set(left_channel = 1, right_channel = 1, threshold = 1.0)
```

Specifies the number of 1 MHz channels to be measured below and above the current measured channel. The threshold is the level that needs to be crossed to search the frequencies fL and fH.

param left_channel
Left adjacent channel relative to the EUT center TX channel

param right_channel
Right adjacent channel relative to the EUT center TX channel

param threshold
Threshold for the spectral power density drop to search the frequencies fL and fH Specification defines - 80 dBm/Hz for equivalent isotropically radiated power or - 30 dBm if measured in a 100 kHz bandwidth.

6.4.1.1.8.8 Limit

SCPI Command :

```
CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIMit:SGACp
```

class LimitCls

Limit commands group definition. 69 total commands, 9 Subgroups, 1 group commands

class SgacpStruct

Structure for setting input parameters. Fields:

- Ptx_Limit: float: No parameter help available
- Exc_Ptx_Limit: float: No parameter help available
- No_Of_Ex_Limit: int: No parameter help available
- Ptxm_26_N_1_Rel_Lim: float: No parameter help available
- Ptxm_26_P_1_Rel_Lim: float: No parameter help available
- Ptx_Enable: bool: No parameter help available
- No_Of_Exc_Enable: bool: No parameter help available
- Ptxm_26_N_1_Rel_Enable: bool: No parameter help available
- Ptxm_26_P_1_Rel_Enable: bool: No parameter help available

get_sgacp() → SgacpStruct

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIMit:SGACp
value: SgacpStruct = driver.configure.bluetooth.measurement.multiEval.limit.get_
↪sgacp()
```

Defines and enables the upper limits for the Spectrum Gated ACP measurement for EDR packets: PTx, Exceptions PTx, No. of Exceptions, PTx–26 dB–1 (rel) , PTx–26 dB +1 (rel) , and limit check enabling.

return
structure: for return value, see the help for SgacpStruct structure arguments.

set_sgacp(*value: SgacpStruct*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIMit:SGACp
structure = driver.configure.bluetooth.measurement.multiEval.limit.SgacpStruct()
structure.Ptx_Limit: float = 1.0
structure.Exc_Ptx_Limit: float = 1.0
structure.No_Of_Ex_Limit: int = 1
structure.Ptxm_26_N_1_Rel_Lim: float = 1.0
structure.Ptxm_26_P_1_Rel_Lim: float = 1.0
structure.Ptx_Enable: bool = False
structure.No_Of_Exc_Enable: bool = False
structure.Ptxm_26_N_1_Rel_Enable: bool = False
structure.Ptxm_26_P_1_Rel_Enable: bool = False
driver.configure.bluetooth.measurement.multiEval.limit.set_sgacp(value =␣
↪structure)
```

Defines and enables the upper limits for the Spectrum Gated ACP measurement for EDR packets: PTx, Exceptions PTx, No. of Exceptions, PTx–26 dB–1 (rel) , PTx–26 dB +1 (rel) , and limit check enabling.

param value
see the help for SgacpStruct structure arguments.

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.configure.bluetooth.measurement.multiEval.limit.clone()
```

Subgroups

6.4.1.1.8.9 Brate

SCPI Commands :

```
CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIMit:BRATe:DAVerage
CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIMit:BRATe:DMINimum
CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIMit:BRATe:DMAXimum
```

class BrateCls

Brate commands group definition. 9 total commands, 5 Subgroups, 3 group commands

class DaverageStruct

Structure for setting input parameters. Fields:

- Freq_Dev_F_1_Lower: float: No parameter help available
- Freq_Dev_F_1_Upper: float: No parameter help available
- Freq_Dev_F_2_Lower: float: No parameter help available
- Freq_Dev_F_2_Upper: float: No parameter help available
- Freq_Dev_F_1_Enable: List[bool]: Disable or enable limits for current, average, maximum, and minimum results (4 values) .
- Freq_Dev_F_2_Enable: List[bool]: Disable or enable limits for current, average, maximum, and minimum results (4 values) .

class DmaximumStruct

Structure for setting input parameters. Fields:

- Freq_Dev_F_1_Lower: float: No parameter help available
- Freq_Dev_F_1_Upper: float: No parameter help available
- Freq_Dev_F_2_Lower: float: No parameter help available
- Freq_Dev_F_2_Upper: float: No parameter help available
- Freq_Dev_F_1_Enable: List[bool]: Disable or enable limits for current, average, maximum, and minimum results (4 values) .
- Freq_Dev_F_2_Enable: List[bool]: Disable or enable limits for current, average, maximum, and minimum results (4 values) .

class DminimumStruct

Structure for setting input parameters. Fields:

- Freq_Dev_F_1_Lower: float: No parameter help available
- Freq_Dev_F_1_Upper: float: No parameter help available
- Freq_Dev_F_2_Lower: float: No parameter help available

- Freq_Dev_F_2_Upper: float: No parameter help available
- Freq_Dev_F_1_Enable: List[bool]: Disable or enable limits for current, average, maximum, and minimum results (4 values) .
- Freq_Dev_F_2_Enable: List[bool]: Disable or enable limits for current, average, maximum, and minimum results (4 values) .

get_daverage() → DaverageStruct

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:MEValuation:LIMit:BRATe:DAVerage
value: DaverageStruct = driver.configure.bluetooth.measurement.multiEval.limit.
↪brate.get_daverage()
```

Defines the lower and upper frequency deviation limits for BR bursts. The mnemonics DAVerage, DMAXimum, DMINimum distinguish average, maximum, and minimum frequency deviations.

return
structure: for return value, see the help for DaverageStruct structure arguments.

get_dmaximum() → DmaximumStruct

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:MEValuation:LIMit:BRATe:DMAXimum
value: DmaximumStruct = driver.configure.bluetooth.measurement.multiEval.limit.
↪brate.get_dmaximum()
```

Defines the lower and upper frequency deviation limits for BR bursts. The mnemonics DAVerage, DMAXimum, DMINimum distinguish average, maximum, and minimum frequency deviations.

return
structure: for return value, see the help for DmaximumStruct structure arguments.

get_dminimum() → DminimumStruct

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:MEValuation:LIMit:BRATe:DMINimum
value: DminimumStruct = driver.configure.bluetooth.measurement.multiEval.limit.
↪brate.get_dminimum()
```

Defines the lower and upper frequency deviation limits for BR bursts. The mnemonics DAVerage, DMAXimum, DMINimum distinguish average, maximum, and minimum frequency deviations.

return
structure: for return value, see the help for DminimumStruct structure arguments.

set_daverage(*value: DaverageStruct*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:MEValuation:LIMit:BRATe:DAVerage
structure = driver.configure.bluetooth.measurement.multiEval.limit.brate.
↪DaverageStruct()
structure.Freq_Dev_F_1_Lower: float = 1.0
structure.Freq_Dev_F_1_Upper: float = 1.0
structure.Freq_Dev_F_2_Lower: float = 1.0
structure.Freq_Dev_F_2_Upper: float = 1.0
structure.Freq_Dev_F_1_Enable: List[bool] = [True, False, True]
```

(continues on next page)

(continued from previous page)

```
structure.Freq_Dev_F_2_Enable: List[bool] = [True, False, True]
driver.configure.bluetooth.measurement.multiEval.limit.brate.set_daverage(value␣
↪= structure)
```

Defines the lower and upper frequency deviation limits for BR bursts. The mnemonics DAVerage, DMAXimum, DMINimum distinguish average, maximum, and minimum frequency deviations.

param value
see the help for DaverageStruct structure arguments.

set_dmaximum(*value: DmaximumStruct*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:MEValuation:LIMit:BRATe:DMAXimum
structure = driver.configure.bluetooth.measurement.multiEval.limit.brate.
↪DmaximumStruct()
structure.Freq_Dev_F_1_Lower: float = 1.0
structure.Freq_Dev_F_1_Upper: float = 1.0
structure.Freq_Dev_F_2_Lower: float = 1.0
structure.Freq_Dev_F_2_Upper: float = 1.0
structure.Freq_Dev_F_1_Enable: List[bool] = [True, False, True]
structure.Freq_Dev_F_2_Enable: List[bool] = [True, False, True]
driver.configure.bluetooth.measurement.multiEval.limit.brate.set_dmaximum(value␣
↪= structure)
```

Defines the lower and upper frequency deviation limits for BR bursts. The mnemonics DAVerage, DMAXimum, DMINimum distinguish average, maximum, and minimum frequency deviations.

param value
see the help for DmaximumStruct structure arguments.

set_dminimum(*value: DminimumStruct*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:MEValuation:LIMit:BRATe:DMINimum
structure = driver.configure.bluetooth.measurement.multiEval.limit.brate.
↪DminimumStruct()
structure.Freq_Dev_F_1_Lower: float = 1.0
structure.Freq_Dev_F_1_Upper: float = 1.0
structure.Freq_Dev_F_2_Lower: float = 1.0
structure.Freq_Dev_F_2_Upper: float = 1.0
structure.Freq_Dev_F_1_Enable: List[bool] = [True, False, True]
structure.Freq_Dev_F_2_Enable: List[bool] = [True, False, True]
driver.configure.bluetooth.measurement.multiEval.limit.brate.set_dminimum(value␣
↪= structure)
```

Defines the lower and upper frequency deviation limits for BR bursts. The mnemonics DAVerage, DMAXimum, DMINimum distinguish average, maximum, and minimum frequency deviations.

param value
see the help for DminimumStruct structure arguments.

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.configure.bluetooth.measurement.multiEval.limit.brate.clone()
```

Subgroups

6.4.1.1.8.10 Delta

SCPI Command :

```
CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIMit:BRATe:DELTa
```

class DeltaCls

Delta commands group definition. 1 total commands, 0 Subgroups, 1 group commands

class DeltaStruct

Response structure. Fields:

- Delta_F_2_P_99_P_9: float: No parameter help available
- Delta_F_2_P_99_Enable: bool: No parameter help available

get() → DeltaStruct

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIMit:BRATe:DELTa
value: DeltaStruct = driver.configure.bluetooth.measurement.multiEval.limit.
↪brate.delta.get()
```

It defines the lower limit for the frequency deviation f2 that must be exceeded by 99.9% of the measured bits.

return
structure: for return value, see the help for DeltaStruct structure arguments.

set(*delta_f_2_p_99_p_9: float*, *delta_f_2_p_99_enable: bool*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIMit:BRATe:DELTa
driver.configure.bluetooth.measurement.multiEval.limit.brate.delta.set(delta_f_
↪2_p_99_p_9 = 1.0, delta_f_2_p_99_enable = False)
```

It defines the lower limit for the frequency deviation f2 that must be exceeded by 99.9% of the measured bits.

param delta_f_2_p_99_p_9
No help available

param delta_f_2_p_99_enable
No help available

6.4.1.1.8.11 Faccuracy

SCPI Command :

```
CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIMit:BRATe:FACCuracy
```

class FaccuracyCls

Faccuracy commands group definition. 1 total commands, 0 Subgroups, 1 group commands

class FaccuracyStruct

Response structure. Fields:

- Freq_Accuracy: float: No parameter help available
- Freq_Acc_Enabled: List[bool]: Disable or enable limit check for current, average, and maximum results (3 values) .

get() → FaccuracyStruct

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:MEValuation:LIMit:BRATe:FACCuracy
value: FaccuracyStruct = driver.configure.bluetooth.measurement.multiEval.limit.
↪brate.faccuracy.get()
```

Defines the limit for the frequency accuracy.

return
structure: for return value, see the help for FaccuracyStruct structure arguments.

set(*freq_accuracy: float*, *freq_acc_enabled: List[bool]*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:MEValuation:LIMit:BRATe:FACCuracy
driver.configure.bluetooth.measurement.multiEval.limit.brate.faccuracy.set(freq_
↪accuracy = 1.0, freq_acc_enabled = [True, False, True])
```

Defines the limit for the frequency accuracy.

param freq_accuracy
No help available

param freq_acc_enabled
Disable or enable limit check for current, average, and maximum results (3 values) .

6.4.1.1.8.12 Fdrift

SCPI Commands :

```
CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIMit:BRATe:FDRift
CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIMit:BRATe:FDRift:APACkets
```

class FdriftCls

Fdrift commands group definition. 2 total commands, 0 Subgroups, 2 group commands

class ApacketsStruct

Structure for setting input parameters. Fields:

- Freq_Drift_Dh_1: float: No parameter help available
- Freq_Drift_Dh_3: float: No parameter help available
- Freq_Drift_Dh_5: float: No parameter help available
- Max_Drift_Rate: float: No parameter help available
- Freq_Drift_Dh_1_Enb: List[bool]: Disable or enable limit check for current, average, and maximum results (3 values) .
- Freq_Drift_Dh_3_Enb: List[bool]: Disable or enable limit check for current, average, and maximum results (3 values) .
- Freq_Drift_Dh_5_Enb: List[bool]: Disable or enable limit check for current, average, and maximum results (3 values) .
- Max_Drift_Rate_Enb: List[bool]: Disable or enable limit check for current, average, and maximum results (3 values) .

class FdriftStruct

Response structure. Fields:

- Frequency_Drift: float: No parameter help available
- Max_Drift_Rate: float: No parameter help available
- Freq_Drift_Enable: List[bool]: Disable or enable limit check for current, average, and maximum results (3 values) .
- Max_Drift_Rate_Enb: List[bool]: Disable or enable limit check for current, average, and maximum results (3 values) .

get() → FdriftStruct

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIMit:BRATe:FDRift
value: FdriftStruct = driver.configure.bluetooth.measurement.multiEval.limit.
↪brate.fdrift.get()
```

Defines the frequency drift limit for DH1 packets and the maximum drift rate limit for all BR packets. Since V2.1. 20, this command is superseded by the command method RsCMPX_BluetoothMeas.Configure.Bluetooth.Measurement.MultiEval.Limit. Brate.Fdrift.apackets that allows to set different limits for different packet types.

return
structure: for return value, see the help for FdriftStruct structure arguments.

get_apackets() → ApacketsStruct

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:MEValuation:LIMit:BRATe:FDRift:APACkets
value: ApacketsStruct = driver.configure.bluetooth.measurement.multiEval.limit.
↪brate.fdrift.get_apackets()
```

Defines the limits for the frequency drift and the maximum drift rate for BR. For each packet type (DH1, DH3, DH5) a different frequency drift limit can be specified.

return
structure: for return value, see the help for ApacketsStruct structure arguments.

set(*frequency_drift: float*, *max_drift_rate: float*, *freq_drift_enable: List[bool]*, *max_drift_rate_enb: List[bool]*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIMit:BRATe:FDRift
driver.configure.bluetooth.measurement.multiEval.limit.brate.fdrift.
↪set(frequency_drift = 1.0, max_drift_rate = 1.0, freq_drift_enable = [True,␣
↪False, True], max_drift_rate_enb = [True, False, True])
```

Defines the frequency drift limit for DH1 packets and the maximum drift rate limit for all BR packets. Since V2.1. 20, this command is superseded by the command method RsCMPX_BluetoothMeas.Configure.Bluetooth.Measurement.MultiEval.Limit. Brate.Fdrift.apackets that allows to set different limits for different packet types.

param frequency_drift
No help available

param max_drift_rate
No help available

param freq_drift_enable
Disable or enable limit check for current, average, and maximum results (3 values) .

param max_drift_rate_enb
Disable or enable limit check for current, average, and maximum results (3 values) .

set_apackets(*value: ApacketsStruct*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:MEValuation:LIMit:BRATe:FDRift:APACkets
structure = driver.configure.bluetooth.measurement.multiEval.limit.brate.fdrift.
↪ApacketsStruct()
structure.Freq_Drift_Dh_1: float = 1.0
structure.Freq_Drift_Dh_3: float = 1.0
structure.Freq_Drift_Dh_5: float = 1.0
structure.Max_Drift_Rate: float = 1.0
structure.Freq_Drift_Dh_1_Enb: List[bool] = [True, False, True]
structure.Freq_Drift_Dh_3_Enb: List[bool] = [True, False, True]
structure.Freq_Drift_Dh_5_Enb: List[bool] = [True, False, True]
structure.Max_Drift_Rate_Enb: List[bool] = [True, False, True]
driver.configure.bluetooth.measurement.multiEval.limit.brate.fdrift.set_
↪apackets(value = structure)
```

Defines the limits for the frequency drift and the maximum drift rate for BR. For each packet type (DH1, DH3, DH5) a different frequency drift limit can be specified.

param value
see the help for ApacketsStruct structure arguments.

6.4.1.1.8.13 Mratio

SCPI Command :

```
CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIMit:BRATe:MRATio
```

class MratioCls

Mratio commands group definition. 1 total commands, 0 Subgroups, 1 group commands

class MratioStruct

Response structure. Fields:

- Mod_Ratio: float: No parameter help available
- Mod_Ratio_Enabled: bool: Disable/enable limit checking

get() → MratioStruct

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIMit:BRATe:MRATio
value: MratioStruct = driver.configure.bluetooth.measurement.multiEval.limit.
↪brate.mratio.get()
```

Specifies the modulation ratio limit f2 avg / f1 avg for BR bursts.

return
structure: for return value, see the help for MratioStruct structure arguments.

set(*mod_ratio: float*, *mod_ratio_enabled: bool*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIMit:BRATe:MRATio
driver.configure.bluetooth.measurement.multiEval.limit.brate.mratio.set(mod_
↪ratio = 1.0, mod_ratio_enabled = False)
```

Specifies the modulation ratio limit f2 avg / f1 avg for BR bursts.

param mod_ratio
No help available

param mod_ratio_enabled
Disable/enable limit checking

6.4.1.1.8.14 PowerVsTime

SCPI Command :

```
CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIMit:BRATe:PVTime
```

class PowerVsTimeCls

PowerVsTime commands group definition. 1 total commands, 0 Subgroups, 1 group commands

class PowerVsTimeStruct

Response structure. Fields:

- Nom_Pow_Lower: float: No parameter help available
- Nom_Pow_Upper: float: No parameter help available

- Peak_Pow_Upper: float: No parameter help available
- Nom_Pow_Enabled: List[bool]: Disables or enables the limit check for the average power, 4 values, corresponding to the current, average, maximum and minimum results.
- Peak_Pow_Enabled: List[bool]: Disables or enables the limit check for the peak power, 4 values, corresponding to the current, average, maximum and minimum results.

get() → PowerVsTimeStruct

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIMit:BRATe:PVTime
value: PowerVsTimeStruct = driver.configure.bluetooth.measurement.multiEval.
↪limit.brate.powerVsTime.get()
```

Defines the power limits for BR: lower and upper average power limits, upper peak power limit, limit check enabling.

return
structure: for return value, see the help for PowerVsTimeStruct structure arguments.

set(*nom_pow_lower: float*, *nom_pow_upper: float*, *peak_pow_upper: float*, *nom_pow_enabled: List[bool]*, *peak_pow_enabled: List[bool]*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIMit:BRATe:PVTime
driver.configure.bluetooth.measurement.multiEval.limit.brate.powerVsTime.
↪set(nom_pow_lower = 1.0, nom_pow_upper = 1.0, peak_pow_upper = 1.0, nom_pow_
↪enabled = [True, False, True], peak_pow_enabled = [True, False, True])
```

Defines the power limits for BR: lower and upper average power limits, upper peak power limit, limit check enabling.

param nom_pow_lower
No help available

param nom_pow_upper
No help available

param peak_pow_upper
No help available

param nom_pow_enabled
Disables or enables the limit check for the average power, 4 values, corresponding to the current, average, maximum and minimum results.

param peak_pow_enabled
Disables or enables the limit check for the peak power, 4 values, corresponding to the current, average, maximum and minimum results.

6.4.1.1.8.15 Cte

class CteCls

Cte commands group definition. 6 total commands, 1 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.configure.bluetooth.measurement.multiEval.limit.cte.clone()
```

Subgroups

6.4.1.1.8.16 LowEnergy

class LowEnergyCls

LowEnergy commands group definition. 6 total commands, 2 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.configure.bluetooth.measurement.multiEval.limit.cte.lowEnergy.clone()
```

Subgroups

6.4.1.1.8.17 Le1M

SCPI Command :

```
CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIMit:CTE:LENergy:LE1M:FDRift
```

class Le1MCls

Le1M commands group definition. 3 total commands, 2 Subgroups, 1 group commands

class FdriftStruct

Structure for setting input parameters. Fields:

- Frequency_Drift: float: No parameter help available
- Max_Drift_Rate: float: No parameter help available
- Initl_Freq_Drift: float: No parameter help available
- Freq_Drift_Enable: List[bool]: Disable or enable limit checking for current, average, and maximum results (3 values) .
- Max_Drift_Rate_Enb: List[bool]: Disable or enable limit checking for current, average, and maximum results (3 values) .
- Init_Freq_Drift_En: List[bool]: Disable or enable limit checking for current, average, and maximum results (3 values) .

get_fdrift() → FdriftStruct

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:MEValuation:LIMit:CTE:LENergy:LE1M:FDRift
value: FdriftStruct = driver.configure.bluetooth.measurement.multiEval.limit.
↪cte.lowEnergy.le1M.get_fdrift()
```

Sets and enables limits for frequency drift, maximum drift rate and initial frequency drift for the CTE portion. Commands for uncoded LE 1M PHY (..:LE1M..) and LE 2M PHY (..:LE2M..) are available.

return
structure: for return value, see the help for FdriftStruct structure arguments.

set_fdrift(*value: FdriftStruct*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:MEValuation:LIMit:CTE:LENergy:LE1M:FDRift
structure = driver.configure.bluetooth.measurement.multiEval.limit.cte.
↪lowEnergy.le1M.FdriftStruct()
structure.Frequency_Drift: float = 1.0
structure.Max_Drift_Rate: float = 1.0
structure.Initl_Freq_Drift: float = 1.0
structure.Freq_Drift_Enable: List[bool] = [True, False, True]
structure.Max_Drift_Rate_Enb: List[bool] = [True, False, True]
structure.Init_Freq_Drift_En: List[bool] = [True, False, True]
driver.configure.bluetooth.measurement.multiEval.limit.cte.lowEnergy.le1M.set_
↪fdrift(value = structure)
```

Sets and enables limits for frequency drift, maximum drift rate and initial frequency drift for the CTE portion. Commands for uncoded LE 1M PHY (..:LE1M..) and LE 2M PHY (..:LE2M..) are available.

param value
see the help for FdriftStruct structure arguments.

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.configure.bluetooth.measurement.multiEval.limit.cte.lowEnergy.le1M.
↪clone()
```

Subgroups

6.4.1.1.8.18 Foffset

SCPI Command :

```
CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIMit:CTE:LENergy:LE1M:FOFFset
```

class FoffsetCls

Foffset commands group definition. 1 total commands, 0 Subgroups, 1 group commands

class FoffsetStruct

Response structure. Fields:

- Freq_Offset: float: No parameter help available
- Freq_Offset_Enable: List[bool]: Disable or enable limit checking for current, average, and maximum results (3 values)

get() → FoffsetStruct

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:MEValuation:LIMit:CTE:LENergy:LE1M:FOFFset
value: FoffsetStruct = driver.configure.bluetooth.measurement.multiEval.limit.
↪cte.lowEnergy.le1M.foffset.get()
```

Sets/gets the frequency offset limit for the CTE portion. Commands for uncoded LE 1M PHY (..:LE1M..) and LE 2M PHY (.. :LE2M..) are available.

return
structure: for return value, see the help for FoffsetStruct structure arguments.

set(*freq_offset: float*, *freq_offset_enable: List[bool]*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:MEValuation:LIMit:CTE:LENergy:LE1M:FOFFset
driver.configure.bluetooth.measurement.multiEval.limit.cte.lowEnergy.le1M.
↪foffset.set(freq_offset = 1.0, freq_offset_enable = [True, False, True])
```

Sets/gets the frequency offset limit for the CTE portion. Commands for uncoded LE 1M PHY (..:LE1M..) and LE 2M PHY (.. :LE2M..) are available.

param freq_offset
No help available

param freq_offset_enable
Disable or enable limit checking for current, average, and maximum results (3 values)

6.4.1.1.8.19 Pdeviation

SCPI Command :

```
CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIMit:CTE:LENergy:LE1M:PDEViation
```

class PdeviationCls

Pdeviation commands group definition. 1 total commands, 0 Subgroups, 1 group commands

class PdeviationStruct

Response structure. Fields:

- Ref_Dev: float: Upper CTE power limit for reference antenna.
- Tx_Dev: float: Upper limit for power deviation in a slot.
- Ref_Dev_Enable: bool: Enables/disables the CTE power limit check for reference antenna.
- Tx_Dev_Enable: bool: Enables/disables the limit check for power deviation in a slot.

get() → PdeviationStruct

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:MEValuation:LIMit:CTE:LENergy:LE1M:PDEViation
value: PdeviationStruct = driver.configure.bluetooth.measurement.multiEval.
↪limit.cte.lowEnergy.le1M.pdeviation.get()
```

Defines the upper CTE power limits and enables/disables the limit check. Commands for uncoded LE 1M PHY (..:LE1M..) and LE 2M PHY (..:LE2M..) are available.

return
structure: for return value, see the help for PdeviationStruct structure arguments.

set(*ref_dev: float*, *tx_dev: float*, *ref_dev_enable: bool*, *tx_dev_enable: bool*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:MEValuation:LIMit:CTE:LENergy:LE1M:PDEViation
driver.configure.bluetooth.measurement.multiEval.limit.cte.lowEnergy.le1M.
↪pdeviation.set(ref_dev = 1.0, tx_dev = 1.0, ref_dev_enable = False, tx_dev_
↪enable = False)
```

Defines the upper CTE power limits and enables/disables the limit check. Commands for uncoded LE 1M PHY (..:LE1M..) and LE 2M PHY (..:LE2M..) are available.

param ref_dev
Upper CTE power limit for reference antenna.

param tx_dev
Upper limit for power deviation in a slot.

param ref_dev_enable
Enables/disables the CTE power limit check for reference antenna.

param tx_dev_enable
Enables/disables the limit check for power deviation in a slot.

6.4.1.1.8.20 Le2M

SCPI Command :

```
CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIMit:CTE:LENergy:LE2M:FDRift
```

class Le2MCls

Le2M commands group definition. 3 total commands, 2 Subgroups, 1 group commands

class FdriftStruct

Structure for setting input parameters. Fields:

- Frequency_Drift: float: No parameter help available
- Max_Drift_Rate: float: No parameter help available
- Initl_Freq_Drift: float: No parameter help available
- Freq_Drift_Enable: List[bool]: Disable or enable limit checking for current, average, and maximum results (3 values) .
- Max_Drift_Rate_Enb: List[bool]: Disable or enable limit checking for current, average, and maximum results (3 values) .
- Init_Freq_Drift_En: List[bool]: Disable or enable limit checking for current, average, and maximum results (3 values) .

get_fdrift() → FdriftStruct

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:MEValuation:LIMit:CTE:LENergy:LE2M:FDRift
value: FdriftStruct = driver.configure.bluetooth.measurement.multiEval.limit.
↪cte.lowEnergy.le2M.get_fdrift()
```

Sets and enables limits for frequency drift, maximum drift rate and initial frequency drift for the CTE portion. Commands for uncoded LE 1M PHY (..:LE1M..) and LE 2M PHY (..:LE2M..) are available.

return
structure: for return value, see the help for FdriftStruct structure arguments.

set_fdrift(*value: FdriftStruct*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:MEValuation:LIMit:CTE:LENergy:LE2M:FDRift
structure = driver.configure.bluetooth.measurement.multiEval.limit.cte.
↪lowEnergy.le2M.FdriftStruct()
structure.Frequency_Drift: float = 1.0
structure.Max_Drift_Rate: float = 1.0
structure.Initl_Freq_Drift: float = 1.0
structure.Freq_Drift_Enable: List[bool] = [True, False, True]
structure.Max_Drift_Rate_Enb: List[bool] = [True, False, True]
structure.Init_Freq_Drift_En: List[bool] = [True, False, True]
driver.configure.bluetooth.measurement.multiEval.limit.cte.lowEnergy.le2M.set_
↪fdrift(value = structure)
```

Sets and enables limits for frequency drift, maximum drift rate and initial frequency drift for the CTE portion. Commands for uncoded LE 1M PHY (..:LE1M..) and LE 2M PHY (..:LE2M..) are available.

param value
see the help for FdriftStruct structure arguments.

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.configure.bluetooth.measurement.multiEval.limit.cte.lowEnergy.le2M.
↪clone()
```

Subgroups

6.4.1.1.8.21 Foffset

SCPI Command :

```
CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIMit:CTE:LENergy:LE2M:FOFFset
```

class FoffsetCls

Foffset commands group definition. 1 total commands, 0 Subgroups, 1 group commands

class FoffsetStruct

Response structure. Fields:

- Freq_Offset: float: No parameter help available
- Freq_Offset_Enable: List[bool]: Disable or enable limit checking for current, average, and maximum results (3 values)

get() → FoffsetStruct

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:MEValuation:LIMit:CTE:LENergy:LE2M:FOFFset
value: FoffsetStruct = driver.configure.bluetooth.measurement.multiEval.limit.
↪cte.lowEnergy.le2M.foffset.get()
```

Sets/gets the frequency offset limit for the CTE portion. Commands for uncoded LE 1M PHY (..:LE1M..) and LE 2M PHY (.. :LE2M..) are available.

return
structure: for return value, see the help for FoffsetStruct structure arguments.

set(*freq_offset: float*, *freq_offset_enable: List[bool]*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:MEValuation:LIMit:CTE:LENergy:LE2M:FOFFset
driver.configure.bluetooth.measurement.multiEval.limit.cte.lowEnergy.le2M.
↪foffset.set(freq_offset = 1.0, freq_offset_enable = [True, False, True])
```

Sets/gets the frequency offset limit for the CTE portion. Commands for uncoded LE 1M PHY (..:LE1M..) and LE 2M PHY (.. :LE2M..) are available.

param freq_offset
No help available

param freq_offset_enable
Disable or enable limit checking for current, average, and maximum results (3 values)

6.4.1.1.8.22 Pdeviation

SCPI Command :

```
CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIMit:CTE:LENergy:LE2M:PDEViation
```

class PdeviationCls

Pdeviation commands group definition. 1 total commands, 0 Subgroups, 1 group commands

class PdeviationStruct

Response structure. Fields:

- Ref_Dev: float: Upper CTE power limit for reference antenna.
- Tx_Dev: float: Upper limit for power deviation in a slot.
- Ref_Dev_Enable: bool: Enables/disables the CTE power limit check for reference antenna.
- Tx_Dev_Enable: bool: Enables/disables the limit check for power deviation in a slot.

get() → PdeviationStruct

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:MEValuation:LIMit:CTE:LENergy:LE2M:PDEViation
value: PdeviationStruct = driver.configure.bluetooth.measurement.multiEval.
↪limit.cte.lowEnergy.le2M.pdeviation.get()
```

Defines the upper CTE power limits and enables/disables the limit check. Commands for uncoded LE 1M PHY (..:LE1M..) and LE 2M PHY (..:LE2M..) are available.

return
structure: for return value, see the help for PdeviationStruct structure arguments.

set(*ref_dev: float*, *tx_dev: float*, *ref_dev_enable: bool*, *tx_dev_enable: bool*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:MEValuation:LIMit:CTE:LENergy:LE2M:PDEViation
driver.configure.bluetooth.measurement.multiEval.limit.cte.lowEnergy.le2M.
↪pdeviation.set(ref_dev = 1.0, tx_dev = 1.0, ref_dev_enable = False, tx_dev_
↪enable = False)
```

Defines the upper CTE power limits and enables/disables the limit check. Commands for uncoded LE 1M PHY (..:LE1M..) and LE 2M PHY (..:LE2M..) are available.

param ref_dev
Upper CTE power limit for reference antenna.

param tx_dev
Upper limit for power deviation in a slot.

param ref_dev_enable
Enables/disables the CTE power limit check for reference antenna.

param tx_dev_enable
Enables/disables the limit check for power deviation in a slot.

6.4.1.1.8.23 Edrate

SCPI Commands :

```
CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIMit:EDRate:PVTime
CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIMit:EDRate:FSTability
```

class EdrateCls

Edrate commands group definition. 6 total commands, 3 Subgroups, 2 group commands

class FstabilityStruct

Structure for setting input parameters. Fields:

- Wi: float: Limit for the initial center frequency error
- Wiplus_W_0_Max: float: Limit for the overall uncompensated frequency error
- W_0_Max: float: Limit for the maximum compensated frequency error in the DPSK portion of the packet
- Wi_Enabled: List[bool]: Enable limits for current, average, and maximum results (3 values) .

- Wi_W_0_Max_Enabled: List[bool]: Enable limits for current, average, and maximum results (3 values) .
- W_0_Max_Enabled: List[bool]: Enable limits for current, average, and maximum results (3 values) .

class PowerVsTimeStruct

Structure for setting input parameters. Fields:

- Dpsk_Minus_Gfsk_Low: float: No parameter help available
- Dpsk_Minus_Gfsk_Upp: float: No parameter help available
- Guard_Period_Low: float: No parameter help available
- Guard_Period_Upp: float: No parameter help available
- Dpsk_Minus_Gfsk_Enable: List[bool]: Disables or enables the limit check for the DPSK minus GFSK power, 4 values, corresponding to the current, average, maximum and minimum results.
- Guard_Period_Enable: List[bool]: Disables or enables the limit check for the guard period, 4 values, corresponding to the current, average, maximum and minimum results.

get_fstability() → FstabilityStruct

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:MEValuation:LIMit:EDRate:FSTability
value: FstabilityStruct = driver.configure.bluetooth.measurement.multiEval.
↪limit.edrate.get_fstability()
```

Defines and activates upper limits for the frequency stability.

return
structure: for return value, see the help for FstabilityStruct structure arguments.

get_power_vs_time() → PowerVsTimeStruct

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:MEValuation:LIMit:EDRate:PVTime
value: PowerVsTimeStruct = driver.configure.bluetooth.measurement.multiEval.
↪limit.edrate.get_power_vs_time()
```

Defines the power limits for EDR: lower and upper limits for DPSK minus GFSK power and for guard period, limit check enabling.

return
structure: for return value, see the help for PowerVsTimeStruct structure arguments.

set_fstability(*value: FstabilityStruct*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:MEValuation:LIMit:EDRate:FSTability
structure = driver.configure.bluetooth.measurement.multiEval.limit.edrate.
↪FstabilityStruct()
structure.Wi: float = 1.0
structure.Wiplus_W_0_Max: float = 1.0
structure.W_0_Max: float = 1.0
structure.Wi_Enabled: List[bool] = [True, False, True]
structure.Wi_W_0_Max_Enabled: List[bool] = [True, False, True]
structure.W_0_Max_Enabled: List[bool] = [True, False, True]
```

(continues on next page)

(continued from previous page)

```
driver.configure.bluetooth.measurement.multiEval.limit.edrate.set_
↪fstability(value = structure)
```

Defines and activates upper limits for the frequency stability.

param value
see the help for FstabilityStruct structure arguments.

set_power_vs_time(*value: PowerVsTimeStruct*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:MEValuation:LIMit:EDRate:PVTime
structure = driver.configure.bluetooth.measurement.multiEval.limit.edrate.
↪PowerVsTimeStruct()
structure.Dpsk_Minus_Gfsk_Low: float = 1.0
structure.Dpsk_Minus_Gfsk_Upp: float = 1.0
structure.Guard_Period_Low: float = 1.0
structure.Guard_Period_Upp: float = 1.0
structure.Dpsk_Minus_Gfsk_Enable: List[bool] = [True, False, True]
structure.Guard_Period_Enable: List[bool] = [True, False, True]
driver.configure.bluetooth.measurement.multiEval.limit.edrate.set_power_vs_
↪time(value = structure)
```

Defines the power limits for EDR: lower and upper limits for DPSK minus GFSK power and for guard period, limit check enabling.

param value
see the help for PowerVsTimeStruct structure arguments.

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.configure.bluetooth.measurement.multiEval.limit.edrate.clone()
```

Subgroups

6.4.1.1.8.24 Dpsk

SCPI Command :

```
CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIMit:EDRate:DPSK:DEVM
```

class DpskCls

Dpsk commands group definition. 1 total commands, 0 Subgroups, 1 group commands

class DevmStruct

Structure for setting input parameters. Fields:

- Rms: float: Limit for RMS DEVM (8DPSK)
- Peak: float: Limit for peak DEVM (8DPSK)
- P_99: float: Limit for 99% DEVM (8DPSK)

- Rms_Enabled: List[bool]: Disable or enable limit check for current, average, and maximum results (3 values)
- Peak_Enabled: List[bool]: Disable or enable limit check for current, average, and maximum results (3 values)
- P_99_Enabled: bool: Disable or enable limit check for current result (1 value)

get_devm() → DevmStruct

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:MEValuation:LIMit:EDRate:DPSK:DEVM
value: DevmStruct = driver.configure.bluetooth.measurement.multiEval.limit.
↪edrate.dpsk.get_devm()
```

Defines and activates upper limits for the differential error vector magnitude for 8DPSK modulated packets (3-DHx) .

return
structure: for return value, see the help for DevmStruct structure arguments.

set_devm(*value: DevmStruct*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:MEValuation:LIMit:EDRate:DPSK:DEVM
structure = driver.configure.bluetooth.measurement.multiEval.limit.edrate.dpsk.
↪DevmStruct()
structure.Rms: float = 1.0
structure.Peak: float = 1.0
structure.P_99: float = 1.0
structure.Rms_Enabled: List[bool] = [True, False, True]
structure.Peak_Enabled: List[bool] = [True, False, True]
structure.P_99_Enabled: bool = False
driver.configure.bluetooth.measurement.multiEval.limit.edrate.dpsk.set_
↪devm(value = structure)
```

Defines and activates upper limits for the differential error vector magnitude for 8DPSK modulated packets (3-DHx) .

param value
see the help for DevmStruct structure arguments.

6.4.1.1.8.25 Dqpsk

SCPI Command :

```
CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIMit:EDRate:DQPSk:DEVM
```

class DqpskCls

Dqpsk commands group definition. 1 total commands, 0 Subgroups, 1 group commands

class DevmStruct

Structure for setting input parameters. Fields:

- Rms: float: Limit for RMS DEVM (/4 DQPSK)
- Peak: float: Limit for peak DEVM (/4 DQPSK)

- P_99: float: Limit for 99% DEVM (/4 DQPSK)
- Rms_Enabled: List[bool]: Disable or enable limit check for current, average, and maximum results (3 values) .
- Peak_Enabled: List[bool]: Disable or enable limit check for current, average, and maximum results (3 values) .
- P_99_Enabled: bool: Disable or enable limit check for current result (1 value) .

get_devm() → DevmStruct

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:MEValuation:LIMit:EDRate:DQPSk:DEVM
value: DevmStruct = driver.configure.bluetooth.measurement.multiEval.limit.
↪edrate.dqpsk.get_devm()
```

Defines and activates upper limits for the differential error vector magnitude for /4 DQPSK modulated packets (2-DHx) .

return
structure: for return value, see the help for DevmStruct structure arguments.

set_devm(*value: DevmStruct*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:MEValuation:LIMit:EDRate:DQPSk:DEVM
structure = driver.configure.bluetooth.measurement.multiEval.limit.edrate.dqpsk.
↪DevmStruct()
structure.Rms: float = 1.0
structure.Peak: float = 1.0
structure.P_99: float = 1.0
structure.Rms_Enabled: List[bool] = [True, False, True]
structure.Peak_Enabled: List[bool] = [True, False, True]
structure.P_99_Enabled: bool = False
driver.configure.bluetooth.measurement.multiEval.limit.edrate.dqpsk.set_
↪devm(value = structure)
```

Defines and activates upper limits for the differential error vector magnitude for /4 DQPSK modulated packets (2-DHx) .

param value
see the help for DevmStruct structure arguments.

6.4.1.1.8.26 Pencoding

SCPI Command :

```
CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIMit:EDRate:PENCoding
```

class PencodingCls

Pencoding commands group definition. 2 total commands, 1 Subgroups, 1 group commands

class PencodingStruct

Response structure. Fields:

- Phase_Encoding: float: No parameter help available

- Phase_Encod_Enable: bool: No parameter help available

get() → PencodingStruct

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:MEValuation:LIMit:EDRate:PENCoding
value: PencodingStruct = driver.configure.bluetooth.measurement.multiEval.limit.
↪edrate.pencoding.get()
```

No command help available

return
structure: for return value, see the help for PencodingStruct structure arguments.

set(*phase_encoding: float*, *phase_encod_enable: bool*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:MEValuation:LIMit:EDRate:PENCoding
driver.configure.bluetooth.measurement.multiEval.limit.edrate.pencoding.
↪set(phase_encoding = 1.0, phase_encod_enable = False)
```

No command help available

param phase_encoding
No help available

param phase_encod_enable
No help available

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.configure.bluetooth.measurement.multiEval.limit.edrate.pencoding.clone()
```

Subgroups

6.4.1.1.8.27 Ssequence

SCPI Command :

```
CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIMit:EDRate:PENCoding:SSEQuence
```

class SsequenceCls

Ssequence commands group definition. 1 total commands, 0 Subgroups, 1 group commands

class SsequenceStruct

Response structure. Fields:

- Sync_Bit_Errors: int: No parameter help available
- Trailer_Bit_Errs: int: No parameter help available
- Sync_Bit_Enable: bool: No parameter help available
- Trailer_Bit_Enable: bool: No parameter help available

get() → SsequenceStruct

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:MEValuation:LIMit:EDRate:PENCoding:SSEQuence
value: SsequenceStruct = driver.configure.bluetooth.measurement.multiEval.limit.
↪edrate.pencoding.ssequence.get()
```

No command help available

return
structure: for return value, see the help for SsequenceStruct structure arguments.

set(*sync_bit_errors: int*, *trailer_bit_errs: int*, *sync_bit_enable: bool*, *trailer_bit_enable: bool*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:MEValuation:LIMit:EDRate:PENCoding:SSEQuence
driver.configure.bluetooth.measurement.multiEval.limit.edrate.pencoding.
↪ssequence.set(sync_bit_errors = 1, trailer_bit_errs = 1, sync_bit_enable =␣
↪False, trailer_bit_enable = False)
```

No command help available

param sync_bit_errors
No help available

param trailer_bit_errs
No help available

param sync_bit_enable
No help available

param trailer_bit_enable
No help available

6.4.1.1.8.28 Frange

SCPI Command :

```
CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIMit:FRANge
```

class FrangeCls

Frange commands group definition. 1 total commands, 0 Subgroups, 1 group commands

class FrangeStruct

Response structure. Fields:

- Flx_Lower: float: Lower limit for the lowest frequency fL relative to center frequency
- Fhx_Upper: float: Upper limit for the highest frequency fH relative to center frequency
- Flx_Lower_Enable: bool: Disable or enable limit check for the lowest frequency fL
- Fhx_Upper_Enable: bool: Disable or enable limit check for the highest frequency fH

get() → FrangeStruct

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIMit:FRANge
value: FrangeStruct = driver.configure.bluetooth.measurement.multiEval.limit.
↪frange.get()
```

Defines the limit for the frequency range measurement.

return
structure: for return value, see the help for FrangeStruct structure arguments.

set(*flx_lower: float*, *fhx_upper: float*, *flx_lower_enable: bool*, *fhx_upper_enable: bool*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIMit:FRANge
driver.configure.bluetooth.measurement.multiEval.limit.frange.set(flx_lower = 1.
↪0, fhx_upper = 1.0, flx_lower_enable = False, fhx_upper_enable = False)
```

Defines the limit for the frequency range measurement.

param flx_lower
Lower limit for the lowest frequency fL relative to center frequency

param fhx_upper
Upper limit for the highest frequency fH relative to center frequency

param flx_lower_enable
Disable or enable limit check for the lowest frequency fL

param fhx_upper_enable
Disable or enable limit check for the highest frequency fH

6.4.1.1.8.29 LowEnergy

class LowEnergyCls

LowEnergy commands group definition. 35 total commands, 7 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.configure.bluetooth.measurement.multiEval.limit.lowEnergy.clone()
```

Subgroups

6.4.1.1.8.30 Daverage

SCPI Command :

```
CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIMit:LENergy:DAVerage
```

class DaverageCls

Daverage commands group definition. 2 total commands, 1 Subgroups, 1 group commands

class DaverageStruct

Response structure. Fields:

- Freq_Dev_F_1_Lower: float: No parameter help available
- Freq_Dev_F_1_Upper: float: No parameter help available
- Freq_Dev_F_1_Enable: List[bool]: Disable or enable limits for current, average, maximum, and minimum results (4 values)

get() → DaverageStruct

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:MEValuation:LIMit:LENergy:DAVerage
value: DaverageStruct = driver.configure.bluetooth.measurement.multiEval.limit.
↪lowEnergy.daverage.get()
```

Defines the lower and upper f1 frequency deviation limits for LE 1M PHY. The mnemonics DAVerage, DMINimum, DMAXimum distinguish average, minimum and maximum frequency deviations.

return
structure: for return value, see the help for DaverageStruct structure arguments.

set(*freq_dev_f_1_lower: float*, *freq_dev_f_1_upper: float*, *freq_dev_f_1_enable: List[bool]*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:MEValuation:LIMit:LENergy:DAVerage
driver.configure.bluetooth.measurement.multiEval.limit.lowEnergy.daverage.
↪set(freq_dev_f_1_lower = 1.0, freq_dev_f_1_upper = 1.0, freq_dev_f_1_enable =␣
↪[True, False, True])
```

Defines the lower and upper f1 frequency deviation limits for LE 1M PHY. The mnemonics DAVerage, DMINimum, DMAXimum distinguish average, minimum and maximum frequency deviations.

param freq_dev_f_1_lower
No help available

param freq_dev_f_1_upper
No help available

param freq_dev_f_1_enable
Disable or enable limits for current, average, maximum, and minimum results (4 values)

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.configure.bluetooth.measurement.multiEval.limit.lowEnergy.daverage.
↪clone()
```

Subgroups

6.4.1.1.8.31 Df2S

SCPI Command :

```
CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIMit:LENergy:DAVerage:DF2S
```

class Df2SCls

Df2S commands group definition. 1 total commands, 0 Subgroups, 1 group commands

class Df2SStruct

Response structure. Fields:

- Freq_Dev_F_2_Lower: float: No parameter help available
- Freq_Dev_F_2_Upper: float: No parameter help available
- Freq_Dev_F_2_Enable: List[bool]: Disable or enable limits for current, average, maximum, and minimum results (4 values)

get() → Df2SStruct

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:MEValuation:LIMit:LENergy:DAVerage:DF2S
value: Df2SStruct = driver.configure.bluetooth.measurement.multiEval.limit.
↪lowEnergy.daverage.df2S.get()
```

Defines the lower and upper f2 frequency deviation limits for uncoded LE 1M PHY. The mnemonics DAVerage, DMINimum, DMAXimum distinguish average, minimum and maximum frequency deviations.

return
structure: for return value, see the help for Df2SStruct structure arguments.

set(*freq_dev_f_2_lower: float*, *freq_dev_f_2_upper: float*, *freq_dev_f_2_enable: List[bool]*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:MEValuation:LIMit:LENergy:DAVerage:DF2S
driver.configure.bluetooth.measurement.multiEval.limit.lowEnergy.daverage.df2S.
↪set(freq_dev_f_2_lower = 1.0, freq_dev_f_2_upper = 1.0, freq_dev_f_2_enable =␣
↪[True, False, True])
```

Defines the lower and upper f2 frequency deviation limits for uncoded LE 1M PHY. The mnemonics DAVerage, DMINimum, DMAXimum distinguish average, minimum and maximum frequency deviations.

param freq_dev_f_2_lower
No help available

param freq_dev_f_2_upper
No help available

param freq_dev_f_2_enable
Disable or enable limits for current, average, maximum, and minimum results (4 values)

6.4.1.1.8.32 Delta

SCPI Command :

```
CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIMit:LENergy:DELTa
```

class DeltaCls

Delta commands group definition. 1 total commands, 0 Subgroups, 1 group commands

class DeltaStruct

Response structure. Fields:

- Delta_F_2_P_99_P_9: float: No parameter help available
- Delta_F_2_P_99_Enable: bool: Disable/enable limit checking

get() → DeltaStruct

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:MEValuation:LIMit:LENergy:DELTa
value: DeltaStruct = driver.configure.bluetooth.measurement.multiEval.limit.
↪lowEnergy.delta.get()
```

Sets/gets the limit for the frequency deviation f2 for LE 1M PHY that must be exceeded by 99.9% of the measured samples.

return
structure: for return value, see the help for DeltaStruct structure arguments.

set(*delta_f_2_p_99_p_9: float*, *delta_f_2_p_99_enable: bool*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:MEValuation:LIMit:LENergy:DELTa
driver.configure.bluetooth.measurement.multiEval.limit.lowEnergy.delta.
↪set(delta_f_2_p_99_p_9 = 1.0, delta_f_2_p_99_enable = False)
```

Sets/gets the limit for the frequency deviation f2 for LE 1M PHY that must be exceeded by 99.9% of the measured samples.

param delta_f_2_p_99_p_9
No help available

param delta_f_2_p_99_enable
Disable/enable limit checking

6.4.1.1.8.33 Dmaximum

SCPI Command :

```
CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIMit:LENergy:DMAXimum
```

class DmaximumCls

Dmaximum commands group definition. 2 total commands, 1 Subgroups, 1 group commands

class DmaximumStruct

Response structure. Fields:

- Freq_Dev_F_1_Lower: float: No parameter help available
- Freq_Dev_F_1_Upper: float: No parameter help available
- Freq_Dev_F_1_Enable: List[bool]: Disable or enable limits for current, average, maximum, and minimum results (4 values)

get() → DmaximumStruct

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:MEValuation:LIMit:LENergy:DMAXimum
value: DmaximumStruct = driver.configure.bluetooth.measurement.multiEval.limit.
↪lowEnergy.dmaximum.get()
```

Defines the lower and upper f1 frequency deviation limits for LE 1M PHY. The mnemonics DAVerage, DMINimum, DMAXimum distinguish average, minimum and maximum frequency deviations.

return
structure: for return value, see the help for DmaximumStruct structure arguments.

set(*freq_dev_f_1_lower: float*, *freq_dev_f_1_upper: float*, *freq_dev_f_1_enable: List[bool]*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:MEValuation:LIMit:LENergy:DMAXimum
driver.configure.bluetooth.measurement.multiEval.limit.lowEnergy.dmaximum.
↪set(freq_dev_f_1_lower = 1.0, freq_dev_f_1_upper = 1.0, freq_dev_f_1_enable =␣
↪[True, False, True])
```

Defines the lower and upper f1 frequency deviation limits for LE 1M PHY. The mnemonics DAVerage, DMINimum, DMAXimum distinguish average, minimum and maximum frequency deviations.

param freq_dev_f_1_lower
No help available

param freq_dev_f_1_upper
No help available

param freq_dev_f_1_enable
Disable or enable limits for current, average, maximum, and minimum results (4 values)

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.configure.bluetooth.measurement.multiEval.limit.lowEnergy.dmaximum.
↪clone()
```

Subgroups

6.4.1.1.8.34 Df2S

SCPI Command :

```
CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIMit:LENergy:DMAXimum:DF2S
```

class Df2SCls

Df2S commands group definition. 1 total commands, 0 Subgroups, 1 group commands

class Df2SStruct

Response structure. Fields:

- Freq_Dev_F_2_Lower: float: No parameter help available
- Freq_Dev_F_2_Upper: float: No parameter help available
- Freq_Dev_F_2_Enable: List[bool]: Disable or enable limits for current, average, maximum, and minimum results (4 values)

get() → Df2SStruct

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:MEValuation:LIMit:LENergy:DMAXimum:DF2S
value: Df2SStruct = driver.configure.bluetooth.measurement.multiEval.limit.
↪lowEnergy.dmaximum.df2S.get()
```

Defines the lower and upper f2 frequency deviation limits for uncoded LE 1M PHY. The mnemonics DAVerage, DMINimum, DMAXimum distinguish average, minimum and maximum frequency deviations.

return
structure: for return value, see the help for Df2SStruct structure arguments.

set(*freq_dev_f_2_lower: float*, *freq_dev_f_2_upper: float*, *freq_dev_f_2_enable: List[bool]*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:MEValuation:LIMit:LENergy:DMAXimum:DF2S
driver.configure.bluetooth.measurement.multiEval.limit.lowEnergy.dmaximum.df2S.
↪set(freq_dev_f_2_lower = 1.0, freq_dev_f_2_upper = 1.0, freq_dev_f_2_enable =␣
↪[True, False, True])
```

Defines the lower and upper f2 frequency deviation limits for uncoded LE 1M PHY. The mnemonics DAVerage, DMINimum, DMAXimum distinguish average, minimum and maximum frequency deviations.

param freq_dev_f_2_lower
No help available

param freq_dev_f_2_upper
No help available

param freq_dev_f_2_enable
Disable or enable limits for current, average, maximum, and minimum results (4 values)

6.4.1.1.8.35 Dminimum

SCPI Command :

```
CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIMit:LENergy:DMINimum
```

class DminimumCls

Dminimum commands group definition. 2 total commands, 1 Subgroups, 1 group commands

class DminimumStruct

Response structure. Fields:

- Freq_Dev_F_1_Lower: float: No parameter help available
- Freq_Dev_F_1_Upper: float: No parameter help available
- Freq_Dev_F_1_Enable: List[bool]: Disable or enable limits for current, average, maximum, and minimum results (4 values)

get() → DminimumStruct

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:MEValuation:LIMit:LENergy:DMINimum
value: DminimumStruct = driver.configure.bluetooth.measurement.multiEval.limit.
↪lowEnergy.dminimum.get()
```

Defines the lower and upper f1 frequency deviation limits for LE 1M PHY. The mnemonics DAVerage, DMINimum, DMAXimum distinguish average, minimum and maximum frequency deviations.

return
structure: for return value, see the help for DminimumStruct structure arguments.

set(*freq_dev_f_1_lower: float*, *freq_dev_f_1_upper: float*, *freq_dev_f_1_enable: List[bool]*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:MEValuation:LIMit:LENergy:DMINimum
driver.configure.bluetooth.measurement.multiEval.limit.lowEnergy.dminimum.
↪set(freq_dev_f_1_lower = 1.0, freq_dev_f_1_upper = 1.0, freq_dev_f_1_enable =␣
↪[True, False, True])
```

Defines the lower and upper f1 frequency deviation limits for LE 1M PHY. The mnemonics DAVerage, DMINimum, DMAXimum distinguish average, minimum and maximum frequency deviations.

param freq_dev_f_1_lower
No help available

param freq_dev_f_1_upper
No help available

param freq_dev_f_1_enable
Disable or enable limits for current, average, maximum, and minimum results (4 values)

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.configure.bluetooth.measurement.multiEval.limit.lowEnergy.dminimum.
↪clone()
```

Subgroups

6.4.1.1.8.36 Df2S

SCPI Command :

```
CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIMit:LENergy:DMINimum:DF2S
```

class Df2SCls

Df2S commands group definition. 1 total commands, 0 Subgroups, 1 group commands

class Df2SStruct

Response structure. Fields:

- Freq_Dev_F_2_Lower: float: No parameter help available
- Freq_Dev_F_2_Upper: float: No parameter help available
- Freq_Dev_F_2_Enable: List[bool]: Disable or enable limits for current, average, maximum, and minimum results (4 values)

get() → Df2SStruct

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:MEValuation:LIMit:LENergy:DMINimum:DF2S
value: Df2SStruct = driver.configure.bluetooth.measurement.multiEval.limit.
↪lowEnergy.dminimum.df2S.get()
```

Defines the lower and upper f2 frequency deviation limits for uncoded LE 1M PHY. The mnemonics DAVerage, DMINimum, DMAXimum distinguish average, minimum and maximum frequency deviations.

return
structure: for return value, see the help for Df2SStruct structure arguments.

set(*freq_dev_f_2_lower: float*, *freq_dev_f_2_upper: float*, *freq_dev_f_2_enable: List[bool]*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:MEValuation:LIMit:LENergy:DMINimum:DF2S
driver.configure.bluetooth.measurement.multiEval.limit.lowEnergy.dminimum.df2S.
↪set(freq_dev_f_2_lower = 1.0, freq_dev_f_2_upper = 1.0, freq_dev_f_2_enable =␣
↪[True, False, True])
```

Defines the lower and upper f2 frequency deviation limits for uncoded LE 1M PHY. The mnemonics DAVerage, DMINimum, DMAXimum distinguish average, minimum and maximum frequency deviations.

param freq_dev_f_2_lower
No help available

param freq_dev_f_2_upper
No help available

param freq_dev_f_2_enable
Disable or enable limits for current, average, maximum, and minimum results (4 values)

6.4.1.1.8.37 Le1M

SCPI Command :

```
CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIMit:LENergy[:LE1M]:FDRift
```

class Le1MCls

Le1M commands group definition. 6 total commands, 5 Subgroups, 1 group commands

class FdriftStruct

Structure for setting input parameters. Fields:

- Frequency_Drift: float: No parameter help available
- Max_Drift_Rate: float: No parameter help available
- Initl_Freq_Drift: float: No parameter help available
- Freq_Drift_Enable: List[bool]: Disable or enable limit checking for current, average, and maximum results (3 values) .
- Max_Drift_Rate_Enb: List[bool]: Disable or enable limit checking for current, average, and maximum results (3 values) .
- Init_Freq_Drift_En: List[bool]: Disable or enable limit checking for current, average, and maximum results (3 values) .

get_fdrift() → FdriftStruct

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:MEValuation:LIMit:LENergy[:LE1M]:FDRift
value: FdriftStruct = driver.configure.bluetooth.measurement.multiEval.limit.
↪lowEnergy.le1M.get_fdrift()
```

Sets and enables limits for frequency drift, maximum drift rate and initial frequency drift. Commands for uncoded LE 1M PHY (..:LE1M..) , LE 2M PHY (..:LE2M..) , and LE coded PHY (..:LRANge..) are available.

return
structure: for return value, see the help for FdriftStruct structure arguments.

set_fdrift(*value: FdriftStruct*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:MEValuation:LIMit:LENergy[:LE1M]:FDRift
structure = driver.configure.bluetooth.measurement.multiEval.limit.lowEnergy.
↪le1M.FdriftStruct()
structure.Frequency_Drift: float = 1.0
structure.Max_Drift_Rate: float = 1.0
structure.Initl_Freq_Drift: float = 1.0
structure.Freq_Drift_Enable: List[bool] = [True, False, True]
structure.Max_Drift_Rate_Enb: List[bool] = [True, False, True]
```

(continues on next page)

(continued from previous page)

```
structure.Init_Freq_Drift_En: List[bool] = [True, False, True]
driver.configure.bluetooth.measurement.multiEval.limit.lowEnergy.le1M.set_
↪fdrift(value = structure)
```

Sets and enables limits for frequency drift, maximum drift rate and initial frequency drift. Commands for uncoded LE 1M PHY (..:LE1M..) , LE 2M PHY (..:LE2M..) , and LE coded PHY (..:LRANge..) are available.

param value
see the help for FdriftStruct structure arguments.

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.configure.bluetooth.measurement.multiEval.limit.lowEnergy.le1M.clone()
```

Subgroups

6.4.1.1.8.38 Faccuracy

SCPI Command :

```
CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIMit:LENergy[:LE1M]:FACCuracy
```

class FaccuracyCls

Faccuracy commands group definition. 1 total commands, 0 Subgroups, 1 group commands

class FaccuracyStruct

Response structure. Fields:

- Freq_Accuracy: float: No parameter help available
- Freq_Acc_Enabled: List[bool]: Disable or enable limit check for current, average, and maximum results (3 values) .

get() → FaccuracyStruct

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:MEValuation:LIMit:LENergy[:LE1M]:FACCuracy
value: FaccuracyStruct = driver.configure.bluetooth.measurement.multiEval.limit.
↪lowEnergy.le1M.faccuracy.get()
```

Defines the limit for the frequency accuracy. Commands for uncoded LE 1M PHY (..:LE1M..) , LE 2M PHY (..:LE2M..) , and LE coded PHY (..:LRANge..) are available.

return
structure: for return value, see the help for FaccuracyStruct structure arguments.

set(*freq_accuracy: float*, *freq_acc_enabled: List[bool]*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:MEValuation:LIMit:LENergy[:LE1M]:FACCuracy
driver.configure.bluetooth.measurement.multiEval.limit.lowEnergy.le1M.faccuracy.
↪set(freq_accuracy = 1.0, freq_acc_enabled = [True, False, True])
```

Defines the limit for the frequency accuracy. Commands for uncoded LE 1M PHY (..:LE1M..) , LE 2M PHY (..:LE2M..) , and LE coded PHY (..:LRANge..) are available.

param freq_accuracy
No help available

param freq_acc_enabled
Disable or enable limit check for current, average, and maximum results (3 values) .

6.4.1.1.8.39 Foffset

SCPI Command :

```
CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIMit:LENergy[:LE1M]:FOFFset
```

class FoffsetCls

Foffset commands group definition. 1 total commands, 0 Subgroups, 1 group commands

class FoffsetStruct

Response structure. Fields:

- Freq_Offset: float: No parameter help available
- Freq_Offset_Enable: List[bool]: Disable or enable limit checking for current, average, and maximum results (3 values)

get() → FoffsetStruct

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:MEValuation:LIMit:LENergy[:LE1M]:FOFFset
value: FoffsetStruct = driver.configure.bluetooth.measurement.multiEval.limit.
↪lowEnergy.le1M.foffset.get()
```

Sets/gets the frequency offset limit. Commands for uncoded LE 1M PHY (..:LE1M..) , LE 2M PHY (..:LE2M..) , and LE coded PHY (..:LRANge..) are available.

return
structure: for return value, see the help for FoffsetStruct structure arguments.

set(*freq_offset: float*, *freq_offset_enable: List[bool]*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:MEValuation:LIMit:LENergy[:LE1M]:FOFFset
driver.configure.bluetooth.measurement.multiEval.limit.lowEnergy.le1M.foffset.
↪set(freq_offset = 1.0, freq_offset_enable = [True, False, True])
```

Sets/gets the frequency offset limit. Commands for uncoded LE 1M PHY (..:LE1M..) , LE 2M PHY (..:LE2M..) , and LE coded PHY (..:LRANge..) are available.

param freq_offset
No help available

param freq_offset_enable
Disable or enable limit checking for current, average, and maximum results (3 values)

6.4.1.1.8.40 Mratio

SCPI Command :

```
CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIMit:LENergy[:LE1M]:MRATio
```

class MratioCls

Mratio commands group definition. 1 total commands, 0 Subgroups, 1 group commands

class MratioStruct

Response structure. Fields:

- Mod_Ratio: float: No parameter help available
- Mod_Ratio_Enabled: bool: Disable/enable limit checking

get() → MratioStruct

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:MEValuation:LIMit:LENergy[:LE1M]:MRATio
value: MratioStruct = driver.configure.bluetooth.measurement.multiEval.limit.
↪lowEnergy.le1M.mratio.get()
```

Sets or queries the modulation ratio limit f2 avg / f1 avg for LE 1M PHY (...:LE1M...) and LE 2M PHY (...:LE2M...) .

return
structure: for return value, see the help for MratioStruct structure arguments.

set(*mod_ratio: float*, *mod_ratio_enabled: bool*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:MEValuation:LIMit:LENergy[:LE1M]:MRATio
driver.configure.bluetooth.measurement.multiEval.limit.lowEnergy.le1M.mratio.
↪set(mod_ratio = 1.0, mod_ratio_enabled = False)
```

Sets or queries the modulation ratio limit f2 avg / f1 avg for LE 1M PHY (...:LE1M...) and LE 2M PHY (...:LE2M...) .

param mod_ratio
No help available

param mod_ratio_enabled
Disable/enable limit checking

6.4.1.1.8.41 PowerVsTime

SCPI Command :

```
CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIMit:LENergy[:LE1M]:PVTime
```

class PowerVsTimeCls

PowerVsTime commands group definition. 1 total commands, 0 Subgroups, 1 group commands

class PowerVsTimeStruct

Response structure. Fields:

- Avg_Pow_Lower: float: No parameter help available
- Avg_Pow_Upper: float: No parameter help available
- Pkm_Avg_Pow_Upper: float: No parameter help available
- Avg_Pow_Enabled: List[bool]: Disables or enables the limit check for the average power, 4 values, corresponding to the current, average, maximum and minimum results.
- Pkm_Avg_Pow_Enable: List[bool]: Disables or enables the limit check for the 'peak minus average power', 4 values, corresponding to the current, average, maximum and minimum results.

get() → PowerVsTimeStruct

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:MEValuation:LIMit:LENergy[:LE1M]:PVTime
value: PowerVsTimeStruct = driver.configure.bluetooth.measurement.multiEval.
↪limit.lowEnergy.le1M.powerVsTime.get()
```

Defines the power limits: lower and upper average power limits, upper limit for 'peak minus average power', limit check enabling. Commands for uncoded LE 1M PHY (..:LE1M..) , LE 2M PHY (..:LE2M..) , and LE coded PHY (..:LRANge..) are available.

return
structure: for return value, see the help for PowerVsTimeStruct structure arguments.

set(*avg_pow_lower: float*, *avg_pow_upper: float*, *pkm_avg_pow_upper: float*, *avg_pow_enabled: List[bool]*, *pkm_avg_pow_enable: List[bool]*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:MEValuation:LIMit:LENergy[:LE1M]:PVTime
driver.configure.bluetooth.measurement.multiEval.limit.lowEnergy.le1M.
↪powerVsTime.set(avg_pow_lower = 1.0, avg_pow_upper = 1.0, pkm_avg_pow_upper =␣
↪1.0, avg_pow_enabled = [True, False, True], pkm_avg_pow_enable = [True, False,
↪ True])
```

Defines the power limits: lower and upper average power limits, upper limit for 'peak minus average power', limit check enabling. Commands for uncoded LE 1M PHY (..:LE1M..) , LE 2M PHY (..:LE2M..) , and LE coded PHY (..:LRANge..) are available.

param avg_pow_lower
No help available

param avg_pow_upper
No help available

param pkm_avg_pow_upper
No help available

param avg_pow_enabled
Disables or enables the limit check for the average power, 4 values, corresponding to the current, average, maximum and minimum results.

param pkm_avg_pow_enable
Disables or enables the limit check for the 'peak minus average power', 4 values, corresponding to the current, average, maximum and minimum results.

6.4.1.1.8.42 Sacp

SCPI Command :

```
CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIMit:LENergy[:LE1M]:SACP
```

class SacpCls

Sacp commands group definition. 1 total commands, 0 Subgroups, 1 group commands

class SacpStruct

Response structure. Fields:

- Ptx_Limit: float: Power limit for 1 MHz channels fTX± 2 MHz
- Exc_Ptx_Limit: float: Power limit for 1 MHz channels fTX±3 MHz, fTX±4 MHz, …
- No_Of_Ex_Limit: int: Maximum number of tolerable exceptions, i.e. 1 MHz channels fTX±3 MHz, fTX±4 MHz, … whose power is above ExcPTxLimit, but below PTxLimit.
- Ptx_Enable: bool: Disables | enables the PTxLimit limit for 1 MHz channels fTX± 2 MHz.
- No_Of_Exc_Enable: bool: Disables | enables the ExcPTxLimit limit for 1 MHz channels fTX±3 MHz, fTX±4 MHz, … with NoOfExLimit tolerable exceptions (per statistic cycle) .

get() → SacpStruct

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:MEValuation:LIMit:LENergy[:LE1M]:SACP
value: SacpStruct = driver.configure.bluetooth.measurement.multiEval.limit.
↪lowEnergy.le1M.sacp.get()
```

These commands define and enable the Spectrum ACP limits for BR (…:LIMit:SACP) , LE 1M PHY (…:LE1M…) , LE 2M PHY (. ..:LE2M…) , and LE coded PHY (…:LRANge…) , respectively.

return
structure: for return value, see the help for SacpStruct structure arguments.

set(*ptx_limit: float*, *exc_ptx_limit: float*, *no_of_ex_limit: int*, *ptx_enable: bool*, *no_of_exc_enable: bool*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:MEValuation:LIMit:LENergy[:LE1M]:SACP
driver.configure.bluetooth.measurement.multiEval.limit.lowEnergy.le1M.sacp.
↪set(ptx_limit = 1.0, exc_ptx_limit = 1.0, no_of_ex_limit = 1, ptx_enable =␣
↪False, no_of_exc_enable = False)
```

These commands define and enable the Spectrum ACP limits for BR (...:LIMit:SACP) , LE 1M PHY (...:LE1M...) , LE 2M PHY (. ..:LE2M...) , and LE coded PHY (...:LRANge...) , respectively.

param ptx_limit
Power limit for 1 MHz channels fTX± 2 MHz

param exc_ptx_limit
Power limit for 1 MHz channels fTX±3 MHz, fTX±4 MHz, ...

param no_of_ex_limit
Maximum number of tolerable exceptions, i.e. 1 MHz channels fTX±3 MHz, fTX±4 MHz, ... whose power is above ExcPTxLimit, but below PTxLimit.

param ptx_enable
Disables | enables the PTxLimit limit for 1 MHz channels fTX± 2 MHz.

param no_of_exc_enable
Disables | enables the ExcPTxLimit limit for 1 MHz channels fTX±3 MHz, fTX±4 MHz, ... with NoOfExLimit tolerable exceptions (per statistic cycle) .

6.4.1.1.8.43 Le2M

SCPI Command :

```
CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIMit:LENergy:LE2M:FDRift
```

`class Le2MCls`

Le2M commands group definition. 13 total commands, 9 Subgroups, 1 group commands

`class FdriftStruct`

Structure for setting input parameters. Fields:

- Frequency_Drift: float: No parameter help available
- Max_Drift_Rate: float: No parameter help available
- Initl_Freq_Drift: float: No parameter help available
- Freq_Drift_Enable: List[bool]: Disable or enable limit checking for current, average, and maximum results (3 values) .
- Max_Drift_Rate_Enb: List[bool]: Disable or enable limit checking for current, average, and maximum results (3 values) .
- Init_Freq_Drift_En: List[bool]: Disable or enable limit checking for current, average, and maximum results (3 values) .

`get_fdrift()` → FdriftStruct

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:MEValuation:LIMit:LENergy:LE2M:FDRift
value: FdriftStruct = driver.configure.bluetooth.measurement.multiEval.limit.
↪lowEnergy.le2M.get_fdrift()
```

Sets and enables limits for frequency drift, maximum drift rate and initial frequency drift. Commands for uncoded LE 1M PHY (..:LE1M..) , LE 2M PHY (..:LE2M..) , and LE coded PHY (..:LRANge..) are available.

return
structure: for return value, see the help for FdriftStruct structure arguments.

set_fdrift(*value: FdriftStruct*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:MEValuation:LIMit:LENergy:LE2M:FDRift
structure = driver.configure.bluetooth.measurement.multiEval.limit.lowEnergy.
↪le2M.FdriftStruct()
structure.Frequency_Drift: float = 1.0
structure.Max_Drift_Rate: float = 1.0
structure.Initl_Freq_Drift: float = 1.0
structure.Freq_Drift_Enable: List[bool] = [True, False, True]
structure.Max_Drift_Rate_Enb: List[bool] = [True, False, True]
structure.Init_Freq_Drift_En: List[bool] = [True, False, True]
driver.configure.bluetooth.measurement.multiEval.limit.lowEnergy.le2M.set_
↪fdrift(value = structure)
```

Sets and enables limits for frequency drift, maximum drift rate and initial frequency drift. Commands for uncoded LE 1M PHY (..:LE1M..) , LE 2M PHY (..:LE2M..) , and LE coded PHY (..:LRANge..) are available.

param value
see the help for FdriftStruct structure arguments.

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.configure.bluetooth.measurement.multiEval.limit.lowEnergy.le2M.clone()
```

Subgroups

6.4.1.1.8.44 Daverage

SCPI Command :

```
CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIMit:LENergy:LE2M:DAVerage
```

class DaverageCls

Daverage commands group definition. 2 total commands, 1 Subgroups, 1 group commands

class DaverageStruct

Response structure. Fields:

- Freq_Dev_F_1_Lower: float: No parameter help available
- Freq_Dev_F_1_Upper: float: No parameter help available
- Freq_Dev_F_1_Enable: List[bool]: Disable or enable limits for current, average, maximum, and minimum results (4 values)

get() → DaverageStruct

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:MEValuation:LIMit:LENergy:LE2M:DAVerage
value: DaverageStruct = driver.configure.bluetooth.measurement.multiEval.limit.
↪lowEnergy.le2M.daverage.get()
```

Defines the lower and upper f1 frequency deviation limits for LE 2M PHY. The mnemonics DAVerage, DMINimum, DMAXimum distinguish average, minimum and maximum frequency deviations.

return
structure: for return value, see the help for DaverageStruct structure arguments.

set(*freq_dev_f_1_lower: float*, *freq_dev_f_1_upper: float*, *freq_dev_f_1_enable: List[bool]*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:MEValuation:LIMit:LENergy:LE2M:DAVerage
driver.configure.bluetooth.measurement.multiEval.limit.lowEnergy.le2M.daverage.
↪set(freq_dev_f_1_lower = 1.0, freq_dev_f_1_upper = 1.0, freq_dev_f_1_enable =␣
↪[True, False, True])
```

Defines the lower and upper f1 frequency deviation limits for LE 2M PHY. The mnemonics DAVerage, DMINimum, DMAXimum distinguish average, minimum and maximum frequency deviations.

param freq_dev_f_1_lower
No help available

param freq_dev_f_1_upper
No help available

param freq_dev_f_1_enable
Disable or enable limits for current, average, maximum, and minimum results (4 values)

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.configure.bluetooth.measurement.multiEval.limit.lowEnergy.le2M.daverage.
↪clone()
```

Subgroups

6.4.1.1.8.45 Df2S

SCPI Command :

```
CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIMit:LENergy:LE2M:DAVerage:DF2S
```

class Df2SCls

Df2S commands group definition. 1 total commands, 0 Subgroups, 1 group commands

class Df2SStruct

Response structure. Fields:

- Freq_Dev_F_2_Lower: float: No parameter help available

- Freq_Dev_F_2_Upper: float: No parameter help available
- Freq_Dev_F_2_Enable: List[bool]: Disable or enable limits for current, average, maximum, and minimum results (4 values)

get() → Df2SStruct

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:MEValuation:LIMit:LENergy:LE2M:DAVerage:DF2S
value: Df2SStruct = driver.configure.bluetooth.measurement.multiEval.limit.
↪lowEnergy.le2M.daverage.df2S.get()
```

Defines the lower and upper f2 frequency deviation limits for LE 2M PHY. The mnemonics DAVerage, DMINimum, DMAXimum distinguish average, minimum and maximum frequency deviations.

return
structure: for return value, see the help for Df2SStruct structure arguments.

set(*freq_dev_f_2_lower: float*, *freq_dev_f_2_upper: float*, *freq_dev_f_2_enable: List[bool]*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:MEValuation:LIMit:LENergy:LE2M:DAVerage:DF2S
driver.configure.bluetooth.measurement.multiEval.limit.lowEnergy.le2M.daverage.
↪df2S.set(freq_dev_f_2_lower = 1.0, freq_dev_f_2_upper = 1.0, freq_dev_f_2_
↪enable = [True, False, True])
```

Defines the lower and upper f2 frequency deviation limits for LE 2M PHY. The mnemonics DAVerage, DMINimum, DMAXimum distinguish average, minimum and maximum frequency deviations.

param freq_dev_f_2_lower
No help available

param freq_dev_f_2_upper
No help available

param freq_dev_f_2_enable
Disable or enable limits for current, average, maximum, and minimum results (4 values)

6.4.1.1.8.46 Delta

SCPI Command :

```
CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIMit:LENergy:LE2M:DELTa
```

class DeltaCls

Delta commands group definition. 1 total commands, 0 Subgroups, 1 group commands

class DeltaStruct

Response structure. Fields:

- Delta_F_2_P_99_P_9: float: No parameter help available
- Delta_F_2_P_99_Enable: bool: Disable/enable limit checking

get() → DeltaStruct

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:MEValuation:LIMit:LENergy:LE2M:DELTa
value: DeltaStruct = driver.configure.bluetooth.measurement.multiEval.limit.
↪lowEnergy.le2M.delta.get()
```

Sets/gets the limit for the frequency deviation f2 for LE 2M PHY that must be exceeded by 99.9% of the measured samples.

return
structure: for return value, see the help for DeltaStruct structure arguments.

set(*delta_f_2_p_99_p_9: float*, *delta_f_2_p_99_enable: bool*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:MEValuation:LIMit:LENergy:LE2M:DELTa
driver.configure.bluetooth.measurement.multiEval.limit.lowEnergy.le2M.delta.
↪set(delta_f_2_p_99_p_9 = 1.0, delta_f_2_p_99_enable = False)
```

Sets/gets the limit for the frequency deviation f2 for LE 2M PHY that must be exceeded by 99.9% of the measured samples.

param delta_f_2_p_99_p_9
No help available

param delta_f_2_p_99_enable
Disable/enable limit checking

6.4.1.1.8.47 Dmaximum

SCPI Command :

```
CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIMit:LENergy:LE2M:DMAXimum
```

class DmaximumCls

Dmaximum commands group definition. 2 total commands, 1 Subgroups, 1 group commands

class DmaximumStruct

Response structure. Fields:

- Freq_Dev_F_1_Lower: float: No parameter help available
- Freq_Dev_F_1_Upper: float: No parameter help available
- Freq_Dev_F_1_Enable: List[bool]: Disable or enable limits for current, average, maximum, and minimum results (4 values)

get() → DmaximumStruct

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:MEValuation:LIMit:LENergy:LE2M:DMAXimum
value: DmaximumStruct = driver.configure.bluetooth.measurement.multiEval.limit.
↪lowEnergy.le2M.dmaximum.get()
```

Defines the lower and upper f1 frequency deviation limits for LE 2M PHY. The mnemonics DAVerage, DMINimum, DMAXimum distinguish average, minimum and maximum frequency deviations.

return
structure: for return value, see the help for DmaximumStruct structure arguments.

set(*freq_dev_f_1_lower: float*, *freq_dev_f_1_upper: float*, *freq_dev_f_1_enable: List[bool]*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:MEValuation:LIMit:LENergy:LE2M:DMAXimum
driver.configure.bluetooth.measurement.multiEval.limit.lowEnergy.le2M.dmaximum.
↪set(freq_dev_f_1_lower = 1.0, freq_dev_f_1_upper = 1.0, freq_dev_f_1_enable =␣
↪[True, False, True])
```

Defines the lower and upper f1 frequency deviation limits for LE 2M PHY. The mnemonics DAVerage, DMINimum, DMAXimum distinguish average, minimum and maximum frequency deviations.

param freq_dev_f_1_lower
No help available

param freq_dev_f_1_upper
No help available

param freq_dev_f_1_enable
Disable or enable limits for current, average, maximum, and minimum results (4 values)

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.configure.bluetooth.measurement.multiEval.limit.lowEnergy.le2M.dmaximum.
↪clone()
```

Subgroups

6.4.1.1.8.48 Df2S

SCPI Command :

```
CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIMit:LENergy:LE2M:DMAXimum:DF2S
```

class Df2SCls

Df2S commands group definition. 1 total commands, 0 Subgroups, 1 group commands

class Df2SStruct

Response structure. Fields:

- Freq_Dev_F_2_Lower: float: No parameter help available
- Freq_Dev_F_2_Upper: float: No parameter help available
- Freq_Dev_F_2_Enable: List[bool]: Disable or enable limits for current, average, maximum, and minimum results (4 values)

get() → Df2SStruct

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:MEValuation:LIMit:LENergy:LE2M:DMAXimum:DF2S
value: Df2SStruct = driver.configure.bluetooth.measurement.multiEval.limit.
↪lowEnergy.le2M.dmaximum.df2S.get()
```

Defines the lower and upper f2 frequency deviation limits for LE 2M PHY. The mnemonics DAVerage, DMINimum, DMAXimum distinguish average, minimum and maximum frequency deviations.

return
structure: for return value, see the help for Df2SStruct structure arguments.

set(*freq_dev_f_2_lower: float*, *freq_dev_f_2_upper: float*, *freq_dev_f_2_enable: List[bool]*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:MEValuation:LIMit:LENergy:LE2M:DMAXimum:DF2S
driver.configure.bluetooth.measurement.multiEval.limit.lowEnergy.le2M.dmaximum.
↪df2S.set(freq_dev_f_2_lower = 1.0, freq_dev_f_2_upper = 1.0, freq_dev_f_2_
↪enable = [True, False, True])
```

Defines the lower and upper f2 frequency deviation limits for LE 2M PHY. The mnemonics DAVerage, DMINimum, DMAXimum distinguish average, minimum and maximum frequency deviations.

param freq_dev_f_2_lower
No help available

param freq_dev_f_2_upper
No help available

param freq_dev_f_2_enable
Disable or enable limits for current, average, maximum, and minimum results (4 values)

6.4.1.1.8.49 Dminimum

SCPI Command :

```
CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIMit:LENergy:LE2M:DMINimum
```

class DminimumCls

Dminimum commands group definition. 2 total commands, 1 Subgroups, 1 group commands

class DminimumStruct

Response structure. Fields:

- Freq_Dev_F_1_Lower: float: No parameter help available
- Freq_Dev_F_1_Upper: float: No parameter help available
- Freq_Dev_F_1_Enable: List[bool]: Disable or enable limits for current, average, maximum, and minimum results (4 values)

get() → DminimumStruct

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:MEValuation:LIMit:LENergy:LE2M:DMINimum
```

(continues on next page)

(continued from previous page)

```
value: DminimumStruct = driver.configure.bluetooth.measurement.multiEval.limit.
↪lowEnergy.le2M.dminimum.get()
```

Defines the lower and upper f1 frequency deviation limits for LE 2M PHY. The mnemonics DAVerage, DMINimum, DMAXimum distinguish average, minimum and maximum frequency deviations.

return
structure: for return value, see the help for DminimumStruct structure arguments.

set(*freq_dev_f_1_lower: float*, *freq_dev_f_1_upper: float*, *freq_dev_f_1_enable: List[bool]*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:MEValuation:LIMit:LENergy:LE2M:DMINimum
driver.configure.bluetooth.measurement.multiEval.limit.lowEnergy.le2M.dminimum.
↪set(freq_dev_f_1_lower = 1.0, freq_dev_f_1_upper = 1.0, freq_dev_f_1_enable =␣
↪[True, False, True])
```

Defines the lower and upper f1 frequency deviation limits for LE 2M PHY. The mnemonics DAVerage, DMINimum, DMAXimum distinguish average, minimum and maximum frequency deviations.

param freq_dev_f_1_lower
No help available

param freq_dev_f_1_upper
No help available

param freq_dev_f_1_enable
Disable or enable limits for current, average, maximum, and minimum results (4 values)

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.configure.bluetooth.measurement.multiEval.limit.lowEnergy.le2M.dminimum.
↪clone()
```

Subgroups

6.4.1.1.8.50 Df2S

SCPI Command :

```
CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIMit:LENergy:LE2M:DMINimum:DF2S
```

class Df2SCls

Df2S commands group definition. 1 total commands, 0 Subgroups, 1 group commands

class Df2SStruct

Response structure. Fields:

- Freq_Dev_F_2_Lower: float: No parameter help available
- Freq_Dev_F_2_Upper: float: No parameter help available

- Freq_Dev_F_2_Enable: List[bool]: Disable or enable limits for current, average, maximum, and minimum results (4 values)

get() → Df2SStruct

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:MEValuation:LIMit:LENergy:LE2M:DMINimum:DF2S
value: Df2SStruct = driver.configure.bluetooth.measurement.multiEval.limit.
↪lowEnergy.le2M.dminimum.df2S.get()
```

Defines the lower and upper f2 frequency deviation limits for LE 2M PHY. The mnemonics DAVerage, DMINimum, DMAXimum distinguish average, minimum and maximum frequency deviations.

return
structure: for return value, see the help for Df2SStruct structure arguments.

set(*freq_dev_f_2_lower: float*, *freq_dev_f_2_upper: float*, *freq_dev_f_2_enable: List[bool]*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:MEValuation:LIMit:LENergy:LE2M:DMINimum:DF2S
driver.configure.bluetooth.measurement.multiEval.limit.lowEnergy.le2M.dminimum.
↪df2S.set(freq_dev_f_2_lower = 1.0, freq_dev_f_2_upper = 1.0, freq_dev_f_2_
↪enable = [True, False, True])
```

Defines the lower and upper f2 frequency deviation limits for LE 2M PHY. The mnemonics DAVerage, DMINimum, DMAXimum distinguish average, minimum and maximum frequency deviations.

param freq_dev_f_2_lower
No help available

param freq_dev_f_2_upper
No help available

param freq_dev_f_2_enable
Disable or enable limits for current, average, maximum, and minimum results (4 values)

6.4.1.1.8.51 Faccuracy

SCPI Command :

```
CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIMit:LENergy:LE2M:FACCuracy
```

class FaccuracyCls

Faccuracy commands group definition. 1 total commands, 0 Subgroups, 1 group commands

class FaccuracyStruct

Response structure. Fields:

- Freq_Accuracy: float: No parameter help available
- Freq_Acc_Enabled: List[bool]: Disable or enable limit check for current, average, and maximum results (3 values) .

get() → FaccuracyStruct

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:MEValuation:LIMit:LENergy:LE2M:FACCuracy
value: FaccuracyStruct = driver.configure.bluetooth.measurement.multiEval.limit.
↪lowEnergy.le2M.faccuracy.get()
```

Defines the limit for the frequency accuracy. Commands for uncoded LE 1M PHY (..:LE1M..) , LE 2M PHY (..:LE2M..) , and LE coded PHY (..:LRANge..) are available.

return
structure: for return value, see the help for FaccuracyStruct structure arguments.

set(*freq_accuracy: float*, *freq_acc_enabled: List[bool]*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:MEValuation:LIMit:LENergy:LE2M:FACCuracy
driver.configure.bluetooth.measurement.multiEval.limit.lowEnergy.le2M.faccuracy.
↪set(freq_accuracy = 1.0, freq_acc_enabled = [True, False, True])
```

Defines the limit for the frequency accuracy. Commands for uncoded LE 1M PHY (..:LE1M..) , LE 2M PHY (..:LE2M..) , and LE coded PHY (..:LRANge..) are available.

param freq_accuracy
No help available

param freq_acc_enabled
Disable or enable limit check for current, average, and maximum results (3 values) .

6.4.1.1.8.52 Foffset

SCPI Command :

```
CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIMit:LENergy:LE2M:FOFFset
```

class FoffsetCls

Foffset commands group definition. 1 total commands, 0 Subgroups, 1 group commands

class FoffsetStruct

Response structure. Fields:

- Freq_Offset: float: No parameter help available
- Freq_Offset_Enable: List[bool]: Disable or enable limit checking for current, average, and maximum results (3 values)

get() → FoffsetStruct

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:MEValuation:LIMit:LENergy:LE2M:FOFFset
value: FoffsetStruct = driver.configure.bluetooth.measurement.multiEval.limit.
↪lowEnergy.le2M.foffset.get()
```

Sets/gets the frequency offset limit. Commands for uncoded LE 1M PHY (..:LE1M..) , LE 2M PHY (..:LE2M..) , and LE coded PHY (..:LRANge..) are available.

return
structure: for return value, see the help for FoffsetStruct structure arguments.

set(*freq_offset: float*, *freq_offset_enable: List[bool]*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:MEValuation:LIMit:LENergy:LE2M:FOFFset
driver.configure.bluetooth.measurement.multiEval.limit.lowEnergy.le2M.foffset.
↪set(freq_offset = 1.0, freq_offset_enable = [True, False, True])
```

Sets/gets the frequency offset limit. Commands for uncoded LE 1M PHY (..:LE1M..) , LE 2M PHY (..:LE2M..) , and LE coded PHY (..:LRANge..) are available.

param freq_offset
No help available

param freq_offset_enable
Disable or enable limit checking for current, average, and maximum results (3 values)

6.4.1.1.8.53 Mratio

SCPI Command :

```
CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIMit:LENergy:LE2M:MRATio
```

class MratioCls

Mratio commands group definition. 1 total commands, 0 Subgroups, 1 group commands

class MratioStruct

Response structure. Fields:

- Mod_Ratio: float: No parameter help available
- Mod_Ratio_Enabled: bool: Disable/enable limit checking

get() → MratioStruct

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:MEValuation:LIMit:LENergy:LE2M:MRATio
value: MratioStruct = driver.configure.bluetooth.measurement.multiEval.limit.
↪lowEnergy.le2M.mratio.get()
```

Sets or queries the modulation ratio limit f2 avg / f1 avg for LE 1M PHY (...:LE1M...) and LE 2M PHY (...:LE2M...) .

return
structure: for return value, see the help for MratioStruct structure arguments.

set(*mod_ratio: float*, *mod_ratio_enabled: bool*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:MEValuation:LIMit:LENergy:LE2M:MRATio
driver.configure.bluetooth.measurement.multiEval.limit.lowEnergy.le2M.mratio.
↪set(mod_ratio = 1.0, mod_ratio_enabled = False)
```

Sets or queries the modulation ratio limit f2 avg / f1 avg for LE 1M PHY (...:LE1M...) and LE 2M PHY (...:LE2M...) .

param mod_ratio
No help available

param mod_ratio_enabled
Disable/enable limit checking

6.4.1.1.8.54 PowerVsTime

SCPI Command :

```
CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIMit:LENergy:LE2M:PVTime
```

class PowerVsTimeCls

PowerVsTime commands group definition. 1 total commands, 0 Subgroups, 1 group commands

class PowerVsTimeStruct

Response structure. Fields:

- Avg_Pow_Lower: float: No parameter help available
- Avg_Pow_Upper: float: No parameter help available
- Pkm_Avg_Pow_Upper: float: No parameter help available
- Avg_Pow_Enabled: List[bool]: Disables or enables the limit check for the average power, 4 values, corresponding to the current, average, maximum and minimum results.
- Pkm_Avg_Pow_Enable: List[bool]: Disables or enables the limit check for the 'peak minus average power', 4 values, corresponding to the current, average, maximum and minimum results.

get() → PowerVsTimeStruct

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:MEValuation:LIMit:LENergy:LE2M:PVTime
value: PowerVsTimeStruct = driver.configure.bluetooth.measurement.multiEval.
↪limit.lowEnergy.le2M.powerVsTime.get()
```

Defines the power limits: lower and upper average power limits, upper limit for 'peak minus average power', limit check enabling. Commands for uncoded LE 1M PHY (..:LE1M..) , LE 2M PHY (..:LE2M..) , and LE coded PHY (..:LRANge..) are available.

return
structure: for return value, see the help for PowerVsTimeStruct structure arguments.

set(*avg_pow_lower: float*, *avg_pow_upper: float*, *pkm_avg_pow_upper: float*, *avg_pow_enabled: List[bool]*, *pkm_avg_pow_enable: List[bool]*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:MEValuation:LIMit:LENergy:LE2M:PVTime
driver.configure.bluetooth.measurement.multiEval.limit.lowEnergy.le2M.
↪powerVsTime.set(avg_pow_lower = 1.0, avg_pow_upper = 1.0, pkm_avg_pow_upper =␣
↪1.0, avg_pow_enabled = [True, False, True], pkm_avg_pow_enable = [True, False,
↪ True])
```

Defines the power limits: lower and upper average power limits, upper limit for 'peak minus average power', limit check enabling. Commands for uncoded LE 1M PHY (..:LE1M..) , LE 2M PHY (..:LE2M..) , and LE coded PHY (..:LRANge..) are available.

param avg_pow_lower
No help available

param avg_pow_upper
No help available

param pkm_avg_pow_upper
No help available

param avg_pow_enabled
Disables or enables the limit check for the average power, 4 values, corresponding to the current, average, maximum and minimum results.

param pkm_avg_pow_enable
Disables or enables the limit check for the 'peak minus average power', 4 values, corresponding to the current, average, maximum and minimum results.

6.4.1.1.8.55 Sacp

SCPI Command :

```
CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIMit:LENergy:LE2M:SACP
```

class SacpCls

Sacp commands group definition. 1 total commands, 0 Subgroups, 1 group commands

class SacpStruct

Response structure. Fields:

- Ptx_Limit: float: Power limit for 1 MHz channels fTX± 2 MHz
- Exc_Ptx_Limit: float: Power limit for 1 MHz channels fTX±3 MHz, fTX±4 MHz, …
- No_Of_Ex_Limit: int: Maximum number of tolerable exceptions, i.e. 1 MHz channels fTX±3 MHz, fTX±4 MHz, … whose power is above ExcPTxLimit, but below PTxLimit.
- Ptx_Enable: bool: Disables | enables the PTxLimit limit for 1 MHz channels fTX± 2 MHz.
- No_Of_Exc_Enable: bool: Disables | enables the ExcPTxLimit limit for 1 MHz channels fTX±3 MHz, fTX±4 MHz, … with NoOfExLimit tolerable exceptions (per statistic cycle) .

get() → SacpStruct

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:MEValuation:LIMit:LENergy:LE2M:SACP
value: SacpStruct = driver.configure.bluetooth.measurement.multiEval.limit.
↪lowEnergy.le2M.sacp.get()
```

These commands define and enable the Spectrum ACP limits for BR (…:LIMit:SACP) , LE 1M PHY (…:LE1M…) , LE 2M PHY (. ..:LE2M…) , and LE coded PHY (…:LRANge…) , respectively.

return
structure: for return value, see the help for SacpStruct structure arguments.

set(*ptx_limit: float*, *exc_ptx_limit: float*, *no_of_ex_limit: int*, *ptx_enable: bool*, *no_of_exc_enable: bool*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:MEValuation:LIMit:LENergy:LE2M:SACP
driver.configure.bluetooth.measurement.multiEval.limit.lowEnergy.le2M.sacp.
```

(continues on next page)

(continued from previous page)

```
↪set(ptx_limit = 1.0, exc_ptx_limit = 1.0, no_of_ex_limit = 1, ptx_enable =␣
↪False, no_of_exc_enable = False)
```

These commands define and enable the Spectrum ACP limits for BR (...:LIMit:SACP) , LE 1M PHY (...:LE1M...) , LE 2M PHY (. ..:LE2M...) , and LE coded PHY (...:LRANge...) , respectively.

param ptx_limit
Power limit for 1 MHz channels fTX± 2 MHz

param exc_ptx_limit
Power limit for 1 MHz channels fTX±3 MHz, fTX±4 MHz, ...

param no_of_ex_limit
Maximum number of tolerable exceptions, i.e. 1 MHz channels fTX±3 MHz, fTX±4 MHz, ... whose power is above ExcPTxLimit, but below PTxLimit.

param ptx_enable
Disables | enables the PTxLimit limit for 1 MHz channels fTX± 2 MHz.

param no_of_exc_enable
Disables | enables the ExcPTxLimit limit for 1 MHz channels fTX±3 MHz, fTX±4 MHz, ... with NoOfExLimit tolerable exceptions (per statistic cycle) .

6.4.1.1.8.56 Lrange

SCPI Command :

```
CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIMit:LENergy:LRANge:FDRift
```

class LrangeCls

Lrange commands group definition. 9 total commands, 8 Subgroups, 1 group commands

class FdriftStruct

Structure for setting input parameters. Fields:

- Frequency_Drift: float: No parameter help available
- Max_Drift_Rate: float: No parameter help available
- Initl_Freq_Drift: float: No parameter help available
- Freq_Drift_Enable: List[bool]: Disable or enable limit checking for current, average, and maximum results (3 values) .
- Max_Drift_Rate_Enb: List[bool]: Disable or enable limit checking for current, average, and maximum results (3 values) .
- Init_Freq_Drift_En: List[bool]: Disable or enable limit checking for current, average, and maximum results (3 values) .

get_fdrift() → FdriftStruct

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:MEValuation:LIMit:LENergy:LRANge:FDRift
value: FdriftStruct = driver.configure.bluetooth.measurement.multiEval.limit.
↪lowEnergy.lrange.get_fdrift()
```

Sets and enables limits for frequency drift, maximum drift rate and initial frequency drift. Commands for uncoded LE 1M PHY (..:LE1M..) , LE 2M PHY (..:LE2M..) , and LE coded PHY (..:LRANge..) are available.

return
structure: for return value, see the help for FdriftStruct structure arguments.

set_fdrift(*value: FdriftStruct*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:MEValuation:LIMit:LENergy:LRANge:FDRift
structure = driver.configure.bluetooth.measurement.multiEval.limit.lowEnergy.
↪lrange.FdriftStruct()
structure.Frequency_Drift: float = 1.0
structure.Max_Drift_Rate: float = 1.0
structure.Initl_Freq_Drift: float = 1.0
structure.Freq_Drift_Enable: List[bool] = [True, False, True]
structure.Max_Drift_Rate_Enb: List[bool] = [True, False, True]
structure.Init_Freq_Drift_En: List[bool] = [True, False, True]
driver.configure.bluetooth.measurement.multiEval.limit.lowEnergy.lrange.set_
↪fdrift(value = structure)
```

Sets and enables limits for frequency drift, maximum drift rate and initial frequency drift. Commands for uncoded LE 1M PHY (..:LE1M..) , LE 2M PHY (..:LE2M..) , and LE coded PHY (..:LRANge..) are available.

param value
see the help for FdriftStruct structure arguments.

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.configure.bluetooth.measurement.multiEval.limit.lowEnergy.lrange.clone()
```

Subgroups

6.4.1.1.8.57 Daverage

SCPI Command :

```
CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIMit:LENergy:LRANge:DAVerage
```

class DaverageCls

Daverage commands group definition. 1 total commands, 0 Subgroups, 1 group commands

class DaverageStruct

Response structure. Fields:

- Freq_Dev_F_1_Lower: float: No parameter help available
- Freq_Dev_F_1_Upper: float: No parameter help available
- Freq_Dev_F_1_Enable: List[bool]: Disable or enable limits for current, average, maximum, and minimum results (4 values)

get() → DaverageStruct

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:MEValuation:LIMit:LENergy:LRANge:DAVerage
value: DaverageStruct = driver.configure.bluetooth.measurement.multiEval.limit.
↪lowEnergy.lrange.daverage.get()
```

Defines the lower and upper f1 frequency deviation limits for LE coded PHY. The mnemonics DAVerage, DMINimum, DMAXimum distinguish average, minimum and maximum frequency deviations.

return
structure: for return value, see the help for DaverageStruct structure arguments.

set(*freq_dev_f_1_lower: float*, *freq_dev_f_1_upper: float*, *freq_dev_f_1_enable: List[bool]*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:MEValuation:LIMit:LENergy:LRANge:DAVerage
driver.configure.bluetooth.measurement.multiEval.limit.lowEnergy.lrange.
↪daverage.set(freq_dev_f_1_lower = 1.0, freq_dev_f_1_upper = 1.0, freq_dev_f_1_
↪enable = [True, False, True])
```

Defines the lower and upper f1 frequency deviation limits for LE coded PHY. The mnemonics DAVerage, DMINimum, DMAXimum distinguish average, minimum and maximum frequency deviations.

param freq_dev_f_1_lower
No help available

param freq_dev_f_1_upper
No help available

param freq_dev_f_1_enable
Disable or enable limits for current, average, maximum, and minimum results (4 values)

6.4.1.1.8.58 Delta

SCPI Command :

```
CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIMit:LENergy:LRANge:DELTa
```

class DeltaCls

Delta commands group definition. 1 total commands, 0 Subgroups, 1 group commands

class DeltaStruct

Response structure. Fields:

- Delta_F_1_P_99_P_9: float: No parameter help available
- Delta_F_1_P_99_Enable: bool: Disable/enable limit checking

get() → DeltaStruct

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:MEValuation:LIMit:LENergy:LRANge:DELTa
value: DeltaStruct = driver.configure.bluetooth.measurement.multiEval.limit.
↪lowEnergy.lrange.delta.get()
```

Sets/gets the limit for the frequency deviation f1 that must be exceeded by 99.9% of the measured samples for LE coded PHY.

return
structure: for return value, see the help for DeltaStruct structure arguments.

set(*delta_f_1_p_99_p_9: float*, *delta_f_1_p_99_enable: bool*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:MEValuation:LIMit:LENergy:LRANge:DELTa
driver.configure.bluetooth.measurement.multiEval.limit.lowEnergy.lrange.delta.
↪set(delta_f_1_p_99_p_9 = 1.0, delta_f_1_p_99_enable = False)
```

Sets/gets the limit for the frequency deviation f1 that must be exceeded by 99.9% of the measured samples for LE coded PHY.

param delta_f_1_p_99_p_9
No help available

param delta_f_1_p_99_enable
Disable/enable limit checking

6.4.1.1.8.59 Dmaximum

SCPI Command :

```
CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIMit:LENergy:LRANge:DMAXimum
```

class DmaximumCls

Dmaximum commands group definition. 1 total commands, 0 Subgroups, 1 group commands

class DmaximumStruct

Response structure. Fields:

- Freq_Dev_F_1_Lower: float: No parameter help available
- Freq_Dev_F_1_Upper: float: No parameter help available
- Freq_Dev_F_1_Enable: List[bool]: Disable or enable limits for current, average, maximum, and minimum results (4 values)

get() → DmaximumStruct

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:MEValuation:LIMit:LENergy:LRANge:DMAXimum
value: DmaximumStruct = driver.configure.bluetooth.measurement.multiEval.limit.
↪lowEnergy.lrange.dmaximum.get()
```

Defines the lower and upper f1 frequency deviation limits for LE coded PHY. The mnemonics DAVerage, DMINimum, DMAXimum distinguish average, minimum and maximum frequency deviations.

return
structure: for return value, see the help for DmaximumStruct structure arguments.

set(*freq_dev_f_1_lower: float*, *freq_dev_f_1_upper: float*, *freq_dev_f_1_enable: List[bool]*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:MEValuation:LIMit:LENergy:LRANge:DMAXimum
driver.configure.bluetooth.measurement.multiEval.limit.lowEnergy.lrange.
↪dmaximum.set(freq_dev_f_1_lower = 1.0, freq_dev_f_1_upper = 1.0, freq_dev_f_1_
↪enable = [True, False, True])
```

Defines the lower and upper f1 frequency deviation limits for LE coded PHY. The mnemonics DAVerage, DMINimum, DMAXimum distinguish average, minimum and maximum frequency deviations.

param freq_dev_f_1_lower
No help available

param freq_dev_f_1_upper
No help available

param freq_dev_f_1_enable
Disable or enable limits for current, average, maximum, and minimum results (4 values)

6.4.1.1.8.60 Dminimum

SCPI Command :

```
CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIMit:LENergy:LRANge:DMINimum
```

class DminimumCls

Dminimum commands group definition. 1 total commands, 0 Subgroups, 1 group commands

class DminimumStruct

Response structure. Fields:

- Freq_Dev_F_1_Lower: float: No parameter help available
- Freq_Dev_F_1_Upper: float: No parameter help available
- Freq_Dev_F_1_Enable: List[bool]: Disable or enable limits for current, average, maximum, and minimum results (4 values)

get() → DminimumStruct

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:MEValuation:LIMit:LENergy:LRANge:DMINimum
value: DminimumStruct = driver.configure.bluetooth.measurement.multiEval.limit.
↪lowEnergy.lrange.dminimum.get()
```

Defines the lower and upper f1 frequency deviation limits for LE coded PHY. The mnemonics DAVerage, DMINimum, DMAXimum distinguish average, minimum and maximum frequency deviations.

return
structure: for return value, see the help for DminimumStruct structure arguments.

set(*freq_dev_f_1_lower: float*, *freq_dev_f_1_upper: float*, *freq_dev_f_1_enable: List[bool]*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:MEValuation:LIMit:LENergy:LRANge:DMINimum
driver.configure.bluetooth.measurement.multiEval.limit.lowEnergy.lrange.
```

(continues on next page)

(continued from previous page)

```
↪dminimum.set(freq_dev_f_1_lower = 1.0, freq_dev_f_1_upper = 1.0, freq_dev_f_1_
↪enable = [True, False, True])
```

Defines the lower and upper f1 frequency deviation limits for LE coded PHY. The mnemonics DAVerage, DMINimum, DMAXimum distinguish average, minimum and maximum frequency deviations.

param freq_dev_f_1_lower
No help available

param freq_dev_f_1_upper
No help available

param freq_dev_f_1_enable
Disable or enable limits for current, average, maximum, and minimum results (4 values)

6.4.1.1.8.61 Faccuracy

SCPI Command :

```
CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIMit:LENergy:LRANge:FACCuracy
```

class FaccuracyCls

Faccuracy commands group definition. 1 total commands, 0 Subgroups, 1 group commands

class FaccuracyStruct

Response structure. Fields:

- Freq_Accuracy: float: No parameter help available
- Freq_Acc_Enabled: List[bool]: Disable or enable limit check for current, average, and maximum results (3 values) .

get() → FaccuracyStruct

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:MEValuation:LIMit:LENergy:LRANge:FACCuracy
value: FaccuracyStruct = driver.configure.bluetooth.measurement.multiEval.limit.
↪lowEnergy.lrange.faccuracy.get()
```

Defines the limit for the frequency accuracy. Commands for uncoded LE 1M PHY (..:LE1M..) , LE 2M PHY (..:LE2M..) , and LE coded PHY (..:LRANge..) are available.

return
structure: for return value, see the help for FaccuracyStruct structure arguments.

set(*freq_accuracy: float*, *freq_acc_enabled: List[bool]*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:MEValuation:LIMit:LENergy:LRANge:FACCuracy
driver.configure.bluetooth.measurement.multiEval.limit.lowEnergy.lrange.
↪faccuracy.set(freq_accuracy = 1.0, freq_acc_enabled = [True, False, True])
```

Defines the limit for the frequency accuracy. Commands for uncoded LE 1M PHY (..:LE1M..) , LE 2M PHY (..:LE2M..) , and LE coded PHY (..:LRANge..) are available.

param freq_accuracy
No help available

param freq_acc_enabled
Disable or enable limit check for current, average, and maximum results (3 values) .

6.4.1.1.8.62 Foffset

SCPI Command :

```
CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIMit:LENergy:LRANge:FOFFset
```

class FoffsetCls

Foffset commands group definition. 1 total commands, 0 Subgroups, 1 group commands

class FoffsetStruct

Response structure. Fields:

- Freq_Offset: float: No parameter help available
- Freq_Offset_Enable: List[bool]: Disable or enable limit checking for current, average, and maximum results (3 values)

get() → FoffsetStruct

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:MEValuation:LIMit:LENergy:LRANge:FOFFset
value: FoffsetStruct = driver.configure.bluetooth.measurement.multiEval.limit.
↪lowEnergy.lrange.foffset.get()
```

Sets/gets the frequency offset limit. Commands for uncoded LE 1M PHY (..:LE1M..) , LE 2M PHY (..:LE2M..) , and LE coded PHY (..:LRANge..) are available.

return
structure: for return value, see the help for FoffsetStruct structure arguments.

set(*freq_offset: float*, *freq_offset_enable: List[bool]*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:MEValuation:LIMit:LENergy:LRANge:FOFFset
driver.configure.bluetooth.measurement.multiEval.limit.lowEnergy.lrange.foffset.
↪set(freq_offset = 1.0, freq_offset_enable = [True, False, True])
```

Sets/gets the frequency offset limit. Commands for uncoded LE 1M PHY (..:LE1M..) , LE 2M PHY (..:LE2M..) , and LE coded PHY (..:LRANge..) are available.

param freq_offset
No help available

param freq_offset_enable
Disable or enable limit checking for current, average, and maximum results (3 values)

6.4.1.1.8.63 PowerVsTime

SCPI Command :

```
CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIMit:LENergy:LRANge:PVTime
```

class PowerVsTimeCls

PowerVsTime commands group definition. 1 total commands, 0 Subgroups, 1 group commands

class PowerVsTimeStruct

Response structure. Fields:

- Avg_Pow_Lower: float: No parameter help available
- Avg_Pow_Upper: float: No parameter help available
- Pkm_Avg_Pow_Upper: float: No parameter help available
- Avg_Pow_Enabled: List[bool]: Disables or enables the limit check for the average power, 4 values, corresponding to the current, average, maximum and minimum results.
- Pkm_Avg_Pow_Enable: List[bool]: Disables or enables the limit check for the 'peak minus average power', 4 values, corresponding to the current, average, maximum and minimum results.

get() → PowerVsTimeStruct

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:MEValuation:LIMit:LENergy:LRANge:PVTime
value: PowerVsTimeStruct = driver.configure.bluetooth.measurement.multiEval.
↪limit.lowEnergy.lrange.powerVsTime.get()
```

Defines the power limits: lower and upper average power limits, upper limit for 'peak minus average power', limit check enabling. Commands for uncoded LE 1M PHY (..:LE1M..) , LE 2M PHY (..:LE2M..) , and LE coded PHY (..:LRANge..) are available.

return
structure: for return value, see the help for PowerVsTimeStruct structure arguments.

set(*avg_pow_lower: float*, *avg_pow_upper: float*, *pkm_avg_pow_upper: float*, *avg_pow_enabled: List[bool]*, *pkm_avg_pow_enable: List[bool]*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:MEValuation:LIMit:LENergy:LRANge:PVTime
driver.configure.bluetooth.measurement.multiEval.limit.lowEnergy.lrange.
↪powerVsTime.set(avg_pow_lower = 1.0, avg_pow_upper = 1.0, pkm_avg_pow_upper =␣
↪1.0, avg_pow_enabled = [True, False, True], pkm_avg_pow_enable = [True, False,
↪ True])
```

Defines the power limits: lower and upper average power limits, upper limit for 'peak minus average power', limit check enabling. Commands for uncoded LE 1M PHY (..:LE1M..) , LE 2M PHY (..:LE2M..) , and LE coded PHY (..:LRANge..) are available.

param avg_pow_lower
No help available

param avg_pow_upper
No help available

param pkm_avg_pow_upper
No help available

param avg_pow_enabled
Disables or enables the limit check for the average power, 4 values, corresponding to the current, average, maximum and minimum results.

param pkm_avg_pow_enable
Disables or enables the limit check for the 'peak minus average power', 4 values, corresponding to the current, average, maximum and minimum results.

6.4.1.1.8.64 Sacp

SCPI Command :

```
CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIMit:LENergy:LRANge:SACP
```

class SacpCls

Sacp commands group definition. 1 total commands, 0 Subgroups, 1 group commands

class SacpStruct

Response structure. Fields:

- Ptx_Limit: float: Power limit for 1 MHz channels fTX± 2 MHz
- Exc_Ptx_Limit: float: Power limit for 1 MHz channels fTX±3 MHz, fTX±4 MHz, …
- No_Of_Ex_Limit: int: Maximum number of tolerable exceptions, i.e. 1 MHz channels fTX±3 MHz, fTX±4 MHz, … whose power is above ExcPTxLimit, but below PTxLimit.
- Ptx_Enable: bool: Disables | enables the PTxLimit limit for 1 MHz channels fTX± 2 MHz.
- No_Of_Exc_Enable: bool: Disables | enables the ExcPTxLimit limit for 1 MHz channels fTX±3 MHz, fTX±4 MHz, … with NoOfExLimit tolerable exceptions (per statistic cycle) .

get() → SacpStruct

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:MEValuation:LIMit:LENergy:LRANge:SACP
value: SacpStruct = driver.configure.bluetooth.measurement.multiEval.limit.
↪lowEnergy.lrange.sacp.get()
```

These commands define and enable the Spectrum ACP limits for BR (…:LIMit:SACP) , LE 1M PHY (…:LE1M…) , LE 2M PHY (. ..:LE2M…) , and LE coded PHY (…:LRANge…) , respectively.

return
structure: for return value, see the help for SacpStruct structure arguments.

set(*ptx_limit: float*, *exc_ptx_limit: float*, *no_of_ex_limit: int*, *ptx_enable: bool*, *no_of_exc_enable: bool*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:MEValuation:LIMit:LENergy:LRANge:SACP
driver.configure.bluetooth.measurement.multiEval.limit.lowEnergy.lrange.sacp.
↪set(ptx_limit = 1.0, exc_ptx_limit = 1.0, no_of_ex_limit = 1, ptx_enable =␣
↪False, no_of_exc_enable = False)
```

These commands define and enable the Spectrum ACP limits for BR (...:LIMit:SACP) , LE 1M PHY (...:LE1M...) , LE 2M PHY (. ..:LE2M...) , and LE coded PHY (...:LRANge...) , respectively.

param ptx_limit
Power limit for 1 MHz channels fTX± 2 MHz

param exc_ptx_limit
Power limit for 1 MHz channels fTX±3 MHz, fTX±4 MHz, ...

param no_of_ex_limit
Maximum number of tolerable exceptions, i.e. 1 MHz channels fTX±3 MHz, fTX±4 MHz, ... whose power is above ExcPTxLimit, but below PTxLimit.

param ptx_enable
Disables | enables the PTxLimit limit for 1 MHz channels fTX± 2 MHz.

param no_of_exc_enable
Disables | enables the ExcPTxLimit limit for 1 MHz channels fTX±3 MHz, fTX±4 MHz, ... with NoOfExLimit tolerable exceptions (per statistic cycle) .

6.4.1.1.8.65 PowerVsTime

SCPI Command :

```
CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIMit:PVTime
```

class PowerVsTimeCls

PowerVsTime commands group definition. 1 total commands, 0 Subgroups, 1 group commands

class PowerVsTimeStruct

Response structure. Fields:

- Pack_Time_Lower: float: No parameter help available
- Pack_Time_Upper: float: No parameter help available
- Pack_Time_Enable: List[bool]: No parameter help available

get() → PowerVsTimeStruct

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIMit:PVTime
value: PowerVsTimeStruct = driver.configure.bluetooth.measurement.multiEval.
↪limit.powerVsTime.get()
```

Sets and enables/disables a lower and upper timing error limit for PVT measurements.

return
structure: for return value, see the help for PowerVsTimeStruct structure arguments.

set(*pack_time_lower: float*, *pack_time_upper: float*, *pack_time_enable: List[bool]*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIMit:PVTime
driver.configure.bluetooth.measurement.multiEval.limit.powerVsTime.set(pack_
↪time_lower = 1.0, pack_time_upper = 1.0, pack_time_enable = [True, False,␣
↪True])
```

Sets and enables/disables a lower and upper timing error limit for PVT measurements.

param pack_time_lower
No help available

param pack_time_upper
No help available

param pack_time_enable
No help available

6.4.1.1.8.66 Qhsl

SCPI Command :

```
CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIMit:QHSL:FSTability
```

class QhslCls

Qhsl commands group definition. 8 total commands, 7 Subgroups, 1 group commands

class FstabilityStruct

Structure for setting input parameters. Fields:

- Wi: float: No parameter help available
- Wiplus_W_0_Max: float: No parameter help available
- W_0_Max: float: No parameter help available
- Wi_Enabled: List[bool]: No parameter help available
- Wi_W_0_Max_Enabled: List[bool]: No parameter help available
- W_0_Max_Enabled: List[bool]: No parameter help available

get_fstability() → FstabilityStruct

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:MEValuation:LIMit:QHSL:FSTability
value: FstabilityStruct = driver.configure.bluetooth.measurement.multiEval.
↪limit.qhsl.get_fstability()
```

No command help available

return
structure: for return value, see the help for FstabilityStruct structure arguments.

set_fstability(*value: FstabilityStruct*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:MEValuation:LIMit:QHSL:FSTability
structure = driver.configure.bluetooth.measurement.multiEval.limit.qhsl.
↪FstabilityStruct()
structure.Wi: float = 1.0
structure.Wiplus_W_0_Max: float = 1.0
structure.W_0_Max: float = 1.0
structure.Wi_Enabled: List[bool] = [True, False, True]
structure.Wi_W_0_Max_Enabled: List[bool] = [True, False, True]
structure.W_0_Max_Enabled: List[bool] = [True, False, True]
```

(continues on next page)

(continued from previous page)

```
driver.configure.bluetooth.measurement.multiEval.limit.qhsl.set_
↪fstability(value = structure)
```

No command help available

param value
see the help for FstabilityStruct structure arguments.

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.configure.bluetooth.measurement.multiEval.limit.qhsl.clone()
```

Subgroups

6.4.1.1.8.67 P2Q

SCPI Command :

```
CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIMit:QHSL:P2Q:DEVM
```

class P2QCls

P2Q commands group definition. 1 total commands, 0 Subgroups, 1 group commands

class DevmStruct

Structure for setting input parameters. Fields:

- Rms: float: No parameter help available
- Peak: float: No parameter help available
- P_99: float: No parameter help available
- Rms_Enabled: List[bool]: No parameter help available
- Peak_Enabled: List[bool]: No parameter help available
- P_99_Enabled: bool: No parameter help available

get_devm() → DevmStruct

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:MEValuation:LIMit:QHSL:P2Q:DEVM
value: DevmStruct = driver.configure.bluetooth.measurement.multiEval.limit.qhsl.
↪p2Q.get_devm()
```

No command help available

return
structure: for return value, see the help for DevmStruct structure arguments.

set_devm(*value: DevmStruct*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:MEValuation:LIMit:QHSL:P2Q:DEVM
structure = driver.configure.bluetooth.measurement.multiEval.limit.qhsl.p2Q.
↪DevmStruct()
structure.Rms: float = 1.0
structure.Peak: float = 1.0
structure.P_99: float = 1.0
structure.Rms_Enabled: List[bool] = [True, False, True]
structure.Peak_Enabled: List[bool] = [True, False, True]
structure.P_99_Enabled: bool = False
driver.configure.bluetooth.measurement.multiEval.limit.qhsl.p2Q.set_devm(value␣
↪= structure)
```

No command help available

param value
see the help for DevmStruct structure arguments.

6.4.1.1.8.68 P3Q

SCPI Command :

```
CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIMit:QHSL:P3Q:DEVM
```

class P3QCls

P3Q commands group definition. 1 total commands, 0 Subgroups, 1 group commands

class DevmStruct

Structure for setting input parameters. Fields:

- Rms: float: No parameter help available
- Peak: float: No parameter help available
- P_99: float: No parameter help available
- Rms_Enabled: List[bool]: No parameter help available
- Peak_Enabled: List[bool]: No parameter help available
- P_99_Enabled: bool: No parameter help available

get_devm() → DevmStruct

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:MEValuation:LIMit:QHSL:P3Q:DEVM
value: DevmStruct = driver.configure.bluetooth.measurement.multiEval.limit.qhsl.
↪p3Q.get_devm()
```

No command help available

return
structure: for return value, see the help for DevmStruct structure arguments.

set_devm(*value: DevmStruct*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:MEValuation:LIMit:QHSL:P3Q:DEVM
structure = driver.configure.bluetooth.measurement.multiEval.limit.qhsl.p3Q.
↪DevmStruct()
structure.Rms: float = 1.0
structure.Peak: float = 1.0
structure.P_99: float = 1.0
structure.Rms_Enabled: List[bool] = [True, False, True]
structure.Peak_Enabled: List[bool] = [True, False, True]
structure.P_99_Enabled: bool = False
driver.configure.bluetooth.measurement.multiEval.limit.qhsl.p3Q.set_devm(value␣
↪= structure)
```

No command help available

param value
see the help for DevmStruct structure arguments.

6.4.1.1.8.69 P4Q

SCPI Command :

```
CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIMit:QHSL:P4Q:DEVM
```

class P4QCls

P4Q commands group definition. 1 total commands, 0 Subgroups, 1 group commands

class DevmStruct

Structure for setting input parameters. Fields:

- Rms: float: No parameter help available
- Peak: float: No parameter help available
- P_99: float: No parameter help available
- Rms_Enabled: List[bool]: No parameter help available
- Peak_Enabled: List[bool]: No parameter help available
- P_99_Enabled: bool: No parameter help available

get_devm() → DevmStruct

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:MEValuation:LIMit:QHSL:P4Q:DEVM
value: DevmStruct = driver.configure.bluetooth.measurement.multiEval.limit.qhsl.
↪p4Q.get_devm()
```

No command help available

return
structure: for return value, see the help for DevmStruct structure arguments.

set_devm(*value: DevmStruct*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:MEValuation:LIMit:QHSL:P4Q:DEVM
structure = driver.configure.bluetooth.measurement.multiEval.limit.qhsl.p4Q.
↪DevmStruct()
structure.Rms: float = 1.0
structure.Peak: float = 1.0
structure.P_99: float = 1.0
structure.Rms_Enabled: List[bool] = [True, False, True]
structure.Peak_Enabled: List[bool] = [True, False, True]
structure.P_99_Enabled: bool = False
driver.configure.bluetooth.measurement.multiEval.limit.qhsl.p4Q.set_devm(value␣
↪= structure)
```

No command help available

param value
see the help for DevmStruct structure arguments.

6.4.1.1.8.70 P5Q

SCPI Command :

```
CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIMit:QHSL:P5Q:DEVM
```

class P5QCls

P5Q commands group definition. 1 total commands, 0 Subgroups, 1 group commands

class DevmStruct

Structure for setting input parameters. Fields:

- Rms: float: No parameter help available
- Peak: float: No parameter help available
- P_99: float: No parameter help available
- Rms_Enabled: List[bool]: No parameter help available
- Peak_Enabled: List[bool]: No parameter help available
- P_99_Enabled: bool: No parameter help available

get_devm() → DevmStruct

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:MEValuation:LIMit:QHSL:P5Q:DEVM
value: DevmStruct = driver.configure.bluetooth.measurement.multiEval.limit.qhsl.
↪p5Q.get_devm()
```

No command help available

return
structure: for return value, see the help for DevmStruct structure arguments.

set_devm(*value: DevmStruct*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:MEValuation:LIMit:QHSL:P5Q:DEVM
structure = driver.configure.bluetooth.measurement.multiEval.limit.qhsl.p5Q.
↪DevmStruct()
structure.Rms: float = 1.0
structure.Peak: float = 1.0
structure.P_99: float = 1.0
structure.Rms_Enabled: List[bool] = [True, False, True]
structure.Peak_Enabled: List[bool] = [True, False, True]
structure.P_99_Enabled: bool = False
driver.configure.bluetooth.measurement.multiEval.limit.qhsl.p5Q.set_devm(value␣
↪= structure)
```

No command help available

param value
see the help for DevmStruct structure arguments.

6.4.1.1.8.71 P6Q

SCPI Command :

```
CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIMit:QHSL:P6Q:DEVM
```

class P6QCls

P6Q commands group definition. 1 total commands, 0 Subgroups, 1 group commands

class DevmStruct

Structure for setting input parameters. Fields:

- Rms: float: No parameter help available
- Peak: float: No parameter help available
- P_99: float: No parameter help available
- Rms_Enabled: List[bool]: No parameter help available
- Peak_Enabled: List[bool]: No parameter help available
- P_99_Enabled: bool: No parameter help available

get_devm() → DevmStruct

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:MEValuation:LIMit:QHSL:P6Q:DEVM
value: DevmStruct = driver.configure.bluetooth.measurement.multiEval.limit.qhsl.
↪p6Q.get_devm()
```

No command help available

return
structure: for return value, see the help for DevmStruct structure arguments.

set_devm(*value: DevmStruct*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:MEValuation:LIMit:QHSL:P6Q:DEVM
structure = driver.configure.bluetooth.measurement.multiEval.limit.qhsl.p6Q.
↪DevmStruct()
structure.Rms: float = 1.0
structure.Peak: float = 1.0
structure.P_99: float = 1.0
structure.Rms_Enabled: List[bool] = [True, False, True]
structure.Peak_Enabled: List[bool] = [True, False, True]
structure.P_99_Enabled: bool = False
driver.configure.bluetooth.measurement.multiEval.limit.qhsl.p6Q.set_devm(value␣
↪= structure)
```

No command help available

param value
see the help for DevmStruct structure arguments.

6.4.1.1.8.72 PowerVsTime

SCPI Command :

```
CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIMit:QHSL:PVTime
```

class PowerVsTimeCls

PowerVsTime commands group definition. 1 total commands, 0 Subgroups, 1 group commands

class PowerVsTimeStruct

Response structure. Fields:

- Avg_Pow_Upper: float: No parameter help available
- Peak_Pow_Upper: float: No parameter help available
- Avg_Pow_Enabled: List[bool]: No parameter help available
- Peak_Pow_Enabled: List[bool]: No parameter help available

get() → PowerVsTimeStruct

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIMit:QHSL:PVTime
value: PowerVsTimeStruct = driver.configure.bluetooth.measurement.multiEval.
↪limit.qhsl.powerVsTime.get()
```

No command help available

return
structure: for return value, see the help for PowerVsTimeStruct structure arguments.

set(*avg_pow_upper: float*, *peak_pow_upper: float*, *avg_pow_enabled: List[bool]*, *peak_pow_enabled: List[bool]*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIMit:QHSL:PVTime
driver.configure.bluetooth.measurement.multiEval.limit.qhsl.powerVsTime.set(avg_
↪pow_upper = 1.0, peak_pow_upper = 1.0, avg_pow_enabled = [True, False, True],␣
↪peak_pow_enabled = [True, False, True])
```

No command help available

param avg_pow_upper
No help available

param peak_pow_upper
No help available

param avg_pow_enabled
No help available

param peak_pow_enabled
No help available

6.4.1.1.8.73 Sacp

SCPI Command :

```
CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIMit:QHSL:SACP
```

class SacpCls

Sacp commands group definition. 1 total commands, 0 Subgroups, 1 group commands

class SacpStruct

Response structure. Fields:

- Ptx_Limit: float: No parameter help available
- Exc_Ptx_Limit: float: No parameter help available
- No_Of_Ex_Limit: int: No parameter help available
- Ptx_Enable: bool: No parameter help available
- No_Of_Exc_Enable: bool: No parameter help available

get() → SacpStruct

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIMit:QHSL:SACP
value: SacpStruct = driver.configure.bluetooth.measurement.multiEval.limit.qhsl.
↪sacp.get()
```

No command help available

return
structure: for return value, see the help for SacpStruct structure arguments.

set(*ptx_limit: float*, *exc_ptx_limit: float*, *no_of_ex_limit: int*, *ptx_enable: bool*, *no_of_exc_enable: bool*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIMit:QHSL:SACP
driver.configure.bluetooth.measurement.multiEval.limit.qhsl.sacp.set(ptx_limit␣
↪= 1.0, exc_ptx_limit = 1.0, no_of_ex_limit = 1, ptx_enable = False, no_of_exc_
↪enable = False)
```

No command help available

param ptx_limit
No help available

param exc_ptx_limit
No help available

param no_of_ex_limit
No help available

param ptx_enable
No help available

param no_of_exc_enable
No help available

6.4.1.1.8.74 Sacp

SCPI Command :

```
CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIMit:SACP
```

class SacpCls

Sacp commands group definition. 1 total commands, 0 Subgroups, 1 group commands

class SacpStruct

Response structure. Fields:

- Ptx_Limit: float: Power limit for 1 MHz channels fTX± 2 MHz
- Exc_Ptx_Limit: float: Power limit for 1 MHz channels fTX±3 MHz, fTX±4 MHz, …
- No_Of_Ex_Limit: int: Maximum number of tolerable exceptions, i.e. 1 MHz channels fTX±3 MHz, fTX±4 MHz, … whose power is above ExcPTxLimit, but below PTxLimit.
- Ptx_Enable: bool: Disables | enables the PTxLimit limit for 1 MHz channels fTX± 2 MHz.
- No_Of_Exc_Enable: bool: Disables | enables the ExcPTxLimit limit for 1 MHz channels fTX±3 MHz, fTX±4 MHz, … with NoOfExLimit tolerable exceptions (per statistic cycle) .

get() → SacpStruct

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIMit:SACP
value: SacpStruct = driver.configure.bluetooth.measurement.multiEval.limit.sacp.
↪get()
```

These commands define and enable the Spectrum ACP limits for BR (…:LIMit:SACP) , LE 1M PHY (…:LE1M…) , LE 2M PHY (. ..:LE2M…) , and LE coded PHY (…:LRANge…) , respectively.

return
structure: for return value, see the help for SacpStruct structure arguments.

set(*ptx_limit: float*, *exc_ptx_limit: float*, *no_of_ex_limit: int*, *ptx_enable: bool*, *no_of_exc_enable: bool*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIMit:SACP
driver.configure.bluetooth.measurement.multiEval.limit.sacp.set(ptx_limit = 1.0,
↪ exc_ptx_limit = 1.0, no_of_ex_limit = 1, ptx_enable = False, no_of_exc_
↪enable = False)
```

These commands define and enable the Spectrum ACP limits for BR (…:LIMit:SACP) , LE 1M PHY (…:LE1M…) , LE 2M PHY (. ..:LE2M…) , and LE coded PHY (…:LRANge…) , respectively.

param ptx_limit
Power limit for 1 MHz channels fTX± 2 MHz

param exc_ptx_limit
Power limit for 1 MHz channels fTX±3 MHz, fTX±4 MHz, …

param no_of_ex_limit
Maximum number of tolerable exceptions, i.e. 1 MHz channels fTX±3 MHz, fTX±4 MHz, … whose power is above ExcPTxLimit, but below PTxLimit.

param ptx_enable
Disables | enables the PTxLimit limit for 1 MHz channels fTX± 2 MHz.

param no_of_exc_enable
Disables | enables the ExcPTxLimit limit for 1 MHz channels fTX±3 MHz, fTX±4 MHz, … with NoOfExLimit tolerable exceptions (per statistic cycle) .

6.4.1.1.8.75 SoBw

SCPI Command :

```
CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIMit:SOBW
```

class SoBwCls

SoBw commands group definition. 1 total commands, 0 Subgroups, 1 group commands

class SoBwStruct

Response structure. Fields:

- Limit_Threshold: float: Threshold value for ‘high’ vs ‘low’ peak emission bursts.
- Eq_High_Peak_Upper: float: 20 dB bandwidth limit for ‘high’ peak emission bursts (LimitThreshold) .
- Low_Peak_Upper: float: 20 dB bandwidth limit for ‘low’ peak emission bursts (LimitThreshold) .
- Eq_High_Peak_Enable: bool: Disable or enable the 20 dB bandwidth limit for ‘high’ peak emission bursts.
- Low_Peak_Enable: bool: Disable or enable the 20 dB bandwidth limit for ‘low’ peak emission bursts.

get() → SoBwStruct

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIMit:SOBW
value: SoBwStruct = driver.configure.bluetooth.measurement.multiEval.limit.soBw.
↪get()
```

Defines and enables the limits for the 20 dB bandwidth measurement (BR only) .

return
structure: for return value, see the help for SoBwStruct structure arguments.

set(*limit_threshold: float*, *eq_high_peak_upper: float*, *low_peak_upper: float*, *eq_high_peak_enable: bool*, *low_peak_enable: bool*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIMit:SOBW
driver.configure.bluetooth.measurement.multiEval.limit.soBw.set(limit_threshold␣
```

(continues on next page)

(continued from previous page)

```
↪= 1.0, eq_high_peak_upper = 1.0, low_peak_upper = 1.0, eq_high_peak_enable =␣
↪False, low_peak_enable = False)
```

Defines and enables the limits for the 20 dB bandwidth measurement (BR only) .

param limit_threshold
Threshold value for 'high' vs 'low' peak emission bursts.

param eq_high_peak_upper
20 dB bandwidth limit for 'high' peak emission bursts (LimitThreshold) .

param low_peak_upper
20 dB bandwidth limit for 'low' peak emission bursts (LimitThreshold) .

param eq_high_peak_enable
Disable or enable the 20 dB bandwidth limit for 'high' peak emission bursts.

param low_peak_enable
Disable or enable the 20 dB bandwidth limit for 'low' peak emission bursts.

6.4.1.1.8.76 ListPy

SCPI Commands :

```
CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIST:NCONnections
CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIST:COUNt
CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIST:MALGorithm
CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIST:CMODe
CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIST
```

class ListPyCls

ListPy commands group definition. 41 total commands, 2 Subgroups, 5 group commands

get_cmode() → ParameterSetMode

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIST:CMODe
value: enums.ParameterSetMode = driver.configure.bluetooth.measurement.
↪multiEval.listPy.get_cmode()
```

Sets the connector mode, selecting whether all list mode segments use the same RF connection.

return
connector_mode: - GLOBal: Use the same RF connection for all segments, see ROUTe:BLUetooth:MEASi:SPATh. - LIST: Assign a connection to each segment, see CONFigure:BLUetooth:MEASi:MEValuation:LIST:SEGMentno:CIDX.

get_count() → int

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIST:COUNt
value: int = driver.configure.bluetooth.measurement.multiEval.listPy.get_count()
```

Defines the number of segments in the entire measurement interval.

return
segments: No help available

get_malgorithm() → PatternIndependent

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIST:MALGorithm
value: enums.PatternIndependent = driver.configure.bluetooth.measurement.
↪multiEval.listPy.get_malgorithm()
```

No command help available

return
pattern_independent: No help available

get_nconnections() → int

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIST:NCONnections
value: int = driver.configure.bluetooth.measurement.multiEval.listPy.get_
↪nconnections()
```

Sets the number of connections to be defined for the list mode, for connector mode LIST. Define the connections via ROUTe:BLUetooth:MEAS<i>:SPATh.

return
no_of_connections: The maximum number of connections is limited by the number of connectors per smart channel.

get_value() → bool

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIST
value: bool = driver.configure.bluetooth.measurement.multiEval.listPy.get_
↪value()
```

Enables or disables the list mode.

return
enable: OFF: disable list mode ON: enable list mode

set_cmode(*connector_mode: ParameterSetMode*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIST:CMODe
driver.configure.bluetooth.measurement.multiEval.listPy.set_cmode(connector_
↪mode = enums.ParameterSetMode.GLOBal)
```

Sets the connector mode, selecting whether all list mode segments use the same RF connection.

param connector_mode

- GLOBal: Use the same RF connection for all segments, see ROUTe:BLUetooth:MEASi:SPATh.
- LIST: Assign a connection to each segment, see CONFigure:BLUetooth:MEASi:MEValuation:LIST:SEGMentno:CIDX.

set_count(*segments: int*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIST:COUNt
driver.configure.bluetooth.measurement.multiEval.listPy.set_count(segments = 1)
```

Defines the number of segments in the entire measurement interval.

param segments
No help available

set_malgorithm(*pattern_independent: PatternIndependent*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIST:MALGorithm
driver.configure.bluetooth.measurement.multiEval.listPy.set_malgorithm(pattern_
↪independent = enums.PatternIndependent.PINDependent)
```

No command help available

param pattern_independent
No help available

set_nconnections(*no_of_connections: int*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIST:NCONnections
driver.configure.bluetooth.measurement.multiEval.listPy.set_nconnections(no_of_
↪connections = 1)
```

Sets the number of connections to be defined for the list mode, for connector mode LIST. Define the connections via ROUTe:BLUetooth:MEAS<i>:SPATh.

param no_of_connections
The maximum number of connections is limited by the number of connectors per smart channel.

set_value(*enable: bool*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIST
driver.configure.bluetooth.measurement.multiEval.listPy.set_value(enable =␣
↪False)
```

Enables or disables the list mode.

param enable
OFF: disable list mode ON: enable list mode

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.configure.bluetooth.measurement.multiEval.listPy.clone()
```

Subgroups

6.4.1.1.8.77 Segment

RepCap Settings

```
# Range: S1 .. S128
rc = driver.configure.bluetooth.measurement.multiEval.listPy.segment.repcap_segment_get()
driver.configure.bluetooth.measurement.multiEval.listPy.segment.repcap_segment_
↪set(repcap.Segment.S1)
```

class SegmentCls

Segment commands group definition. 35 total commands, 4 Subgroups, 0 group commands Repeated Capability: Segment, default value after init: Segment.S1

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.configure.bluetooth.measurement.multiEval.listPy.segment.clone()
```

Subgroups

6.4.1.1.8.78 Cidx

SCPI Command :

```
CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIST:SEGMent<nr>:CIDX
```

class CidxCls

Cidx commands group definition. 1 total commands, 0 Subgroups, 1 group commands

get(*segment=Segment.Default*) → int

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIST:SEGMent<nr>
↪:CIDX
value: int = driver.configure.bluetooth.measurement.multiEval.listPy.segment.
↪cidx.get(segment = repcap.Segment.Default)
```

Selects the RF connection index for segment <no>. For a definition of the connection indices, see ROUTe:BLUetooth:MEAS<i>:SPATh.

param segment
optional repeated capability selector. Default value: S1 (settable in the interface 'Segment')

return
connection_index: Index of the connection to be used for the segment.

set(*connection_index: int*, *segment=Segment.Default*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIST:SEGMent<nr>
↪:CIDX
driver.configure.bluetooth.measurement.multiEval.listPy.segment.cidx.
↪set(connection_index = 1, segment = repcap.Segment.Default)
```

Selects the RF connection index for segment <no>. For a definition of the connection indices, see ROUTe:BLUetooth:MEAS<i>:SPATh.

param connection_index
Index of the connection to be used for the segment.

param segment
optional repeated capability selector. Default value: S1 (settable in the interface 'Segment')

6.4.1.1.8.79 Results

SCPI Command :

```
CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIST:SEGMent<nr>:RESults
```

class ResultsCls

Results commands group definition. 7 total commands, 6 Subgroups, 1 group commands

class ResultsStruct

Response structure. Fields:

- Enable_Mod_Scalar: bool: Enable/disable statistical modulation results
- Enable_Pow_Scalar: bool: Enable/disable statistical power results
- Enable_Spec_Obw: bool: Enable/disable the spectrum 20 dB bandwidth results (BR)
- Enable_Spec_Acp: bool: Enable/disable the spectrum ACP results (BR, LE)
- Enable_Spec_Gat_Acp: bool: Enable/disable the spectrum gated ACP results (EDR)

get(*segment=Segment.Default*) → ResultsStruct

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIST:SEGMent<nr>
↪:RESults
value: ResultsStruct = driver.configure.bluetooth.measurement.multiEval.listPy.
↪segment.results.get(segment = repcap.Segment.Default)
```

Enables or disables the evaluation of the particular measurement type in the segment.

param segment
optional repeated capability selector. Default value: S1 (settable in the interface 'Segment')

return
structure: for return value, see the help for ResultsStruct structure arguments.

set(*enable_mod_scalar: bool*, *enable_pow_scalar: bool*, *enable_spec_obw: bool*, *enable_spec_acp: bool*, *enable_spec_gat_acp: bool*, *segment=Segment.Default*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIST:SEGMent<nr>
↪:RESults
driver.configure.bluetooth.measurement.multiEval.listPy.segment.results.
↪set(enable_mod_scalar = False, enable_pow_scalar = False, enable_spec_obw =␣
↪False, enable_spec_acp = False, enable_spec_gat_acp = False, segment = repcap.
↪Segment.Default)
```

Enables or disables the evaluation of the particular measurement type in the segment.

param enable_mod_scalar
Enable/disable statistical modulation results

param enable_pow_scalar
Enable/disable statistical power results

param enable_spec_obw
Enable/disable the spectrum 20 dB bandwidth results (BR)

param enable_spec_acp
Enable/disable the spectrum ACP results (BR, LE)

param enable_spec_gat_acp
Enable/disable the spectrum gated ACP results (EDR)

param segment
optional repeated capability selector. Default value: S1 (settable in the interface 'Segment')

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.configure.bluetooth.measurement.multiEval.listPy.segment.results.clone()
```

Subgroups

6.4.1.1.8.80 Mscalar

SCPI Command :

```
CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIST:SEGMent<nr>:RESults:MSCalar
```

class MscalarCls

Mscalar commands group definition. 1 total commands, 0 Subgroups, 1 group commands

get(*segment=Segment.Default*) → bool

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIST:SEGMent<nr>
↪:RESults:MSCalar
value: bool = driver.configure.bluetooth.measurement.multiEval.listPy.segment.
↪results.mscalar.get(segment = repcap.Segment.Default)
```

Enables or disables the evaluation of results for the segment<no> in list mode. The last mnemonic denotes the measurement type: statistical modulation results, statistical power results, spectrum ACP (BR, LE) , spectrum gated ACP (EDR) , spectrum 20 dB bandwidth (BR) .

param segment
optional repeated capability selector. Default value: S1 (settable in the interface 'Segment')

return
enable_mod_scalar: No help available

set(*enable_mod_scalar: bool*, *segment=Segment.Default*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIST:SEGMent<nr>
↪:RESults:MSCalar
driver.configure.bluetooth.measurement.multiEval.listPy.segment.results.mscalar.
↪set(enable_mod_scalar = False, segment = repcap.Segment.Default)
```

Enables or disables the evaluation of results for the segment<no> in list mode. The last mnemonic denotes the measurement type: statistical modulation results, statistical power results, spectrum ACP (BR, LE) , spectrum gated ACP (EDR) , spectrum 20 dB bandwidth (BR) .

param enable_mod_scalar
No help available

param segment
optional repeated capability selector. Default value: S1 (settable in the interface 'Segment')

6.4.1.1.8.81 Pencoding

SCPI Command :

```
CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIST:SEGMent<nr>:RESults:PENCoding
```

class PencodingCls

Pencoding commands group definition. 1 total commands, 0 Subgroups, 1 group commands

get(*segment=Segment.Default*) → bool

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIST:SEGMent<nr>
↪:RESults:PENCoding
value: bool = driver.configure.bluetooth.measurement.multiEval.listPy.segment.
↪results.pencoding.get(segment = repcap.Segment.Default)
```

No command help available

param segment
optional repeated capability selector. Default value: S1 (settable in the interface 'Segment')

return
enable_phase_enc: No help available

set(*enable_phase_enc: bool*, *segment=Segment.Default*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIST:SEGMent<nr>
↪:RESults:PENCoding
driver.configure.bluetooth.measurement.multiEval.listPy.segment.results.
↪pencoding.set(enable_phase_enc = False, segment = repcap.Segment.Default)
```

No command help available

param enable_phase_enc
No help available

param segment
optional repeated capability selector. Default value: S1 (settable in the interface 'Segment')

6.4.1.1.8.82 Pscalar

SCPI Command :

```
CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIST:SEGMent<nr>:RESults:PSCalar
```

class PscalarCls

Pscalar commands group definition. 1 total commands, 0 Subgroups, 1 group commands

get(*segment=Segment.Default*) → bool

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIST:SEGMent<nr>
↪:RESults:PSCalar
value: bool = driver.configure.bluetooth.measurement.multiEval.listPy.segment.
↪results.pscalar.get(segment = repcap.Segment.Default)
```

Enables or disables the evaluation of results for the segment<no> in list mode. The last mnemonic denotes the measurement type: statistical modulation results, statistical power results, spectrum ACP (BR, LE) , spectrum gated ACP (EDR) , spectrum 20 dB bandwidth (BR) .

param segment
optional repeated capability selector. Default value: S1 (settable in the interface 'Segment')

return
enable_pow_scalar: No help available

set(*enable_pow_scalar: bool*, *segment=Segment.Default*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIST:SEGMent<nr>
↪:RESults:PSCalar
driver.configure.bluetooth.measurement.multiEval.listPy.segment.results.pscalar.
↪set(enable_pow_scalar = False, segment = repcap.Segment.Default)
```

Enables or disables the evaluation of results for the segment<no> in list mode. The last mnemonic denotes the measurement type: statistical modulation results, statistical power results, spectrum ACP (BR, LE) , spectrum gated ACP (EDR) , spectrum 20 dB bandwidth (BR) .

param enable_pow_scalar
No help available

param segment
optional repeated capability selector. Default value: S1 (settable in the interface 'Segment')

6.4.1.1.8.83 Sacp

SCPI Command :

```
CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIST:SEGMent<nr>:RESults:SACP
```

class SacpCls

Sacp commands group definition. 1 total commands, 0 Subgroups, 1 group commands

get(*segment=Segment.Default*) → bool

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIST:SEGMent<nr>
↪:RESults:SACP
value: bool = driver.configure.bluetooth.measurement.multiEval.listPy.segment.
↪results.sacp.get(segment = repcap.Segment.Default)
```

Enables or disables the evaluation of results for the segment<no> in list mode. The last mnemonic denotes the measurement type: statistical modulation results, statistical power results, spectrum ACP (BR, LE) , spectrum gated ACP (EDR) , spectrum 20 dB bandwidth (BR) .

param segment
optional repeated capability selector. Default value: S1 (settable in the interface 'Segment')

return
enable_spec_acp: No help available

set(*enable_spec_acp: bool*, *segment=Segment.Default*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIST:SEGMent<nr>
↪:RESults:SACP
driver.configure.bluetooth.measurement.multiEval.listPy.segment.results.sacp.
↪set(enable_spec_acp = False, segment = repcap.Segment.Default)
```

Enables or disables the evaluation of results for the segment<no> in list mode. The last mnemonic denotes the measurement type: statistical modulation results, statistical power results, spectrum ACP (BR, LE) , spectrum gated ACP (EDR) , spectrum 20 dB bandwidth (BR) .

param enable_spec_acp
No help available

param segment
optional repeated capability selector. Default value: S1 (settable in the interface 'Segment')

6.4.1.1.8.84 Sgacp

SCPI Command :

```
CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIST:SEGMent<nr>:RESults:SGACp
```

class SgacpCls

Sgacp commands group definition. 1 total commands, 0 Subgroups, 1 group commands

get(*segment=Segment.Default*) → bool

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIST:SEGMent<nr>
↪:RESults:SGACp
value: bool = driver.configure.bluetooth.measurement.multiEval.listPy.segment.
↪results.sgacp.get(segment = repcap.Segment.Default)
```

Enables or disables the evaluation of results for the segment<no> in list mode. The last mnemonic denotes the measurement type: statistical modulation results, statistical power results, spectrum ACP (BR, LE) , spectrum gated ACP (EDR) , spectrum 20 dB bandwidth (BR) .

param segment
optional repeated capability selector. Default value: S1 (settable in the interface 'Segment')

return
enable_spec_gat_acp: No help available

`set`(*enable_spec_gat_acp: bool*, *segment=Segment.Default*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIST:SEGMent<nr>
↪:RESults:SGACp
driver.configure.bluetooth.measurement.multiEval.listPy.segment.results.sgacp.
↪set(enable_spec_gat_acp = False, segment = repcap.Segment.Default)
```

Enables or disables the evaluation of results for the segment<no> in list mode. The last mnemonic denotes the measurement type: statistical modulation results, statistical power results, spectrum ACP (BR, LE) , spectrum gated ACP (EDR) , spectrum 20 dB bandwidth (BR) .

param enable_spec_gat_acp
No help available

param segment
optional repeated capability selector. Default value: S1 (settable in the interface 'Segment')

6.4.1.1.8.85 SoBw

SCPI Command :

```
CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIST:SEGMent<nr>:RESults:SOBW
```

`class SoBwCls`

SoBw commands group definition. 1 total commands, 0 Subgroups, 1 group commands

`get`(*segment=Segment.Default*) → bool

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIST:SEGMent<nr>
↪:RESults:SOBW
value: bool = driver.configure.bluetooth.measurement.multiEval.listPy.segment.
↪results.soBw.get(segment = repcap.Segment.Default)
```

Enables or disables the evaluation of results for the segment<no> in list mode. The last mnemonic denotes the measurement type: statistical modulation results, statistical power results, spectrum ACP (BR, LE) , spectrum gated ACP (EDR) , spectrum 20 dB bandwidth (BR) .

param segment
optional repeated capability selector. Default value: S1 (settable in the interface 'Segment')

return
enable_spec_obw: No help available

`set`(*enable_spec_obw: bool*, *segment=Segment.Default*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIST:SEGMent<nr>
↪:RESults:SOBW
driver.configure.bluetooth.measurement.multiEval.listPy.segment.results.soBw.
↪set(enable_spec_obw = False, segment = repcap.Segment.Default)
```

Enables or disables the evaluation of results for the segment<no> in list mode. The last mnemonic denotes the measurement type: statistical modulation results, statistical power results, spectrum ACP (BR, LE) , spectrum gated ACP (EDR) , spectrum 20 dB bandwidth (BR) .

param enable_spec_obw
No help available

param segment
optional repeated capability selector. Default value: S1 (settable in the interface 'Segment')

6.4.1.1.8.86 Scount

SCPI Command :

```
CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIST:SEGMent<nr>:SCOunt
```

class ScountCls

Scount commands group definition. 7 total commands, 6 Subgroups, 1 group commands

class ScountStruct

Response structure. Fields:

- Mod_Stat_Count: int: Statistic count for statistical modulation measurement
- Power_Stat_Count: int: Statistic count for statistical power measurement
- Spec_Obw_Stat_Cnt: int: Statistic count for spectrum 20 dB bandwidth measurement (BR)
- Spec_Acp_Stat_Cnt: int: Statistic count for spectrum ACP (BR, LE)
- Spec_Gat_Acp_Stat_Cnt: int: Statistic count for spectrum gated ACP measurement (EDR)

get(*segment=Segment.Default*) → ScountStruct

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIST:SEGMent<nr>
↪:SCOunt
value: ScountStruct = driver.configure.bluetooth.measurement.multiEval.listPy.
↪segment.scount.get(segment = repcap.Segment.Default)
```

Defines the statistic count for the particular measurement type in the segment.

param segment
optional repeated capability selector. Default value: S1 (settable in the interface 'Segment')

return
structure: for return value, see the help for ScountStruct structure arguments.

set(*mod_stat_count: int*, *power_stat_count: int*, *spec_obw_stat_cnt: int*, *spec_acp_stat_cnt: int*, *spec_gat_acp_stat_cnt: int*, *segment=Segment.Default*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIST:SEGMent<nr>
↪:SCOunt
driver.configure.bluetooth.measurement.multiEval.listPy.segment.scount.set(mod_
↪stat_count = 1, power_stat_count = 1, spec_obw_stat_cnt = 1, spec_acp_stat_
↪cnt = 1, spec_gat_acp_stat_cnt = 1, segment = repcap.Segment.Default)
```

Defines the statistic count for the particular measurement type in the segment.

param mod_stat_count
Statistic count for statistical modulation measurement

param power_stat_count
Statistic count for statistical power measurement

param spec_obw_stat_cnt
Statistic count for spectrum 20 dB bandwidth measurement (BR)

param spec_acp_stat_cnt
Statistic count for spectrum ACP (BR, LE)

param spec_gat_acp_stat_cnt
Statistic count for spectrum gated ACP measurement (EDR)

param segment
optional repeated capability selector. Default value: S1 (settable in the interface 'Segment')

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.configure.bluetooth.measurement.multiEval.listPy.segment.scount.clone()
```

Subgroups

6.4.1.1.8.87 Mscalar

SCPI Command :

```
CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIST:SEGMent<nr>:SCOunt:MSCalar
```

class MscalarCls

Mscalar commands group definition. 1 total commands, 0 Subgroups, 1 group commands

get(*segment=Segment.Default*) → int

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIST:SEGMent<nr>
↪:SCOunt:MSCalar
value: int = driver.configure.bluetooth.measurement.multiEval.listPy.segment.
↪scount.mscalar.get(segment = repcap.Segment.Default)
```

Defines the statistic count in the segment. The last mnemonic denotes the measurement type: statistical modulation measurement, statistical power measurement, spectrum ACP measurement (BR, LE) , spectrum gated ACP measurement (EDR) , spectrum 20 dB bandwidth (occupied bandwidth) measurement (BR) .

param segment
optional repeated capability selector. Default value: S1 (settable in the interface 'Segment')

return
mod_stat_count: Statistic count

`set`(*mod_stat_count: int*, *segment=Segment.Default*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIST:SEGMent<nr>
↪:SCOunt:MSCalar
driver.configure.bluetooth.measurement.multiEval.listPy.segment.scount.mscalar.
↪set(mod_stat_count = 1, segment = repcap.Segment.Default)
```

Defines the statistic count in the segment. The last mnemonic denotes the measurement type: statistical modulation measurement, statistical power measurement, spectrum ACP measurement (BR, LE) , spectrum gated ACP measurement (EDR) , spectrum 20 dB bandwidth (occupied bandwidth) measurement (BR) .

param mod_stat_count
Statistic count

param segment
optional repeated capability selector. Default value: S1 (settable in the interface 'Segment')

6.4.1.1.8.88 Pencoding

SCPI Command :

```
CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIST:SEGMent<nr>:SCOunt:PENCoding
```

`class PencodingCls`

Pencoding commands group definition. 1 total commands, 0 Subgroups, 1 group commands

`get`(*segment=Segment.Default*) → int

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIST:SEGMent<nr>
↪:SCOunt:PENCoding
value: int = driver.configure.bluetooth.measurement.multiEval.listPy.segment.
↪scount.pencoding.get(segment = repcap.Segment.Default)
```

No command help available

param segment
optional repeated capability selector. Default value: S1 (settable in the interface 'Segment')

return
phase_enc_stat_cnt: No help available

`set`(*phase_enc_stat_cnt: int*, *segment=Segment.Default*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIST:SEGMent<nr>
↪:SCOunt:PENCoding
driver.configure.bluetooth.measurement.multiEval.listPy.segment.scount.
↪pencoding.set(phase_enc_stat_cnt = 1, segment = repcap.Segment.Default)
```

No command help available

param phase_enc_stat_cnt
No help available

param segment
optional repeated capability selector. Default value: S1 (settable in the interface 'Segment')

6.4.1.1.8.89 Pscalar

SCPI Command :

```
CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIST:SEGMent<nr>:SCOunt:PSCalar
```

class PscalarCls

Pscalar commands group definition. 1 total commands, 0 Subgroups, 1 group commands

get(*segment=Segment.Default*) → int

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIST:SEGMent<nr>
↪:SCOunt:PSCalar
value: int = driver.configure.bluetooth.measurement.multiEval.listPy.segment.
↪scount.pscalar.get(segment = repcap.Segment.Default)
```

Defines the statistic count in the segment. The last mnemonic denotes the measurement type: statistical modulation measurement, statistical power measurement, spectrum ACP measurement (BR, LE) , spectrum gated ACP measurement (EDR) , spectrum 20 dB bandwidth (occupied bandwidth) measurement (BR) .

param segment
optional repeated capability selector. Default value: S1 (settable in the interface 'Segment')

return
power_stat_count: Statistic count

set(*power_stat_count: int*, *segment=Segment.Default*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIST:SEGMent<nr>
↪:SCOunt:PSCalar
driver.configure.bluetooth.measurement.multiEval.listPy.segment.scount.pscalar.
↪set(power_stat_count = 1, segment = repcap.Segment.Default)
```

Defines the statistic count in the segment. The last mnemonic denotes the measurement type: statistical modulation measurement, statistical power measurement, spectrum ACP measurement (BR, LE) , spectrum gated ACP measurement (EDR) , spectrum 20 dB bandwidth (occupied bandwidth) measurement (BR) .

param power_stat_count
Statistic count

param segment
optional repeated capability selector. Default value: S1 (settable in the interface 'Segment')

6.4.1.1.8.90 Sacp

SCPI Command :

```
CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIST:SEGMent<nr>:SCOunt:SACP
```

class SacpCls

Sacp commands group definition. 1 total commands, 0 Subgroups, 1 group commands

get(*segment=Segment.Default*) → int

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIST:SEGMent<nr>
↪:SCOunt:SACP
value: int = driver.configure.bluetooth.measurement.multiEval.listPy.segment.
↪scount.sacp.get(segment = repcap.Segment.Default)
```

Defines the statistic count in the segment. The last mnemonic denotes the measurement type: statistical modulation measurement, statistical power measurement, spectrum ACP measurement (BR, LE) , spectrum gated ACP measurement (EDR) , spectrum 20 dB bandwidth (occupied bandwidth) measurement (BR) .

param segment
optional repeated capability selector. Default value: S1 (settable in the interface 'Segment')

return
spec_acp_stat_cnt: Statistic count

set(*spec_acp_stat_cnt: int*, *segment=Segment.Default*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIST:SEGMent<nr>
↪:SCOunt:SACP
driver.configure.bluetooth.measurement.multiEval.listPy.segment.scount.sacp.
↪set(spec_acp_stat_cnt = 1, segment = repcap.Segment.Default)
```

Defines the statistic count in the segment. The last mnemonic denotes the measurement type: statistical modulation measurement, statistical power measurement, spectrum ACP measurement (BR, LE) , spectrum gated ACP measurement (EDR) , spectrum 20 dB bandwidth (occupied bandwidth) measurement (BR) .

param spec_acp_stat_cnt
Statistic count

param segment
optional repeated capability selector. Default value: S1 (settable in the interface 'Segment')

6.4.1.1.8.91 Sgacp

SCPI Command :

```
CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIST:SEGMent<nr>:SCOunt:SGACp
```

class SgacpCls

Sgacp commands group definition. 1 total commands, 0 Subgroups, 1 group commands

get(*segment=Segment.Default*) → int

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIST:SEGMent<nr>
↪:SCOunt:SGACp
value: int = driver.configure.bluetooth.measurement.multiEval.listPy.segment.
↪scount.sgacp.get(segment = repcap.Segment.Default)
```

Defines the statistic count in the segment. The last mnemonic denotes the measurement type: statistical modulation measurement, statistical power measurement, spectrum ACP measurement (BR, LE) , spectrum gated ACP measurement (EDR) , spectrum 20 dB bandwidth (occupied bandwidth) measurement (BR) .

param segment
optional repeated capability selector. Default value: S1 (settable in the interface 'Segment')

return
spec_gat_acp_stat_cnt: Statistic count

set(*spec_gat_acp_stat_cnt: int*, *segment=Segment.Default*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIST:SEGMent<nr>
↪:SCOunt:SGACp
driver.configure.bluetooth.measurement.multiEval.listPy.segment.scount.sgacp.
↪set(spec_gat_acp_stat_cnt = 1, segment = repcap.Segment.Default)
```

Defines the statistic count in the segment. The last mnemonic denotes the measurement type: statistical modulation measurement, statistical power measurement, spectrum ACP measurement (BR, LE) , spectrum gated ACP measurement (EDR) , spectrum 20 dB bandwidth (occupied bandwidth) measurement (BR) .

param spec_gat_acp_stat_cnt
Statistic count

param segment
optional repeated capability selector. Default value: S1 (settable in the interface 'Segment')

6.4.1.1.8.92 SoBw

SCPI Command :

```
CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIST:SEGMent<nr>:SCOunt:SOBW
```

class SoBwCls

SoBw commands group definition. 1 total commands, 0 Subgroups, 1 group commands

get(*segment=Segment.Default*) → int

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIST:SEGMent<nr>
↪:SCOunt:SOBW
value: int = driver.configure.bluetooth.measurement.multiEval.listPy.segment.
↪scount.soBw.get(segment = repcap.Segment.Default)
```

Defines the statistic count in the segment. The last mnemonic denotes the measurement type: statistical modulation measurement, statistical power measurement, spectrum ACP measurement (BR, LE) , spectrum gated ACP measurement (EDR) , spectrum 20 dB bandwidth (occupied bandwidth) measurement (BR) .

param segment
optional repeated capability selector. Default value: S1 (settable in the interface 'Segment')

return
spec_obw_stat_cnt: Statistic count

`set`(*spec_obw_stat_cnt: int*, *segment=Segment.Default*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIST:SEGMent<nr>
↪:SCOunt:SOBW
driver.configure.bluetooth.measurement.multiEval.listPy.segment.scount.soBw.
↪set(spec_obw_stat_cnt = 1, segment = repcap.Segment.Default)
```

Defines the statistic count in the segment. The last mnemonic denotes the measurement type: statistical modulation measurement, statistical power measurement, spectrum ACP measurement (BR, LE) , spectrum gated ACP measurement (EDR) , spectrum 20 dB bandwidth (occupied bandwidth) measurement (BR) .

param spec_obw_stat_cnt
Statistic count

param segment
optional repeated capability selector. Default value: S1 (settable in the interface 'Segment')

6.4.1.1.8.93 Setup

SCPI Command :

```
CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIST:SEGMent<nr>[:SETup]
```

`class SetupCls`

Setup commands group definition. 20 total commands, 17 Subgroups, 1 group commands

`class SetupStruct`

Structure for setting input parameters. Contains optional setting parameters. Fields:

- Burst_Type: enums.BurstType: BR: Basic Rate EDR: Enhanced Data Rate LE: Low Energy
- Packet_Type: enums.SegmentPacketType: Packet type expected in the segment DH1, DH3, DH5: BR packet E21P, E23P, E25P, E31P, E33P, E35P: 2-DH1, 2-DH3, 2-DH5, 3-DH1, 3-DH3, 3-DH5 EDR packet RFPHytest: LE test packet ADVertiser: LE advertiser RFCTe: LE with CTE test packet
- Pattern_Type: enums.MevPatternType: Payload pattern type expected in the segment: ALL1: 11111111 P11: 10101010 OTHer: any pattern except P11, P44 and ALL1 ALTernating: the periodical change of the pattern P11, P44 P44: 11110000
- Payload_Length: int: Payload length expected in the segment
- No_Of_Off_Slots: int: Number of unused slots between any two occupied slots or slot sequences expected in the segment.
- Segment_Length: int: Number of measured bursts in the segment. The sum of the length of all active segments must not exceed 6700 timeslots (1 timeslot = 625 s duration) .
- Meas_On_Exception: bool: Specifies whether the segment results that the CMP180 identifies as faulty or inaccurate are rejected. ON: include the erroneous bursts OFF: exclude the erroneous bursts

- Level: float: Expected nominal power in the segment. The range of the expected nominal power can be calculated as follows: Range (Expected Nominal Power) = Range (Input Power) + External Attenuation - User Margin The input power range is stated in the specifications document.
- Frequency: float: Center frequency for the segment
- Meas_Filter: enums.FilterWidth: Filter bandwidth for the segment NARRow: Narrowband filter WIDE: Wideband filter
- Retrigger: bool: Optional setting parameter. Specifies whether a trigger event is required for the segment or not. The setting is ignored for the first segment of a measurement. OFF: measure the segment without retrigger ON: trigger event required

get(*segment=Segment.Default*) → SetupStruct

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIST:SEGMent<nr>
↪[:SETup]
value: SetupStruct = driver.configure.bluetooth.measurement.multiEval.listPy.
↪segment.setup.get(segment = repcap.Segment.Default)
```

Defines the segment length, the signal properties and the analyzer settings for a selected segment. In general, this command must be sent for all segments to be measured.

param segment
optional repeated capability selector. Default value: S1 (settable in the interface 'Segment')

return
structure: for return value, see the help for SetupStruct structure arguments.

set(*structure: SetupStruct*, *segment=Segment.Default*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIST:SEGMent<nr>
↪[:SETup]
structure = driver.configure.bluetooth.measurement.multiEval.listPy.segment.
↪setup.SetupStruct()
structure.Burst_Type: enums.BurstType = enums.BurstType.BR
structure.Packet_Type: enums.SegmentPacketType = enums.SegmentPacketType.
↪ADVertiser
structure.Pattern_Type: enums.MevPatternType = enums.MevPatternType.ALL1
structure.Payload_Length: int = 1
structure.No_Of_Off_Slots: int = 1
structure.Segment_Length: int = 1
structure.Meas_On_Exception: bool = False
structure.Level: float = 1.0
structure.Frequency: float = 1.0
structure.Meas_Filter: enums.FilterWidth = enums.FilterWidth.NARRow
structure.Retrigger: bool = False
driver.configure.bluetooth.measurement.multiEval.listPy.segment.setup.
↪set(structure, segment = repcap.Segment.Default)
```

Defines the segment length, the signal properties and the analyzer settings for a selected segment. In general, this command must be sent for all segments to be measured.

param structure
for set value, see the help for SetupStruct structure arguments.

param segment
optional repeated capability selector. Default value: S1 (settable in the interface 'Segment')

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.configure.bluetooth.measurement.multiEval.listPy.segment.setup.clone()
```

Subgroups

6.4.1.1.8.94 Btype

SCPI Command :

```
CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIST:SEGMent<nr>[:SETup]:BTYPe
```

class BtypeCls

Btype commands group definition. 1 total commands, 0 Subgroups, 1 group commands

get(*segment=Segment.Default*) → BurstType

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIST:SEGMent<nr>
↪[:SETup]:BTYPe
value: enums.BurstType = driver.configure.bluetooth.measurement.multiEval.
↪listPy.segment.setup.btype.get(segment = repcap.Segment.Default)
```

Specifies the burst type of the signal expected in the segment.

param segment
optional repeated capability selector. Default value: S1 (settable in the interface 'Segment')

return
burst_type: BR: Basic Rate EDR: Enhanced Data Rate LE: Low Energy

set(*burst_type: BurstType*, *segment=Segment.Default*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIST:SEGMent<nr>
↪[:SETup]:BTYPe
driver.configure.bluetooth.measurement.multiEval.listPy.segment.setup.btype.
↪set(burst_type = enums.BurstType.BR, segment = repcap.Segment.Default)
```

Specifies the burst type of the signal expected in the segment.

param burst_type
BR: Basic Rate EDR: Enhanced Data Rate LE: Low Energy

param segment
optional repeated capability selector. Default value: S1 (settable in the interface 'Segment')

6.4.1.1.8.95 Cscheme

SCPI Command :

```
CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIST:SEGMent<nr>[:SETup]:CSCHeme
```

class CschemeCls

Cscheme commands group definition. 1 total commands, 0 Subgroups, 1 group commands

get(*segment=Segment.Default*) → CodingScheme

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIST:SEGMent<nr>
↪[:SETup]:CSCHeme
value: enums.CodingScheme = driver.configure.bluetooth.measurement.multiEval.
↪listPy.segment.setup.cscheme.get(segment = repcap.Segment.Default)
```

Defines coding scheme S for LE coded PHY according to the core specification version 5.0 for Bluetooth wireless technology.

param segment
optional repeated capability selector. Default value: S1 (settable in the interface 'Segment')

return
le_lr_coding: Coding S = 8 or S = 2

set(*le_lr_coding: CodingScheme*, *segment=Segment.Default*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIST:SEGMent<nr>
↪[:SETup]:CSCHeme
driver.configure.bluetooth.measurement.multiEval.listPy.segment.setup.cscheme.
↪set(le_lr_coding = enums.CodingScheme.S2, segment = repcap.Segment.Default)
```

Defines coding scheme S for LE coded PHY according to the core specification version 5.0 for Bluetooth wireless technology.

param le_lr_coding
Coding S = 8 or S = 2

param segment
optional repeated capability selector. Default value: S1 (settable in the interface 'Segment')

6.4.1.1.8.96 Cte

class CteCls

Cte commands group definition. 2 total commands, 2 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.configure.bluetooth.measurement.multiEval.listPy.segment.setup.cte.
↪clone()
```

Subgroups

6.4.1.1.8.97 TypePy

SCPI Command :

```
CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIST:SEGMent<nr>[:SETup]:CTE:TYPE
```

class TypePyCls

TypePy commands group definition. 1 total commands, 0 Subgroups, 1 group commands

get(*segment=Segment.Default*) → CteType

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIST:SEGMent<nr>
↪[:SETup]:CTE:TYPE
value: enums.CteType = driver.configure.bluetooth.measurement.multiEval.listPy.
↪segment.setup.cte.typePy.get(segment = repcap.Segment.Default)
```

Defines the CTE type for segment.

param segment
optional repeated capability selector. Default value: S1 (settable in the interface 'Segment')

return
cte_type: CTE slot type for LE with CTE AOA: CTE type angle of arrival, 2 s slot AOD1: CTE type angle of departure, 1 s slot AOD2: CTE type angle of departure, 2 s slot

set(*cte_type: CteType*, *segment=Segment.Default*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIST:SEGMent<nr>
↪[:SETup]:CTE:TYPE
driver.configure.bluetooth.measurement.multiEval.listPy.segment.setup.cte.
↪typePy.set(cte_type = enums.CteType.AOA, segment = repcap.Segment.Default)
```

Defines the CTE type for segment.

param cte_type
CTE slot type for LE with CTE AOA: CTE type angle of arrival, 2 s slot AOD1: CTE type angle of departure, 1 s slot AOD2: CTE type angle of departure, 2 s slot

param segment
optional repeated capability selector. Default value: S1 (settable in the interface 'Segment')

6.4.1.1.8.98 Units

SCPI Command :

```
CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIST:SEGMent<nr>[:SETup]:CTE:UNITs
```

class UnitsCls

Units commands group definition. 1 total commands, 0 Subgroups, 1 group commands

get(*segment=Segment.Default*) → int

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIST:SEGMent<nr>
↪[:SETup]:CTE:UNITs
value: int = driver.configure.bluetooth.measurement.multiEval.listPy.segment.
↪setup.cte.units.get(segment = repcap.Segment.Default)
```

Defines the No. of CTE units for segment.

param segment
optional repeated capability selector. Default value: S1 (settable in the interface 'Segment')

return
cte_units: No. of CTE units for LE with CTE, one unit corresponds to 8 s.

set(*cte_units: int*, *segment=Segment.Default*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIST:SEGMent<nr>
↪[:SETup]:CTE:UNITs
driver.configure.bluetooth.measurement.multiEval.listPy.segment.setup.cte.units.
↪set(cte_units = 1, segment = repcap.Segment.Default)
```

Defines the No. of CTE units for segment.

param cte_units
No. of CTE units for LE with CTE, one unit corresponds to 8 s.

param segment
optional repeated capability selector. Default value: S1 (settable in the interface 'Segment')

6.4.1.1.8.99 EnvelopePower

SCPI Command :

```
CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIST:SEGMent<nr>[:SETup]:ENPower
```

class EnvelopePowerCls

EnvelopePower commands group definition. 1 total commands, 0 Subgroups, 1 group commands

get(*segment=Segment.Default*) → float

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIST:SEGMent<nr>
↪[:SETup]:ENPower
value: float = driver.configure.bluetooth.measurement.multiEval.listPy.segment.
↪setup.envelopePower.get(segment = repcap.Segment.Default)
```

Specifies the expected nominal power in the segment. The range of the expected nominal power can be calculated as follows: Range (Expected Nominal Power) = Range (Input Power) + External Attenuation - User Margin

param segment
optional repeated capability selector. Default value: S1 (settable in the interface 'Segment')

return
level: The input power range is stated in the specifications document.

set(*level: float*, *segment=Segment.Default*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIST:SEGMent<nr>
↪[:SETup]:ENPower
driver.configure.bluetooth.measurement.multiEval.listPy.segment.setup.
↪envelopePower.set(level = 1.0, segment = repcap.Segment.Default)
```

Specifies the expected nominal power in the segment. The range of the expected nominal power can be calculated as follows: Range (Expected Nominal Power) = Range (Input Power) + External Attenuation - User Margin

param level
The input power range is stated in the specifications document.

param segment
optional repeated capability selector. Default value: S1 (settable in the interface 'Segment')

6.4.1.1.8.100 Extended

SCPI Command :

```
CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIST:SEGMent<nr>[:SETup]:EXTended
```

class ExtendedCls

Extended commands group definition. 1 total commands, 0 Subgroups, 1 group commands

class ExtendedStruct

Structure for setting input parameters. Contains optional setting parameters. Fields:

- Burst_Type: enums.BurstType: BR: Basic Rate EDR: Enhanced Data Rate LE: Low Energy
- Phy: enums.LePhyType: LE1M: LE 1 Msymbol/s uncoded PHY LE2M: LE 2 Msymbol/s uncoded PHY LELR: LE 1 Msymbol/s long range (LE coded PHY)
- Coding: enums.CodingScheme: Coding S = 8 or S = 2 is relevant only for LE coded PHY.
- Packet_Type: enums.SegmentPacketType: Packet type expected in the segment DH1, DH3, DH5: BR packet E21P, E23P, E25P, E31P, E33P, E35P: 2-DH1, 2-DH3, 2-DH5, 3-DH1, 3-DH3, 3-DH5 EDR packet RFPHytest: LE test packet ADVertiser: LE advertiser RFCTe: LE with CTE test packet

- Pattern_Type: enums.MevPatternType: Payload pattern type expected in the segment. ALL1: 11111111 P11: 10101010 OTHer: any pattern except P11, P44 and ALL1 ALTernating: the periodical change of the pattern P11, P44 P44: 11110000
- Payload_Length: int: Payload length expected in the segment
- No_Of_Off_Slots: int: Number of unused slots between any two occupied slots or slot sequences expected in the segment.
- Segment_Length: int: Number of measured bursts in the segment. The sum of the length of all active segments must not exceed 6700 timeslots (1 timeslot = 625 s duration) .
- Meas_On_Exception: bool: Specifies whether the segment results that the CMP180 identifies as faulty or inaccurate are rejected. ON: include the erroneous bursts OFF: exclude the erroneous bursts
- Level: float: Expected nominal power in the segment. The range of the expected nominal power can be calculated as follows: Range (Expected Nominal Power) = Range (Input Power) + External Attenuation - User Margin The input power range is stated in the specifications document.
- Frequency: float: Center frequency for the segment
- Meas_Filter: enums.FilterWidth: Filter bandwidth for the segment NARRow: Narrowband filter WIDE: Wideband filter
- Retrigger: bool: Specifies whether a trigger event is required for the segment or not. The setting is ignored for the first segment of a measurement. OFF: measure the segment without retrigger ON: trigger event required
- Cte_Units: int: Optional setting parameter. No. of CTE units for LE with CTE, one unit corresponds to 8 s.
- Cte_Type: enums.CteType: Optional setting parameter. CTE slot type for LE with CTE AOA: CTE type angle of arrival, 2 s slot AOD1: CTE type angle of departure, 1 s slot AOD2: CTE type angle of departure, 2 s slot

get(*segment=Segment.Default*) → ExtendedStruct

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIST:SEGMent<nr>
↪[:SETup]:EXTended
value: ExtendedStruct = driver.configure.bluetooth.measurement.multiEval.listPy.
↪segment.setup.extended.get(segment = repcap.Segment.Default)
```

Defines the segment length, the signal properties including Bluetooth version 5.0 and higher, and the analyzer settings for a selected segment. In general, this command must be sent for all segments to be measured.

param segment
optional repeated capability selector. Default value: S1 (settable in the interface 'Segment')

return
structure: for return value, see the help for ExtendedStruct structure arguments.

set(*structure: ExtendedStruct*, *segment=Segment.Default*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIST:SEGMent<nr>
↪[:SETup]:EXTended
structure = driver.configure.bluetooth.measurement.multiEval.listPy.segment.
↪setup.extended.ExtendedStruct()
structure.Burst_Type: enums.BurstType = enums.BurstType.BR
structure.Phy: enums.LePhyType = enums.LePhyType.LE1M
```

(continues on next page)

(continued from previous page)

```
structure.Coding: enums.CodingScheme = enums.CodingScheme.S2
structure.Packet_Type: enums.SegmentPacketType = enums.SegmentPacketType.
↪ADVertiser
structure.Pattern_Type: enums.MevPatternType = enums.MevPatternType.ALL1
structure.Payload_Length: int = 1
structure.No_Of_Off_Slots: int = 1
structure.Segment_Length: int = 1
structure.Meas_On_Exception: bool = False
structure.Level: float = 1.0
structure.Frequency: float = 1.0
structure.Meas_Filter: enums.FilterWidth = enums.FilterWidth.NARRow
structure.Retrigger: bool = False
structure.Cte_Units: int = 1
structure.Cte_Type: enums.CteType = enums.CteType.AOA
driver.configure.bluetooth.measurement.multiEval.listPy.segment.setup.extended.
↪set(structure, segment = repcap.Segment.Default)
```

Defines the segment length, the signal properties including Bluetooth version 5.0 and higher, and the analyzer settings for a selected segment. In general, this command must be sent for all segments to be measured.

param structure
for set value, see the help for ExtendedStruct structure arguments.

param segment
optional repeated capability selector. Default value: S1 (settable in the interface 'Segment')

6.4.1.1.8.101 FilterPy

SCPI Command :

```
CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIST:SEGMent<nr>[:SETup]:FILTer
```

class FilterPyCls

FilterPy commands group definition. 1 total commands, 0 Subgroups, 1 group commands

get(*segment=Segment.Default*) → FilterWidth

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIST:SEGMent<nr>
↪[:SETup]:FILTer
value: enums.FilterWidth = driver.configure.bluetooth.measurement.multiEval.
↪listPy.segment.setup.filterPy.get(segment = repcap.Segment.Default)
```

Specifies the measurement filter bandwidth for the segment.

param segment
optional repeated capability selector. Default value: S1 (settable in the interface 'Segment')

return
meas_filter: NARRow: Narrowband filter WIDE: Wideband filter

set(*meas_filter: FilterWidth*, *segment=Segment.Default*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIST:SEGMent<nr>
↪[:SETup]:FILTer
driver.configure.bluetooth.measurement.multiEval.listPy.segment.setup.filterPy.
↪set(meas_filter = enums.FilterWidth.NARRow, segment = repcap.Segment.Default)
```

Specifies the measurement filter bandwidth for the segment.

param meas_filter
NARRow: Narrowband filter WIDE: Wideband filter

param segment
optional repeated capability selector. Default value: S1 (settable in the interface 'Segment')

6.4.1.1.8.102 Frequency

SCPI Command :

```
CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIST:SEGMent<nr>[:SETup]:FREQuency
```

class FrequencyCls

Frequency commands group definition. 1 total commands, 0 Subgroups, 1 group commands

get(*segment=Segment.Default*) → float

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIST:SEGMent<nr>
↪[:SETup]:FREQuency
value: float = driver.configure.bluetooth.measurement.multiEval.listPy.segment.
↪setup.frequency.get(segment = repcap.Segment.Default)
```

Specifies the center frequency of the signal expected in the segment.

param segment
optional repeated capability selector. Default value: S1 (settable in the interface 'Segment')

return
frequency: No help available

set(*frequency: float*, *segment=Segment.Default*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIST:SEGMent<nr>
↪[:SETup]:FREQuency
driver.configure.bluetooth.measurement.multiEval.listPy.segment.setup.frequency.
↪set(frequency = 1.0, segment = repcap.Segment.Default)
```

Specifies the center frequency of the signal expected in the segment.

param frequency
No help available

param segment
optional repeated capability selector. Default value: S1 (settable in the interface 'Segment')

6.4.1.1.8.103 MoException

SCPI Command :

```
CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIST:SEGMent<nr>
↪[:SETup]:MOEXception
```

class MoExceptionCls

MoException commands group definition. 1 total commands, 0 Subgroups, 1 group commands

get(*segment=Segment.Default*) → bool

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIST:SEGMent<nr>
↪[:SETup]:MOEXception
value: bool = driver.configure.bluetooth.measurement.multiEval.listPy.segment.
↪setup.moException.get(segment = repcap.Segment.Default)
```

Specifies whether the segment results that the CMP180 identifies as faulty or inaccurate are rejected.

param segment
optional repeated capability selector. Default value: S1 (settable in the interface 'Segment')

return
meas_on_exception: ON: include the erroneous bursts OFF: exclude the erroneous bursts

set(*meas_on_exception: bool*, *segment=Segment.Default*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIST:SEGMent<nr>
↪[:SETup]:MOEXception
driver.configure.bluetooth.measurement.multiEval.listPy.segment.setup.
↪moException.set(meas_on_exception = False, segment = repcap.Segment.Default)
```

Specifies whether the segment results that the CMP180 identifies as faulty or inaccurate are rejected.

param meas_on_exception
ON: include the erroneous bursts OFF: exclude the erroneous bursts

param segment
optional repeated capability selector. Default value: S1 (settable in the interface 'Segment')

6.4.1.1.8.104 Oslots

SCPI Command :

```
CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIST:SEGMent<nr>[:SETup]:OSLots
```

class OslotsCls

Oslots commands group definition. 1 total commands, 0 Subgroups, 1 group commands

get(*segment=Segment.Default*) → int

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIST:SEGMent<nr>
↪[:SETup]:OSLots
value: int = driver.configure.bluetooth.measurement.multiEval.listPy.segment.
↪setup.oslots.get(segment = repcap.Segment.Default)
```

Specifies the number of unused slots between any two occupied slots or slot sequences expected in the segment.

param segment
optional repeated capability selector. Default value: S1 (settable in the interface 'Segment')

return
no_of_off_slots: No help available

set(*no_of_off_slots: int*, *segment=Segment.Default*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIST:SEGMent<nr>
↪[:SETup]:OSLots
driver.configure.bluetooth.measurement.multiEval.listPy.segment.setup.oslots.
↪set(no_of_off_slots = 1, segment = repcap.Segment.Default)
```

Specifies the number of unused slots between any two occupied slots or slot sequences expected in the segment.

param no_of_off_slots
No help available

param segment
optional repeated capability selector. Default value: S1 (settable in the interface 'Segment')

6.4.1.1.8.105 Pattern

SCPI Command :

```
CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIST:SEGMent<nr>[:SETup]:PATTern
```

class PatternCls

Pattern commands group definition. 1 total commands, 0 Subgroups, 1 group commands

get(*segment=Segment.Default*) → MevPatternType

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIST:SEGMent<nr>
↪[:SETup]:PATTern
value: enums.MevPatternType = driver.configure.bluetooth.measurement.multiEval.
↪listPy.segment.setup.pattern.get(segment = repcap.Segment.Default)
```

Specifies the payload pattern type expected in the segment.

param segment
optional repeated capability selector. Default value: S1 (settable in the interface 'Segment')

return
pattern_type: ALL1: 11111111 P11: 10101010 OTHer: any pattern except P11, P44, ALL1 ALTernating: the periodical change between the pattern P11 and P44 P44: 11110000

set(*pattern_type: MevPatternType*, *segment=Segment.Default*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIST:SEGMent<nr>
↪[:SETup]:PATTern
driver.configure.bluetooth.measurement.multiEval.listPy.segment.setup.pattern.
↪set(pattern_type = enums.MevPatternType.ALL1, segment = repcap.Segment.
↪Default)
```

Specifies the payload pattern type expected in the segment.

param pattern_type
ALL1: 11111111 P11: 10101010 OTHer: any pattern except P11, P44, ALL1 ALTernating: the periodical change between the pattern P11 and P44 P44: 11110000

param segment
optional repeated capability selector. Default value: S1 (settable in the interface 'Segment')

6.4.1.1.8.106 Phy

SCPI Command :

```
CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIST:SEGMent<nr>[:SETup]:PHY
```

class PhyCls

Phy commands group definition. 1 total commands, 0 Subgroups, 1 group commands

get(*segment=Segment.Default*) → LePhyType

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIST:SEGMent<nr>
↪[:SETup]:PHY
value: enums.LePhyType = driver.configure.bluetooth.measurement.multiEval.
↪listPy.segment.setup.phy.get(segment = repcap.Segment.Default)
```

Defines the physical layer (PHY) for segment.

param segment
optional repeated capability selector. Default value: S1 (settable in the interface 'Segment')

return
le_phy_type: LE1M: LE 1 Msymbol/s uncoded PHY LE2M: LE 2 Msymbol/s uncoded PHY LELR: LE 1 Msymbol/s long range (LE coded PHY)

set(*le_phy_type: LePhyType*, *segment=Segment.Default*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIST:SEGMent<nr>
↪[:SETup]:PHY
driver.configure.bluetooth.measurement.multiEval.listPy.segment.setup.phy.
↪set(le_phy_type = enums.LePhyType.LE1M, segment = repcap.Segment.Default)
```

Defines the physical layer (PHY) for segment.

param le_phy_type
LE1M: LE 1 Msymbol/s uncoded PHY LE2M: LE 2 Msymbol/s uncoded PHY LELR: LE 1 Msymbol/s long range (LE coded PHY)

param segment
optional repeated capability selector. Default value: S1 (settable in the interface 'Segment')

6.4.1.1.8.107 Plength

SCPI Command :

```
CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIST:SEGMent<nr>[:SETup]:PLENgth
```

class PlengthCls

Plength commands group definition. 1 total commands, 0 Subgroups, 1 group commands

get(*segment=Segment.Default*) → int

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIST:SEGMent<nr>
↪[:SETup]:PLENgth
value: int = driver.configure.bluetooth.measurement.multiEval.listPy.segment.
↪setup.plength.get(segment = repcap.Segment.Default)
```

Specifies the payload length expected in the segment.

param segment
optional repeated capability selector. Default value: S1 (settable in the interface 'Segment')

return
payload_length: No help available

set(*payload_length: int*, *segment=Segment.Default*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIST:SEGMent<nr>
↪[:SETup]:PLENgth
driver.configure.bluetooth.measurement.multiEval.listPy.segment.setup.plength.
↪set(payload_length = 1, segment = repcap.Segment.Default)
```

Specifies the payload length expected in the segment.

param payload_length
No help available

param segment
optional repeated capability selector. Default value: S1 (settable in the interface 'Segment')

6.4.1.1.8.108 Ptype

SCPI Command :

```
CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIST:SEGMent<nr>[:SETup]:PTYPe
```

class PtypeCls

Ptype commands group definition. 1 total commands, 0 Subgroups, 1 group commands

get(*segment=Segment.Default*) → SegmentPacketType

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIST:SEGMent<nr>
↪[:SETup]:PTYPe
value: enums.SegmentPacketType = driver.configure.bluetooth.measurement.
↪multiEval.listPy.segment.setup.ptype.get(segment = repcap.Segment.Default)
```

Specifies the packet type expected in the segment.

param segment
optional repeated capability selector. Default value: S1 (settable in the interface 'Segment')

return
packet_type: DH1, DH3, DH5: BR packet E21P, E23P, E25P, E31P, E33P, E35P: 2-DH1, 2-DH3, 2-DH5, 3-DH1, 3-DH3, 3-DH5 EDR packet RFPHytest: LE test packet ADVertiser: LE advertiser RFCTe: LE with CTE test packet

set(*packet_type: SegmentPacketType*, *segment=Segment.Default*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIST:SEGMent<nr>
↪[:SETup]:PTYPe
driver.configure.bluetooth.measurement.multiEval.listPy.segment.setup.ptype.
↪set(packet_type = enums.SegmentPacketType.ADVertiser, segment = repcap.
↪Segment.Default)
```

Specifies the packet type expected in the segment.

param packet_type
DH1, DH3, DH5: BR packet E21P, E23P, E25P, E31P, E33P, E35P: 2-DH1, 2-DH3, 2-DH5, 3-DH1, 3-DH3, 3-DH5 EDR packet RFPHytest: LE test packet ADVertiser: LE advertiser RFCTe: LE with CTE test packet

param segment
optional repeated capability selector. Default value: S1 (settable in the interface 'Segment')

6.4.1.1.8.109 Qhsl

SCPI Command :

```
CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIST:SEGMent<nr>[:SETup]:QHSL
```

class QhslCls

Qhsl commands group definition. 2 total commands, 1 Subgroups, 1 group commands

class QhslStruct

Structure for setting input parameters. Contains optional setting parameters. Fields:

- Burst_Type: enums.BurstType: No parameter help available
- Phy: enums.LePhyType: No parameter help available
- Coding: enums.CodingScheme: No parameter help available
- Packet_Type: enums.SegmentPacketType: No parameter help available
- Pattern_Type: enums.MevPatternType: No parameter help available
- Payload_Length: int: No parameter help available
- No_Of_Off_Slots: int: No parameter help available
- Segment_Length: int: No parameter help available
- Meas_On_Exception: bool: No parameter help available
- Level: float: No parameter help available
- Frequency: float: No parameter help available
- Meas_Filter: enums.FilterWidth: No parameter help available
- Retrigger: bool: No parameter help available
- Cte_Units: int: No parameter help available
- Cte_Type: enums.CteType: No parameter help available

get(*segment*=*Segment.Default*) → QhslStruct

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIST:SEGMent<nr>
↪[:SETup]:QHSL
value: QhslStruct = driver.configure.bluetooth.measurement.multiEval.listPy.
↪segment.setup.qhsl.get(segment = repcap.Segment.Default)
```

No command help available

param segment
optional repeated capability selector. Default value: S1 (settable in the interface 'Segment')

return
structure: for return value, see the help for QhslStruct structure arguments.

set(*structure: QhslStruct*, *segment*=*Segment.Default*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIST:SEGMent<nr>
↪[:SETup]:QHSL
structure = driver.configure.bluetooth.measurement.multiEval.listPy.segment.
↪setup.qhsl.QhslStruct()
structure.Burst_Type: enums.BurstType = enums.BurstType.BR
structure.Phy: enums.LePhyType = enums.LePhyType.LE1M
structure.Coding: enums.CodingScheme = enums.CodingScheme.S2
structure.Packet_Type: enums.SegmentPacketType = enums.SegmentPacketType.
↪ADVertiser
structure.Pattern_Type: enums.MevPatternType = enums.MevPatternType.ALL1
structure.Payload_Length: int = 1
structure.No_Of_Off_Slots: int = 1
structure.Segment_Length: int = 1
structure.Meas_On_Exception: bool = False
structure.Level: float = 1.0
structure.Frequency: float = 1.0
structure.Meas_Filter: enums.FilterWidth = enums.FilterWidth.NARRow
structure.Retrigger: bool = False
structure.Cte_Units: int = 1
structure.Cte_Type: enums.CteType = enums.CteType.AOA
driver.configure.bluetooth.measurement.multiEval.listPy.segment.setup.qhsl.
↪set(structure, segment = repcap.Segment.Default)
```

No command help available

param structure
for set value, see the help for QhslStruct structure arguments.

param segment
optional repeated capability selector. Default value: S1 (settable in the interface 'Segment')

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.configure.bluetooth.measurement.multiEval.listPy.segment.setup.qhsl.
↪clone()
```

Subgroups

6.4.1.1.8.110 Phy

SCPI Command :

```
CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIST:SEGMent<nr>[:SETup]:QHSL:PHY
```

class PhyCls

Phy commands group definition. 1 total commands, 0 Subgroups, 1 group commands

get(*segment=Segment.Default*) → LePhyType

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIST:SEGMent<nr>
↪[:SETup]:QHSL:PHY
value: enums.LePhyType = driver.configure.bluetooth.measurement.multiEval.
↪listPy.segment.setup.qhsl.phy.get(segment = repcap.Segment.Default)
```

No command help available

param segment
optional repeated capability selector. Default value: S1 (settable in the interface 'Segment')

return
le_phy_type: No help available

set(*le_phy_type: LePhyType*, *segment=Segment.Default*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIST:SEGMent<nr>
↪[:SETup]:QHSL:PHY
driver.configure.bluetooth.measurement.multiEval.listPy.segment.setup.qhsl.phy.
↪set(le_phy_type = enums.LePhyType.LE1M, segment = repcap.Segment.Default)
```

No command help available

param le_phy_type
No help available

param segment
optional repeated capability selector. Default value: S1 (settable in the interface 'Segment')

6.4.1.1.8.111 Rtrigger

SCPI Command :

```
CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIST:SEGMent<nr>[:SETup]:RTRigger
```

class RtriggerCls

Rtrigger commands group definition. 1 total commands, 0 Subgroups, 1 group commands

get(*segment=Segment.Default*) → bool

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIST:SEGMent<nr>
↪[:SETup]:RTRigger
value: bool = driver.configure.bluetooth.measurement.multiEval.listPy.segment.
↪setup.rtrigger.get(segment = repcap.Segment.Default)
```

Specifies whether a trigger event is required for the segment or not. The setting is ignored for the first segment of a measurement.

param segment
optional repeated capability selector. Default value: S1 (settable in the interface 'Segment')

return
retrigger: OFF: measure the segment without retrigger ON: trigger event required

set(*retrigger: bool*, *segment=Segment.Default*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIST:SEGMent<nr>
↪[:SETup]:RTRigger
driver.configure.bluetooth.measurement.multiEval.listPy.segment.setup.rtrigger.
↪set(retrigger = False, segment = repcap.Segment.Default)
```

Specifies whether a trigger event is required for the segment or not. The setting is ignored for the first segment of a measurement.

param retrigger
OFF: measure the segment without retrigger ON: trigger event required

param segment
optional repeated capability selector. Default value: S1 (settable in the interface 'Segment')

6.4.1.1.8.112 SingleCmw

class SingleCmwCls

SingleCmw commands group definition. 1 total commands, 1 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.configure.bluetooth.measurement.multiEval.listPy.segment.setup.singleCmw.
↪clone()
```

Subgroups

6.4.1.1.8.113 Connector

SCPI Command :

```
CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIST:SEGMent<nr>
↪[:SETup]:CMWS:CONNector
```

class ConnectorCls

Connector commands group definition. 1 total commands, 0 Subgroups, 1 group commands

get(*segment=Segment.Default*) → CmwSingleConnector

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIST:SEGMent<nr>
↪[:SETup]:CMWS:CONNector
value: enums.CmwSingleConnector = driver.configure.bluetooth.measurement.
↪multiEval.listPy.segment.setup.singleCmw.connector.get(segment = repcap.
↪Segment.Default)
```

No command help available

param segment
optional repeated capability selector. Default value: S1 (settable in the interface 'Segment')

return
cmws_connector: No help available

set(*cmws_connector: CmwSingleConnector*, *segment=Segment.Default*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIST:SEGMent<nr>
↪[:SETup]:CMWS:CONNector
driver.configure.bluetooth.measurement.multiEval.listPy.segment.setup.singleCmw.
↪connector.set(cmws_connector = enums.CmwSingleConnector.R11, segment = repcap.
↪Segment.Default)
```

No command help available

param cmws_connector
No help available

param segment
optional repeated capability selector. Default value: S1 (settable in the interface 'Segment')

6.4.1.1.8.114 Slength

SCPI Command :

```
CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIST:SEGMent<nr>[:SETup]:SLENgth
```

class SlengthCls

Slength commands group definition. 1 total commands, 0 Subgroups, 1 group commands

get(*segment=Segment.Default*) → int

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIST:SEGMent<nr>
↪[:SETup]:SLENgth
value: int = driver.configure.bluetooth.measurement.multiEval.listPy.segment.
↪setup.slength.get(segment = repcap.Segment.Default)
```

Defines the number of measured bursts in the segment

param segment
optional repeated capability selector. Default value: S1 (settable in the interface 'Segment')

return
segment_length: The sum of the length of all active segments must not exceed 6700 timeslots (1 timeslot = 625 s duration) .

set(*segment_length: int*, *segment=Segment.Default*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIST:SEGMent<nr>
↪[:SETup]:SLENgth
driver.configure.bluetooth.measurement.multiEval.listPy.segment.setup.slength.
↪set(segment_length = 1, segment = repcap.Segment.Default)
```

Defines the number of measured bursts in the segment

param segment_length
The sum of the length of all active segments must not exceed 6700 timeslots (1 timeslot = 625 s duration) .

param segment
optional repeated capability selector. Default value: S1 (settable in the interface 'Segment')

6.4.1.1.8.115 SingleCmw

SCPI Command :

```
CONFigure:BLUetooth:MEASurement<instance>:MEValuation:LIST:CMWS:CMODe
```

class SingleCmwCls

SingleCmw commands group definition. 1 total commands, 0 Subgroups, 1 group commands

get_cmode() → ParameterSetMode

```
# SCPI: CONFigure:BLUetooth:MEASurement<instance>:MEValuation:LIST:CMWS:CMODe
value: enums.ParameterSetMode = driver.configure.bluetooth.measurement.
↪multiEval.listPy.singleCmw.get_cmode()
```

No command help available

return
connector_mode: No help available

set_cmode(*connector_mode: ParameterSetMode*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<instance>:MEValuation:LIST:CMWS:CMODe
driver.configure.bluetooth.measurement.multiEval.listPy.singleCmw.set_
↪cmode(connector_mode = enums.ParameterSetMode.GLOBal)
```

No command help available

param connector_mode
No help available

6.4.1.1.8.116 LowEnergy

class LowEnergyCls

LowEnergy commands group definition. 3 total commands, 3 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.configure.bluetooth.measurement.multiEval.lowEnergy.clone()
```

Subgroups

6.4.1.1.8.117 Le1M

class Le1MCls

Le1M commands group definition. 1 total commands, 1 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.configure.bluetooth.measurement.multiEval.lowEnergy.le1M.clone()
```

Subgroups

6.4.1.1.8.118 FilterPy

SCPI Command :

```
CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LENergy[:LE1M]:FILTer:BWIDth
```

class FilterPyCls

FilterPy commands group definition. 1 total commands, 0 Subgroups, 1 group commands

get_bandwidth() → FilterWidth

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:MEValuation:LENergy[:LE1M]:FILTer:BWIDth
value: enums.FilterWidth = driver.configure.bluetooth.measurement.multiEval.
↪lowEnergy.le1M.filterPy.get_bandwidth()
```

Selects the filter bandwidth. Commands for uncoded LE 1M PHY (..:LE1M..) , LE 2M PHY (..:LE2M..) , and LE coded PHY (.. :LRANge..) are available.

return
filter_bandwidth: NARRow: Narrowband filter WIDE: Wideband filter

set_bandwidth(*filter_bandwidth: FilterWidth*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:MEValuation:LENergy[:LE1M]:FILTer:BWIDth
driver.configure.bluetooth.measurement.multiEval.lowEnergy.le1M.filterPy.set_
↪bandwidth(filter_bandwidth = enums.FilterWidth.NARRow)
```

Selects the filter bandwidth. Commands for uncoded LE 1M PHY (..:LE1M..) , LE 2M PHY (..:LE2M..) , and LE coded PHY (.. :LRANge..) are available.

param filter_bandwidth
NARRow: Narrowband filter WIDE: Wideband filter

6.4.1.1.8.119 Le2M

class Le2MCls

Le2M commands group definition. 1 total commands, 1 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.configure.bluetooth.measurement.multiEval.lowEnergy.le2M.clone()
```

Subgroups

6.4.1.1.8.120 FilterPy

SCPI Command :

```
CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LENergy:LE2M:FILTer:BWIDth
```

class FilterPyCls

FilterPy commands group definition. 1 total commands, 0 Subgroups, 1 group commands

get_bandwidth() → FilterWidth

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:MEValuation:LENergy:LE2M:FILTer:BWIDth
value: enums.FilterWidth = driver.configure.bluetooth.measurement.multiEval.
↪lowEnergy.le2M.filterPy.get_bandwidth()
```

Selects the filter bandwidth. Commands for uncoded LE 1M PHY (..:LE1M..) , LE 2M PHY (..:LE2M..) , and LE coded PHY (.. :LRANge..) are available.

return
filter_bandwidth: NARRow: Narrowband filter WIDE: Wideband filter

set_bandwidth(*filter_bandwidth: FilterWidth*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:MEValuation:LENergy:LE2M:FILTer:BWIDth
driver.configure.bluetooth.measurement.multiEval.lowEnergy.le2M.filterPy.set_
↪bandwidth(filter_bandwidth = enums.FilterWidth.NARRow)
```

Selects the filter bandwidth. Commands for uncoded LE 1M PHY (..:LE1M..) , LE 2M PHY (..:LE2M..) , and LE coded PHY (.. :LRANge..) are available.

param filter_bandwidth
NARRow: Narrowband filter WIDE: Wideband filter

6.4.1.1.8.121 Lrange

class LrangeCls

Lrange commands group definition. 1 total commands, 1 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.configure.bluetooth.measurement.multiEval.lowEnergy.lrange.clone()
```

Subgroups

6.4.1.1.8.122 FilterPy

SCPI Command :

```
CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LENergy:LRANge:FILTer:BWIDth
```

class FilterPyCls

FilterPy commands group definition. 1 total commands, 0 Subgroups, 1 group commands

get_bandwidth() → FilterWidth

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:MEValuation:LENergy:LRANge:FILTer:BWIDth
value: enums.FilterWidth = driver.configure.bluetooth.measurement.multiEval.
↪lowEnergy.lrange.filterPy.get_bandwidth()
```

Selects the filter bandwidth. Commands for uncoded LE 1M PHY (..:LE1M..) , LE 2M PHY (..:LE2M..) , and LE coded PHY (.. :LRANge..) are available.

return
filter_bandwidth: NARRow: Narrowband filter WIDE: Wideband filter

set_bandwidth(*filter_bandwidth: FilterWidth*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:MEValuation:LENergy:LRANge:FILTer:BWIDth
driver.configure.bluetooth.measurement.multiEval.lowEnergy.lrange.filterPy.set_
↪bandwidth(filter_bandwidth = enums.FilterWidth.NARRow)
```

Selects the filter bandwidth. Commands for uncoded LE 1M PHY (..:LE1M..) , LE 2M PHY (..:LE2M..) , and LE coded PHY (.. :LRANge..) are available.

param filter_bandwidth
NARRow: Narrowband filter WIDE: Wideband filter

6.4.1.1.8.123 Malgorithm

SCPI Commands :

```
CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:MALGorithm:LENergy
CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:MALGorithm:BRATe
```

class MalgorithmCls

Malgorithm commands group definition. 2 total commands, 0 Subgroups, 2 group commands

get_brate() → PatternIndependent

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:MALGorithm:BRATe
value: enums.PatternIndependent = driver.configure.bluetooth.measurement.
↪multiEval.malgorithm.get_brate()
```

No command help available

return
pattern_independent: No help available

get_low_energy() → PatternIndependent

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:MALGorithm:LENergy
value: enums.PatternIndependent = driver.configure.bluetooth.measurement.
↪multiEval.malgorithm.get_low_energy()
```

No command help available

return
pattern_independent: No help available

set_brate(*pattern_independent: PatternIndependent*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:MALGorithm:BRATe
driver.configure.bluetooth.measurement.multiEval.malgorithm.set_brate(pattern_
↪independent = enums.PatternIndependent.PINDependent)
```

No command help available

param pattern_independent
No help available

set_low_energy(*pattern_independent: PatternIndependent*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:MALGorithm:LENergy
driver.configure.bluetooth.measurement.multiEval.malgorithm.set_low_
↪energy(pattern_independent = enums.PatternIndependent.PINDependent)
```

No command help available

param pattern_independent
No help available

6.4.1.1.8.124 Measurement

SCPI Command :

```
CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:MEASurement:MECount
```

class MeasurementCls

Measurement commands group definition. 1 total commands, 0 Subgroups, 1 group commands

get_me_count() → int

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:MEValuation:MEASurement:MECount
value: int = driver.configure.bluetooth.measurement.multiEval.measurement.get_
↪me_count()
```

No command help available

return
max_error_count: No help available

set_me_count(*max_error_count: int*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:MEValuation:MEASurement:MECount
driver.configure.bluetooth.measurement.multiEval.measurement.set_me_count(max_
↪error_count = 1)
```

No command help available

param max_error_count
No help available

6.4.1.1.8.125 Qhsl

class QhslCls

Qhsl commands group definition. 1 total commands, 1 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.configure.bluetooth.measurement.multiEval.qhsl.clone()
```

Subgroups

6.4.1.1.8.126 FilterPy

SCPI Command :

```
CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:QHSL:FILTer:BWIDth
```

class FilterPyCls

FilterPy commands group definition. 1 total commands, 0 Subgroups, 1 group commands

get_bandwidth() → FilterWidth

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:QHSL:FILTer:BWIDth
value: enums.FilterWidth = driver.configure.bluetooth.measurement.multiEval.
↪qhsl.filterPy.get_bandwidth()
```

No command help available

return
filter_bandwidth: No help available

set_bandwidth(*filter_bandwidth: FilterWidth*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:QHSL:FILTer:BWIDth
driver.configure.bluetooth.measurement.multiEval.qhsl.filterPy.set_
↪bandwidth(filter_bandwidth = enums.FilterWidth.NARRow)
```

No command help available

param filter_bandwidth
No help available

6.4.1.1.8.127 Result

SCPI Commands :

```
CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:RESult:SPOWer
CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:RESult:PENCoding
CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:RESult:FRANge
CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:RESult:SGACp
CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:RESult:SOBW
CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:RESult:SACP
CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:RESult[:ALL]
CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:RESult:PSCalar
CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:RESult:IQABsolute
CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:RESult:IQERror
CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:RESult:IQDiff
CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:RESult:PVTime
CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:RESult:DEVMagnitude
CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:RESult:PDIFference
CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:RESult:MSCalar
```

(continues on next page)

(continued from previous page)

```
CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:RESult:FDEViation
CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:RESult:PVSLot
```

class ResultCls

Result commands group definition. 17 total commands, 0 Subgroups, 17 group commands

class AllStruct

Structure for setting input parameters. Contains optional set arguments. Fields:

- Devm: bool: Differential error vector magnitude (only for EDR) ON: Evaluate the results. OFF: Do not evaluate results.
- Phase_Diff: bool: Phase difference (only for EDR)
- Mod_Scalars: bool: Statistical modulation results
- Iq_Absolute: bool: I/Q constellation absolute (only for EDR)
- Iq_Differential: bool: I/Q constellation differential (only for EDR)
- Iq_Error: bool: I/Q constellation error (only for EDR)
- Freq_Dev: bool: Frequency deviation (only for BR and LE)
- Power_Vs_Time: bool: Power vs time
- Power_Scalars: bool: Statistical power results
- Spectrum_Obw: bool: Spectrum 20 dB bandwidth (only for BR)
- Spectrum_Acp: bool: Spectrum ACP (only for BR and LE)
- Spectrum_Gat_Acp: bool: Spectrum gated ACP (only for EDR)
- Spec_Freq_Range: bool: Optional setting parameter. Spectrum frequency range (only for BR)
- Phase_Encoding: bool: No parameter help available
- Power_Vs_Slot: bool: Optional setting parameter. Power versus slot (only for LE with CTE)

get_all() → AllStruct

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:RESult[:ALL]
value: AllStruct = driver.configure.bluetooth.measurement.multiEval.result.get_
↪all()
```

Enables or disables the evaluation of results in the multi-evaluation measurement. This command combines all other CONFigure:BLUetooth:MEAS<i>:MEValuation:RESult… commands.

return
structure: for return value, see the help for AllStruct structure arguments.

get_dev_magnitude() → bool

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:MEValuation:RESult:DEVMagnitude
value: bool = driver.configure.bluetooth.measurement.multiEval.result.get_dev_
↪magnitude()
```

Enables or disables the evaluation of results in the multi-evaluation measurement. The last mnemonic denotes the measurement type: Statistical modulation results, statistical power results, power vs time results, power vs slot results (LE with CTE) , DEVM (EDR) , phase difference (EDR) , I/Q constellation

absolute (EDR) , I/Q constellation differential (EDR) , I/Q constellation error (EDR) , frequency deviation (BR, LE) , frequency range results (BR) , spectrum 20 dB bandwidth (BR) , spectrum ACP (BR, LE) , spectrum gated ACP (EDR) . Use method RsCMPX_BluetoothMeas.Configure.Bluetooth. Measurement.MultiEval.Result.all to enable/disable all result types. Tip: Use READ...? queries to retrieve results for disabled measurements.

return
enable: ON: Evaluate the results. OFF: Do not evaluate results.

get_fdeviation() → bool

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:RESult:FDEViation
value: bool = driver.configure.bluetooth.measurement.multiEval.result.get_
↪fdeviation()
```

Enables or disables the evaluation of results in the multi-evaluation measurement. The last mnemonic denotes the measurement type: Statistical modulation results, statistical power results, power vs time results, power vs slot results (LE with CTE) , DEVM (EDR) , phase difference (EDR) , I/Q constellation absolute (EDR) , I/Q constellation differential (EDR) , I/Q constellation error (EDR) , frequency deviation (BR, LE) , frequency range results (BR) , spectrum 20 dB bandwidth (BR) , spectrum ACP (BR, LE) , spectrum gated ACP (EDR) . Use method RsCMPX_BluetoothMeas.Configure.Bluetooth. Measurement.MultiEval.Result.all to enable/disable all result types. Tip: Use READ...? queries to retrieve results for disabled measurements.

return
enable: ON: Evaluate the results. OFF: Do not evaluate results.

get_frange() → bool

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:RESult:FRANge
value: bool = driver.configure.bluetooth.measurement.multiEval.result.get_
↪frange()
```

Enables or disables the evaluation of results in the multi-evaluation measurement. The last mnemonic denotes the measurement type: Statistical modulation results, statistical power results, power vs time results, power vs slot results (LE with CTE) , DEVM (EDR) , phase difference (EDR) , I/Q constellation absolute (EDR) , I/Q constellation differential (EDR) , I/Q constellation error (EDR) , frequency deviation (BR, LE) , frequency range results (BR) , spectrum 20 dB bandwidth (BR) , spectrum ACP (BR, LE) , spectrum gated ACP (EDR) . Use method RsCMPX_BluetoothMeas.Configure.Bluetooth. Measurement.MultiEval.Result.all to enable/disable all result types. Tip: Use READ...? queries to retrieve results for disabled measurements.

return
enable: ON: Evaluate the results. OFF: Do not evaluate results.

get_iq_absolute() → bool

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:RESult:IQABsolute
value: bool = driver.configure.bluetooth.measurement.multiEval.result.get_iq_
↪absolute()
```

Enables or disables the evaluation of results in the multi-evaluation measurement. The last mnemonic denotes the measurement type: Statistical modulation results, statistical power results, power vs time results, power vs slot results (LE with CTE) , DEVM (EDR) , phase difference (EDR) , I/Q constellation absolute (EDR) , I/Q constellation differential (EDR) , I/Q constellation error (EDR) , frequency deviation (BR, LE) , frequency range results (BR) , spectrum 20 dB bandwidth (BR) , spectrum ACP (BR,

LE) , spectrum gated ACP (EDR) . Use method RsCMPX_BluetoothMeas.Configure.Bluetooth. Measurement.MultiEval.Result.all to enable/disable all result types. Tip: Use READ…? queries to retrieve results for disabled measurements.

return
enable: ON: Evaluate the results. OFF: Do not evaluate results.

get_iq_difference() → bool

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:RESult:IQDiff
value: bool = driver.configure.bluetooth.measurement.multiEval.result.get_iq_
↪difference()
```

Enables or disables the evaluation of results in the multi-evaluation measurement. The last mnemonic denotes the measurement type: Statistical modulation results, statistical power results, power vs time results, power vs slot results (LE with CTE) , DEVM (EDR) , phase difference (EDR) , I/Q constellation absolute (EDR) , I/Q constellation differential (EDR) , I/Q constellation error (EDR) , frequency deviation (BR, LE) , frequency range results (BR) , spectrum 20 dB bandwidth (BR) , spectrum ACP (BR, LE) , spectrum gated ACP (EDR) . Use method RsCMPX_BluetoothMeas.Configure.Bluetooth. Measurement.MultiEval.Result.all to enable/disable all result types. Tip: Use READ…? queries to retrieve results for disabled measurements.

return
enable: ON: Evaluate the results. OFF: Do not evaluate results.

get_iq_error() → bool

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:RESult:IQERror
value: bool = driver.configure.bluetooth.measurement.multiEval.result.get_iq_
↪error()
```

Enables or disables the evaluation of results in the multi-evaluation measurement. The last mnemonic denotes the measurement type: Statistical modulation results, statistical power results, power vs time results, power vs slot results (LE with CTE) , DEVM (EDR) , phase difference (EDR) , I/Q constellation absolute (EDR) , I/Q constellation differential (EDR) , I/Q constellation error (EDR) , frequency deviation (BR, LE) , frequency range results (BR) , spectrum 20 dB bandwidth (BR) , spectrum ACP (BR, LE) , spectrum gated ACP (EDR) . Use method RsCMPX_BluetoothMeas.Configure.Bluetooth. Measurement.MultiEval.Result.all to enable/disable all result types. Tip: Use READ…? queries to retrieve results for disabled measurements.

return
enable: ON: Evaluate the results. OFF: Do not evaluate results.

get_mscalar() → bool

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:RESult:MSCalar
value: bool = driver.configure.bluetooth.measurement.multiEval.result.get_
↪mscalar()
```

Enables or disables the evaluation of results in the multi-evaluation measurement. The last mnemonic denotes the measurement type: Statistical modulation results, statistical power results, power vs time results, power vs slot results (LE with CTE) , DEVM (EDR) , phase difference (EDR) , I/Q constellation absolute (EDR) , I/Q constellation differential (EDR) , I/Q constellation error (EDR) , frequency deviation (BR, LE) , frequency range results (BR) , spectrum 20 dB bandwidth (BR) , spectrum ACP (BR, LE) , spectrum gated ACP (EDR) . Use method RsCMPX_BluetoothMeas.Configure.Bluetooth. Measurement.MultiEval.Result.all to enable/disable all result types. Tip: Use READ…? queries to retrieve results for disabled measurements.

return
enable: ON: Evaluate the results. OFF: Do not evaluate results.

get_pdifference() → bool

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:RESult:PDIFference
value: bool = driver.configure.bluetooth.measurement.multiEval.result.get_
↪pdifference()
```

Enables or disables the evaluation of results in the multi-evaluation measurement. The last mnemonic denotes the measurement type: Statistical modulation results, statistical power results, power vs time results, power vs slot results (LE with CTE) , DEVM (EDR) , phase difference (EDR) , I/Q constellation absolute (EDR) , I/Q constellation differential (EDR) , I/Q constellation error (EDR) , frequency deviation (BR, LE) , frequency range results (BR) , spectrum 20 dB bandwidth (BR) , spectrum ACP (BR, LE) , spectrum gated ACP (EDR) . Use method RsCMPX_BluetoothMeas.Configure.Bluetooth. Measurement.MultiEval.Result.all to enable/disable all result types. Tip: Use READ...? queries to retrieve results for disabled measurements.

return
enable: ON: Evaluate the results. OFF: Do not evaluate results.

get_pencoding() → bool

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:RESult:PENCoding
value: bool = driver.configure.bluetooth.measurement.multiEval.result.get_
↪pencoding()
```

No command help available

return
enable: No help available

get_power_vs_time() → bool

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:RESult:PVTime
value: bool = driver.configure.bluetooth.measurement.multiEval.result.get_power_
↪vs_time()
```

Enables or disables the evaluation of results in the multi-evaluation measurement. The last mnemonic denotes the measurement type: Statistical modulation results, statistical power results, power vs time results, power vs slot results (LE with CTE) , DEVM (EDR) , phase difference (EDR) , I/Q constellation absolute (EDR) , I/Q constellation differential (EDR) , I/Q constellation error (EDR) , frequency deviation (BR, LE) , frequency range results (BR) , spectrum 20 dB bandwidth (BR) , spectrum ACP (BR, LE) , spectrum gated ACP (EDR) . Use method RsCMPX_BluetoothMeas.Configure.Bluetooth. Measurement.MultiEval.Result.all to enable/disable all result types. Tip: Use READ...? queries to retrieve results for disabled measurements.

return
enable: ON: Evaluate the results. OFF: Do not evaluate results.

get_pscalar() → bool

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:RESult:PSCalar
value: bool = driver.configure.bluetooth.measurement.multiEval.result.get_
↪pscalar()
```

Enables or disables the evaluation of results in the multi-evaluation measurement. The last mnemonic denotes the measurement type: Statistical modulation results, statistical power results, power vs time results, power vs slot results (LE with CTE) , DEVM (EDR) , phase difference (EDR) , I/Q constellation absolute (EDR) , I/Q constellation differential (EDR) , I/Q constellation error (EDR) , frequency deviation (BR, LE) , frequency range results (BR) , spectrum 20 dB bandwidth (BR) , spectrum ACP (BR, LE) , spectrum gated ACP (EDR) . Use method RsCMPX_BluetoothMeas.Configure.Bluetooth. Measurement.MultiEval.Result.all to enable/disable all result types. Tip: Use READ...? queries to retrieve results for disabled measurements.

return
enable: ON: Evaluate the results. OFF: Do not evaluate results.

get_pv_slot() → bool

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:RESult:PVSLot
value: bool = driver.configure.bluetooth.measurement.multiEval.result.get_pv_
↪slot()
```

Enables or disables the evaluation of results in the multi-evaluation measurement. The last mnemonic denotes the measurement type: Statistical modulation results, statistical power results, power vs time results, power vs slot results (LE with CTE) , DEVM (EDR) , phase difference (EDR) , I/Q constellation absolute (EDR) , I/Q constellation differential (EDR) , I/Q constellation error (EDR) , frequency deviation (BR, LE) , frequency range results (BR) , spectrum 20 dB bandwidth (BR) , spectrum ACP (BR, LE) , spectrum gated ACP (EDR) . Use method RsCMPX_BluetoothMeas.Configure.Bluetooth. Measurement.MultiEval.Result.all to enable/disable all result types. Tip: Use READ...? queries to retrieve results for disabled measurements.

return
enable: ON: Evaluate the results. OFF: Do not evaluate results.

get_sacp() → bool

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:RESult:SACP
value: bool = driver.configure.bluetooth.measurement.multiEval.result.get_sacp()
```

Enables or disables the evaluation of results in the multi-evaluation measurement. The last mnemonic denotes the measurement type: Statistical modulation results, statistical power results, power vs time results, power vs slot results (LE with CTE) , DEVM (EDR) , phase difference (EDR) , I/Q constellation absolute (EDR) , I/Q constellation differential (EDR) , I/Q constellation error (EDR) , frequency deviation (BR, LE) , frequency range results (BR) , spectrum 20 dB bandwidth (BR) , spectrum ACP (BR, LE) , spectrum gated ACP (EDR) . Use method RsCMPX_BluetoothMeas.Configure.Bluetooth. Measurement.MultiEval.Result.all to enable/disable all result types. Tip: Use READ...? queries to retrieve results for disabled measurements.

return
enable: ON: Evaluate the results. OFF: Do not evaluate results.

get_sgacp() → bool

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:RESult:SGACp
value: bool = driver.configure.bluetooth.measurement.multiEval.result.get_
↪sgacp()
```

Enables or disables the evaluation of results in the multi-evaluation measurement. The last mnemonic denotes the measurement type: Statistical modulation results, statistical power results, power vs time results, power vs slot results (LE with CTE) , DEVM (EDR) , phase difference (EDR) , I/Q constellation

absolute (EDR) , I/Q constellation differential (EDR) , I/Q constellation error (EDR) , frequency deviation (BR, LE) , frequency range results (BR) , spectrum 20 dB bandwidth (BR) , spectrum ACP (BR, LE) , spectrum gated ACP (EDR) . Use method RsCMPX_BluetoothMeas.Configure.Bluetooth. Measurement.MultiEval.Result.all to enable/disable all result types. Tip: Use READ...? queries to retrieve results for disabled measurements.

return
enable: ON: Evaluate the results. OFF: Do not evaluate results.

get_so_bw() → bool

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:RESult:SOBW
value: bool = driver.configure.bluetooth.measurement.multiEval.result.get_so_
↪bw()
```

Enables or disables the evaluation of results in the multi-evaluation measurement. The last mnemonic denotes the measurement type: Statistical modulation results, statistical power results, power vs time results, power vs slot results (LE with CTE) , DEVM (EDR) , phase difference (EDR) , I/Q constellation absolute (EDR) , I/Q constellation differential (EDR) , I/Q constellation error (EDR) , frequency deviation (BR, LE) , frequency range results (BR) , spectrum 20 dB bandwidth (BR) , spectrum ACP (BR, LE) , spectrum gated ACP (EDR) . Use method RsCMPX_BluetoothMeas.Configure.Bluetooth. Measurement.MultiEval.Result.all to enable/disable all result types. Tip: Use READ...? queries to retrieve results for disabled measurements.

return
enable: ON: Evaluate the results. OFF: Do not evaluate results.

get_spower() → bool

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:RESult:SPOWer
value: bool = driver.configure.bluetooth.measurement.multiEval.result.get_
↪spower()
```

No command help available

return
enable: No help available

set_all(*value: AllStruct*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:RESult[:ALL]
structure = driver.configure.bluetooth.measurement.multiEval.result.AllStruct()
structure.Devm: bool = False
structure.Phase_Diff: bool = False
structure.Mod_Scalars: bool = False
structure.Iq_Absolute: bool = False
structure.Iq_Differential: bool = False
structure.Iq_Error: bool = False
structure.Freq_Dev: bool = False
structure.Power_Vs_Time: bool = False
structure.Power_Scalars: bool = False
structure.Spectrum_Obw: bool = False
structure.Spectrum_Acp: bool = False
structure.Spectrum_Gat_Acp: bool = False
structure.Spec_Freq_Range: bool = False
structure.Phase_Encoding: bool = False
```

(continues on next page)

(continued from previous page)

```
structure.Power_Vs_Slot: bool = False
driver.configure.bluetooth.measurement.multiEval.result.set_all(value =␣
↪structure)
```

Enables or disables the evaluation of results in the multi-evaluation measurement. This command combines all other CONFigure:BLUetooth:MEAS<i>:MEValuation:RESult... commands.

param value
see the help for AllStruct structure arguments.

set_dev_magnitude(*enable: bool*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:MEValuation:RESult:DEVMagnitude
driver.configure.bluetooth.measurement.multiEval.result.set_dev_
↪magnitude(enable = False)
```

Enables or disables the evaluation of results in the multi-evaluation measurement. The last mnemonic denotes the measurement type: Statistical modulation results, statistical power results, power vs time results, power vs slot results (LE with CTE) , DEVM (EDR) , phase difference (EDR) , I/Q constellation absolute (EDR) , I/Q constellation differential (EDR) , I/Q constellation error (EDR) , frequency deviation (BR, LE) , frequency range results (BR) , spectrum 20 dB bandwidth (BR) , spectrum ACP (BR, LE) , spectrum gated ACP (EDR) . Use method RsCMPX_BluetoothMeas.Configure.Bluetooth. Measurement.MultiEval.Result.all to enable/disable all result types. Tip: Use READ...? queries to retrieve results for disabled measurements.

param enable
ON: Evaluate the results. OFF: Do not evaluate results.

set_fdeviation(*enable: bool*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:RESult:FDEViation
driver.configure.bluetooth.measurement.multiEval.result.set_fdeviation(enable =␣
↪False)
```

Enables or disables the evaluation of results in the multi-evaluation measurement. The last mnemonic denotes the measurement type: Statistical modulation results, statistical power results, power vs time results, power vs slot results (LE with CTE) , DEVM (EDR) , phase difference (EDR) , I/Q constellation absolute (EDR) , I/Q constellation differential (EDR) , I/Q constellation error (EDR) , frequency deviation (BR, LE) , frequency range results (BR) , spectrum 20 dB bandwidth (BR) , spectrum ACP (BR, LE) , spectrum gated ACP (EDR) . Use method RsCMPX_BluetoothMeas.Configure.Bluetooth. Measurement.MultiEval.Result.all to enable/disable all result types. Tip: Use READ...? queries to retrieve results for disabled measurements.

param enable
ON: Evaluate the results. OFF: Do not evaluate results.

set_frange(*enable: bool*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:RESult:FRANge
driver.configure.bluetooth.measurement.multiEval.result.set_frange(enable =␣
↪False)
```

Enables or disables the evaluation of results in the multi-evaluation measurement. The last mnemonic denotes the measurement type: Statistical modulation results, statistical power results, power vs time results, power vs slot results (LE with CTE) , DEVM (EDR) , phase difference (EDR) , I/Q constellation

absolute (EDR) , I/Q constellation differential (EDR) , I/Q constellation error (EDR) , frequency deviation (BR, LE) , frequency range results (BR) , spectrum 20 dB bandwidth (BR) , spectrum ACP (BR, LE) , spectrum gated ACP (EDR) . Use method RsCMPX_BluetoothMeas.Configure.Bluetooth. Measurement.MultiEval.Result.all to enable/disable all result types. Tip: Use READ…? queries to retrieve results for disabled measurements.

param enable
ON: Evaluate the results. OFF: Do not evaluate results.

set_iq_absolute(*enable: bool*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:RESult:IQABsolute
driver.configure.bluetooth.measurement.multiEval.result.set_iq_absolute(enable␣
↪= False)
```

Enables or disables the evaluation of results in the multi-evaluation measurement. The last mnemonic denotes the measurement type: Statistical modulation results, statistical power results, power vs time results, power vs slot results (LE with CTE) , DEVM (EDR) , phase difference (EDR) , I/Q constellation absolute (EDR) , I/Q constellation differential (EDR) , I/Q constellation error (EDR) , frequency deviation (BR, LE) , frequency range results (BR) , spectrum 20 dB bandwidth (BR) , spectrum ACP (BR, LE) , spectrum gated ACP (EDR) . Use method RsCMPX_BluetoothMeas.Configure.Bluetooth. Measurement.MultiEval.Result.all to enable/disable all result types. Tip: Use READ…? queries to retrieve results for disabled measurements.

param enable
ON: Evaluate the results. OFF: Do not evaluate results.

set_iq_difference(*enable: bool*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:RESult:IQDiff
driver.configure.bluetooth.measurement.multiEval.result.set_iq_
↪difference(enable = False)
```

Enables or disables the evaluation of results in the multi-evaluation measurement. The last mnemonic denotes the measurement type: Statistical modulation results, statistical power results, power vs time results, power vs slot results (LE with CTE) , DEVM (EDR) , phase difference (EDR) , I/Q constellation absolute (EDR) , I/Q constellation differential (EDR) , I/Q constellation error (EDR) , frequency deviation (BR, LE) , frequency range results (BR) , spectrum 20 dB bandwidth (BR) , spectrum ACP (BR, LE) , spectrum gated ACP (EDR) . Use method RsCMPX_BluetoothMeas.Configure.Bluetooth. Measurement.MultiEval.Result.all to enable/disable all result types. Tip: Use READ…? queries to retrieve results for disabled measurements.

param enable
ON: Evaluate the results. OFF: Do not evaluate results.

set_iq_error(*enable: bool*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:RESult:IQERror
driver.configure.bluetooth.measurement.multiEval.result.set_iq_error(enable =␣
↪False)
```

Enables or disables the evaluation of results in the multi-evaluation measurement. The last mnemonic denotes the measurement type: Statistical modulation results, statistical power results, power vs time results, power vs slot results (LE with CTE) , DEVM (EDR) , phase difference (EDR) , I/Q constellation absolute (EDR) , I/Q constellation differential (EDR) , I/Q constellation error (EDR) , frequency deviation (BR, LE) , frequency range results (BR) , spectrum 20 dB bandwidth (BR) , spectrum ACP (BR,

LE) , spectrum gated ACP (EDR) . Use method RsCMPX_BluetoothMeas.Configure.Bluetooth. Measurement.MultiEval.Result.all to enable/disable all result types. Tip: Use READ…? queries to retrieve results for disabled measurements.

param enable
ON: Evaluate the results. OFF: Do not evaluate results.

set_mscalar(*enable: bool*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:RESult:MSCalar
driver.configure.bluetooth.measurement.multiEval.result.set_mscalar(enable =␣
↪False)
```

Enables or disables the evaluation of results in the multi-evaluation measurement. The last mnemonic denotes the measurement type: Statistical modulation results, statistical power results, power vs time results, power vs slot results (LE with CTE) , DEVM (EDR) , phase difference (EDR) , I/Q constellation absolute (EDR) , I/Q constellation differential (EDR) , I/Q constellation error (EDR) , frequency deviation (BR, LE) , frequency range results (BR) , spectrum 20 dB bandwidth (BR) , spectrum ACP (BR, LE) , spectrum gated ACP (EDR) . Use method RsCMPX_BluetoothMeas.Configure.Bluetooth. Measurement.MultiEval.Result.all to enable/disable all result types. Tip: Use READ…? queries to retrieve results for disabled measurements.

param enable
ON: Evaluate the results. OFF: Do not evaluate results.

set_pdifference(*enable: bool*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:RESult:PDIFference
driver.configure.bluetooth.measurement.multiEval.result.set_pdifference(enable␣
↪= False)
```

Enables or disables the evaluation of results in the multi-evaluation measurement. The last mnemonic denotes the measurement type: Statistical modulation results, statistical power results, power vs time results, power vs slot results (LE with CTE) , DEVM (EDR) , phase difference (EDR) , I/Q constellation absolute (EDR) , I/Q constellation differential (EDR) , I/Q constellation error (EDR) , frequency deviation (BR, LE) , frequency range results (BR) , spectrum 20 dB bandwidth (BR) , spectrum ACP (BR, LE) , spectrum gated ACP (EDR) . Use method RsCMPX_BluetoothMeas.Configure.Bluetooth. Measurement.MultiEval.Result.all to enable/disable all result types. Tip: Use READ…? queries to retrieve results for disabled measurements.

param enable
ON: Evaluate the results. OFF: Do not evaluate results.

set_pencoding(*enable: bool*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:RESult:PENCoding
driver.configure.bluetooth.measurement.multiEval.result.set_pencoding(enable =␣
↪False)
```

No command help available

param enable
No help available

set_power_vs_time(*enable: bool*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:RESult:PVTime
driver.configure.bluetooth.measurement.multiEval.result.set_power_vs_
↪time(enable = False)
```

Enables or disables the evaluation of results in the multi-evaluation measurement. The last mnemonic denotes the measurement type: Statistical modulation results, statistical power results, power vs time results, power vs slot results (LE with CTE) , DEVM (EDR) , phase difference (EDR) , I/Q constellation absolute (EDR) , I/Q constellation differential (EDR) , I/Q constellation error (EDR) , frequency deviation (BR, LE) , frequency range results (BR) , spectrum 20 dB bandwidth (BR) , spectrum ACP (BR, LE) , spectrum gated ACP (EDR) . Use method RsCMPX_BluetoothMeas.Configure.Bluetooth. Measurement.MultiEval.Result.all to enable/disable all result types. Tip: Use READ…? queries to retrieve results for disabled measurements.

param enable
ON: Evaluate the results. OFF: Do not evaluate results.

set_pscalar(*enable: bool*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:RESult:PSCalar
driver.configure.bluetooth.measurement.multiEval.result.set_pscalar(enable =␣
↪False)
```

Enables or disables the evaluation of results in the multi-evaluation measurement. The last mnemonic denotes the measurement type: Statistical modulation results, statistical power results, power vs time results, power vs slot results (LE with CTE) , DEVM (EDR) , phase difference (EDR) , I/Q constellation absolute (EDR) , I/Q constellation differential (EDR) , I/Q constellation error (EDR) , frequency deviation (BR, LE) , frequency range results (BR) , spectrum 20 dB bandwidth (BR) , spectrum ACP (BR, LE) , spectrum gated ACP (EDR) . Use method RsCMPX_BluetoothMeas.Configure.Bluetooth. Measurement.MultiEval.Result.all to enable/disable all result types. Tip: Use READ…? queries to retrieve results for disabled measurements.

param enable
ON: Evaluate the results. OFF: Do not evaluate results.

set_pv_slot(*enable: bool*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:RESult:PVSLot
driver.configure.bluetooth.measurement.multiEval.result.set_pv_slot(enable =␣
↪False)
```

Enables or disables the evaluation of results in the multi-evaluation measurement. The last mnemonic denotes the measurement type: Statistical modulation results, statistical power results, power vs time results, power vs slot results (LE with CTE) , DEVM (EDR) , phase difference (EDR) , I/Q constellation absolute (EDR) , I/Q constellation differential (EDR) , I/Q constellation error (EDR) , frequency deviation (BR, LE) , frequency range results (BR) , spectrum 20 dB bandwidth (BR) , spectrum ACP (BR, LE) , spectrum gated ACP (EDR) . Use method RsCMPX_BluetoothMeas.Configure.Bluetooth. Measurement.MultiEval.Result.all to enable/disable all result types. Tip: Use READ…? queries to retrieve results for disabled measurements.

param enable
ON: Evaluate the results. OFF: Do not evaluate results.

set_sacp(*enable: bool*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:RESult:SACP
driver.configure.bluetooth.measurement.multiEval.result.set_sacp(enable = False)
```

Enables or disables the evaluation of results in the multi-evaluation measurement. The last mnemonic denotes the measurement type: Statistical modulation results, statistical power results, power vs time results, power vs slot results (LE with CTE) , DEVM (EDR) , phase difference (EDR) , I/Q constellation absolute (EDR) , I/Q constellation differential (EDR) , I/Q constellation error (EDR) , frequency deviation (BR, LE) , frequency range results (BR) , spectrum 20 dB bandwidth (BR) , spectrum ACP (BR, LE) , spectrum gated ACP (EDR) . Use method RsCMPX_BluetoothMeas.Configure.Bluetooth. Measurement.MultiEval.Result.all to enable/disable all result types. Tip: Use READ...? queries to retrieve results for disabled measurements.

param enable
ON: Evaluate the results. OFF: Do not evaluate results.

set_sgacp(*enable: bool*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:RESult:SGACp
driver.configure.bluetooth.measurement.multiEval.result.set_sgacp(enable =␣
↪False)
```

Enables or disables the evaluation of results in the multi-evaluation measurement. The last mnemonic denotes the measurement type: Statistical modulation results, statistical power results, power vs time results, power vs slot results (LE with CTE) , DEVM (EDR) , phase difference (EDR) , I/Q constellation absolute (EDR) , I/Q constellation differential (EDR) , I/Q constellation error (EDR) , frequency deviation (BR, LE) , frequency range results (BR) , spectrum 20 dB bandwidth (BR) , spectrum ACP (BR, LE) , spectrum gated ACP (EDR) . Use method RsCMPX_BluetoothMeas.Configure.Bluetooth. Measurement.MultiEval.Result.all to enable/disable all result types. Tip: Use READ...? queries to retrieve results for disabled measurements.

param enable
ON: Evaluate the results. OFF: Do not evaluate results.

set_so_bw(*enable: bool*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:RESult:SOBW
driver.configure.bluetooth.measurement.multiEval.result.set_so_bw(enable =␣
↪False)
```

Enables or disables the evaluation of results in the multi-evaluation measurement. The last mnemonic denotes the measurement type: Statistical modulation results, statistical power results, power vs time results, power vs slot results (LE with CTE) , DEVM (EDR) , phase difference (EDR) , I/Q constellation absolute (EDR) , I/Q constellation differential (EDR) , I/Q constellation error (EDR) , frequency deviation (BR, LE) , frequency range results (BR) , spectrum 20 dB bandwidth (BR) , spectrum ACP (BR, LE) , spectrum gated ACP (EDR) . Use method RsCMPX_BluetoothMeas.Configure.Bluetooth. Measurement.MultiEval.Result.all to enable/disable all result types. Tip: Use READ...? queries to retrieve results for disabled measurements.

param enable
ON: Evaluate the results. OFF: Do not evaluate results.

set_spower(*enable: bool*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:RESult:SPOWer
driver.configure.bluetooth.measurement.multiEval.result.set_spower(enable =␣
↪False)
```

No command help available

param enable
No help available

6.4.1.1.8.128 Sacp

class SacpCls

Sacp commands group definition. 4 total commands, 3 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.configure.bluetooth.measurement.multiEval.sacp.clone()
```

Subgroups

6.4.1.1.8.129 Brate

class BrateCls

Brate commands group definition. 1 total commands, 1 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.configure.bluetooth.measurement.multiEval.sacp.brate.clone()
```

Subgroups

6.4.1.1.8.130 Measurement

SCPI Command :

```
CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:SACP:BRATe:MEASurement:MODE
```

class MeasurementCls

Measurement commands group definition. 1 total commands, 0 Subgroups, 1 group commands

get_mode() → BrEdrChannelsRange

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:MEValuation:SACP:BRATe:MEASurement:MODE
value: enums.BrEdrChannelsRange = driver.configure.bluetooth.measurement.
↪multiEval.sacp.brate.measurement.get_mode()
```

Selects the measured ACP channel range for BR or EDR packets. The ACP can be measured over the expected transmit channel +/- 10 channels (21 channels in total) or over the entire Bluetooth regulatory range (79 channels) .

return

meas_mode: Measure 79 or 21 channels

set_mode(*meas_mode: BrEdrChannelsRange*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:MEValuation:SACP:BRATe:MEASurement:MODE
driver.configure.bluetooth.measurement.multiEval.sacp.brate.measurement.set_
↪mode(meas_mode = enums.BrEdrChannelsRange.CH21)
```

Selects the measured ACP channel range for BR or EDR packets. The ACP can be measured over the expected transmit channel +/- 10 channels (21 channels in total) or over the entire Bluetooth regulatory range (79 channels) .

param meas_mode
Measure 79 or 21 channels

6.4.1.1.8.131 LowEnergy

class LowEnergyCls

LowEnergy commands group definition. 2 total commands, 2 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.configure.bluetooth.measurement.multiEval.sacp.lowEnergy.clone()
```

Subgroups

6.4.1.1.8.132 Le1M

class Le1MCls

Le1M commands group definition. 1 total commands, 1 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.configure.bluetooth.measurement.multiEval.sacp.lowEnergy.le1M.clone()
```

Subgroups

6.4.1.1.8.133 Measurement

SCPI Command :

```
CONFigure:BLUetooth:MEASurement<Instance>
↪:MEValuation:SACP:LENergy[:LE1M]:MEASurement:MODE
```

class MeasurementCls

Measurement commands group definition. 1 total commands, 0 Subgroups, 1 group commands

get_mode() → LeChannelsRange

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:MEValuation:SACP:LENergy[:LE1M]:MEASurement:MODE
value: enums.LeChannelsRange = driver.configure.bluetooth.measurement.multiEval.
↪sacp.lowEnergy.le1M.measurement.get_mode()
```

Specifies the channel range for ACP measurements. It can be selected to cover either the full LE frequency band (forty 2 MHz channels) or only the adjacency of the current LE channel (ten 2 MHz channels) . The commands for LE 1M PHY (...:LE1M. ..) and LE 2M PHY (...:LE2M...) are available. Note: Although LE channels are 2 MHz wide, the channel width in ACP measurements is always 1 MHz ('half-channel') .

return
meas_mode: CH10: ACP +/- 5 Channels - covers the current and its 10 adjacent 2 MHz LE channels (5 to the left, 5 to the right) . The CMP180 measures the 1 MHz channels centered at fTX – 10 MHz, ..., fTX + 10 MHz. CH40: LE All Channels - covers all 40 LE channels. The CMP180 measures the 81 half-channels centered at 2401 MHz, 2402 MHz, ..., 2481 MHz.

set_mode(*meas_mode: LeChannelsRange*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:MEValuation:SACP:LENergy[:LE1M]:MEASurement:MODE
driver.configure.bluetooth.measurement.multiEval.sacp.lowEnergy.le1M.
↪measurement.set_mode(meas_mode = enums.LeChannelsRange.CH10)
```

Specifies the channel range for ACP measurements. It can be selected to cover either the full LE frequency band (forty 2 MHz channels) or only the adjacency of the current LE channel (ten 2 MHz channels) . The commands for LE 1M PHY (...:LE1M. ..) and LE 2M PHY (...:LE2M...) are available. Note: Although LE channels are 2 MHz wide, the channel width in ACP measurements is always 1 MHz ('half-channel') .

param meas_mode
CH10: ACP +/- 5 Channels - covers the current and its 10 adjacent 2 MHz LE channels (5 to the left, 5 to the right) . The CMP180 measures the 1 MHz channels centered at fTX – 10 MHz, ..., fTX + 10 MHz. CH40: LE All Channels - covers all 40 LE channels. The CMP180 measures the 81 half-channels centered at 2401 MHz, 2402 MHz, ..., 2481 MHz.

6.4.1.1.8.134 Le2M

class Le2MCls

Le2M commands group definition. 1 total commands, 1 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.configure.bluetooth.measurement.multiEval.sacp.lowEnergy.le2M.clone()
```

Subgroups

6.4.1.1.8.135 Measurement

SCPI Command :

```
CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:SACP:LENergy:LE2M:MEASurement:MODE
```

class MeasurementCls

Measurement commands group definition. 1 total commands, 0 Subgroups, 1 group commands

get_mode() → LeChannelsRange

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:MEValuation:SACP:LENergy:LE2M:MEASurement:MODE
value: enums.LeChannelsRange = driver.configure.bluetooth.measurement.multiEval.
↪sacp.lowEnergy.le2M.measurement.get_mode()
```

Specifies the channel range for ACP measurements. It can be selected to cover either the full LE frequency band (forty 2 MHz channels) or only the adjacency of the current LE channel (ten 2 MHz channels) . The commands for LE 1M PHY (...:LE1M. ..) and LE 2M PHY (...:LE2M...) are available. Note: Although LE channels are 2 MHz wide, the channel width in ACP measurements is always 1 MHz ('half-channel') .

return
meas_mode: CH10: ACP +/- 5 Channels - covers the current and its 10 adjacent 2 MHz LE channels (5 to the left, 5 to the right) . The CMP180 measures the 1 MHz channels centered at fTX – 10 MHz, ..., fTX + 10 MHz. CH40: LE All Channels - covers all 40 LE channels. The CMP180 measures the 81 half-channels centered at 2401 MHz, 2402 MHz, ..., 2481 MHz.

set_mode(*meas_mode: LeChannelsRange*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:MEValuation:SACP:LENergy:LE2M:MEASurement:MODE
driver.configure.bluetooth.measurement.multiEval.sacp.lowEnergy.le2M.
↪measurement.set_mode(meas_mode = enums.LeChannelsRange.CH10)
```

Specifies the channel range for ACP measurements. It can be selected to cover either the full LE frequency band (forty 2 MHz channels) or only the adjacency of the current LE channel (ten 2 MHz channels) . The commands for LE 1M PHY (...:LE1M. ..) and LE 2M PHY (...:LE2M...) are available. Note: Although LE channels are 2 MHz wide, the channel width in ACP measurements is always 1 MHz ('half-channel') .

param meas_mode
CH10: ACP +/- 5 Channels - covers the current and its 10 adjacent 2 MHz LE channels (5 to the left, 5 to the right) . The CMP180 measures the 1 MHz channels centered at fTX – 10 MHz, ..., fTX + 10 MHz. CH40: LE All Channels - covers all 40 LE channels. The CMP180 measures the 81 half-channels centered at 2401 MHz, 2402 MHz, ..., 2481 MHz.

6.4.1.1.8.136 Qhsl

class QhslCls

Qhsl commands group definition. 1 total commands, 1 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.configure.bluetooth.measurement.multiEval.sacp.qhsl.clone()
```

Subgroups

6.4.1.1.8.137 Measurement

SCPI Command :

```
CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:SACP:QHSL:MEASurement:MODE
```

class MeasurementCls

Measurement commands group definition. 1 total commands, 0 Subgroups, 1 group commands

get_mode() → LeChannelsRange

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:MEValuation:SACP:QHSL:MEASurement:MODE
value: enums.LeChannelsRange = driver.configure.bluetooth.measurement.multiEval.
↪sacp.qhsl.measurement.get_mode()
```

No command help available

return
meas_mode: No help available

set_mode(*meas_mode: LeChannelsRange*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:MEValuation:SACP:QHSL:MEASurement:MODE
driver.configure.bluetooth.measurement.multiEval.sacp.qhsl.measurement.set_
↪mode(meas_mode = enums.LeChannelsRange.CH10)
```

No command help available

param meas_mode
No help available

6.4.1.1.8.138 Scount

SCPI Commands :

```
CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:SCOunt:PENCoding
CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:SCOunt:FRANge
CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:SCOunt:SGACp
CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:SCOunt:SOBW
CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:SCOunt:SACP
CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:SCOunt:PVTime
CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:SCOunt:MODulation
```

class ScountCls

Scount commands group definition. 7 total commands, 0 Subgroups, 7 group commands

get_frange() → int

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:SCOunt:FRANge
value: int = driver.configure.bluetooth.measurement.multiEval.scount.get_
↪frange()
```

Specifies the statistic count of the measurement. The statistic count is equal to the number of measurement intervals per single shot. The last mnemonic denotes the measurement type: spectrum frequency range, spectrum ACP and spectrum gated ACP.

return
statistic_count: Statistic count for the measurement

get_modulation() → int

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:SCOunt:MODulation
value: int = driver.configure.bluetooth.measurement.multiEval.scount.get_
↪modulation()
```

Specifies the statistic count of the measurement. The statistic count is equal to the number of measurement intervals per single shot. The last mnemonic denotes the measurement type: statistical modulation, statistical power, and spectrum 20 dB bandwidth (occupied bandwidth) measurement.

return
statistic_count: Number of measurement intervals

get_pencoding() → int

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:SCOunt:PENCoding
value: int = driver.configure.bluetooth.measurement.multiEval.scount.get_
↪pencoding()
```

No command help available

return
statistic_count: No help available

get_power_vs_time() → int

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:SCOunt:PVTime
value: int = driver.configure.bluetooth.measurement.multiEval.scount.get_power_
↪vs_time()
```

Specifies the statistic count of the measurement. The statistic count is equal to the number of measurement intervals per single shot. The last mnemonic denotes the measurement type: statistical modulation, statistical power, and spectrum 20 dB bandwidth (occupied bandwidth) measurement.

return
statistic_count: Number of measurement intervals

get_sacp() → int

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:SCOunt:SACP
value: int = driver.configure.bluetooth.measurement.multiEval.scount.get_sacp()
```

Specifies the statistic count of the measurement. The statistic count is equal to the number of measurement intervals per single shot. The last mnemonic denotes the measurement type: spectrum frequency range, spectrum ACP and spectrum gated ACP.

return
statistic_count: Statistic count for the measurement

get_sgacp() → int

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:SCOunt:SGACp
value: int = driver.configure.bluetooth.measurement.multiEval.scount.get_sgacp()
```

Specifies the statistic count of the measurement. The statistic count is equal to the number of measurement intervals per single shot. The last mnemonic denotes the measurement type: spectrum frequency range, spectrum ACP and spectrum gated ACP.

return
statistic_count: Statistic count for the measurement

get_so_bw() → int

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:SCOunt:SOBW
value: int = driver.configure.bluetooth.measurement.multiEval.scount.get_so_bw()
```

Specifies the statistic count of the measurement. The statistic count is equal to the number of measurement intervals per single shot. The last mnemonic denotes the measurement type: statistical modulation, statistical power, and spectrum 20 dB bandwidth (occupied bandwidth) measurement.

return
statistic_count: Number of measurement intervals

set_frange(*statistic_count: int*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:SCOunt:FRANge
driver.configure.bluetooth.measurement.multiEval.scount.set_frange(statistic_
↪count = 1)
```

Specifies the statistic count of the measurement. The statistic count is equal to the number of measurement intervals per single shot. The last mnemonic denotes the measurement type: spectrum frequency range, spectrum ACP and spectrum gated ACP.

param statistic_count
Statistic count for the measurement

set_modulation(*statistic_count: int*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:SCOunt:MODulation
driver.configure.bluetooth.measurement.multiEval.scount.set_
↪modulation(statistic_count = 1)
```

Specifies the statistic count of the measurement. The statistic count is equal to the number of measurement intervals per single shot. The last mnemonic denotes the measurement type: statistical modulation, statistical power, and spectrum 20 dB bandwidth (occupied bandwidth) measurement.

param statistic_count
Number of measurement intervals

set_pencoding(*statistic_count: int*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:SCOunt:PENCoding
driver.configure.bluetooth.measurement.multiEval.scount.set_pencoding(statistic_
↪count = 1)
```

No command help available

param statistic_count
No help available

set_power_vs_time(*statistic_count: int*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:SCOunt:PVTime
driver.configure.bluetooth.measurement.multiEval.scount.set_power_vs_
↪time(statistic_count = 1)
```

Specifies the statistic count of the measurement. The statistic count is equal to the number of measurement intervals per single shot. The last mnemonic denotes the measurement type: statistical modulation, statistical power, and spectrum 20 dB bandwidth (occupied bandwidth) measurement.

param statistic_count
Number of measurement intervals

set_sacp(*statistic_count: int*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:SCOunt:SACP
driver.configure.bluetooth.measurement.multiEval.scount.set_sacp(statistic_
↪count = 1)
```

Specifies the statistic count of the measurement. The statistic count is equal to the number of measurement intervals per single shot. The last mnemonic denotes the measurement type: spectrum frequency range, spectrum ACP and spectrum gated ACP.

param statistic_count
Statistic count for the measurement

set_sgacp(*statistic_count: int*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:SCOunt:SGACp
driver.configure.bluetooth.measurement.multiEval.scount.set_sgacp(statistic_
↪count = 1)
```

Specifies the statistic count of the measurement. The statistic count is equal to the number of measurement intervals per single shot. The last mnemonic denotes the measurement type: spectrum frequency range, spectrum ACP and spectrum gated ACP.

param statistic_count
Statistic count for the measurement

set_so_bw(*statistic_count: int*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:SCOunt:SOBW
driver.configure.bluetooth.measurement.multiEval.scount.set_so_bw(statistic_
↪count = 1)
```

Specifies the statistic count of the measurement. The statistic count is equal to the number of measurement intervals per single shot. The last mnemonic denotes the measurement type: statistical modulation, statistical power, and spectrum 20 dB bandwidth (occupied bandwidth) measurement.

param statistic_count
Number of measurement intervals

6.4.1.1.8.139 Sgacp

class SgacpCls

Sgacp commands group definition. 1 total commands, 1 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.configure.bluetooth.measurement.multiEval.sgacp.clone()
```

Subgroups

6.4.1.1.8.140 Edrate

class EdrateCls

Edrate commands group definition. 1 total commands, 1 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.configure.bluetooth.measurement.multiEval.sgacp.edrate.clone()
```

Subgroups

6.4.1.1.8.141 Measurement

SCPI Command :

```
CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:SGACp:EDRate:MEASurement:MODE
```

class MeasurementCls

Measurement commands group definition. 1 total commands, 0 Subgroups, 1 group commands

get_mode() → BrEdrChannelsRange

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:MEValuation:SGACp:EDRate:MEASurement:MODE
value: enums.BrEdrChannelsRange = driver.configure.bluetooth.measurement.
↪multiEval.sgacp.edrate.measurement.get_mode()
```

Selects the measured ACP channel range for BR or EDR packets. The ACP can be measured over the expected transmit channel +/- 10 channels (21 channels in total) or over the entire Bluetooth regulatory range (79 channels) .

return
meas_mode: Measure 79 or 21 channels

set_mode(*meas_mode: BrEdrChannelsRange*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:MEValuation:SGACp:EDRate:MEASurement:MODE
driver.configure.bluetooth.measurement.multiEval.sgacp.edrate.measurement.set_
↪mode(meas_mode = enums.BrEdrChannelsRange.CH21)
```

Selects the measured ACP channel range for BR or EDR packets. The ACP can be measured over the expected transmit channel +/- 10 channels (21 channels in total) or over the entire Bluetooth regulatory range (79 channels) .

param meas_mode
Measure 79 or 21 channels

6.4.1.1.8.142 Synchronise

SCPI Command :

```
CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:SYNChronise
```

class SynchroniseCls

Synchronise commands group definition. 1 total commands, 0 Subgroups, 1 group commands

class SynchroniseStruct

Response structure. Fields:

- Min_No_Valid_Bursts: int: No parameter help available
- Syn_Check_Filter: int: No parameter help available

- Max_Invalid_Burst: int: No parameter help available

get() → SynchroniseStruct

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:SYNChronise
value: SynchroniseStruct = driver.configure.bluetooth.measurement.multiEval.
↪synchronise.get()
```

No command help available

return
structure: for return value, see the help for SynchroniseStruct structure arguments.

set(*min_no_valid_bursts: int*, *syn_check_filter: int*, *max_invalid_burst: int*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:SYNChronise
driver.configure.bluetooth.measurement.multiEval.synchronise.set(min_no_valid_
↪bursts = 1, syn_check_filter = 1, max_invalid_burst = 1)
```

No command help available

param min_no_valid_bursts
No help available

param syn_check_filter
No help available

param max_invalid_burst
No help available

6.4.1.1.9 RfSettings

SCPI Commands :

```
CONFigure:BLUetooth:MEASurement<Instance>:RFSettings:EATTenuation
CONFigure:BLUetooth:MEASurement<Instance>:RFSettings:UMARgin
CONFigure:BLUetooth:MEASurement<Instance>:RFSettings:ENPower
CONFigure:BLUetooth:MEASurement<Instance>:RFSettings:FREQuency
CONFigure:BLUetooth:MEASurement<Instance>:RFSettings:RLEVel
CONFigure:BLUetooth:MEASurement<Instance>:RFSettings:LRINterval
```

class RfSettingsCls

RfSettings commands group definition. 18 total commands, 5 Subgroups, 6 group commands

get_eattenuation() → float

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:RFSettings:EATTenuation
value: float = driver.configure.bluetooth.measurement.rfSettings.get_
↪eattenuation()
```

Defines an external attenuation (or gain, if the value is negative) , to be applied to the input connector.

return
external_att: No help available

get_envelope_power() → float

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:RFSettings:ENPower
value: float = driver.configure.bluetooth.measurement.rfSettings.get_envelope_
↪power()
```

Sets the expected nominal power of the measured RF signal.

return
exp_nominal_power: The range of the expected nominal power can be calculated as follows: Range (Expected Nominal Power) = Range (Input Power) + External Attenuation - User Margin The input power range is stated in the specifications document.

get_frequency() → float

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:RFSettings:FREQuency
value: float = driver.configure.bluetooth.measurement.rfSettings.get_frequency()
```

Selects the center frequency of the RF analyzer.

return
analyzer_freq: No help available

get_lr_interval() → float

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:RFSettings:LRINterval
value: float = driver.configure.bluetooth.measurement.rfSettings.get_lr_
↪interval()
```

Defines the measurement interval for level adjustment.

return
lvl_rang_interval: No help available

get_rlevel() → int

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:RFSettings:RLEVel
value: int = driver.configure.bluetooth.measurement.rfSettings.get_rlevel()
```

Queries the reference level of the measured RF signal. The value is calculated as the expected peak power at the output of the DUT: Reference level = Expected Nominal Power + User Margin

return
reference_level: No help available

get_umargin() → float

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:RFSettings:UMARgin
value: float = driver.configure.bluetooth.measurement.rfSettings.get_umargin()
```

Sets the margin that the measurement adds to the expected nominal power to determine the reference power. The reference power minus the external input attenuation must be within the power range of the selected input connector. Refer to the specifications document.

return
user_margin: No help available

set_eattenuation(*external_att: float*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:RFSettings:EATTenuation
driver.configure.bluetooth.measurement.rfSettings.set_eattenuation(external_att␣
↪= 1.0)
```

Defines an external attenuation (or gain, if the value is negative) , to be applied to the input connector.

param external_att
No help available

set_envelope_power(*exp_nominal_power: float*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:RFSettings:ENPower
driver.configure.bluetooth.measurement.rfSettings.set_envelope_power(exp_
↪nominal_power = 1.0)
```

Sets the expected nominal power of the measured RF signal.

param exp_nominal_power
The range of the expected nominal power can be calculated as follows: Range (Expected Nominal Power) = Range (Input Power) + External Attenuation - User Margin
The input power range is stated in the specifications document.

set_frequency(*analyzer_freq: float*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:RFSettings:FREQuency
driver.configure.bluetooth.measurement.rfSettings.set_frequency(analyzer_freq =␣
↪1.0)
```

Selects the center frequency of the RF analyzer.

param analyzer_freq
No help available

set_lr_interval(*lvl_rang_interval: float*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:RFSettings:LRINterval
driver.configure.bluetooth.measurement.rfSettings.set_lr_interval(lvl_rang_
↪interval = 1.0)
```

Defines the measurement interval for level adjustment.

param lvl_rang_interval
No help available

set_umargin(*user_margin: float*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:RFSettings:UMARgin
driver.configure.bluetooth.measurement.rfSettings.set_umargin(user_margin = 1.0)
```

Sets the margin that the measurement adds to the expected nominal power to determine the reference power. The reference power minus the external input attenuation must be within the power range of the selected input connector. Refer to the specifications document.

param user_margin
No help available

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.configure.bluetooth.measurement.rfSettings.clone()
```

Subgroups

6.4.1.1.9.1 Cte

class CteCls

Cte commands group definition. 3 total commands, 1 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.configure.bluetooth.measurement.rfSettings.cte.clone()
```

Subgroups

6.4.1.1.9.2 LowEnergy

SCPI Commands :

```
CONFigure:BLUetooth:MEASurement<Instance>:RFSettings:CTE:LENergy:NANTenna
CONFigure:BLUetooth:MEASurement<Instance>:RFSettings:CTE:LENergy:ROFFset
```

class LowEnergyCls

LowEnergy commands group definition. 3 total commands, 1 Subgroups, 2 group commands

get_nantenna() → int

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:RFSettings:CTE:LENergy:NANTenna
value: int = driver.configure.bluetooth.measurement.rfSettings.cte.lowEnergy.
↪get_nantenna()
```

No command help available

return
nof_antennas: No help available

get_roffset() → float

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:RFSettings:CTE:LENergy:ROFFset
value: float = driver.configure.bluetooth.measurement.rfSettings.cte.lowEnergy.
↪get_roffset()
```

No command help available

return
ant_ref: No help available

set_nantenna(*nof_antennas: int*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:RFSettings:CTE:LENergy:NANTenna
driver.configure.bluetooth.measurement.rfSettings.cte.lowEnergy.set_
↪nantenna(nof_antennas = 1)
```

No command help available

param nof_antennas
No help available

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.configure.bluetooth.measurement.rfSettings.cte.lowEnergy.clone()
```

Subgroups

6.4.1.1.9.3 Aoffset

SCPI Command :

```
CONFigure:BLUetooth:MEASurement<Instance>:RFSettings:CTE:LENergy:AOFFset
```

class AoffsetCls

Aoffset commands group definition. 1 total commands, 0 Subgroups, 1 group commands

class AoffsetStruct

Response structure. Fields:

- Ant_Ref_1: float: No parameter help available
- Ant_Ref_2: float: No parameter help available
- Ant_Ref_3: float: No parameter help available

get() → AoffsetStruct

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:RFSettings:CTE:LENergy:AOFFset
value: AoffsetStruct = driver.configure.bluetooth.measurement.rfSettings.cte.
↪lowEnergy.aoffset.get()
```

No command help available

return
structure: for return value, see the help for AoffsetStruct structure arguments.

set(*ant_ref_1: float*, *ant_ref_2: float*, *ant_ref_3: float*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:RFSettings:CTE:LENergy:AOFFset
driver.configure.bluetooth.measurement.rfSettings.cte.lowEnergy.aoffset.set(ant_
↪ref_1 = 1.0, ant_ref_2 = 1.0, ant_ref_3 = 1.0)
```

No command help available

param ant_ref_1
No help available

param ant_ref_2
No help available

param ant_ref_3
No help available

6.4.1.1.9.4 Dtx

SCPI Commands :

```
CONFigure:BLUetooth:MEASurement<Instance>:RFSettings:DTX:STERror
CONFigure:BLUetooth:MEASurement<Instance>:RFSettings:DTX:FOFFset
CONFigure:BLUetooth:MEASurement<Instance>:RFSettings:DTX:MINDex
CONFigure:BLUetooth:MEASurement<Instance>:RFSettings:DTX
```

class DtxCls

Dtx commands group definition. 4 total commands, 0 Subgroups, 4 group commands

get_foffset() → float

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:RFSettings:DTX:FOFFset
value: float = driver.configure.bluetooth.measurement.rfSettings.dtx.get_
↪foffset()
```

No command help available

return
level: No help available

get_mindex() → float

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:RFSettings:DTX:MINDex
value: float = driver.configure.bluetooth.measurement.rfSettings.dtx.get_
↪mindex()
```

No command help available

return
level: No help available

get_st_error() → LeSymolTimeError

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:RFSettings:DTX:STERror
value: enums.LeSymolTimeError = driver.configure.bluetooth.measurement.
↪rfSettings.dtx.get_st_error()
```

No command help available

return
sym_tim_err: No help available

get_value() → bool

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:RFSettings:DTX
value: bool = driver.configure.bluetooth.measurement.rfSettings.dtx.get_value()
```

No command help available

return
dtx_state: No help available

set_foffset(*level: float*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:RFSettings:DTX:FOFFset
driver.configure.bluetooth.measurement.rfSettings.dtx.set_foffset(level = 1.0)
```

No command help available

param level
No help available

set_mindex(*level: float*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:RFSettings:DTX:MINDex
driver.configure.bluetooth.measurement.rfSettings.dtx.set_mindex(level = 1.0)
```

No command help available

param level
No help available

set_st_error(*sym_tim_err: LeSymolTimeError*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:RFSettings:DTX:STERror
driver.configure.bluetooth.measurement.rfSettings.dtx.set_st_error(sym_tim_err␣
↪= enums.LeSymolTimeError.NEG50)
```

No command help available

param sym_tim_err
No help available

set_value(*dtx_state: bool*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:RFSettings:DTX
driver.configure.bluetooth.measurement.rfSettings.dtx.set_value(dtx_state =␣
↪False)
```

No command help available

param dtx_state
No help available

6.4.1.1.9.5 LrStart

SCPI Command :

```
CONFigure:BLUetooth:MEASurement<Instance>:RFSettings:LRSTart
```

class LrStartCls

LrStart commands group definition. 1 total commands, 0 Subgroups, 1 group commands

set(*opc_timeout_ms: int = -1*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:RFSettings:LRSTart
driver.configure.bluetooth.measurement.rfSettings.lrStart.set()
```

Starts level adjustment.

param opc_timeout_ms
Maximum time to wait in milliseconds, valid only for this call.

6.4.1.1.9.6 Mchannel

SCPI Commands :

```
CONFigure:BLUetooth:MEASurement<Instance>:RFSettings:MCHannel[:CLASsic]
CONFigure:BLUetooth:MEASurement<Instance>:RFSettings:MCHannel:LENergy
```

class MchannelCls

Mchannel commands group definition. 2 total commands, 0 Subgroups, 2 group commands

get_classic() → int

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:RFSettings:MCHannel[:CLASsic]
value: int = driver.configure.bluetooth.measurement.rfSettings.mchannel.get_
↪classic()
```

No command help available

return
measured_channel: No help available

get_low_energy() → int

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:RFSettings:MCHannel:LENergy
value: int = driver.configure.bluetooth.measurement.rfSettings.mchannel.get_low_
↪energy()
```

No command help available

return
measured_channel: No help available

set_classic(*measured_channel: int*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:RFSettings:MCHannel[:CLASsic]
driver.configure.bluetooth.measurement.rfSettings.mchannel.set_classic(measured_
↪channel = 1)
```

No command help available

param measured_channel
No help available

set_low_energy(*measured_channel: int*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:RFSettings:MCHannel:LENergy
driver.configure.bluetooth.measurement.rfSettings.mchannel.set_low_
↪energy(measured_channel = 1)
```

No command help available

param measured_channel
No help available

6.4.1.1.9.7 Mmode

SCPI Command :

```
CONFigure:BLUetooth:MEASurement<Instance>:RFSettings:MMODe
```

class MmodeCls

Mmode commands group definition. 2 total commands, 1 Subgroups, 1 group commands

get_value() → MeasureScope

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:RFSettings:MMODe
value: enums.MeasureScope = driver.configure.bluetooth.measurement.rfSettings.
↪mmode.get_value()
```

No command help available

return
measure_mode: No help available

set_value(*measure_mode: MeasureScope*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:RFSettings:MMODe
driver.configure.bluetooth.measurement.rfSettings.mmode.set_value(measure_mode␣
↪= enums.MeasureScope.ALL)
```

No command help available

param measure_mode
No help available

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.configure.bluetooth.measurement.rfSettings.mmode.clone()
```

Subgroups

6.4.1.1.9.8 Nmode

SCPI Command :

```
CONFigure:BLUetooth:MEASurement<Instance>:RFSettings:MMODe:NMODe:LENergy
```

class NmodeCls

Nmode commands group definition. 1 total commands, 0 Subgroups, 1 group commands

get_low_energy() → MeasureScope

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:RFSettings:MMODe:NMODe:LENergy
value: enums.MeasureScope = driver.configure.bluetooth.measurement.rfSettings.
↪mmode.nmode.get_low_energy()
```

No command help available

return
measure_mode: No help available

set_low_energy(*measure_mode: MeasureScope*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:RFSettings:MMODe:NMODe:LENergy
driver.configure.bluetooth.measurement.rfSettings.mmode.nmode.set_low_
↪energy(measure_mode = enums.MeasureScope.ALL)
```

No command help available

param measure_mode
No help available

6.4.1.1.10 RxQuality

SCPI Commands :

```
CONFigure:BLUetooth:MEASurement<Instance>:RXQuality:DOFFset
CONFigure:BLUetooth:MEASurement<Instance>:RXQuality:SADDress
CONFigure:BLUetooth:MEASurement<Instance>:RXQuality:SATYpe
CONFigure:BLUetooth:MEASurement<Instance>:RXQuality:ADETect
CONFigure:BLUetooth:MEASurement<Instance>:RXQuality:MMODe
CONFigure:BLUetooth:MEASurement<Instance>:RXQuality:GARB
CONFigure:BLUetooth:MEASurement<Instance>:RXQuality:AINDex
```

class RxQualityCls

RxQuality commands group definition. 16 total commands, 5 Subgroups, 7 group commands

get_adetect() → bool

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:RXQuality:ADETect
value: bool = driver.configure.bluetooth.measurement.rxQuality.get_adetect()
```

No command help available

return
addr_auto_user: No help available

get_aindex() → int

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:RXQuality:AINDex
value: int = driver.configure.bluetooth.measurement.rxQuality.get_aindex()
```

Specifies the advertiser channel index to be measured. See also Figure 'RF channel index'.

return
adv_chan_index: No help available

get_doffset() → int

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:RXQuality:DOFFset
value: int = driver.configure.bluetooth.measurement.rxQuality.get_doffset()
```

No command help available

return
delay_offset: No help available

get_garb() → bool

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:RXQuality:GARB
value: bool = driver.configure.bluetooth.measurement.rxQuality.get_garb()
```

No command help available

return
arb_during_tx: No help available

get_mmode() → RxQualityMeasMode

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:RXQuality:MMODe
value: enums.RxQualityMeasMode = driver.configure.bluetooth.measurement.
↪rxQuality.get_mmode()
```

No command help available

return
meas_mode: No help available

get_sa_type() → AddressType

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:RXQuality:SATYpe
value: enums.AddressType = driver.configure.bluetooth.measurement.rxQuality.get_
↪sa_type()
```

No command help available

return
scanner_address_type: No help available

get_saddress() → str

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:RXQuality:SADDress
value: str = driver.configure.bluetooth.measurement.rxQuality.get_saddress()
```

No command help available

return
scanner_address: No help available

set_adetect(*addr_auto_user: bool*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:RXQuality:ADETect
driver.configure.bluetooth.measurement.rxQuality.set_adetect(addr_auto_user =␣
↪False)
```

No command help available

param addr_auto_user
No help available

set_aindex(*adv_chan_index: int*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:RXQuality:AINDex
driver.configure.bluetooth.measurement.rxQuality.set_aindex(adv_chan_index = 1)
```

Specifies the advertiser channel index to be measured. See also Figure 'RF channel index'.

param adv_chan_index
No help available

set_doffset(*delay_offset: int*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:RXQuality:DOFFset
driver.configure.bluetooth.measurement.rxQuality.set_doffset(delay_offset = 1)
```

No command help available

param delay_offset
No help available

set_garb(*arb_during_tx: bool*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:RXQuality:GARB
driver.configure.bluetooth.measurement.rxQuality.set_garb(arb_during_tx = False)
```

No command help available

param arb_during_tx
No help available

set_mmode(*meas_mode: RxQualityMeasMode*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:RXQuality:MMODe
driver.configure.bluetooth.measurement.rxQuality.set_mmode(meas_mode = enums.
↪RxQualityMeasMode.PER)
```

No command help available

param meas_mode
No help available

set_sa_type(*scanner_address_type: AddressType*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:RXQuality:SATYpe
driver.configure.bluetooth.measurement.rxQuality.set_sa_type(scanner_address_
↪type = enums.AddressType.PUBLic)
```

No command help available

param scanner_address_type
No help available

set_saddress(*scanner_address: str*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:RXQuality:SADDress
driver.configure.bluetooth.measurement.rxQuality.set_saddress(scanner_address =␣
↪rawAbc)
```

No command help available

param scanner_address
No help available

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.configure.bluetooth.measurement.rxQuality.clone()
```

Subgroups

6.4.1.1.10.1 Eattenuation

SCPI Command :

```
CONFigure:BLUetooth:MEASurement<Instance>:RXQuality:EATTenuation:OUTPut
```

class EattenuationCls

Eattenuation commands group definition. 1 total commands, 0 Subgroups, 1 group commands

get_output() → float

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:RXQuality:EATTenuation:OUTPut
value: float = driver.configure.bluetooth.measurement.rxQuality.eattenuation.
↪get_output()
```

No command help available

return
atten: No help available

set_output(*atten: float*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:RXQuality:EATTenuation:OUTPut
driver.configure.bluetooth.measurement.rxQuality.eattenuation.set_output(atten
↪= 1.0)
```

No command help available

param atten
No help available

6.4.1.1.10.2 Per

SCPI Commands :

```
CONFigure:BLUetooth:MEASurement<Instance>:RXQuality:PER:LEVel
CONFigure:BLUetooth:MEASurement<Instance>:RXQuality:PER:TXPackets
```

class PerCls

Per commands group definition. 2 total commands, 0 Subgroups, 2 group commands

get_level() → float

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:RXQuality:PER:LEVel
value: float = driver.configure.bluetooth.measurement.rxQuality.per.get_level()
```

No command help available

return
level: No help available

get_tx_packets() → int

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:RXQuality:PER:TXPackets
value: int = driver.configure.bluetooth.measurement.rxQuality.per.get_tx_
↪packets()
```

No command help available

return
packets_to_send: No help available

set_level(*level: float*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:RXQuality:PER:LEVel
driver.configure.bluetooth.measurement.rxQuality.per.set_level(level = 1.0)
```

No command help available

param level
No help available

set_tx_packets(*packets_to_send: int*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:RXQuality:PER:TXPackets
driver.configure.bluetooth.measurement.rxQuality.per.set_tx_packets(packets_to_
↪send = 1)
```

No command help available

param packets_to_send
No help available

6.4.1.1.10.3 Route

SCPI Command :

```
CONFigure:BLUetooth:MEASurement<Instance>:RXQuality:ROUTe
```

class RouteCls

Route commands group definition. 2 total commands, 1 Subgroups, 1 group commands

class RouteStruct

Response structure. Fields:

- Tx_Connector: enums.TxConnector: No parameter help available
- Rf_Converter: enums.TxConverter: No parameter help available

get() → RouteStruct

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:RXQuality:ROUTe
value: RouteStruct = driver.configure.bluetooth.measurement.rxQuality.route.
↪get()
```

No command help available

return
structure: for return value, see the help for RouteStruct structure arguments.

set(*tx_connector: TxConnector*, *rf_converter: TxConverter*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:RXQuality:ROUTe
driver.configure.bluetooth.measurement.rxQuality.route.set(tx_connector = enums.
↪TxConnector.I120, rf_converter = enums.TxConverter.ITX1)
```

No command help available

param tx_connector
No help available

param rf_converter
No help available

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.configure.bluetooth.measurement.rxQuality.route.clone()
```

Subgroups

6.4.1.1.10.4 Usage

class UsageCls

Usage commands group definition. 1 total commands, 1 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.configure.bluetooth.measurement.rxQuality.route.usage.clone()
```

Subgroups

6.4.1.1.10.5 All

SCPI Command :

```
CONFigure:BLUetooth:MEASurement<Instance>:RXQuality:ROUTe:USAGe:ALL
```

class AllCls

All commands group definition. 1 total commands, 0 Subgroups, 1 group commands

get(*tx_connector_bench: TxConnectorBench*) → List[bool]

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:RXQuality:ROUTe:USAGe:ALL
value: List[bool] = driver.configure.bluetooth.measurement.rxQuality.route.
↪usage.all.get(tx_connector_bench = enums.TxConnectorBench.R118)
```

No command help available

param tx_connector_bench
No help available

return
usage: No help available

set(*tx_connector_bench: TxConnectorBench*, *usage: List[bool]*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:RXQuality:ROUTe:USAGe:ALL
driver.configure.bluetooth.measurement.rxQuality.route.usage.all.set(tx_
↪connector_bench = enums.TxConnectorBench.R118, usage = [True, False, True])
```

No command help available

param tx_connector_bench
No help available

param usage
No help available

6.4.1.1.10.6 Sensitivity

SCPI Commands :

```
CONFigure:BLUetooth:MEASurement<Instance>:RXQuality:SENSitivity:STARtlevel
CONFigure:BLUetooth:MEASurement<Instance>:RXQuality:SENSitivity:STEPsize
CONFigure:BLUetooth:MEASurement<Instance>:RXQuality:SENSitivity:RETRy
```

class SensitivityCls

Sensitivity commands group definition. 3 total commands, 0 Subgroups, 3 group commands

get_retry() → int

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:RXQuality:SENSitivity:RETRy
value: int = driver.configure.bluetooth.measurement.rxQuality.sensitivity.get_
↪retry()
```

No command help available

return
retry_count: No help available

get_start_level() → float

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:RXQuality:SENSitivity:STARtlevel
value: float = driver.configure.bluetooth.measurement.rxQuality.sensitivity.get_
↪start_level()
```

No command help available

return
start_level: No help available

get_stepsize() → float

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:RXQuality:SENSitivity:STEPsize
value: float = driver.configure.bluetooth.measurement.rxQuality.sensitivity.get_
↪stepsize()
```

No command help available

return
stepsize: No help available

set_retry(*retry_count: int*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:RXQuality:SENSitivity:RETRy
driver.configure.bluetooth.measurement.rxQuality.sensitivity.set_retry(retry_
↪count = 1)
```

No command help available

param retry_count
No help available

set_start_level(*start_level: float*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>
↪:RXQuality:SENSitivity:STARtlevel
driver.configure.bluetooth.measurement.rxQuality.sensitivity.set_start_
↪level(start_level = 1.0)
```

No command help available

param start_level
No help available

set_stepsize(*stepsize: float*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:RXQuality:SENSitivity:STEPsize
driver.configure.bluetooth.measurement.rxQuality.sensitivity.set_
↪stepsize(stepsize = 1.0)
```

No command help available

param stepsize
No help available

6.4.1.1.10.7 SpotCheck

SCPI Command :

```
CONFigure:BLUetooth:MEASurement<Instance>:RXQuality:SPOTcheck:LEVel
```

class SpotCheckCls

SpotCheck commands group definition. 1 total commands, 0 Subgroups, 1 group commands

get_level() → float

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:RXQuality:SPOTcheck:LEVel
value: float = driver.configure.bluetooth.measurement.rxQuality.spotCheck.get_
↪level()
```

No command help available

return
level: No help available

set_level(*level: float*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:RXQuality:SPOTcheck:LEVel
driver.configure.bluetooth.measurement.rxQuality.spotCheck.set_level(level = 1.
↪0)
```

No command help available

param level
No help available

6.4.1.1.11 Trx

class TrxCls

Trx commands group definition. 1 total commands, 1 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.configure.bluetooth.measurement.trx.clone()
```

Subgroups

6.4.1.1.11.1 Result

class ResultCls

Result commands group definition. 1 total commands, 1 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.configure.bluetooth.measurement.trx.result.clone()
```

Subgroups

6.4.1.1.11.2 All

SCPI Command :

```
CONFigure:BLUetooth:MEASurement<Instance>:TRX:RESult[:ALL]
```

class AllCls

All commands group definition. 1 total commands, 0 Subgroups, 1 group commands

class AllStruct

Response structure. Fields:

- Spot_Check: bool: No parameter help available
- Power: bool: No parameter help available
- Modulation: bool: No parameter help available
- Spectrum_Acp: bool: No parameter help available

get() → AllStruct

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:TRX:RESult[:ALL]
value: AllStruct = driver.configure.bluetooth.measurement.trx.result.all.get()
```

No command help available

return
structure: for return value, see the help for AllStruct structure arguments.

set(*spot_check: bool*, *power: bool*, *modulation: bool*, *spectrum_acp: bool*) → None

```
# SCPI: CONFigure:BLUetooth:MEASurement<Instance>:TRX:RESult[:ALL]
driver.configure.bluetooth.measurement.trx.result.all.set(spot_check = False,␣
↪power = False, modulation = False, spectrum_acp = False)
```

No command help available

param spot_check
No help available

param power
No help available

param modulation
No help available

param spectrum_acp
No help available

6.5 Diagnostic

class DiagnosticCls

Diagnostic commands group definition. 2 total commands, 1 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.diagnostic.clone()
```

Subgroups

6.5.1 Bluetooth

class BluetoothCls

Bluetooth commands group definition. 2 total commands, 2 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.diagnostic.bluetooth.clone()
```

Subgroups

6.5.1.1 Measurement

class MeasurementCls

Measurement commands group definition. 1 total commands, 1 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.diagnostic.bluetooth.measurement.clone()
```

Subgroups

6.5.1.1.1 RfControl

SCPI Command :

```
DIAGnostic:BLUetooth:MEASurement<Instance>:RFControl:TXENable
```

class RfControlCls

RfControl commands group definition. 1 total commands, 0 Subgroups, 1 group commands

get_tx_enable() → bool

```
# SCPI: DIAGnostic:BLUetooth:MEASurement<Instance>:RFControl:TXENable
value: bool = driver.diagnostic.bluetooth.measurement.rfControl.get_tx_enable()
```

No command help available

return
set_ctrl_bit: No help available

set_tx_enable(*set_ctrl_bit: bool*) → None

```
# SCPI: DIAGnostic:BLUetooth:MEASurement<Instance>:RFControl:TXENable
driver.diagnostic.bluetooth.measurement.rfControl.set_tx_enable(set_ctrl_bit =␣
↪False)
```

No command help available

param set_ctrl_bit
No help available

6.5.1.2 Synchronise

SCPI Command :

```
DIAGnostic:BLUetooth:SYNChronise
```

class SynchroniseCls

Synchronise commands group definition. 1 total commands, 0 Subgroups, 1 group commands

class SynchroniseStruct

Response structure. Fields:

- Min_No_Valid_Bursts: int: No parameter help available
- Syn_Check_Filter: int: No parameter help available
- Max_Invalid_Burst: int: No parameter help available

get() → SynchroniseStruct

```
# SCPI: DIAGnostic:BLUetooth:SYNChronise
value: SynchroniseStruct = driver.diagnostic.bluetooth.synchronise.get()
```

No command help available

return
structure: for return value, see the help for SynchroniseStruct structure arguments.

set(*min_no_valid_bursts: int*, *syn_check_filter: int*, *max_invalid_burst: int*) → None

```
# SCPI: DIAGnostic:BLUetooth:SYNChronise
driver.diagnostic.bluetooth.synchronise.set(min_no_valid_bursts = 1, syn_check_
↪filter = 1, max_invalid_burst = 1)
```

No command help available

param min_no_valid_bursts
No help available

param syn_check_filter
No help available

param max_invalid_burst
No help available

6.6 Route

class RouteCls

Route commands group definition. 6 total commands, 1 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.route.clone()
```

Subgroups

6.6.1 Bluetooth

class BluetoothCls

Bluetooth commands group definition. 6 total commands, 1 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.route.bluetooth.clone()
```

Subgroups

6.6.1.1 Measurement

SCPI Command :

```
ROUTe:BLUetooth:MEASurement<Instance>
```

class MeasurementCls

Measurement commands group definition. 6 total commands, 2 Subgroups, 1 group commands

class ValueStruct

Structure for reading output parameters. Fields:

- Scenario: enums.TestScenario: No parameter help available
- Master: str: No parameter help available
- Rf_Connector: enums.RxConnector: No parameter help available
- Rf_Converter: enums.RxConverter: No parameter help available

get_value() → ValueStruct

```
# SCPI: ROUTe:BLUetooth:MEASurement<Instance>
value: ValueStruct = driver.route.bluetooth.measurement.get_value()
```

No command help available

return

structure: for return value, see the help for ValueStruct structure arguments.

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.route.bluetooth.measurement.clone()
```

Subgroups

6.6.1.1.1 RfSettings

SCPI Command :

```
ROUTe:BLUetooth:MEASurement<Instance>:RFSettings:CONNector
```

class RfSettingsCls

RfSettings commands group definition. 1 total commands, 0 Subgroups, 1 group commands

get_connector() → RxConnector

```
# SCPI: ROUTe:BLUetooth:MEASurement<Instance>:RFSettings:CONNector
value: enums.RxConnector = driver.route.bluetooth.measurement.rfSettings.get_
↪connector()
```

No command help available

return
rf_input_con: No help available

set_connector(*rf_input_con: RxConnector*) → None

```
# SCPI: ROUTe:BLUetooth:MEASurement<Instance>:RFSettings:CONNector
driver.route.bluetooth.measurement.rfSettings.set_connector(rf_input_con =␣
↪enums.RxConnector.I11I)
```

No command help available

param rf_input_con
No help available

6.6.1.1.2 Scenario

SCPI Commands :

```
ROUTe:BLUetooth:MEASurement<Instance>:SCENario:CSPath
ROUTe:BLUetooth:MEASurement<Instance>:SCENario
```

class ScenarioCls

Scenario commands group definition. 4 total commands, 2 Subgroups, 2 group commands

get_cspath() → str

```
# SCPI: ROUTe:BLUetooth:MEASurement<Instance>:SCENario:CSPath
value: str = driver.route.bluetooth.measurement.scenario.get_cspath()
```

No command help available

return
master: No help available

get_value() → TestScenario

```
# SCPI: ROUTe:BLUetooth:MEASurement<Instance>:SCENario
value: enums.TestScenario = driver.route.bluetooth.measurement.scenario.get_
↪value()
```

No command help available

return
scenario: No help available

set_cspath(*master: str*) → None

```
# SCPI: ROUTe:BLUetooth:MEASurement<Instance>:SCENario:CSPath
driver.route.bluetooth.measurement.scenario.set_cspath(master = 'abc')
```

No command help available

param master
No help available

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.route.bluetooth.measurement.scenario.clone()
```

Subgroups

6.6.1.1.2.1 MaProtocol

SCPI Command :

```
ROUTe:BLUetooth:MEASurement<Instance>:SCENario:MAPRotocol
```

class MaProtocolCls

MaProtocol commands group definition. 1 total commands, 0 Subgroups, 1 group commands

set() → None

```
# SCPI: ROUTe:BLUetooth:MEASurement<Instance>:SCENario:MAPRotocol
driver.route.bluetooth.measurement.scenario.maProtocol.set()
```

No command help available

set_with_opc(*opc_timeout_ms: int = -1*) → None

```
# SCPI: ROUTe:BLUetooth:MEASurement<Instance>:SCENario:MAPRotocol
driver.route.bluetooth.measurement.scenario.maProtocol.set_with_opc()
```

No command help available

Same as set, but waits for the operation to complete before continuing further. Use the RsCMPX_BluetoothMeas.utilities.opc_timeout_set() to set the timeout value.

param opc_timeout_ms
Maximum time to wait in milliseconds, valid only for this call.

6.6.1.1.2.2 Salone

SCPI Command :

```
ROUTe:BLUetooth:MEASurement<Instance>:SCENario:SALone
```

class SaloneCls

Salone commands group definition. 1 total commands, 0 Subgroups, 1 group commands

class SaloneStruct

Response structure. Fields:

- Rx_Connector: enums.RxConnector: No parameter help available
- Rf_Converter: enums.RxConverter: No parameter help available

get() → SaloneStruct

```
# SCPI: ROUTe:BLUetooth:MEASurement<Instance>:SCENario:SALone
value: SaloneStruct = driver.route.bluetooth.measurement.scenario.salone.get()
```

No command help available

return
structure: for return value, see the help for SaloneStruct structure arguments.

set(*rx_connector: RxConnector*, *rf_converter: RxConverter*) → None

```
# SCPI: ROUTe:BLUetooth:MEASurement<Instance>:SCENario:SALone
driver.route.bluetooth.measurement.scenario.salone.set(rx_connector = enums.
↪RxConnector.I11I, rf_converter = enums.RxConverter.IRX1)
```

No command help available

param rx_connector
No help available

param rf_converter
No help available

6.7 Sense

class SenseCls

Sense commands group definition. 2 total commands, 1 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.sense.clone()
```

Subgroups

6.7.1 Bluetooth

class BluetoothCls

Bluetooth commands group definition. 2 total commands, 1 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.sense.bluetooth.clone()
```

Subgroups

6.7.1.1 Measurement

class MeasurementCls

Measurement commands group definition. 2 total commands, 1 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.sense.bluetooth.measurement.clone()
```

Subgroups

6.7.1.1.1 Elogging

SCPI Commands :

```
SENSe:BLUetooth:MEASurement<Instance>:ELOGging:LAST
SENSe:BLUetooth:MEASurement<Instance>:ELOGging:ALL
```

class EloggingCls

Elogging commands group definition. 2 total commands, 0 Subgroups, 2 group commands

class AllStruct

Structure for reading output parameters. Fields:

- Timestamp: List[str]: No parameter help available
- Category: List[enums.LogCategory]: No parameter help available
- Event: List[str]: No parameter help available

class LastStruct

Structure for reading output parameters. Fields:

- Timestamp: str: No parameter help available
- Category: enums.LogCategory: No parameter help available
- Event: str: No parameter help available

get_all() → AllStruct

```
# SCPI: SENSe:BLUetooth:MEASurement<Instance>:ELOGging:ALL
value: AllStruct = driver.sense.bluetooth.measurement.elogging.get_all()
```

No command help available

return
structure: for return value, see the help for AllStruct structure arguments.

get_last() → LastStruct

```
# SCPI: SENSe:BLUetooth:MEASurement<Instance>:ELOGging:LAST
value: LastStruct = driver.sense.bluetooth.measurement.elogging.get_last()
```

No command help available

return
structure: for return value, see the help for LastStruct structure arguments.

6.8 Trigger

class TriggerCls

Trigger commands group definition. 14 total commands, 1 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.trigger.clone()
```

Subgroups

6.8.1 Bluetooth

class BluetoothCls

Bluetooth commands group definition. 14 total commands, 1 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.trigger.bluetooth.clone()
```

Subgroups

6.8.1.1 Measurement

class MeasurementCls

Measurement commands group definition. 14 total commands, 4 Subgroups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.trigger.bluetooth.measurement.clone()
```

Subgroups

6.8.1.1.1 BhRate

SCPI Commands :

```
TRIGger:BLUetooth:MEASurement<Instance>:BHRate:SOURce
TRIGger:BLUetooth:MEASurement<Instance>:BHRate:THReshold
TRIGger:BLUetooth:MEASurement<Instance>:BHRate:TOUT
```

class BhRateCls

BhRate commands group definition. 4 total commands, 1 Subgroups, 3 group commands

get_source() → str

```
# SCPI: TRIGger:BLUetooth:MEASurement<Instance>:BHRate:SOURce
value: str = driver.trigger.bluetooth.measurement.bhRate.get_source()
```

No command help available

return
source: No help available

get_threshold() → float

```
# SCPI: TRIGger:BLUetooth:MEASurement<Instance>:BHRate:THReshold
value: float or bool = driver.trigger.bluetooth.measurement.bhRate.get_
↪threshold()
```

No command help available

return
power: (float or boolean) No help available

get_timeout() → float

```
# SCPI: TRIGger:BLUetooth:MEASurement<Instance>:BHRate:TOUT
value: float or bool = driver.trigger.bluetooth.measurement.bhRate.get_timeout()
```

No command help available

return
timeout: (float or boolean) No help available

set_source(*source: str*) → None

```
# SCPI: TRIGger:BLUetooth:MEASurement<Instance>:BHRate:SOURce
driver.trigger.bluetooth.measurement.bhRate.set_source(source = 'abc')
```

No command help available

param source
No help available

set_threshold(*power: float*) → None

```
# SCPI: TRIGger:BLUetooth:MEASurement<Instance>:BHRate:THReshold
driver.trigger.bluetooth.measurement.bhRate.set_threshold(power = 1.0)
```

No command help available

param power
(float or boolean) No help available

set_timeout(*timeout: float*) → None

```
# SCPI: TRIGger:BLUetooth:MEASurement<Instance>:BHRate:TOUT
driver.trigger.bluetooth.measurement.bhRate.set_timeout(timeout = 1.0)
```

No command help available

param timeout
(float or boolean) No help available

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.trigger.bluetooth.measurement.bhRate.clone()
```

Subgroups

6.8.1.1.1.1 Catalog

SCPI Command :

```
TRIGger:BLUetooth:MEASurement<Instance>:BHRate:CATalog:SOURce
```

class CatalogCls

Catalog commands group definition. 1 total commands, 0 Subgroups, 1 group commands

get_source() → str

```
# SCPI: TRIGger:BLUetooth:MEASurement<Instance>:BHRate:CATalog:SOURce
value: str = driver.trigger.bluetooth.measurement.bhRate.catalog.get_source()
```

No command help available

return
source: No help available

6.8.1.1.2 Hdr

SCPI Commands :

```
TRIGger:BLUetooth:MEASurement<Instance>:HDR:THReshold
TRIGger:BLUetooth:MEASurement<Instance>:HDR:TOUT
TRIGger:BLUetooth:MEASurement<Instance>:HDR:SOURce
```

class HdrCls

Hdr commands group definition. 4 total commands, 1 Subgroups, 3 group commands

get_source() → str

```
# SCPI: TRIGger:BLUetooth:MEASurement<Instance>:HDR:SOURce
value: str = driver.trigger.bluetooth.measurement.hdr.get_source()
```

No command help available

return
source: No help available

get_threshold() → float

```
# SCPI: TRIGger:BLUetooth:MEASurement<Instance>:HDR:THReshold
value: float or bool = driver.trigger.bluetooth.measurement.hdr.get_threshold()
```

No command help available

return
power: (float or boolean) No help available

get_timeout() → float

```
# SCPI: TRIGger:BLUetooth:MEASurement<Instance>:HDR:TOUT
value: float or bool = driver.trigger.bluetooth.measurement.hdr.get_timeout()
```

No command help available

return
timeout: (float or boolean) No help available

set_source(*source: str*) → None

```
# SCPI: TRIGger:BLUetooth:MEASurement<Instance>:HDR:SOURce
driver.trigger.bluetooth.measurement.hdr.set_source(source = 'abc')
```

No command help available

param source
No help available

set_threshold(*power: float*) → None

```
# SCPI: TRIGger:BLUetooth:MEASurement<Instance>:HDR:THReshold
driver.trigger.bluetooth.measurement.hdr.set_threshold(power = 1.0)
```

No command help available

param power
(float or boolean) No help available

set_timeout(*timeout: float*) → None

```
# SCPI: TRIGger:BLUetooth:MEASurement<Instance>:HDR:TOUT
driver.trigger.bluetooth.measurement.hdr.set_timeout(timeout = 1.0)
```

No command help available

param timeout
(float or boolean) No help available

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.trigger.bluetooth.measurement.hdr.clone()
```

Subgroups

6.8.1.1.2.1 Catalog

SCPI Command :

```
TRIGger:BLUetooth:MEASurement<Instance>:HDR:CATalog:SOURce
```

class CatalogCls

Catalog commands group definition. 1 total commands, 0 Subgroups, 1 group commands

get_source() → str

```
# SCPI: TRIGger:BLUetooth:MEASurement<Instance>:HDR:CATalog:SOURce
value: str = driver.trigger.bluetooth.measurement.hdr.catalog.get_source()
```

No command help available

return
source: No help available

6.8.1.1.3 Hdrp

SCPI Commands :

```
TRIGger:BLUetooth:MEASurement<Instance>:HDRP:THReshold
TRIGger:BLUetooth:MEASurement<Instance>:HDRP:TOUT
TRIGger:BLUetooth:MEASurement<Instance>:HDRP:SOURce
```

class HdrpCls

Hdrp commands group definition. 4 total commands, 1 Subgroups, 3 group commands

get_source() → str

```
# SCPI: TRIGger:BLUetooth:MEASurement<Instance>:HDRP:SOURce
value: str = driver.trigger.bluetooth.measurement.hdrp.get_source()
```

No command help available

return
source: No help available

get_threshold() → float

```
# SCPI: TRIGger:BLUetooth:MEASurement<Instance>:HDRP:THReshold
value: float or bool = driver.trigger.bluetooth.measurement.hdrp.get_threshold()
```

No command help available

return
power: (float or boolean) No help available

get_timeout() → float

```
# SCPI: TRIGger:BLUetooth:MEASurement<Instance>:HDRP:TOUT
value: float or bool = driver.trigger.bluetooth.measurement.hdrp.get_timeout()
```

No command help available

return
timeout: (float or boolean) No help available

set_source(*source: str*) → None

```
# SCPI: TRIGger:BLUetooth:MEASurement<Instance>:HDRP:SOURce
driver.trigger.bluetooth.measurement.hdrp.set_source(source = 'abc')
```

No command help available

param source
No help available

set_threshold(*power: float*) → None

```
# SCPI: TRIGger:BLUetooth:MEASurement<Instance>:HDRP:THReshold
driver.trigger.bluetooth.measurement.hdrp.set_threshold(power = 1.0)
```

No command help available

param power
(float or boolean) No help available

set_timeout(*timeout: float*) → None

```
# SCPI: TRIGger:BLUetooth:MEASurement<Instance>:HDRP:TOUT
driver.trigger.bluetooth.measurement.hdrp.set_timeout(timeout = 1.0)
```

No command help available

param timeout
(float or boolean) No help available

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.trigger.bluetooth.measurement.hdrp.clone()
```

Subgroups

6.8.1.1.3.1 Catalog

SCPI Command :

```
TRIGger:BLUetooth:MEASurement<Instance>:HDRP:CATalog:SOURce
```

class CatalogCls

Catalog commands group definition. 1 total commands, 0 Subgroups, 1 group commands

get_source() → str

```
# SCPI: TRIGger:BLUetooth:MEASurement<Instance>:HDRP:CATalog:SOURce
value: str = driver.trigger.bluetooth.measurement.hdrp.catalog.get_source()
```

No command help available

return
source: No help available

6.8.1.1.4 MultiEval

SCPI Commands :

```
TRIGger:BLUetooth:MEASurement<Instance>:MEValuation:THReshold
TRIGger:BLUetooth:MEASurement<Instance>:MEValuation:TOUT
```

class MultiEvalCls

MultiEval commands group definition. 2 total commands, 0 Subgroups, 2 group commands

get_threshold() → float

```
# SCPI: TRIGger:BLUetooth:MEASurement<Instance>:MEValuation:THReshold
value: float or bool = driver.trigger.bluetooth.measurement.multiEval.get_
↪threshold()
```

Defines the trigger threshold for the power trigger.

return
power: (float or boolean) No help available

get_timeout() → float

```
# SCPI: TRIGger:BLUetooth:MEASurement<Instance>:MEValuation:TOUT
value: float or bool = driver.trigger.bluetooth.measurement.multiEval.get_
↪timeout()
```

Selects the maximum time that the measurement waits for a trigger event before it stops in remote control mode or indicates a trigger timeout in manual operation mode.

return
trigger_timeout: (float or boolean) No help available

set_threshold(*power: float*) → None

```
# SCPI: TRIGger:BLUetooth:MEASurement<Instance>:MEValuation:THReshold
driver.trigger.bluetooth.measurement.multiEval.set_threshold(power = 1.0)
```

Defines the trigger threshold for the power trigger.

param power
(float or boolean) No help available

set_timeout(*trigger_timeout: float*) → None

```
# SCPI: TRIGger:BLUetooth:MEASurement<Instance>:MEValuation:TOUT
driver.trigger.bluetooth.measurement.multiEval.set_timeout(trigger_timeout = 1.
↪0)
```

Selects the maximum time that the measurement waits for a trigger event before it stops in remote control mode or indicates a trigger timeout in manual operation mode.

param trigger_timeout
(float or boolean) No help available

CHAPTER

SEVEN

RSCMPX_BLUETOOTHMEAS UTILITIES

class Utilities

Common utility class. Utility functions common for all types of drivers.

Access snippet: `utils = RsCMPX_BluetoothMeas.utilities`

property logger: *ScpiLogger*

Scpi Logger interface, see *here*

Access snippet: `logger = RsCMPX_BluetoothMeas.utilities.logger`

property driver_version: str

Returns the instrument driver version.

property idn_string: str

Returns instrument's identification string - the response on the SCPI command *IDN?

property manufacturer: str

Returns manufacturer of the instrument.

property full_instrument_model_name: str

Returns the current instrument's full name e.g. 'FSW26'.

property instrument_model_name: str

Returns the current instrument's family name e.g. 'FSW'.

property supported_models: List[str]

Returns a list of the instrument models supported by this instrument driver.

property instrument_firmware_version: str

Returns instrument's firmware version.

property instrument_serial_number: str

Returns instrument's serial_number.

query_opc(*timeout: int = 0*) → int

SCPI command: *OPC? Queries the instrument's OPC bit and hence it waits until the instrument reports operation complete. If you define timeout > 0, the VISA timeout is set to that value just for this method call.

property instrument_status_checking: bool

Sets / returns Instrument Status Checking. When True (default is True), all the driver methods and properties are sending "SYSTem:ERRor?" at the end to immediately react on error that might have occurred. We recommend to keep the state checking ON all the time. Switch it OFF only in rare cases when you require maximum speed. The default state after initializing the session is ON.

property encoding: str

Returns string<=>bytes encoding of the session.

property opc_query_after_write: bool

Sets / returns Instrument *OPC? query sending after each command write. When True, (default is False) the driver sends *OPC? every time a write command is performed. Use this if you want to make sure your sequence is performed command-after-command.

property bin_float_numbers_format: BinFloatFormat

Sets / returns format of float numbers when transferred as binary data.

property bin_int_numbers_format: BinIntFormat

Sets / returns format of integer numbers when transferred as binary data.

clear_status() → None

Clears instrument's status system, the session's I/O buffers and the instrument's error queue.

query_all_errors() → List[str]

Queries and clears all the errors from the instrument's error queue. The method returns list of strings as error messages. If no error is detected, the return value is None. The process is: querying 'SYSTem:ERRor?' in a loop until the error queue is empty. If you want to include the error codes, call the query_all_errors_with_codes()

query_all_errors_with_codes() → List[Tuple[int, str]]

Queries and clears all the errors from the instrument's error queue. The method returns list of tuples (code: int, message: str). If no error is detected, the return value is None. The process is: querying 'SYSTem:ERRor?' in a loop until the error queue is empty.

property instrument_options: List[str]

Returns all the instrument options. The options are sorted in the ascending order starting with K-options and continuing with B-options.

reset() → None

SCPI command: *RST Sends *RST command + calls the clear_status().

default_instrument_setup() → None

Custom steps performed at the init and at the reset().

self_test(*timeout: int = None*) → Tuple[int, str]

SCPI command: *TST? Performs instrument's self-test. Returns tuple (code:int, message: str). Code 0 means the self-test passed. You can define the custom timeout in milliseconds. If you do not define it, the default selftest timeout is used (usually 60 secs).

is_connection_active() → bool

Returns true, if the VISA connection is active and the communication with the instrument still works.

reconnect(*force_close: bool = False*) → bool

If the connection is not active, the method tries to reconnect to the device If the connection is active, and force_close is False, the method does nothing. If the connection is active, and force_close is True, the method closes, and opens the session again. Returns True, if the reconnection has been performed.

property resource_name: int

Returns the resource name used in the constructor

property opc_timeout: int

Sets / returns timeout in milliseconds for all the operations that use OPC synchronization.

property visa_timeout: int

Sets / returns visa IO timeout in milliseconds.

property data_chunk_size: int

Sets / returns the maximum size of one block transferred during write/read operations

property visa_manufacturer: int

Returns the manufacturer of the current VISA session.

process_all_commands() → None

SCPI command: *WAI Stops further commands processing until all commands sent before *WAI have been executed.

write_str(*cmd: str*) → None

Writes the command to the instrument.

write(*cmd: str*) → None

This method is an alias to the write_str(). Writes the command to the instrument as string.

write_int(*cmd: str*, *param: int*) → None

Writes the command to the instrument followed by the integer parameter: e.g.: cmd = 'SELECT:INPUT' param = '2', result command = 'SELECT:INPUT 2'

write_int_with_opc(*cmd: str*, *param: int*, *timeout: int = None*) → None

Writes the command with OPC to the instrument followed by the integer parameter: e.g.: cmd = 'SELECT:INPUT' param = '2', result command = 'SELECT:INPUT 2' If you do not provide timeout, the method uses current opc_timeout.

write_float(*cmd: str*, *param: float*) → None

Writes the command to the instrument followed by the boolean parameter: e.g.: cmd = 'CENTER:FREQ' param = '10E6', result command = 'CENTER:FREQ 10E6'

write_float_with_opc(*cmd: str*, *param: float*, *timeout: int = None*) → None

Writes the command with OPC to the instrument followed by the boolean parameter: e.g.: cmd = 'CENTER:FREQ' param = '10E6', result command = 'CENTER:FREQ 10E6' If you do not provide timeout, the method uses current opc_timeout.

write_bool(*cmd: str*, *param: bool*) → None

Writes the command to the instrument followed by the boolean parameter: e.g.: cmd = 'OUTPUT' param = 'True', result command = 'OUTPUT ON'

write_bool_with_opc(*cmd: str*, *param: bool*, *timeout: int = None*) → None

Writes the command with OPC to the instrument followed by the boolean parameter: e.g.: cmd = 'OUTPUT' param = 'True', result command = 'OUTPUT ON' If you do not provide timeout, the method uses current opc_timeout.

query_str(*query: str*) → str

Sends the query to the instrument and returns the response as string. The response is trimmed of any trailing LF characters and has no length limit.

query(*query: str*) → str

This method is an alias to the query_str(). Sends the query to the instrument and returns the response as string. The response is trimmed of any trailing LF characters and has no length limit.

query_bool(*query: str*) → bool

Sends the query to the instrument and returns the response as boolean.

query_int(*query: str*) → int

Sends the query to the instrument and returns the response as integer.

query_float(*query: str*) → float

Sends the query to the instrument and returns the response as float.

write_str_with_opc(*cmd: str*, *timeout: int = None*) → None

Writes the opc-synced command to the instrument. If you do not provide timeout, the method uses current opc_timeout.

write_with_opc(*cmd: str*, *timeout: int = None*) → None

This method is an alias to the write_str_with_opc(). Writes the opc-synced command to the instrument. If you do not provide timeout, the method uses current opc_timeout.

query_str_with_opc(*query: str*, *timeout: int = None*) → str

Sends the opc-synced query to the instrument and returns the response as string. The response is trimmed of any trailing LF characters and has no length limit. If you do not provide timeout, the method uses current opc_timeout.

query_with_opc(*query: str*, *timeout: int = None*) → str

This method is an alias to the query_str_with_opc(). Sends the opc-synced query to the instrument and returns the response as string. The response is trimmed of any trailing LF characters and has no length limit. If you do not provide timeout, the method uses current opc_timeout.

query_bool_with_opc(*query: str*, *timeout: int = None*) → bool

Sends the opc-synced query to the instrument and returns the response as boolean. If you do not provide timeout, the method uses current opc_timeout.

query_int_with_opc(*query: str*, *timeout: int = None*) → int

Sends the opc-synced query to the instrument and returns the response as integer. If you do not provide timeout, the method uses current opc_timeout.

query_float_with_opc(*query: str*, *timeout: int = None*) → float

Sends the opc-synced query to the instrument and returns the response as float. If you do not provide timeout, the method uses current opc_timeout.

write_bin_block(*cmd: str*, *payload: bytes*) → None

Writes all the payload as binary data block to the instrument. The binary data header is added at the beginning of the transmission automatically, do not include it in the payload!!!

query_bin_block(*query: str*) → bytes

Queries binary data block to bytes. Throws an exception if the returned data was not a binary data. Returns data:bytes

query_bin_block_with_opc(*query: str*, *timeout: int = None*) → bytes

Sends a OPC-synced query and returns binary data block to bytes. If you do not provide timeout, the method uses current opc_timeout.

query_bin_or_ascii_float_list(*query: str*) → List[float]

Queries a list of floating-point numbers that can be returned in ASCII format or in binary format. - For ASCII format, the list numbers are decoded as comma-separated values. - For Binary Format, the numbers are decoded based on the property BinFloatFormat, usually float 32-bit (FORM REAL,32).

query_bin_or_ascii_float_list_with_opc(*query: str*, *timeout: int = None*) → List[float]

Sends a OPC-synced query and reads a list of floating-point numbers that can be returned in ASCII format or in binary format. - For ASCII format, the list numbers are decoded as comma-separated values. - For Binary Format, the numbers are decoded based on the property BinFloatFormat, usually float 32-bit (FORM REAL,32). If you do not provide timeout, the method uses current opc_timeout.

query_bin_or_ascii_int_list(*query: str*) → List[int]

Queries a list of floating-point numbers that can be returned in ASCII format or in binary format. - For ASCII format, the list numbers are decoded as comma-separated values. - For Binary Format, the numbers are decoded based on the property BinFloatFormat, usually float 32-bit (FORM REAL,32).

query_bin_or_ascii_int_list_with_opc(*query: str*, *timeout: int = None*) → List[int]

Sends a OPC-synced query and reads a list of floating-point numbers that can be returned in ASCII format or in binary format. - For ASCII format, the list numbers are decoded as comma-separated values. - For Binary Format, the numbers are decoded based on the property BinFloatFormat, usually float 32-bit (FORM REAL,32). If you do not provide timeout, the method uses current opc_timeout.

query_bin_block_to_file(*query: str*, *file_path: str*, *append: bool = False*) → None

Queries binary data block to the provided file. If append is False, any existing file content is discarded. If append is True, the new content is added to the end of the existing file, or if the file does not exit, it is created. Throws an exception if the returned data was not a binary data. Example for transferring a file from Instrument -> PC: query = f'MMEM:DATA? '{INSTR_FILE_PATH}'". Alternatively, use the dedicated methods for this purpose:

- `send_file_from_pc_to_instrument()`
- `read_file_from_instrument_to_pc()`

query_bin_block_to_file_with_opc(*query: str*, *file_path: str*, *append: bool = False*, *timeout: int = None*) → None

Sends a OPC-synced query and writes the returned data to the provided file. If append is False, any existing file content is discarded. If append is True, the new content is added to the end of the existing file, or if the file does not exit, it is created. Throws an exception if the returned data was not a binary data.

write_bin_block_from_file(*cmd: str*, *file_path: str*) → None

Writes data from the file as binary data block to the instrument using the provided command. Example for transferring a file from PC -> Instrument: cmd = f'MMEM:DATA '{INSTR_FILE_PATH}',". Alternatively, use the dedicated methods for this purpose:

- `send_file_from_pc_to_instrument()`
- `read_file_from_instrument_to_pc()`

send_file_from_pc_to_instrument(*source_pc_file: str*, *target_instr_file: str*) → None

SCPI Command: MMEM:DATA

Sends file from PC to the instrument

read_file_from_instrument_to_pc(*source_instr_file: str*, *target_pc_file: str*, *append_to_pc_file: bool = False*) → None

SCPI Command: MMEM:DATA?

Reads file from instrument to the PC.

Set the `append_to_pc_file` to True if you want to append the read content to the end of the existing PC file

get_last_sent_cmd() → str

Returns the last commands sent to the instrument. Only works in simulation mode

go_to_local() → None

Puts the instrument into local state.

go_to_remote() → None

Puts the instrument into remote state.

get_lock() → RLock

Returns the thread lock for the current session.

By default:

- If you create standard new RsCMPX_BluetoothMeas instance with new VISA session, the session gets a new thread lock. You can assign it to other RsCMPX_BluetoothMeas sessions in order to share one physical instrument with a multi-thread access.
- If you create new RsCMPX_BluetoothMeas from an existing session, the thread lock is shared automatically making both instances multi-thread safe.

You can always assign new thread lock by calling `driver.utilities.assign_lock()`

assign_lock(*lock: RLock*) → None

Assigns the provided thread lock.

clear_lock()

Clears the existing thread lock, making the current session thread-independent from others that might share the current thread lock.

sync_from(*source: Utilities*) → None

Synchronises these Utils with the source.

CHAPTER

EIGHT

RSCMPX_BLUETOOTHMEAS LOGGER

Check the usage in the Getting Started chapter *here*.

class ScpiLogger

Base class for SCPI logging

mode

Sets the logging ON or OFF. Additionally, you can set the logging ON only for errors. Possible values:

- LoggingMode.Off - logging is switched OFF
- LoggingMode.On - logging is switched ON
- LoggingMode.Errors - logging is switched ON, but only for error entries
- LoggingMode.Default - sets the logging to default - the value you have set with logger.default_mode

default_mode

Sets / returns the default logging mode. You can recall the default mode by calling the logger.mode = LoggingMode.Default.

Data Type
LoggingMode

device_name: str

Use this property to change the resource name in the log from the default Resource Name (e.g. TCPIP::192.168.2.101::INSTR) to another name e.g. 'MySigGen1'.

set_logging_target(*target*, *console_log: bool = None*, *udp_log: bool = None*) → None

Sets logging target - the target must implement write() and flush(). You can optionally set the console and UDP logging ON or OFF. This method switches the logging target global OFF.

get_logging_target()

Based on the global_mode, it returns the logging target: either the local or the global one.

set_logging_target_global(*console_log: bool = None*, *udp_log: bool = None*) → None

Sets logging target to global. The global target must be defined. You can optionally set the console and UDP logging ON or OFF.

log_to_console

Returns logging to console status.

log_to_udp

Returns logging to UDP status.

log_to_console_and_udp

Returns true, if both logging to UDP and console in are True.

info_raw(*log_entry: str*, *add_new_line: bool = True*) → None

Method for logging the raw string without any formatting.

info(*start_time: datetime*, *end_time: datetime*, *log_string_info: str*, *log_string: str*) → None

Method for logging one info entry. For binary log_string, use the info_bin()

error(*start_time: datetime*, *end_time: datetime*, *log_string_info: str*, *log_string: str*) → None

Method for logging one error entry.

set_relative_timestamp(*timestamp: datetime*) → None

If set, the further timestamps will be relative to the entered time.

set_relative_timestamp_now() → None

Sets the relative timestamp to the current time.

get_relative_timestamp() → datetime

Based on the global_mode, it returns the relative timestamp: either the local or the global one.

clear_relative_timestamp() → None

Clears the reference time, and the further logging continues with absolute times.

flush() → None

Flush all the entries.

log_status_check_ok

Sets / returns the current status of status checking OK. If True (default), the log contains logging of the status checking 'Status check: OK'. If False, the 'Status check: OK' is skipped - the log is more compact. Errors will still be logged.

clear_cached_entries() → None

Clears potential cached log entries. Cached log entries are generated when the Logging is ON, but no target has been defined yet.

set_format_string(*value: str*, *line_divider: str = '\n'*) → None

Sets new format string and line divider. If you just want to set the line divider, set the format string value=None The original format string is: `PAD_LEFT12(%START_TIME%) PAD_LEFT25(%DEVICE_NAME%) PAD_LEFT12(%DURATION%) %LOG_STRING_INFO%: %LOG_STRING%`

restore_format_string() → None

Restores the original format string and the line divider to LF

abbreviated_max_len_ascii: int

Defines the maximum length of one ASCII log entry. Default value is 200 characters.

abbreviated_max_len_bin: int

Defines the maximum length of one Binary log entry. Default value is 2048 bytes.

abbreviated_max_len_list: int

Defines the maximum length of one list entry. Default value is 100 elements.

bin_line_block_size: int

Defines number of bytes to display in one line. Default value is 16 bytes.

udp_port

Returns udp logging port.

target_auto_flushing

Returns status of the auto-flushing for the logging target.

CHAPTER

NINE

RSCMPX_BLUETOOTHMEAS EVENTS

Check the usage in the Getting Started chapter *here*.

class Events

Common Events class. Event-related methods and properties. Here you can set all the event handlers.

property before_query_handler: Callable

Returns the handler of before_query events.

Returns
current `before_query_handler`

property before_write_handler: Callable

Returns the handler of before_write events.

Returns
current `before_write_handler`

property io_events_include_data: bool

Returns the current state of the io_events_include_data See the setter for more details.

property on_read_handler: Callable

Returns the handler of on_read events.

Returns
current `on_read_handler`

property on_write_handler: Callable

Returns the handler of on_write events.

Returns
current `on_write_handler`

sync_from(*source: Events*) → None

Synchronises these Events with the source.

CHAPTER

TEN

INDEX

INDEX

CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:AVERage, 298
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:CURRent, 300
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:MAXimum, 301
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:MINimum, 303
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:SDEViation, 304
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:XMAXimum, 306
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:XMINimum, 307
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:EVM:AVERage, 322
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:EVM:CURRent, 324
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:EVM:MAXimum, 326
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:EVM:MINimum, 328
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:EVM:SDEViation, 329
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:EVM:XMAXimum, 331
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:EVM:XMINimum, 333
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:FERRor:AVERage, 335
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:FERRor:CURRent, 337
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:FERRor:MAXimum, 339
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:FERRor:MINimum, 340
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:FERRor:SDEViation, 342
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:FERRor:STDev, 343
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:FERRor:XMAXimum, 345
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:FERRor:XMINimum, 346
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:PENCoding:AVERage, 309
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:PENCoding:CURRent, 311
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:MAXimum, 313
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:MINimum, 315
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:PVTime:SDEViation, 317
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:RESult:AVERage, 318
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:RESult:CURRent, 320
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:RESult:MAXimum, 348
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:RESult:MINimum, 350
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:RESult:SDEViation, 351
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:RESult:XMAXimum, 353
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:RESult:XMINimum, 355
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:SYNC:AVERage, 356
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:SYNC:CURRent, 358
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:SYNC:MAXimum, 359
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:SYNC:MINimum, 361
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:SYNC:SDEViation, 362
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:SYNC:XMAXimum, 364
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:SYNC:XMINimum, 365
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:TRANge:AVERage, 367
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:TRANge:CURRent, 368
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:TRANge:MAXimum, 370
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:TRANge:MINimum, 371
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:TRANge:SDEViation, 373
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:TRANge:STDev, 374
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:TRANge:XMAXimum, 376
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:TRANge:XMINimum, 377
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:TSLot:AVERage, 379
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:TSLot:CURRent, 382
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:TSLot:MAXimum, 384
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:TSLot:MINimum, 386
CALCulate:BLUetooth:MEASurement<Instance>:MEValuation:TSLot:SDEViation, 388

```
CONFigure:BLUetooth:MEASurement<Instance>:BHRate:ECOunt:BCAP,
CONFigure:BLUetooth:MEASurement<Instance>:BHRate:ECOunt:MEASurement:MPAYload,
CONFigure:BLUetooth:MEASurement<Instance>:CFILter,
CONFigure:BLUetooth:MEASurement<Instance>:COMSignal:BADDress,
CONFigure:BLUetooth:MEASurement<Instance>:COMSignal:COMPort,
CONFigure:BLUetooth:MEASurement<Instance>:COMSignal:DBITs,
CONFigure:BLUetooth:MEASurement<Instance>:COMSignal:PARity,
CONFigure:BLUetooth:MEASurement<Instance>:COMSignal:PORTocatalog,
CONFigure:BLUetooth:MEASurement<Instance>:COMSignal:PROTocol,
CONFigure:BLUetooth:MEASurement<Instance>:COMSignal:SBITs,
CONFigure:BLUetooth:MEASurement<Instance>:CPRotocol,
CONFigure:BLUetooth:MEASurement<Instance>:DISPlay,
CONFigure:BLUetooth:MEASurement<Instance>:DTMode:RXQuality:BYEThreshold,
CONFigure:BLUetooth:MEASurement<Instance>:DTMode:RXQuality:BYTHreshold:MEASurement,
CONFigure:BLUetooth:MEASurement<Instance>:DTMode:RXQuality:BYTHreshold:MEASurement:LE1M,
CONFigure:BLUetooth:MEASurement<Instance>:DTMode:RXQuality:BYTHreshold:MEASurement:LE2M,
CONFigure:BLUetooth:MEASurement<Instance>:DTMode:RXQuality:BYTHreshold:MEASurement:LRANge,
CONFigure:BLUetooth:MEASurement<Instance>:DTMode:RXQuality:BYTHreshold:TELegram,
CONFigure:BLUetooth:MEASurement<Instance>:DTMode:RXQuality:BYTHreshold:TELegram:LE1M,
CONFigure:BLUetooth:MEASurement<Instance>:DTMode:RXQuality:BYTHreshold:TELegram:LE2M,
CONFigure:BLUetooth:MEASurement<Instance>:DTMode:RXQuality:BYTHreshold:TELegram:LRANge,
CONFigure:BLUetooth:MEASurement<Instance>:DTMode:RXQuality:SEARch:MEASurement:LENGth:LE1M,
CONFigure:BLUetooth:MEASurement<Instance>:DTMode:RXQuality:SEARch:MEASurement:LENGth:LE2M,
CONFigure:BLUetooth:MEASurement<Instance>:DTMode:RXQuality:SEARch:MEASurement:LENGth:LRANge,
CONFigure:BLUetooth:MEASurement<Instance>:DTMode:RXQuality:SEARch:PACKets:LENGth:LE1M,
```

```
CONFigure:BLUetooth:MEASurement<Instance>:DTMode:RXQuality:SEARch:PACKets:LENGth:LE2M,
CONFigure:BLUetooth:MEASurement<Instance>:DTMode:RXQuality:SEARch:PACKets:LENGth:LRANge,
CONFigure:BLUetooth:MEASurement<Instance>:DTMode:RXQuality:SEARch:TELegram:LENGth:LE1M,
CONFigure:BLUetooth:MEASurement<Instance>:DTMode:RXQuality:SEARch:TELegram:LENGth:LE2M,
CONFigure:BLUetooth:MEASurement<Instance>:DTMode:RXQuality:SEARch:TELegram:LENGth:LRANge,
CONFigure:BLUetooth:MEASurement<Instance>:DTMode:RXQuality:SEARch:TELegram:MOEXception,
CONFigure:BLUetooth:MEASurement<Instance>:DTMode:RXQuality:SEARch:TELegram:REPetition,
```

```
CONFigure:BLUetooth:MEASurement<Instance>:DTMode:RXQuality:SEARch:TELegram:TOEXception,
CONFigure:BLUetooth:MEASurement<Instance>:DTMode:RXQuality:SEARch:TIMing:BLENgth,
CONFigure:BLUetooth:MEASurement<Instance>:DTMode:RXQuality:SEARch:TIMing:BLOCk,
CONFigure:BLUetooth:MEASurement<Instance>:DTMode:RXQuality:SEARch:TIMing:PLENgth,
CONFigure:BLUetooth:MEASurement<Instance>:DTMode:RXQuality:SEARch:TIMing:RESPonse,
CONFigure:BLUetooth:MEASurement<Instance>:DTMode:RXQuality:SEARch:TIMing:SLOT,
CONFigure:BLUetooth:MEASurement<Instance>:DTMode:RXQuality:SEARch:TIMing:STEP,
CONFigure:BLUetooth:MEASurement<Instance>:DTMode:RXQuality:SEARch:TIMing:TOUT,
CONFigure:BLUetooth:MEASurement<Instance>:DTMode:RXQuality:SEARch:TYPE,
CONFigure:BLUetooth:MEASurement<Instance>:DTMode:RXQuality:SEARch:WLENgth,
CONFigure:BLUetooth:MEASurement<Instance>:DTMode:RXQuality:SEARch:WSYNc,
CONFigure:BLUetooth:MEASurement<Instance>:DTMode:RXQuality:SEARch:WSYNc:LENGth,
CONFigure:BLUetooth:MEASurement<Instance>:DTMode:RXQuality:SEARch:WSYNc:MOEXception,
CONFigure:BLUetooth:MEASurement<Instance>:DTMode:RXQuality:SEARch:WSYNc:REPetition,
```

```
CONFigure:BLUetooth:MEASurement<Instance>:DTMode:RXQuality:SEARch:WSYNc:TOEXception,
CONFigure:BLUetooth:MEASurement<Instance>:DTMode:RXQuality:THReshold,
CONFigure:BLUetooth:MEASurement<Instance>:DTMode:RXQuality:THReshold:MEASurement,
CONFigure:BLUetooth:MEASurement<Instance>:DTMode:RXQuality:THReshold:MEASurement:LE1M,
CONFigure:BLUetooth:MEASurement<Instance>:DTMode:RXQuality:THReshold:MEASurement:LE2M,
CONFigure:BLUetooth:MEASurement<Instance>:DTMode:RXQuality:THReshold:MEASurement:LRANge,
CONFigure:BLUetooth:MEASurement<Instance>:DTMode:RXQuality:THReshold:TELegram,
CONFigure:BLUetooth:MEASurement<Instance>:DTMode:RXQuality:THReshold:TELegram:LE1M,
CONFigure:BLUetooth:MEASurement<Instance>:DTMode:RXQuality:THReshold:TELegram:LE2M,
CONFigure:BLUetooth:MEASurement<Instance>:DTMode:RXQuality:THReshold:TELegram:LRANge,
CONFigure:BLUetooth:MEASurement<Instance>:GDELay,
CONFigure:BLUetooth:MEASurement<Instance>:HDR:ISIGnal:ASYNc,
CONFigure:BLUetooth:MEASurement<Instance>:HDR:ISIGnal:BDADdress,
CONFigure:BLUetooth:MEASurement<Instance>:HDR:ISIGnal:DMODulation,
CONFigure:BLUetooth:MEASurement<Instance>:HDR:ISIGnal:LAP,
CONFigure:BLUetooth:MEASurement<Instance>:HDR:ISIGnal:NAP,
CONFigure:BLUetooth:MEASurement<Instance>:HDR:ISIGnal:PATTern,
CONFigure:BLUetooth:MEASurement<Instance>:HDR:ISIGnal:PLENgth,
CONFigure:BLUetooth:MEASurement<Instance>:HDR:ISIGnal:PTYPe,
CONFigure:BLUetooth:MEASurement<Instance>:HDR:ISIGnal:UAP,
CONFigure:BLUetooth:MEASurement<Instance>:HDR:LIMit:FSTability,
CONFigure:BLUetooth:MEASurement<Instance>:HDR:LIMit:P4H:DPSK,
CONFigure:BLUetooth:MEASurement<Instance>:HDR:LIMit:P4H:SCORr,
CONFigure:BLUetooth:MEASurement<Instance>:HDR:LIMit:P8H:DPSK,
CONFigure:BLUetooth:MEASurement<Instance>:HDR:LIMit:P8H:SCORr,
CONFigure:BLUetooth:MEASurement<Instance>:HDR:LIMit:PENCode,
CONFigure:BLUetooth:MEASurement<Instance>:HDR:LIMit:PVTime,
CONFigure:BLUetooth:MEASurement<Instance>:HDR:MOEXception,
CONFigure:BLUetooth:MEASurement<Instance>:HDR:REPetition,
```

```
637
CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:OSLots:L
637
CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:OSLots:L
638
CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:OSLots:L
638
CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:PATTern,
639
CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:PATTern:
639
CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:PATTern:
640
CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:PATTern:
640
CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:PLENgth:
641
CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:PLENgth:
629
CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:PLENgth:
643
CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:PLENgth:
643
CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:PLENgth:
643
CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:PLENgth:
644
CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:PLENgth:
644
CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:PLENgth:
644
CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:PLENgth:
646
CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:PLENgth:
646
CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:PTYPe:BR
646
CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:PTYPe:ED
648
CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:PTYPe:LE
649
CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:PTYPe:LE
649
CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:PTYPe:LE
650
CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:PTYPe:QH
650
CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:PTYPe:QH
651
CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:PTYPe:QH
651
CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:PTYPe:QH
652
CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:PTYPe:QH
652
CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:OSLots:L
653
CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:OSLots:L
653
CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:OSLots:L
653
CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:PATTern,
654
CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:PATTern:
655
CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:PATTern:
655
CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:PATTern:
655
CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:PLENgth:
657
CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:PLENgth:
657
CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:PLENgth:
658
CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:PLENgth:
658
CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:PLENgth:
658
CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:PLENgth:
660
CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:PLENgth:
660
CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:PLENgth:
660
CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:PLENgth:
660
CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:PTYPe:BR
662
CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:PTYPe:ED
662
CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:PTYPe:LE
663
CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:PTYPe:LE
663
CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:PTYPe:LE
663
CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:PTYPe:QH
664
CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:PTYPe:QH
664
CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:PTYPe:QH
664
CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:PTYPe:QH
664
CONFigure:BLUetooth:MEASurement<Instance>:ISIGnal:PTYPe:QH
```

```
727 CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIST
728 CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIST
729 CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIST
730 CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIST
725 CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIST
731 CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIST
732 CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIST
733 CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIST
707 CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIST
706 CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIST
708 CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIST
709 CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIST
710 CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIST
711 CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIST
734 CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIST
735 CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIST
736 CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIST
737 CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIST
738 CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIST
739 CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIST
740 CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIST
741 CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIST
742 CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIST
743 CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIST
674 CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIST
744 CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIST
745 CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIST
```

```
745 CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIST
781 CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIST
745 CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIST
745 CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIST
745 CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIST
748 CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIST
749 CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIST
750 CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIST
751 CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIST
752 CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIST
752 CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIST
753 CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIST
754 CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIST
755 CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIST
756 CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIST
757 CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIST
758 CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIST
759 CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIST
759 CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIST
760 CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIST
761 CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIST
763 CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIST
779 CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIST
764 CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIST
765 CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIST
766 CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIST
766 CONFigure:BLUetooth:MEASurement<Instance>:MEValuation:LIST
```

D

E

F

G

I

L

M

R

READ:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:AVERage, 233
READ:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:CURRent, 235
READ:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:EXTReme, 237
READ:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:MAXimum, 238
READ:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:MINimum, 240
READ:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:SDEViation, 241
READ:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:STDDev, 243
READ:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:AVERage:ALL, 245
READ:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:CURRent:ALL, 246
READ:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:EXTReme:ALL, 248
READ:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:MAXimum:ALL, 249
READ:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:SDEViation:ALL, 251
READ:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:STDDev:ALL, 253
READ:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:E1M:AVERage, 255
READ:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:E1M:CURRent, 258
READ:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:E1M:MAXimum, 260
READ:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:E1M:SDEViation, 262
READ:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:E1M:STDDev, 263
READ:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:E1M:MINimum, 266
READ:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:E2M:AVERage, 298
READ:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:E2M:CURRent, 300
READ:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:E2M:CURRent:EXTended, 301
READ:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:E2M:MAXimum, 303
READ:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:E2M:SDEViation, 304
READ:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:E2M:STDDev, 306
READ:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:E2M:CURRent, 307
READ:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:E2M:MAXimum, 322
READ:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:E2M:MINimum, 324
READ:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:E2M:SDEViation, 326
READ:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:E2M:XMAXimum, 328
READ:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:E2M:XMINimum, 329
READ:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:LRANge:AVERage, 331
READ:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:LRANge:CURRent, 333
READ:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:LRANge:MAXimum, 335
READ:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:LRANge:MINimum, 337
READ:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:LRANge:SDEViation, 339
READ:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:LRANge:STDDev, 340
READ:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:LRANge:XMAXimum, 342
READ:BLUetooth:MEASurement<Instance>:MEValuation:MODulation:LRANge:XMINimum, 343

S

T

U