RsCmwEvdoMeas

Release 3.8.10.6

Rohde & Schwarz

May 27, 2021

CONTENTS:


CMW

CHAPTER

ONE

GETTING STARTED

1.1 Introduction

RsCmwEvdoMeas 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 RsCmwBase example:

```
""" Example on how to use the python RsCmw auto-generated instrument driver showing:
- usage of basic properties of the cmw_base object
- basic concept of setting commands and repcaps: DISPlay:WINDow<n>:SELect
- cmw_xxx drivers reliability interface usage
"""

from RsCmwBase import *  # install from pypi.org

RsCmwBase.assert_minimum_version('3.7.90.32')
cmw_base = RsCmwBase('TCPIP::10.112.1.116::INSTR', True, False)
print(f'CMW Base IND: {cmw_base.utilities.idn_string}')
print(f'CMW Instrument options:\n{",".join(cmw_base.utilities.instrument_options)}')
cmw_base.utilities.visa_timeout = 5000

# Sends OPC after each command
cmw_base.utilities.opc_query_after_write = False
```

(continues on next page)

(continued from previous page)

```
# Checks for syst:err? after each command / query
cmw_base.utilities.instrument_status_checking = True

# DISPlay:WINDow<n>:SELect
cmw_base.display.window.select.set(repcap.Window.Win1)
cmw_base.display.window.repcap_window_set(repcap.Window.Win2)
cmw_base.display.window.select.set()

# Self-test
self_test = cmw_base.utilities.self_test()
print(f'CMW self-test result: {self_test} - {"Passed" if self_test[0] == 0 else "Failed"}
↪"')

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


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


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

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

# Reference Frequency Source
cmw_base.system.reference.frequency.source_set(enums.SourceIntExt.INTernal)

# Close the session
cmw_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 (in case of big data transfer)

- Multithreading session locking - you can use multiple threads talking to one instrument at the same time

1.2 Installation

RsCmwEvdoMeas 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 RsCmwEvdoMeas`

Option 2 - Installing in Pycharm

- In Pycharm Menu `File->Settings->Project->Project Interpreter` click on the **'+'** button on the bottom left
- Type `RsCmwEvdoMeas` 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 5 easy step for installing the RsCmwEvdoMeas offline:

- Download this python script (**Save target as**): rsinstrument_offline_install.py This installs all the preconditions that the RsCmwEvdoMeas needs.
- Execute the script in your offline computer (supported is python 3.6 or newer)
- Download the RsCmwEvdoMeas package to your computer from the pypi.org: https://pypi.org/project/RsCmwEvdoMeas/#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\RsCmwEvdoMeas-3.8.10.6.tar`

1.3 Finding Available Instruments

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

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

from RsCmwEvdoMeas import *

# Use the instr_list string items as resource names in the RsCmwEvdoMeas constructor
instr_list = RsCmwEvdoMeas.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 RsCmwEvdoMeas 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 = RsCmwEvdoMeas.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

1.4 Initiating Instrument Session

RsCmwEvdoMeas 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 RsCmwEvdoMeas object. Below, is a simple Hello World example. Different resource names are examples for different physical interfaces.

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

- Installed RsCmwEvdoMeas Python module Version 3.8.10 or newer from pypi.org
- Installed VISA, for example R&S Visa 5.12 or newer
"""

from RsCmwEvdoMeas import *

# A good practice is to assure that you have a certain minimum version installed
RsCmwEvdoMeas.assert_minimum_version('3.8.10')
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 = RsCmwEvdoMeas(resource_string_1)

idn = driver.utilities.query_str('*IDN?')
print(f"\nHello, I am: '{idn}'")
print(f'RsCmwEvdoMeas 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 2021.

Do not care about specialty of each session kind; RsCmwEvdoMeas 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 = RsCmwEvdoMeas('TCPIP::192.168.56.101::HISLIP', id_query=True, reset=True)
```

- Setting id_query to True (default is True) checks, whether your instrument can be used with the RsCmwEvdoMeas 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 RsCmwEvdoMeas allows you to choose which VISA to use:

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

from RsCmwEvdoMeas import *

# Force use of the Rs Visa. For NI Visa, use the "SelectVisa='ni'"
driver = RsCmwEvdoMeas('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, RsCmwEvdoMeas has something for you too - **no Visa installation raw LAN socket**:

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

from RsCmwEvdoMeas import *

driver = RsCmwEvdoMeas('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 = RsCmwEvdoMeas('TCPIP::192.168.56.101::HISLIP', True, True, "Simulate=True")
```

More `option_string` tokens are separated by comma:

```
driver = RsCmwEvdoMeas('TCPIP::192.168.56.101::HISLIP', 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 RsCmwEvdoMeas objects:

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

from RsCmwEvdoMeas import *

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

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

1.5 Plain SCPI Communication

After you have opened the session, you can use the instrument-specific part described in the RsCmwEvdoMeas 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 RsCmwEvdoMeas import *

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

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

(continues on next page)

(continued from previous page)

```
# 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 RsCmwEvdoMeas raises an exception. Speaking of exceptions, an important feature of the RsCmwEvdoMeas 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 RsCmwEvdoMeas import *

driver = RsCmwEvdoMeas('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
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
```

1.6 Error Checking

RsCmwEvdoMeas 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.

1.7 Exception Handling

The base class for all the exceptions raised by the RsCmwEvdoMeas 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 RsCmwEvdoMeas import *

driver = None
# Try-catch for initialization. If an error occures, the ResourceError is raised
```

(continues on next page)

(continued from previous page)

```
try:
    driver = RsCmwEvdoMeas('TCPIP::10.112.1.179::HISLIP')
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...')

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

1.8 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 RsCmwEvdoMeas, 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 RsCmwEvdoMeas 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')
```

1.9 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")
```

1.10 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 RsCmwEvdoMeas has this covered. And, this feature is quite universal - not just for big files transfer, but for any data in both directions.

RsCmwEvdoMeas 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 RsCmwEvdoMeas import *
import time


def my_transfer_handler(args):
    """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 = RsCmwEvdoMeas('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 RsCmwEvdoMeas 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
```

1.11 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, RsCmwEvdoMeas 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 RsCmwEvdoMeas object
"""

import threading
from RsCmwEvdoMeas import *


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


driver = RsCmwEvdoMeas('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')
```

(continues on next page)

(continued from previous page)

```
# 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 RsCmwEvdoMeas objects with shared session
"""

import threading
from RsCmwEvdoMeas import *


def execute(session: RsCmwEvdoMeas, 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 = RsCmwEvdoMeas('TCPIP::192.168.56.101::INSTR')
driver2 = RsCmwEvdoMeas.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()
```

(continues on next page)

(continued from previous page)

```
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 RsCmwEvdoMeas takes care of it for you. The text below describes this scenario.

Run the following example:

```
"""
Multiple threads are accessing two RsCmwEvdoMeas objects with two separate sessions
"""

import threading
from RsCmwEvdoMeas import *


def execute(session: RsCmwEvdoMeas, 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 = RsCmwEvdoMeas('TCPIP::192.168.56.101::INSTR')
driver2 = RsCmwEvdoMeas('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,))
```

(continues on next page)

(continued from previous page)

```
    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.

CHAPTER

TWO

REVISION HISTORY

Rohde & Schwarz CMW Base System RsCmwBase instrument driver.

Supported instruments: CMW500, CMW100, CMW270, CMW280

The package is hosted here: https://pypi.org/project/RsCmwBase/

Documentation: https://RsCmwBase.readthedocs.io/

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

Currently supported CMW subsystems:

- Base: RsCmwBase
- Global Purpose RF: RsCmwGprfGen, RsCmwGprfMeas
- Bluetooth: RsCmwBluetoothSig, RsCmwBluetoothMeas
- LTE: RsCmwLteSig, RsCmwLteMeas
- CDMA2000: RsCdma2kSig, RsCdma2kMeas
- 1xEVDO: RsCmwEvdoSig, RsCmwEvdoMeas
- WCDMA: RsCmwWcdmaSig, RsCmwWcdmaMeas
- GSM: RsCmwGsmSig, RsCmwGsmMeas
- WLAN: RsCmwWlanSig, RscmwWlanMeas
- DAU: RsCMwDau

In case you require support for more subsystems, please contact our customer support on customersupport@rohde-schwarz.com with the topic "Auto-generated Python drivers" in the email subject. This will speed up the response process

---

Examples: Download the file 'CMW Python instrument drivers' from https://www.rohde-schwarz.com/driver/cmw500_overview/ The zip file contains the examples on how to use these drivers. Remember to adjust the resourceName string to fit your instrument.

Release Notes for the whole RsCmwXXX group:

Latest release notes summary: <INVALID>

Version 3.7.90.39

- <INVALID>

Version 3.8.xx2

- Fixed several misspelled arguments and command headers

Version 3.8.xx1

- Bluetooth and WLAN update for FW versions 3.8.xxx

Version 3.7.xx8

- Added documentation on ReadTheDocs

Version 3.7.xx7

- Added 3G measurement subsystems RsCmwGsmMeas, RsCmwCdma2kMeas, RsCmwEvdoMeas, RsCmwWcdmaMeas
- Added new data types for commands accepting numbers or ON/OFF:
- int or bool
- float or bool

Version 3.7.xx6

- Added new UDF integer number recognition

Version 3.7.xx5

- Added RsCmwDau

Version 3.7.xx4

- Fixed several interface names
- New release for CMW Base 3.7.90
- New release for CMW Bluetooth 3.7.90

Version 3.7.xx3

- Second release of the CMW python drivers packet
- New core component RsInstrument
- Previously, the groups starting with CATalog: e.g. 'CATalog:SIGNaling:TOPology:PLMN' were reordered to 'SIGNaling:TOPology:PLMN:CATALOG' give more contextual meaning to the method/property name. This is now reverted back, since it was hard to find the desired functionality.
- Reorganized Utilities interface to sub-groups

Version 3.7.xx2

- Fixed some misspeling errors
- Changed enum and repCap types names
- All the assemblies are signed with Rohde & Schwarz signature

Version 1.0.0.0

- First released version

CHAPTER

THREE

ENUMS

3.1 AckDsc

```
# Example value:
value = enums.AckDsc.ACK
# All values (4x):
ACK | DNCare | DSC | OFF
```

3.2 BandClass

```
# First value:
value = enums.BandClass.AWS
# Last value:
value = enums.BandClass.USPC
# All values (22x):
AWS | B18M | IEXT | IM2K | JTAC | KCEL | KPCS | LBANd
LO7C | N45T | NA7C | NA8S | NA9C | NAPC | PA4M | PA8M
PS7C | SBANd | TACS | U25B | USC | USPC
```

3.3 CmwsConnector

```
# First value:
value = enums.CmwsConnector.R11
# Last value:
value = enums.CmwsConnector.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
```

3.4 Dmodulation

```
# Example value:
value = enums.Dmodulation.AUTO
# All values (6x):
AUTO | B4 | E4E2 | Q2 | Q4 | Q4Q2
```

3.5 HalfSlot

```
# Example value:
value = enums.HalfSlot.BHSLots
# All values (3x):
BHSLots | FHSLot | SHSLot
```

3.6 MeasCondition

```
# Example value:
value = enums.MeasCondition.DNCare
# All values (3x):
DNCare | OFF | ON
```

3.7 ObwUsedLimitSet

```
# Example value:
value = enums.ObwUsedLimitSet.SETA
# All values (2x):
SETA | SETB
```

3.8 ParameterSetMode

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

3.9 PlSubtype

```
# Example value:
value = enums.PlSubtype.ST01
# All values (3x):
ST01 | ST2 | ST3
```

3.10 Rbw

```
# First value:
value = enums.Rbw.F100k
# Last value:
value = enums.Rbw.F6K25
# All values (10x):
F100k | F10K | F12K5 | F1K0 | F1M0 | F1M23 | F25K | F30K
F50K | F6K25
```

3.11 RefPowerMode

```
# Example value:
value = enums.RefPowerMode.ATPower
# All values (2x):
ATPower | PPOWer
```

3.12 Repeat

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

3.13 ResourceState

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

3.14 ResultStateA

```
# Example value:
value = enums.ResultStateA.ACTive
# All values (3x):
ACTive | IACTive | INVisible
```

3.15 ResultStateB

```
# Example value:
value = enums.ResultStateB.ACTive
# All values (3x):
ACTive | INACtive | NAV
```

3.16 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
```

3.17 RetriggerMode

```
# Example value:
value = enums.RetriggerMode.ONCE
# All values (2x):
ONCE | SEGMent
```

3.18 RetriggerOption

```
# Example value:
value = enums.RetriggerOption.IFPower
# All values (4x):
IFPower | IFPSync | OFF | ON
```

3.19 RxConnector

```
# First value:
value = enums.RxConnector.I11I
# Last value:
value = enums.RxConnector.RH8
# All values (154x):
I11I | I13I | I15I | I17I | I21I | I23I | I25I | I27I
I31I | I33I | I35I | I37I | I41I | I43I | I45I | I47I
IF1 | IF2 | IF3 | IQ1I | IQ3I | IQ5I | IQ7I | R11
R11C | R12 | R12C | 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 | 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 | RF8 | RFAC
RFBC | RFBI | RG1 | RG2 | RG3 | RG4 | RG5 | RG6
RG7 | RG8 | RH1 | RH2 | RH3 | RH4 | RH5 | RH6
RH7 | RH8
```

3.20 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
```

3.21 Srate

```
# Example value:
value = enums.Srate.SF16
# All values (2x):
SF16 | SF32
```

3.22 StatePower

```
# Example value:
value = enums.StatePower.BOTH
# All values (4x):
BOTH | LOWer | OK | UPPer
```

3.23 StopConditionB

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

3.24 Tab

```
# Example value:
value = enums.Tab.MEVA
# All values (2x):
MEVA | OLTR
```

3.25 TestScenarioB

```
# Example value:
value = enums.TestScenarioB.CSPath
# All values (4x):
CSPath | MAPRotocol | SALone | UNDefined
```

3.26 TriggerSlope

```
# Example value:
value = enums.TriggerSlope.FEDGe
# All values (4x):
FEDGe | OFF | ON | REDGe
```

3.27 UpDownDirection

```
# Example value:
value = enums.UpDownDirection.DOWN
# All values (2x):
DOWN | UP
```

3.28 WbFilter

```
# Example value:
value = enums.WbFilter.F16M0
# All values (2x):
F16M0 | F8M0
```

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 AcpMinus

```
# First value:
value = repcap.AcpMinus.Ch1
# Range:
Ch1 .. Ch20
# All values (20x):
Ch1 | Ch2 | Ch3 | Ch4 | Ch5 | Ch6 | Ch7 | Ch8
Ch9 | Ch10 | Ch11 | Ch12 | Ch13 | Ch14 | Ch15 | Ch16
Ch17 | Ch18 | Ch19 | Ch20
```

4.3 AcpPlus

```
# First value:
value = repcap.AcpPlus.Ch1
# Range:
Ch1 .. Ch20
# All values (20x):
Ch1 | Ch2 | Ch3 | Ch4 | Ch5 | Ch6 | Ch7 | Ch8
Ch9 | Ch10 | Ch11 | Ch12 | Ch13 | Ch14 | Ch15 | Ch16
Ch17 | Ch18 | Ch19 | Ch20
```

4.4 Obw

```
# First value:
value = repcap.Obw.Nr1
# Values (4x):
Nr1 | Nr2 | Nr3 | Nr4
```

4.5 Segment

```
# First value:
value = repcap.Segment.Nr1
# Range:
Nr1 .. Nr200
# All values (200x):
Nr1 | Nr2 | Nr3 | Nr4 | Nr5 | Nr6 | Nr7 | Nr8
Nr9 | Nr10 | Nr11 | Nr12 | Nr13 | Nr14 | Nr15 | Nr16
Nr17 | Nr18 | Nr19 | Nr20 | Nr21 | Nr22 | Nr23 | Nr24
Nr25 | Nr26 | Nr27 | Nr28 | Nr29 | Nr30 | Nr31 | Nr32
Nr33 | Nr34 | Nr35 | Nr36 | Nr37 | Nr38 | Nr39 | Nr40
Nr41 | Nr42 | Nr43 | Nr44 | Nr45 | Nr46 | Nr47 | Nr48
Nr49 | Nr50 | Nr51 | Nr52 | Nr53 | Nr54 | Nr55 | Nr56
Nr57 | Nr58 | Nr59 | Nr60 | Nr61 | Nr62 | Nr63 | Nr64
Nr65 | Nr66 | Nr67 | Nr68 | Nr69 | Nr70 | Nr71 | Nr72
Nr73 | Nr74 | Nr75 | Nr76 | Nr77 | Nr78 | Nr79 | Nr80
Nr81 | Nr82 | Nr83 | Nr84 | Nr85 | Nr86 | Nr87 | Nr88
Nr89 | Nr90 | Nr91 | Nr92 | Nr93 | Nr94 | Nr95 | Nr96
Nr97 | Nr98 | Nr99 | Nr100 | Nr101 | Nr102 | Nr103 | Nr104
Nr105 | Nr106 | Nr107 | Nr108 | Nr109 | Nr110 | Nr111 | Nr112
Nr113 | Nr114 | Nr115 | Nr116 | Nr117 | Nr118 | Nr119 | Nr120
Nr121 | Nr122 | Nr123 | Nr124 | Nr125 | Nr126 | Nr127 | Nr128
Nr129 | Nr130 | Nr131 | Nr132 | Nr133 | Nr134 | Nr135 | Nr136
Nr137 | Nr138 | Nr139 | Nr140 | Nr141 | Nr142 | Nr143 | Nr144
Nr145 | Nr146 | Nr147 | Nr148 | Nr149 | Nr150 | Nr151 | Nr152
Nr153 | Nr154 | Nr155 | Nr156 | Nr157 | Nr158 | Nr159 | Nr160
Nr161 | Nr162 | Nr163 | Nr164 | Nr165 | Nr166 | Nr167 | Nr168
Nr169 | Nr170 | Nr171 | Nr172 | Nr173 | Nr174 | Nr175 | Nr176
Nr177 | Nr178 | Nr179 | Nr180 | Nr181 | Nr182 | Nr183 | Nr184
Nr185 | Nr186 | Nr187 | Nr188 | Nr189 | Nr190 | Nr191 | Nr192
Nr193 | Nr194 | Nr195 | Nr196 | Nr197 | Nr198 | Nr199 | Nr200
```

4.6 Sequence

```
# First value:
value = repcap.Sequence.Nr1
# Range:
Nr1 .. Nr5
# All values (5x):
Nr1 | Nr2 | Nr3 | Nr4 | Nr5
```

CHAPTER

FIVE

EXAMPLES

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

```
""" Example on how to use the python RsCmw auto-generated instrument driver showing:
- usage of basic properties of the cmw_base object
- basic concept of setting commands and repcaps: DISPlay:WINDow<n>:SELect
- cmw_xxx drivers reliability interface usage
"""

from RsCmwBase import *  # install from pypi.org

RsCmwBase.assert_minimum_version('3.7.90.32')
cmw_base = RsCmwBase('TCPIP::10.112.1.116::INSTR', True, False)
print(f'CMW Base IND: {cmw_base.utilities.idn_string}')
print(f'CMW Instrument options:\n{",".join(cmw_base.utilities.instrument_options)}')
cmw_base.utilities.visa_timeout = 5000

# Sends OPC after each command
cmw_base.utilities.opc_query_after_write = False

# Checks for syst:err? after each command / query
cmw_base.utilities.instrument_status_checking = True

# DISPlay:WINDow<n>:SELect
cmw_base.display.window.select.set(repcap.Window.Win1)
cmw_base.display.window.repcap_window_set(repcap.Window.Win2)
cmw_base.display.window.select.set()

# Self-test
self_test = cmw_base.utilities.self_test()
print(f'CMW self-test result: {self_test} - {"Passed" if self_test[0] == 0 else "Failed"}
↪"')

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


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

(continues on next page)

(continued from previous page)

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

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

# Reference Frequency Source
cmw_base.system.reference.frequency.source_set(enums.SourceIntExt.INTernal)

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

CHAPTER

SIX

INDEX

CHAPTER

SEVEN

RSCMWEVDOMEAS API STRUCTURE

Global RepCaps

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

class RsCmwEvdoMeas(*resource_name: str*, *id_query: bool = True*, *reset: bool = False*, *options: Optional[str] = None*, *direct_session: Optional[object] = None*)

727 total commands, 6 Sub-groups, 0 group commands

Initializes new RsCmwEvdoMeas 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=ni' - forces usage of National Instruments Visa
- 'QueryInstrumentStatus = False' - same as driver.utilities.instrument_status_checking = False
- 'DriverSetup=(WriteDelay = 20, ReadDelay = 5)' - Introduces delay of 20ms before each write and 5ms before each read
- 'DriverSetup=(OpcWaitMode = OpcQuery)' - mode for all the opc-synchronised write/reads. Other modes: StbPolling, StbPollingSlow, StbPollingSuperSlow
- 'DriverSetup=(AddTermCharToWriteBinBLock = True)' - Adds one additional LF to the end of the binary data (some instruments require that)
- 'DriverSetup=(AssureWriteWithTermChar = True)' - Makes sure each command/query is terminated with termination character. Default: Interface dependent
- 'DriverSetup=(TerminationCharacter = 'x')' - Sets the termination character for reading. Default: '<LF>' (LineFeed)
- 'DriverSetup=(IoSegmentSize = 10E3)' - Maximum size of one write/read segment. If transferred data is bigger, it is split to more segments
- 'DriverSetup=(OpcTimeout = 10000)' - same as driver.utilities.opc_timeout = 10000
- 'DriverSetup=(VisaTimeout = 5000)' - same as driver.utilities.visa_timeout = 5000

- 'DriverSetup=(ViClearExeMode = 255)' - Binary combination where 1 means performing viClear() on a certain interface as the very first command in init
- 'DriverSetup=(OpcQueryAfterWrite = True)' - same as driver.utilities.opc_query_after_write = True

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.

close() → None

Closes the active RsCmwEvdoMeas session.

classmethod from_existing_session(*session: object*, *options: Optional[str] = None*) → RsCmwEvdoMeas

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

Parameters

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

get_session_handle() → object

Returns the underlying session handle.

static list_resources(*expression: str = '?*::INSTR'*, *visa_select: Optional[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: '@ni', '@rs'

restore_all_repcaps_to_default() → None

Sets all the Group and Global repcaps to their initial values

Subgroups

7.1 Route

SCPI Commands

```
ROUTe:EVDO:MEASurement<Instance>
```

class Route

Route commands group definition. 6 total commands, 1 Sub-groups, 1 group commands

class ValueStruct

Structure for reading output parameters. Fields:

- Scenario: enums.TestScenarioB: SALone | CSPath SALone: Standalone (non-signaling) CSPath: Combined signal path
- Controller: str: string Controlling application while scenario CSPath is active.
- Rx_Connector: enums.RxConnector: RF connector for the input path
- Rx_Converter: enums.RxConverter: RX module for the input path

get_value() → ValueStruct

```
# SCPI: ROUTe:EVDO:MEASurement<instance>
value: ValueStruct = driver.route.get_value()
```

Returns the configured routing settings. For possible connector and converter values, see 'Values for RF Path Selection'.

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.clone()
```

Subgroups

7.1.1 Scenario

SCPI Commands

```
ROUTe:EVDO:MEASurement<Instance>:SCENario:SALone
ROUTe:EVDO:MEASurement<Instance>:SCENario:CSPath
ROUTe:EVDO:MEASurement<Instance>:SCENario
```

class Scenario

Scenario commands group definition. 5 total commands, 2 Sub-groups, 3 group commands

class SaloneStruct

Structure for reading output parameters. Fields:

- Rx_Connector: enums.RxConnector: RF connector for the input path
- Rx_Converter: enums.RxConverter: RX module for the input path

get_cspath() → str

```
# SCPI: ROUTe:EVDO:MEASurement<instance>:SCENario:CSPath
value: str = driver.route.scenario.get_cspath()
```

Activates the combined signal path scenario and selects a master. The master controls the signal routing, analyzer settings and AT signal info settings while the combined signal path scenario is active. Configure the connector and converter settings via ROUTe:EVDO:SIGN<i>:SCENario:.... Depending on the installed options, the set of masters available at your instrument can differ from the values listed below. A complete list of all supported values can be displayed using method RsCmwEvdoMeas.Route.Scenario.Catalog.cspath.

return master: string String parameter containing the master application, e.g. '1xEV-DO Sig1' or '1xEV-DO Sig2'

get_salone() → SaloneStruct

```
# SCPI: ROUTe:EVDO:MEASurement<instance>:SCENario:SALone
value: SaloneStruct = driver.route.scenario.get_salone()
```

Activates the standalone scenario and selects the RF input path for the measured RF signal. For possible connector and converter values, see 'Values for RF Path Selection'.

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

get_value() → RsCmwEvdoMeas.enums.TestScenarioB

```
# SCPI: ROUTe:EVDO:MEASurement<instance>:SCENario
value: enums.TestScenarioB = driver.route.scenario.get_value()
```

Returns the active scenario.

return scenario: SALone | CSPath SALone: Standalone (non-signaling) CSPath: Combined signal path

set_cspath(*master: str*) → None

```
# SCPI: ROUTe:EVDO:MEASurement<instance>:SCENario:CSPath
driver.route.scenario.set_cspath(master = '1')
```

Activates the combined signal path scenario and selects a master. The master controls the signal routing, analyzer settings and AT signal info settings while the combined signal path scenario is active. Configure the connector and converter settings via ROUTe:EVDO:SIGN<i>:SCENario:.... Depending on the installed options, the set of masters available at your instrument can differ from the values listed below. A complete list of all supported values can be displayed using method RsCmwEvdoMeas.Route.Scenario.Catalog.cspath.

param master string String parameter containing the master application, e.g. '1xEV-DO Sig1' or '1xEV-DO Sig2'

set_salone(*value: RsCmwEvdoMeas.Implementations.Route_.Scenario.Scenario.SaloneStruct*) → None

```
# SCPI: ROUTe:EVDO:MEASurement<instance>:SCENario:SALone
driver.route.scenario.set_salone(value = SaloneStruct())
```

Activates the standalone scenario and selects the RF input path for the measured RF signal. For possible connector and converter values, see 'Values for RF Path Selection'.

param value see the help for SaloneStruct structure arguments.

Cloning the Group

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

Subgroups

7.1.1.1 MaProtocol

SCPI Commands

```
ROUTe:EVDO:MEASurement<Instance>:SCENario:MAPRotocol
```

class MaProtocol

MaProtocol commands group definition. 1 total commands, 0 Sub-groups, 1 group commands

set() → None

```
# SCPI: ROUTe:EVDO:MEASurement<instance>:SCENario:MAPRotocol
driver.route.scenario.maProtocol.set()
```

No command help available

set_with_opc() → None

```
# SCPI: ROUTe:EVDO:MEASurement<instance>:SCENario:MAPRotocol
driver.route.scenario.maProtocol.set_with_opc()
```

No command help available

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

7.1.1.2 Catalog

SCPI Commands

```
ROUTe:EVDO:MEASurement<Instance>:SCENario:CATalog:CSPath
```

class Catalog

Catalog commands group definition. 1 total commands, 0 Sub-groups, 1 group commands

get_cspath() → List[str]

```
# SCPI: ROUTe:EVDO:MEASurement<instance>:SCENario:CATalog:CSPath
value: List[str] = driver.route.scenario.catalog.get_cspath()
```

Lists all applications that can be set as master for the combined signal path scenario using method RsCmwEvdoMeas.Route. Scenario.cspath.

return source_list: string Comma-separated list. Each supported value is represented as a string.

7.2 Configure

SCPI Commands

```
CONFigure:EVDO:MEASurement<Instance>:DISPlay
```

class Configure

Configure commands group definition. 101 total commands, 3 Sub-groups, 1 group commands

get_display() → RsCmwEvdoMeas.enums.Tab

```
# SCPI: CONFigure:EVDO:MEASurement<Instance>:DISPlay
value: enums.Tab = driver.configure.get_display()
```

Selects the view to be shown when the display is switched on during remote control.

return tab: MEVA | OLTR Multi-evaluation - overview, OLTR view

set_display(*tab: RsCmwEvdoMeas.enums.Tab*) → None

```
# SCPI: CONFigure:EVDO:MEASurement<Instance>:DISPlay
driver.configure.set_display(tab = enums.Tab.MEVA)
```

Selects the view to be shown when the display is switched on during remote control.

param tab MEVA | OLTR Multi-evaluation - overview, OLTR view

Cloning the Group

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

Subgroups

7.2.1 RfSettings

SCPI Commands

```
CONFigure:EVDO:MEASurement<Instance>:RFSettings:EATTenuation
CONFigure:EVDO:MEASurement<Instance>:RFSettings:UMARgin
CONFigure:EVDO:MEASurement<Instance>:RFSettings:ENPower
CONFigure:EVDO:MEASurement<Instance>:RFSettings:FREQuency
CONFigure:EVDO:MEASurement<Instance>:RFSettings:CHANnel
CONFigure:EVDO:MEASurement<Instance>:RFSettings:FOFFset
CONFigure:EVDO:MEASurement<Instance>:RFSettings:BCLass
```

class RfSettings

RfSettings commands group definition. 7 total commands, 0 Sub-groups, 7 group commands

get_bclass() → RsCmwEvdoMeas.enums.BandClass

```
# SCPI: CONFigure:EVDO:MEASurement<instance>:RFSettings:BCLass
value: enums.BandClass = driver.configure.rfSettings.get_bclass()
```

Selects the band class (BC) . If the current center frequency (method RsCmwEvdoMeas.Configure.RfSettings.frequency) is valid for this band class, the corresponding channel number (method RsCmwEvdoMeas.Configure.RfSettings.channel) is also calculated and set. See also 'Band Classes' For the combined signal path scenario, useCONFigure:EVDO:SIGN<i>:RFSettings:BCLass.

return band_class: USC | KCEL | NAPC | TACS | JTAC | KPCS | N45T | IM2K | NA7C | B18M | NA9C | NA8S | PA4M | PA8M | IEXT | USPC | AWS | U25B | PS7C | LO7C | LBANd | SBANd USC: BC 0, US-Cellular KCEL: BC 0, Korean Cellular NAPC: BC 1, North American PCS TACS: BC 2, TACS Band JTAC: BC 3, JTACS Band KPCS: BC 4, Korean PCS N45T: BC 5, NMT-450 IM2K: BC 6, IMT-2000 NA7C: BC 7, Upper 700 MHz B18M: BC 8, 1800 MHz Band NA9C: BC 9, North American 900 MHz NA8S: BC 10, Secondary 800 MHz PA4M: BC 11, European 400 MHz PAMR PA8M: BC 12, 800 MHz PAMR IEXT: BC 13, IMT-2000 2.5 GHz Extension USPC: BC 14, US PCS 1900 MHz AWS: BC 15, AWS Band U25B: BC 16, US 2.5 GHz Band PS7C: BC 18, Public Safety Band 700 MHz LO7C: BC 19, Lower 700 MHz LBAN: BC 20, L-Band SBAN: BC 21, S-Band

get_channel() → int

```
# SCPI: CONFigure:EVDO:MEASurement<instance>:RFSettings:CHANnel
value: int = driver.configure.rfSettings.get_channel()
```

Selects the channel number. The channel number must be valid for the current band class, for dependencies see 'Band Classes'. The corresponding center frequency (method RsCmwEv-

doMeas.Configure.RfSettings.frequency) is calculated and set. For the combined signal path scenario, useCONFigure:EVDO:SIGN<i>:RFSettings:CHANnel.

return channel: integer Range: depends on selected band class

get_eattenuation() → float

```
# SCPI: CONFigure:EVDO:MEASurement<instance>:RFSettings:EATTenuation
value: float = driver.configure.rfSettings.get_eattenuation()
```

Defines an external attenuation (or gain, if the value is negative) , to be applied to the input connector. INTRO_CMD_HELP: For the combined signal path scenario, use:

- SOURce:EVDO:SIGN<i>:RFSettings:RX:EATTenuation
- CONFigure:EVDO:SIGN<i>:RFSettings:EATTenuation

return rf_input_ext_att: numeric Range: -50 dB to 90 dB, Unit: dB

get_envelope_power() → float

```
# SCPI: CONFigure:EVDO:MEASurement<instance>:RFSettings:ENPower
value: float = driver.configure.rfSettings.get_envelope_power()
```

Sets the expected nominal power of the measured RF signal. INTRO_CMD_HELP: For the combined signal path scenario, use:

- CONFigure:EVDO:SIGN<i>:RFPower:EPMode
- CONFigure:EVDO:SIGN<i>:RFPower:MANual
- CONFigure:EVDO:SIGN<i>:RFPower:EXPected

return exp_nom_power: numeric 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 data sheet. Unit: dBm

get_foffset() → float

```
# SCPI: CONFigure:EVDO:MEASurement<instance>:RFSettings:FOFFset
value: float = driver.configure.rfSettings.get_foffset()
```

Selects a positive or negative offset frequency to be added to the center frequency (method RsCmwEvdoMeas.Configure. RfSettings.frequency) . For the combined signal path scenario, useCONFigure:EVDO:SIGN<i>:RFSettings:FOFFset.

return freq_offset: numeric Range: -50 kHz to 50 kHz, Unit: Hz

get_frequency() → float

```
# SCPI: CONFigure:EVDO:MEASurement<instance>:RFSettings:FREQuency
value: float = driver.configure.rfSettings.get_frequency()
```

Selects the center frequency of the RF analyzer. If the center frequency is valid for the current band class, the corresponding channel number is also calculated and set.

INTRO_CMD_HELP: For the combined signal path scenario, use:

- CONFigure:EVDO:SIGN<i>:RFSettings:FREQuency
- CONFigure:EVDO:SIGN<i>:RFSettings:RLFRequency or
- CONFigure:EVDO:SIGN<i>:RFSettings:CHANnel

The supported frequency range depends on the instrument model and the available options. The supported range can be smaller than stated here. Refer to the preface of your model-specific base unit manual.

return frequency: numeric Range: 100 MHz to 6 GHz, Unit: Hz

get_umargin() → float

```
# SCPI: CONFigure:EVDO:MEASurement<instance>:RFSettings:UMARgin
value: float = driver.configure.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 data sheet.

return user_margin: numeric Range: 0 dB to (55 dB + External Attenuation - Expected Nominal Power) , Unit: dB

set_bclass(*band_class: RsCmwEvdoMeas.enums.BandClass*) → None

```
# SCPI: CONFigure:EVDO:MEASurement<instance>:RFSettings:BCLass
driver.configure.rfSettings.set_bclass(band_class = enums.BandClass.AWS)
```

Selects the band class (BC) . If the current center frequency (method RsCmwEvdoMeas.Configure.RfSettings.frequency) is valid for this band class, the corresponding channel number (method RsCmwEvdoMeas.Configure.RfSettings.channel) is also calculated and set. See also 'Band Classes' For the combined signal path scenario, useCONFigure:EVDO:SIGN<i>:RFSettings:BCLass.

param band_class USC | KCEL | NAPC | TACS | JTAC | KPCS | N45T | IM2K | NA7C | B18M | NA9C | NA8S | PA4M | PA8M | IEXT | USPC | AWS | U25B | PS7C | LO7C | LBANd | SBANd USC: BC 0, US-Cellular KCEL: BC 0, Korean Cellular NAPC: BC 1, North American PCS TACS: BC 2, TACS Band JTAC: BC 3, JTACS Band KPCS: BC 4, Korean PCS N45T: BC 5, NMT-450 IM2K: BC 6, IMT-2000 NA7C: BC 7, Upper 700 MHz B18M: BC 8, 1800 MHz Band NA9C: BC 9, North American 900 MHz NA8S: BC 10, Secondary 800 MHz PA4M: BC 11, European 400 MHz PAMR PA8M: BC 12, 800 MHz PAMR IEXT: BC 13, IMT-2000 2.5 GHz Extension USPC: BC 14, US PCS 1900 MHz AWS: BC 15, AWS Band U25B: BC 16, US 2.5 GHz Band PS7C: BC 18, Public Safety Band 700 MHz LO7C: BC 19, Lower 700 MHz LBAN: BC 20, L-Band SBAN: BC 21, S-Band

set_channel(*channel: int*) → None

```
# SCPI: CONFigure:EVDO:MEASurement<instance>:RFSettings:CHANnel
driver.configure.rfSettings.set_channel(channel = 1)
```

Selects the channel number. The channel number must be valid for the current band class, for dependencies see 'Band Classes'. The corresponding center frequency (method RsCmwEvdoMeas.Configure.RfSettings.frequency) is calculated and set. For the combined signal path scenario, useCONFigure:EVDO:SIGN<i>:RFSettings:CHANnel.

param channel integer Range: depends on selected band class

set_eattenuation(*rf_input_ext_att: float*) → None

```
# SCPI: CONFigure:EVDO:MEASurement<instance>:RFSettings:EATTenuation
driver.configure.rfSettings.set_eattenuation(rf_input_ext_att = 1.0)
```

Defines an external attenuation (or gain, if the value is negative) , to be applied to the input connector. INTRO_CMD_HELP: For the combined signal path scenario, use:

- SOURce:EVDO:SIGN<i>:RFSettings:RX:EATTenuation
- CONFigure:EVDO:SIGN<i>:RFSettings:EATTenuation

param rf_input_ext_att numeric Range: -50 dB to 90 dB, Unit: dB

set_envelope_power(*exp_nom_power: float*) → None

```
# SCPI: CONFigure:EVDO:MEASurement<instance>:RFSettings:ENPower
driver.configure.rfSettings.set_envelope_power(exp_nom_power = 1.0)
```

Sets the expected nominal power of the measured RF signal. INTRO_CMD_HELP: For the combined signal path scenario, use:

- CONFigure:EVDO:SIGN<i>:RFPower:EPMode
- CONFigure:EVDO:SIGN<i>:RFPower:MANual
- CONFigure:EVDO:SIGN<i>:RFPower:EXPected

param exp_nom_power numeric 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 data sheet. Unit: dBm

set_foffset(*freq_offset: float*) → None

```
# SCPI: CONFigure:EVDO:MEASurement<instance>:RFSettings:FOFFset
driver.configure.rfSettings.set_foffset(freq_offset = 1.0)
```

Selects a positive or negative offset frequency to be added to the center frequency (method RsCmwEvdoMeas.Configure. RfSettings.frequency) . For the combined signal path scenario, useCONFigure:EVDO:SIGN<i>:RFSettings:FOFFset.

param freq_offset numeric Range: -50 kHz to 50 kHz, Unit: Hz

set_frequency(*frequency: float*) → None

```
# SCPI: CONFigure:EVDO:MEASurement<instance>:RFSettings:FREQuency
driver.configure.rfSettings.set_frequency(frequency = 1.0)
```

Selects the center frequency of the RF analyzer. If the center frequency is valid for the current band class, the corresponding channel number is also calculated and set.

INTRO_CMD_HELP: For the combined signal path scenario, use:

- CONFigure:EVDO:SIGN<i>:RFSettings:FREQuency

- CONFigure:EVDO:SIGN<i>:RFSettings:RLFRequency or
- CONFigure:EVDO:SIGN<i>:RFSettings:CHANnel

The supported frequency range depends on the instrument model and the available options. The supported range can be smaller than stated here. Refer to the preface of your model-specific base unit manual.

param frequency numeric Range: 100 MHz to 6 GHz, Unit: Hz

set_umargin(*user_margin: float*) → None

```
# SCPI: CONFigure:EVDO:MEASurement<instance>:RFSettings:UMARgin
driver.configure.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 data sheet.

param user_margin numeric Range: 0 dB to (55 dB + External Attenuation - Expected Nominal Power) , Unit: dB

7.2.2 MultiEval

SCPI Commands

```
CONFigure:EVDO:MEASurement<Instance>:MEValuation:TOUT
CONFigure:EVDO:MEASurement<Instance>:MEValuation:DMODulation
CONFigure:EVDO:MEASurement<Instance>:MEValuation:HSLot
CONFigure:EVDO:MEASurement<Instance>:MEValuation:DRC
CONFigure:EVDO:MEASurement<Instance>:MEValuation:ACK
CONFigure:EVDO:MEASurement<Instance>:MEValuation:ACKDsc
CONFigure:EVDO:MEASurement<Instance>:MEValuation:DATA
CONFigure:EVDO:MEASurement<Instance>:MEValuation:APILot
CONFigure:EVDO:MEASurement<Instance>:MEValuation:REPetition
CONFigure:EVDO:MEASurement<Instance>:MEValuation:MOEXception
CONFigure:EVDO:MEASurement<Instance>:MEValuation:PLSubtype
CONFigure:EVDO:MEASurement<Instance>:MEValuation:SCONdition
CONFigure:EVDO:MEASurement<Instance>:MEValuation:SFACtor
CONFigure:EVDO:MEASurement<Instance>:MEValuation:ILCMask
CONFigure:EVDO:MEASurement<Instance>:MEValuation:QLCMask
CONFigure:EVDO:MEASurement<Instance>:MEValuation:RPMode
CONFigure:EVDO:MEASurement<Instance>:MEValuation:IQLCheck
```

class MultiEval

MultiEval commands group definition. 82 total commands, 6 Sub-groups, 17 group commands

get_ack() → RsCmwEvdoMeas.enums.MeasCondition

```
# SCPI: CONFigure:EVDO:MEASurement<instance>:MEValuation:ACK
value: enums.MeasCondition = driver.configure.multiEval.get_ack()
```

Specify a measurement condition for the selected carrier (see method RsCmwEvdoMeas.Configure.MultiEval.Carrier.select) based on the presence of the acknowledgment chan-

nel. Value ACK is only relevant for physical layer protocol subtype 0/1 (see method RsCmwEvdoMeas.Configure.MultiEval.plSubtype) .

return ack: DNCare | ON | OFF OFF: do not evaluate the signal regardless of whether it is active or not. ON: evaluate the signal only when the ACK channel is present. Otherwise the CMW returns invalid results (INV) . DNCare: evaluate the signal irrespective of the presence of the channel.

get_ack_dsc() → RsCmwEvdoMeas.enums.AckDsc

```
# SCPI: CONFigure:EVDO:MEASurement<instance>:MEValuation:ACKDsc
value: enums.AckDsc = driver.configure.multiEval.get_ack_dsc()
```

Specify a measurement condition for the selected carrier (see method RsCmwEvdoMeas.Configure.MultiEval.Carrier.select) based on the presence of the acknowledgment channel / data source control channel. Value DSC is only relevant for physical layer protocol subtypes 2 and 3 (see method RsCmwEvdoMeas.Configure.MultiEval.plSubtype) .

return ack_dsc: DNCare | DSC | ACK | OFF OFF: evaluate the signal only when no DSC or ACK channels are present DSC: evaluate the signal only when the DSC channel is present ACK: evaluate the signal only when the ACK channel is present DNCare: evaluate the signal irrespective of the presence of the channels

get_apilot() → RsCmwEvdoMeas.enums.MeasCondition

```
# SCPI: CONFigure:EVDO:MEASurement<instance>:MEValuation:APILot
value: enums.MeasCondition = driver.configure.multiEval.get_apilot()
```

Specifies a measurement condition for the selected carrier (see method RsCmwEvdoMeas.Configure.MultiEval.Carrier.select) based on the presence of the auxiliary pilot channel. The condition is only relevant for physical layer protocol subtypes 2 and 3 (see method RsCmwEvdoMeas.Configure.MultiEval.plSubtype) .

return apilot: DNCare | ON | OFF DNCare: evaluate the signal irrespective of the presence of the channel ON: evaluate the signal only when the channel is present OFF: evaluate the signal only when the channel is not present

get_data() → RsCmwEvdoMeas.enums.MeasCondition

```
# SCPI: CONFigure:EVDO:MEASurement<instance>:MEValuation:DATA
value: enums.MeasCondition = driver.configure.multiEval.get_data()
```

Specifies a measurement condition for the selected carrier (see method RsCmwEvdoMeas.Configure.MultiEval.Carrier.select) based on the presence of the data channel.

return code_chs_data: DNCare | ON | OFF DNCare: evaluate the signal irrespective of the presence of the channel ON: evaluate the signal only when the channel is present OFF: evaluate the signal only when the channel is not present

get_dmodulation() → RsCmwEvdoMeas.enums.Dmodulation

```
# SCPI: CONFigure:EVDO:MEASurement<instance>:MEValuation:DMODulation
value: enums.Dmodulation = driver.configure.multiEval.get_dmodulation()
```

Specifies the data channel modulation type of the selected carrier (see method RsCmwEvdoMeas.Configure.MultiEval.Carrier. select) . This setting is only relevant for measurements using physical layer subtype 2 or 3 (see method RsCmwEvdoMeas. Configure.MultiEval.plSubtype) .

return dmodulation: AUTO | B4 | Q4 | Q2 | Q4Q2 | E4E2 AUTO: automatic detection of the modulation type. Signals with unrecognized modulation type are ignored. B4: BPSK modulation with 4-ary Walsh cover (W24) Q4: QPSK modulation with 4-ary Walsh cover (W24) Q2: QPSK modulation with 2-ary Walsh cover (W12) Q4Q2: (QPSK, W24) + (QPSK, W12) E4E2: (8-PSK, W24) + (8-PSK, W12)

get_drc() → RsCmwEvdoMeas.enums.MeasCondition

```
# SCPI: CONFigure:EVDO:MEASurement<instance>:MEValuation:DRC
value: enums.MeasCondition = driver.configure.multiEval.get_drc()
```

Specifies a measurement condition for the selected carrier (see method RsCmwEvdoMeas.Configure.MultiEval.Carrier.select) based on the presence of the data rate control (DRC) channel.

return dr_control: DNCare | ON | OFF DNCare: evaluate the signal irrespective of the presence of the channel ON: evaluate the signal only when the channel is present OFF: evaluate the signal only when the channel is not present

get_hslot() → RsCmwEvdoMeas.enums.HalfSlot

```
# SCPI: CONFigure:EVDO:MEASurement<instance>:MEValuation:HSLot
value: enums.HalfSlot = driver.configure.multiEval.get_hslot()
```

Specifies which half-slots of the code channel is/are evaluated for measurements using physical layer subtype 2 or 3 (see method RsCmwEvdoMeas.Configure.MultiEval.plSubtype) . Consider that the DSC channel and the ACK channel are transmitted time-multiplexed on Walsh channel W1232. The ACK is transmitted on the first half-slot and the DSC on the second half-slot.

return hslot: FHSLot | SHSLot | BHSLots FHSLot: evaluate the first half-slot SHSLot: evaluate the second half-slot BHSLots: evaluate both half-slots

get_ilc_mask() → float

```
# SCPI: CONFigure:EVDO:MEASurement<instance>:MEValuation:ILCMask
value: float = driver.configure.multiEval.get_ilc_mask()
```

Specifies the long code mask for the I branch.

return lc_mask_i: numeric Range: #H0 to #H3FFFFFFFFFF

get_iql_check() → bool

```
# SCPI: CONFigure:EVDO:MEASurement<instance>:MEValuation:IQLCheck
value: bool = driver.configure.multiEval.get_iql_check()
```

Enables or disables the CDP I/Q leakage check.

return iq_leakage_check: OFF | ON ON: enable check OFF: disable check

get_mo_exception() → bool

```
# SCPI: CONFigure:EVDO:MEASurement<instance>:MEValuation:MOEXception
value: bool = driver.configure.multiEval.get_mo_exception()
```

Specifies whether measurement results that the R&S CMW identifies as faulty or inaccurate are rejected.

return meas_on_exception: OFF | ON ON: Results are never rejected OFF: Faulty results are rejected

get_pl_subtype() → RsCmwEvdoMeas.enums.PlSubtype

```
# SCPI: CONFigure:EVDO:MEASurement<instance>:MEValuation:PLSubtype
value: enums.PlSubtype = driver.configure.multiEval.get_pl_subtype()
```

Selects the physical layer protocol subtype. For the combined signal path scenario, use CONFigure:EVDO:SIGN<i>:NETWork:RELease.

return pl_subtype: ST01 | ST2 | ST3 ST01: subtype 0 or 1 ST2: subtype 2 ST3: subtype 3

get_qlcmask() → float

```
# SCPI: CONFigure:EVDO:MEASurement<instance>:MEValuation:QLCMask
value: float = driver.configure.multiEval.get_qlcmask()
```

Specifies the long code mask for the Q branch.

return lc_mask_q: numeric Range: #H0 to #H3FFFFFFFFF

get_repetition() → RsCmwEvdoMeas.enums.Repeat

```
# SCPI: CONFigure:EVDO:MEASurement<instance>:MEValuation:REPetition
value: enums.Repeat = driver.configure.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 | CONTinuous SINGleshot: Single-shot measurement CONTinuous: Continuous measurement

get_rp_mode() → RsCmwEvdoMeas.enums.RefPowerMode

```
# SCPI: CONFigure:EVDO:MEASurement<instance>:MEValuation:RPMode
value: enums.RefPowerMode = driver.configure.multiEval.get_rp_mode()
```

Sets the reference power relative to which the power (in dB) of the reverse link physical channels of both the I and Q signal are measured.

return ref_power_mode: ATPower | PPOWer ATPower: total channel power PPOWer: pilot power

get_scondition() → RsCmwEvdoMeas.enums.StopConditionB

```
# SCPI: CONFigure:EVDO:MEASurement<instance>:MEValuation:SCONdition
value: enums.StopConditionB = driver.configure.multiEval.get_scondition()
```

Qualifies whether the measurement is stopped after a failed limit check or continued. OLFail means that the measurement is stopped (STOP:...MEAS<i>...) and reaches the RDY state when one of the results exceeds the limits.

return stop_condition: NONE | OLFail NONE: Continue measurement irrespective of the limit check OLFail: Stop measurement on limit failure

get_sfactor() → RsCmwEvdoMeas.enums.Srate

```
# SCPI: CONFigure:EVDO:MEASurement<instance>:MEValuation:SFACtor
value: enums.Srate = driver.configure.multiEval.get_sfactor()
```

Queries the spreading factor. The spreading factor cannot be set directly but depends on the physical layer protocol subtype (method RsCmwEvdoMeas.Configure.MultiEval.plSubtype) .

return srate: SF16 | SF32

get_timeout() → float

```
# SCPI: CONFigure:EVDO:MEASurement<instance>:MEValuation:TOUT
value: float = driver.configure.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 ([ON | OFF] key or [RESTART | STOP] key) . 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: numeric Unit: s

set_ack(*ack: RsCmwEvdoMeas.enums.MeasCondition*) → None

```
# SCPI: CONFigure:EVDO:MEASurement<instance>:MEValuation:ACK
driver.configure.multiEval.set_ack(ack = enums.MeasCondition.DNCare)
```

Specify a measurement condition for the selected carrier (see method RsCmwEvdoMeas.Configure.MultiEval.Carrier.select) based on the presence of the acknowledgment channel. Value ACK is only relevant for physical layer protocol subtype 0/1 (see method RsCmwEvdoMeas.Configure.MultiEval.plSubtype) .

param ack DNCare | ON | OFF OFF: do not evaluate the signal regardless of whether it is active or not. ON: evaluate the signal only when the ACK channel is present. Otherwise the CMW returns invalid results (INV) . DNCare: evaluate the signal irrespective of the presence of the channel.

set_ack_dsc(*ack_dsc: RsCmwEvdoMeas.enums.AckDsc*) → None

```
# SCPI: CONFigure:EVDO:MEASurement<instance>:MEValuation:ACKDsc
driver.configure.multiEval.set_ack_dsc(ack_dsc = enums.AckDsc.ACK)
```

Specify a measurement condition for the selected carrier (see method RsCmwEvdoMeas.Configure.MultiEval.Carrier.select) based on the presence of the acknowledgment channel / data source control channel. Value DSC is only relevant for physical layer protocol subtypes 2 and 3 (see method RsCmwEvdoMeas.Configure.MultiEval.plSubtype) .

param ack_dsc DNCare | DSC | ACK | OFF OFF: evaluate the signal only when no DSC or ACK channels are present DSC: evaluate the signal only when the DSC channel is present ACK: evaluate the signal only when the ACK channel is present DNCare: evaluate the signal irrespective of the presence of the channels

set_apilot(*apilot: RsCmwEvdoMeas.enums.MeasCondition*) → None

```
# SCPI: CONFigure:EVDO:MEASurement<instance>:MEValuation:APILot
driver.configure.multiEval.set_apilot(apilot = enums.MeasCondition.DNCare)
```

Specifies a measurement condition for the selected carrier (see method RsCmwEvdoMeas.Configure.MultiEval.Carrier.select) based on the presence of the auxiliary pilot channel. The condition is only relevant for physical layer protocol subtypes 2 and 3 (see method RsCmwEvdoMeas.Configure.MultiEval.plSubtype) .

param apilot DNCare | ON | OFF DNCare: evaluate the signal irrespective of the presence of the channel ON: evaluate the signal only when the channel is present OFF: evaluate the signal only when the channel is not present

set_data(*code_chs_data: RsCmwEvdoMeas.enums.MeasCondition*) → None

```
# SCPI: CONFigure:EVDO:MEASurement<instance>:MEValuation:DATA
driver.configure.multiEval.set_data(code_chs_data = enums.MeasCondition.DNCare)
```

Specifies a measurement condition for the selected carrier (see method RsCmwEvdoMeas.Configure.MultiEval.Carrier.select) based on the presence of the data channel.

param code_chs_data DNCare | ON | OFF DNCare: evaluate the signal irrespective of the presence of the channel ON: evaluate the signal only when the channel is present OFF: evaluate the signal only when the channel is not present

set_dmodulation(*dmodulation: RsCmwEvdoMeas.enums.Dmodulation*) → None

```
# SCPI: CONFigure:EVDO:MEASurement<instance>:MEValuation:DMODulation
driver.configure.multiEval.set_dmodulation(dmodulation = enums.Dmodulation.AUTO)
```

Specifies the data channel modulation type of the selected carrier (see method RsCmwEvdoMeas.Configure.MultiEval.Carrier. select) . This setting is only relevant for measurements using physical layer subtype 2 or 3 (see method RsCmwEvdoMeas. Configure.MultiEval.plSubtype) .

param dmodulation AUTO | B4 | Q4 | Q2 | Q4Q2 | E4E2 AUTO: automatic detection of the modulation type. Signals with unrecognized modulation type are ignored. B4: BPSK modulation with 4-ary Walsh cover (W24) Q4: QPSK modulation with 4-ary Walsh cover (W24) Q2: QPSK modulation with 2-ary Walsh cover (W12) Q4Q2: (QPSK, W24) + (QPSK, W12) E4E2: (8-PSK, W24) + (8-PSK, W12)

set_drc(*dr_control: RsCmwEvdoMeas.enums.MeasCondition*) → None

```
# SCPI: CONFigure:EVDO:MEASurement<instance>:MEValuation:DRC
driver.configure.multiEval.set_drc(dr_control = enums.MeasCondition.DNCare)
```

Specifies a measurement condition for the selected carrier (see method RsCmwEvdoMeas.Configure.MultiEval.Carrier.select) based on the presence of the data rate control (DRC) channel.

param dr_control DNCare | ON | OFF DNCare: evaluate the signal irrespective of the presence of the channel ON: evaluate the signal only when the channel is present OFF: evaluate the signal only when the channel is not present

set_hslot(*hslot: RsCmwEvdoMeas.enums.HalfSlot*) → None

```
# SCPI: CONFigure:EVDO:MEASurement<instance>:MEValuation:HSLot
driver.configure.multiEval.set_hslot(hslot = enums.HalfSlot.BHSLots)
```

Specifies which half-slots of the code channel is/are evaluated for measurements using physical layer subtype 2 or 3 (see method RsCmwEvdoMeas.Configure.MultiEval.plSubtype) . Consider that the DSC channel and the ACK channel are transmitted time-multiplexed on Walsh channel W1232. The ACK is transmitted on the first half-slot and the DSC on the second half-slot.

param hslot FHSLot | SHSLot | BHSLots FHSLot: evaluate the first half-slot SHSLot: evaluate the second half-slot BHSLots: evaluate both half-slots

set_ilc_mask(*lc_mask_i: float*) → None

```
# SCPI: CONFigure:EVDO:MEASurement<instance>:MEValuation:ILCMask
driver.configure.multiEval.set_ilc_mask(lc_mask_i = 1.0)
```

Specifies the long code mask for the I branch.

param lc_mask_i numeric Range: #H0 to #H3FFFFFFFFFF

set_iql_check(*iq_leakage_check: bool*) → None

```
# SCPI: CONFigure:EVDO:MEASurement<instance>:MEValuation:IQLCheck
driver.configure.multiEval.set_iql_check(iq_leakage_check = False)
```

Enables or disables the CDP I/Q leakage check.

param iq_leakage_check OFF | ON ON: enable check OFF: disable check

set_mo_exception(*meas_on_exception: bool*) → None

```
# SCPI: CONFigure:EVDO:MEASurement<instance>:MEValuation:MOEXception
driver.configure.multiEval.set_mo_exception(meas_on_exception = False)
```

Specifies whether measurement results that the R&S CMW identifies as faulty or inaccurate are rejected.

param meas_on_exception OFF | ON ON: Results are never rejected OFF: Faulty results are rejected

set_pl_subtype(*pl_subtype: RsCmwEvdoMeas.enums.PlSubtype*) → None

```
# SCPI: CONFigure:EVDO:MEASurement<instance>:MEValuation:PLSubtype
driver.configure.multiEval.set_pl_subtype(pl_subtype = enums.PlSubtype.ST01)
```

Selects the physical layer protocol subtype. For the combined signal path scenario, use CONFigure:EVDO:SIGN<i>:NETWork:RELease.

param pl_subtype ST01 | ST2 | ST3 ST01: subtype 0 or 1 ST2: subtype 2 ST3: subtype 3

set_qlcmask(*lc_mask_q: float*) → None

```
# SCPI: CONFigure:EVDO:MEASurement<instance>:MEValuation:QLCMask
driver.configure.multiEval.set_qlcmask(lc_mask_q = 1.0)
```

Specifies the long code mask for the Q branch.

param lc_mask_q numeric Range: #H0 to #H3FFFFFFFFF

set_repetition(*repetition: RsCmwEvdoMeas.enums.Repeat*) → None

```
# SCPI: CONFigure:EVDO:MEASurement<instance>:MEValuation:REPetition
driver.configure.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 | CONTinuous SINGleshot: Single-shot measurement CONTinuous: Continuous measurement

set_rp_mode(*ref_power_mode: RsCmwEvdoMeas.enums.RefPowerMode*) → None

```
# SCPI: CONFigure:EVDO:MEASurement<instance>:MEValuation:RPMode
driver.configure.multiEval.set_rp_mode(ref_power_mode = enums.RefPowerMode.
↪ATPower)
```

Sets the reference power relative to which the power (in dB) of the reverse link physical channels of both the I and Q signal are measured.

param ref_power_mode ATPower | PPOWer ATPower: total channel power PPOWer: pilot power

set_scondition(*stop_condition: RsCmwEvdoMeas.enums.StopConditionB*) → None

```
# SCPI: CONFigure:EVDO:MEASurement<instance>:MEValuation:SCONdition
driver.configure.multiEval.set_scondition(stop_condition = enums.StopConditionB.
↪NONE)
```

Qualifies whether the measurement is stopped after a failed limit check or continued. OLFail means that the measurement is stopped (STOP:...MEAS<i>...) and reaches the RDY state when one of the results exceeds the limits.

param stop_condition NONE | OLFail NONE: Continue measurement irrespective of the limit check OLFail: Stop measurement on limit failure

set_sfactor(*srate: RsCmwEvdoMeas.enums.Srate*) → None

```
# SCPI: CONFigure:EVDO:MEASurement<instance>:MEValuation:SFACtor
driver.configure.multiEval.set_sfactor(srate = enums.Srate.SF16)
```

Queries the spreading factor. The spreading factor cannot be set directly but depends on the physical layer protocol subtype (method RsCmwEvdoMeas.Configure.MultiEval.plSubtype) .

param srate No help available

set_timeout(*timeout: float*) → None

```
# SCPI: CONFigure:EVDO:MEASurement<instance>:MEValuation:TOUT
driver.configure.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 ([ON | OFF] key or [RESTART | STOP] key) . 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 numeric Unit: s

Cloning the Group

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

Subgroups

7.2.2.1 Scount

SCPI Commands

```
CONFigure:EVDO:MEASurement<Instance>:MEValuation:SCOunt:MODulation
CONFigure:EVDO:MEASurement<Instance>:MEValuation:SCOunt:SPECtrum
```

class Scount

Scount commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

get_modulation() → int

```
# SCPI: CONFigure:EVDO:MEASurement<instance>:MEValuation:SCOunt:MODulation
value: int = driver.configure.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.

return scount_mod: numeric Number of measurement intervals. Range: 1 to 1000

get_spectrum() → int

```
# SCPI: CONFigure:EVDO:MEASurement<instance>:MEValuation:SCOunt:SPECtrum
value: int = driver.configure.multiEval.scount.get_spectrum()
```

Specifies the statistic count of the measurement. The statistic count is equal to the number of measurement intervals per single shot.

return scount_spectrum: numeric Number of measurement intervals. Range: 1 to 1000

set_modulation(*scount_mod: int*) → None

```
# SCPI: CONFigure:EVDO:MEASurement<instance>:MEValuation:SCOunt:MODulation
driver.configure.multiEval.scount.set_modulation(scount_mod = 1)
```

Specifies the statistic count of the measurement. The statistic count is equal to the number of measurement intervals per single shot.

param scount_mod numeric Number of measurement intervals. Range: 1 to 1000

set_spectrum(*scount_spectrum: int*) → None

```
# SCPI: CONFigure:EVDO:MEASurement<instance>:MEValuation:SCOunt:SPECtrum
driver.configure.multiEval.scount.set_spectrum(scount_spectrum = 1)
```

Specifies the statistic count of the measurement. The statistic count is equal to the number of measurement intervals per single shot.

param scount_spectrum numeric Number of measurement intervals. Range: 1 to 1000

7.2.2.2 Carrier

SCPI Commands

```
CONFigure:EVDO:MEASurement<Instance>:MEValuation:CARRier:SETTing
CONFigure:EVDO:MEASurement<Instance>:MEValuation:CARRier:ENABle
CONFigure:EVDO:MEASurement<Instance>:MEValuation:CARRier:SELect
CONFigure:EVDO:MEASurement<Instance>:MEValuation:CARRier:FOFFset
CONFigure:EVDO:MEASurement<Instance>:MEValuation:CARRier:FREQuency
CONFigure:EVDO:MEASurement<Instance>:MEValuation:CARRier:WBFilter
```

class Carrier

Carrier commands group definition. 6 total commands, 0 Sub-groups, 6 group commands

get_enable() → bool

```
# SCPI: CONFigure:EVDO:MEASurement<instance>:MEValuation:CARRier:ENABle
value: bool = driver.configure.multiEval.carrier.get_enable()
```

Defines whether a carrier is measured (ON) or not (OFF) . The related carrier has to be pre-set using the method RsCmwEvdoMeas.Configure.MultiEval.Carrier.setting command. All carriers can be queried, but carrier 0 cannot be set (fix set to ON) .

INTRO_CMD_HELP: For the combined signal path scenario, use:

- CONFigure:EVDO:SIGN<i>:NETWork:PILot:AN:ACTive
- CONFigure:EVDO:SIGN<i>:NETWork:PILot:AT:ASSigned

return cenable: OFF | ON

get_foffset() → float

```
# SCPI: CONFigure:EVDO:MEASurement<instance>:MEValuation:CARRier:FOFFset
value: float = driver.configure.multiEval.carrier.get_foffset()
```

Gets/sets the frequency offset of a selected carrier relative to carrier 0. The related carrier has to be pre-set using the method RsCmwEvdoMeas.Configure.MultiEval.Carrier.setting command. All carriers can be queried, but carrier 0 cannot be set. This command is relevant only for standalone mode. While the combined signal path scenario is active, the command for carrier frequency offset is not used.

return cf_offset: numeric The offset relative to carrier 0. The maximum distance between carriers is restricted to 8 MHz. Range: - 8 MHz to + 8 MHz, Unit: Hz

get_frequency() → float

```
# SCPI: CONFigure:EVDO:MEASurement<instance>:MEValuation:CARRier:FREQuency
value: float = driver.configure.multiEval.carrier.get_frequency()
```

Gets/sets the frequency of a selected carrier. The related carrier has to be pre-set using the method RsCmwEvdoMeas. Configure.MultiEval.Carrier.setting command. All carriers can be queried, but only carrier 0 can be set. The frequencies of the other carriers are set implicitly via method RsCmwEvdoMeas.Configure.MultiEval.Carrier.foffset.

INTRO_CMD_HELP: For the combined signal path scenario, use:

- CONFigure:EVDO:SIGN<i>:CARRier:CHANnel or
- CONFigure:EVDO:SIGN<i>:CARRier:RLFRequency

The supported frequency range depends on the instrument model and the available options. The supported range can be smaller than stated here. Refer to the preface of your model-specific base unit manual.

return cfrequency: numeric Range: 100 MHz to 6 GHz, Unit: Hz

get_select() → int

```
# SCPI: CONFigure:EVDO:MEASurement<instance>:MEValuation:CARRier:SELect
value: int = driver.configure.multiEval.carrier.get_select()


    INTRO_CMD_HELP: Defines the selected carrier, also displayed at the GUI.␣
↪The GUI displays the results for this carrier (if the carrier is enabled) .␣
↪Results retrieved via remote command and the following remote commands are␣
↪also related to this carrier:
```

(continues on next page)

(continued from previous page)

```
    - method RsCmwEvdoMeas.Configure.MultiEval.drc
    - method RsCmwEvdoMeas.Configure.MultiEval.data
    - method RsCmwEvdoMeas.Configure.MultiEval.apilot
    - method RsCmwEvdoMeas.Configure.MultiEval.ackDsc
    - method RsCmwEvdoMeas.Configure.MultiEval.dmodulation
    - method RsCmwEvdoMeas.Configure.MultiEval.Acp.Foffsets.lower
    - method RsCmwEvdoMeas.Configure.MultiEval.Acp.Foffsets.upper
    - method RsCmwEvdoMeas.Configure.MultiEval.Acp.Extended.Foffsets.lower
    - method RsCmwEvdoMeas.Configure.MultiEval.Acp.Extended.Foffsets.upper
    - CONFigure:EVDO:MEAS<i>:MEValuation:LIMit:ACP
    - CONFigure:EVDO:MEAS<i>:MEValuation:LIMit:ACP
    - CONFigure:EVDO:MEAS<i>:MEValuation:LIMit:ACP:EXTended
    - CONFigure:EVDO:MEAS<i>:MEValuation:LIMit:ACP:EXTended
    - method RsCmwEvdoMeas.Configure.MultiEval.Acp.Rbw.lower
    - method RsCmwEvdoMeas.Configure.MultiEval.Acp.Rbw.upper
    - method RsCmwEvdoMeas.Configure.MultiEval.Acp.Extended.Rbw.lower
    - method RsCmwEvdoMeas.Configure.MultiEval.Acp.Extended.Rbw.upper

    :return: selected_carrier: integer Range: 0 to 2
```

get_setting() → int

```
# SCPI: CONFigure:EVDO:MEASurement<instance>:MEValuation:CARRier:SETTing
value: int = driver.configure.multiEval.carrier.get_setting()


    INTRO_CMD_HELP: Selects a carrier for the following carrier settings:

    - method RsCmwEvdoMeas.Configure.MultiEval.Carrier.enable
    - method RsCmwEvdoMeas.Configure.MultiEval.Carrier.foffset
    - method RsCmwEvdoMeas.Configure.MultiEval.Carrier.frequency
    INTRO_CMD_HELP: For the combined signal path scenario, use:

    - CONFigure:EVDO:SIGN<i>:CARRier:SETTing or
    - CONFigure:EVDO:SIGN<i>:PILot:SETTing

    :return: set_carrier: numeric Range: 0 to 2
```

get_wb_filter() → RsCmwEvdoMeas.enums.WbFilter

```
# SCPI: CONFigure:EVDO:MEASurement<instance>:MEValuation:CARRier:WBFilter
value: enums.WbFilter = driver.configure.multiEval.carrier.get_wb_filter()
```

Selects the bandwidth of the wideband filter, used to measure the 'AT Power (wideband) ' of a single-carrier configuration. For a multi-carrier configuration, the bandwidth can only be queried (equals 16 MHz) .

return wb_filter: F8M0 | F16M0 F8M0: 8 MHz filter bandwidth F16M0: 16 MHz filter bandwidth

set_enable(*cenable: bool*) → None

```
# SCPI: CONFigure:EVDO:MEASurement<instance>:MEValuation:CARRier:ENABle
driver.configure.multiEval.carrier.set_enable(cenable = False)
```

Defines whether a carrier is measured (ON) or not (OFF) . The related carrier has to be pre-set using the method RsCmwEvdoMeas.Configure.MultiEval.Carrier.setting command. All carriers can be queried, but carrier 0 cannot be set (fix set to ON) .

INTRO_CMD_HELP: For the combined signal path scenario, use:

- CONFigure:EVDO:SIGN<i>:NETWork:PILot:AN:ACTive
- CONFigure:EVDO:SIGN<i>:NETWork:PILot:AT:ASSigned

param cenable OFF | ON

set_foffset(*cf_offset: float*) → None

```
# SCPI: CONFigure:EVDO:MEASurement<instance>:MEValuation:CARRier:FOFFset
driver.configure.multiEval.carrier.set_foffset(cf_offset = 1.0)
```

Gets/sets the frequency offset of a selected carrier relative to carrier 0. The related carrier has to be pre-set using the method RsCmwEvdoMeas.Configure.MultiEval.Carrier.setting command. All carriers can be queried, but carrier 0 cannot be set. This command is relevant only for standalone mode. While the combined signal path scenario is active, the command for carrier frequency offset is not used.

param cf_offset numeric The offset relative to carrier 0. The maximum distance between carriers is restricted to 8 MHz. Range: - 8 MHz to + 8 MHz, Unit: Hz

set_frequency(*cfrequency: float*) → None

```
# SCPI: CONFigure:EVDO:MEASurement<instance>:MEValuation:CARRier:FREQuency
driver.configure.multiEval.carrier.set_frequency(cfrequency = 1.0)
```

Gets/sets the frequency of a selected carrier. The related carrier has to be pre-set using the method RsCmwEvdoMeas. Configure.MultiEval.Carrier.setting command. All carriers can be queried, but only carrier 0 can be set. The frequencies of the other carriers are set implicitly via method RsCmwEvdoMeas.Configure.MultiEval.Carrier.foffset.

INTRO_CMD_HELP: For the combined signal path scenario, use:

- CONFigure:EVDO:SIGN<i>:CARRier:CHANnel or
- CONFigure:EVDO:SIGN<i>:CARRier:RLFRequency

The supported frequency range depends on the instrument model and the available options. The supported range can be smaller than stated here. Refer to the preface of your model-specific base unit manual.

param cfrequency numeric Range: 100 MHz to 6 GHz, Unit: Hz

set_select(*selected_carrier: int*) → None

```
# SCPI: CONFigure:EVDO:MEASurement<instance>:MEValuation:CARRier:SELect
driver.configure.multiEval.carrier.set_select(selected_carrier = 1)


    INTRO_CMD_HELP: Defines the selected carrier, also displayed at the GUI.␣
↪The GUI displays the results for this carrier (if the carrier is enabled) ␣
↪Results retrieved via remote command and the following remote commands are␣
↪also related to this carrier:
```

(continues on next page)

(continued from previous page)

```
    - method RsCmwEvdoMeas.Configure.MultiEval.drc
    - method RsCmwEvdoMeas.Configure.MultiEval.data
    - method RsCmwEvdoMeas.Configure.MultiEval.apilot
    - method RsCmwEvdoMeas.Configure.MultiEval.ackDsc
    - method RsCmwEvdoMeas.Configure.MultiEval.dmodulation
    - method RsCmwEvdoMeas.Configure.MultiEval.Acp.Foffsets.lower
    - method RsCmwEvdoMeas.Configure.MultiEval.Acp.Foffsets.upper
    - method RsCmwEvdoMeas.Configure.MultiEval.Acp.Extended.Foffsets.lower
    - method RsCmwEvdoMeas.Configure.MultiEval.Acp.Extended.Foffsets.upper
    - CONFigure:EVDO:MEAS<i>:MEValuation:LIMit:ACP
    - CONFigure:EVDO:MEAS<i>:MEValuation:LIMit:ACP
    - CONFigure:EVDO:MEAS<i>:MEValuation:LIMit:ACP:EXTended
    - CONFigure:EVDO:MEAS<i>:MEValuation:LIMit:ACP:EXTended
    - method RsCmwEvdoMeas.Configure.MultiEval.Acp.Rbw.lower
    - method RsCmwEvdoMeas.Configure.MultiEval.Acp.Rbw.upper
    - method RsCmwEvdoMeas.Configure.MultiEval.Acp.Extended.Rbw.lower
    - method RsCmwEvdoMeas.Configure.MultiEval.Acp.Extended.Rbw.upper

    :param selected_carrier: integer Range: 0 to 2
```

set_setting(*set_carrier: int*) → None

```
# SCPI: CONFigure:EVDO:MEASurement<instance>:MEValuation:CARRier:SETTing
driver.configure.multiEval.carrier.set_setting(set_carrier = 1)


    INTRO_CMD_HELP: Selects a carrier for the following carrier settings:

    - method RsCmwEvdoMeas.Configure.MultiEval.Carrier.enable
    - method RsCmwEvdoMeas.Configure.MultiEval.Carrier.foffset
    - method RsCmwEvdoMeas.Configure.MultiEval.Carrier.frequency
    INTRO_CMD_HELP: For the combined signal path scenario, use:

    - CONFigure:EVDO:SIGN<i>:CARRier:SETTing or
    - CONFigure:EVDO:SIGN<i>:PILot:SETTing

    :param set_carrier: numeric Range: 0 to 2
```

set_wb_filter(*wb_filter: RsCmwEvdoMeas.enums.WbFilter*) → None

```
# SCPI: CONFigure:EVDO:MEASurement<instance>:MEValuation:CARRier:WBFilter
driver.configure.multiEval.carrier.set_wb_filter(wb_filter = enums.WbFilter.
↪F16M0)
```

Selects the bandwidth of the wideband filter, used to measure the 'AT Power (wideband) ' of a single-carrier configuration. For a multi-carrier configuration, the bandwidth can only be queried (equals 16 MHz) .

param wb_filter F8M0 | F16M0 F8M0: 8 MHz filter bandwidth F16M0: 16 MHz filter bandwidth

7.2.2.3 Acp

class Acp
Acp commands group definition. 8 total commands, 3 Sub-groups, 0 group commands

Cloning the Group

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

Subgroups

7.2.2.3.1 Foffsets

SCPI Commands

```
CONFigure:EVDO:MEASurement<Instance>:MEValuation:ACP:FOFFsets:LOWer
CONFigure:EVDO:MEASurement<Instance>:MEValuation:ACP:FOFFsets:UPPer
```

class Foffsets
Foffsets commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

get_lower() → List[float]

```
# SCPI: CONFigure:EVDO:MEASurement<instance>:MEValuation:ACP:FOFFsets:LOWer
value: List[float or bool] = driver.configure.multiEval.acp.foffsets.get_lower()
```

Defines the negative (lower) frequency offsets to be used for ACP measurements of the selected carrier (see method RsCmwEvdoMeas.Configure.MultiEval.Carrier.select) . The offsets are defined relative to the analyzer frequency. Up to 10 offsets can be defined and enabled. The offset index 0 to 9 corresponds to the index used in manual control.

return frequency_offset: No help available

get_upper() → List[float]

```
# SCPI: CONFigure:EVDO:MEASurement<instance>:MEValuation:ACP:FOFFsets:UPPer
value: List[float or bool] = driver.configure.multiEval.acp.foffsets.get_upper()
```

Defines the positive (upper) frequency offsets to be used for ACP measurements of the selected carrier (see method RsCmwEvdoMeas.Configure.MultiEval.Carrier.select) . The offsets are defined relative to the analyzer frequency. Up to 10 offsets can be defined and enabled. The offset index 0 to 9 corresponds to the index used in manual control.

return frequency_offset: No help available

set_lower(*frequency_offset: List[float]*) → None

```
# SCPI: CONFigure:EVDO:MEASurement<instance>:MEValuation:ACP:FOFFsets:LOWer
driver.configure.multiEval.acp.foffsets.set_lower(frequency_offset = [1.1, True,
↪ 2.2, False, 3.3])
```

Defines the negative (lower) frequency offsets to be used for ACP measurements of the selected carrier (see method RsCmwEvdoMeas.Configure.MultiEval.Carrier.select) . The offsets are defined relative to the analyzer frequency. Up to 10 offsets can be defined and enabled. The offset index 0 to 9 corresponds to the index used in manual control.

param frequency_offset numeric | ON | OFF Range: -4 MHz to 0 MHz, Unit: MHz
Additional parameters: OFF | ON (disables | enables the offset)

set_upper(*frequency_offset: List[float]*) → None

```
# SCPI: CONFigure:EVDO:MEASurement<instance>:MEValuation:ACP:FOFFsets:UPPer
driver.configure.multiEval.acp.foffsets.set_upper(frequency_offset = [1.1, True,
↪ 2.2, False, 3.3])
```

Defines the positive (upper) frequency offsets to be used for ACP measurements of the selected carrier (see method RsCmwEvdoMeas.Configure.MultiEval.Carrier.select) . The offsets are defined relative to the analyzer frequency. Up to 10 offsets can be defined and enabled. The offset index 0 to 9 corresponds to the index used in manual control.

param frequency_offset numeric | ON | OFF Range: 0 MHz to 4 MHz, Unit: MHz
Additional parameters: OFF | ON (disables | enables the offset)

7.2.2.3.2 Extended

class Extended
Extended commands group definition. 4 total commands, 2 Sub-groups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.configure.multiEval.acp.extended.clone()
```

Subgroups

7.2.2.3.2.1 Foffsets

SCPI Commands

```
CONFigure:EVDO:MEASurement<Instance>:MEValuation:ACP:EXTended:FOFFsets:LOWer
CONFigure:EVDO:MEASurement<Instance>:MEValuation:ACP:EXTended:FOFFsets:UPPer
```

class Foffsets
Foffsets commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

get_lower() → List[float]

```
# SCPI: CONFigure:EVDO:MEASurement<instance>
↪:MEValuation:ACP:EXTended:FOFFsets:LOWer
value: List[float or bool] = driver.configure.multiEval.acp.extended.foffsets.
↪get_lower()
```

Defines the negative (lower) frequency offsets 19 to 0, to be used for extended ACP measurements of the selected carrier (see method RsCmwEvdoMeas.Configure.MultiEval.Carrier.select) . The offsets are defined relative to the analyzer frequency. Up to 20 offsets can be defined and enabled. The offset index 19 to 0 corresponds to the index used in manual control.

return frequency_offset: Range: -4 MHz to 0 MHz Additional parameters: OFF | ON (disables | enables the offset)

get_upper() → List[float]

```
# SCPI: CONFigure:EVDO:MEASurement<instance>
↪:MEValuation:ACP:EXTended:FOFFsets:UPPer
value: List[float or bool] = driver.configure.multiEval.acp.extended.foffsets.
↪get_upper()
```

Defines the positive (upper) frequency offsets 0 to 19, to be used for extended ACP measurements of the selected carrier (see method RsCmwEvdoMeas.Configure.MultiEval.Carrier.select) . The offsets are defined relative to the analyzer frequency. Up to 20 offsets can be defined and enabled. The offset index 0 to 19 corresponds to the index used in manual control.

return frequency_offset: Range: 0 MHz to 4 MHz , Unit: Hz Additional parameters: OFF | ON (disables | enables the offset)

set_lower(*frequency_offset: List[float]*) → None

```
# SCPI: CONFigure:EVDO:MEASurement<instance>
↪:MEValuation:ACP:EXTended:FOFFsets:LOWer
driver.configure.multiEval.acp.extended.foffsets.set_lower(frequency_offset =␣
↪[1.1, True, 2.2, False, 3.3])
```

Defines the negative (lower) frequency offsets 19 to 0, to be used for extended ACP measurements of the selected carrier (see method RsCmwEvdoMeas.Configure.MultiEval.Carrier.select) . The offsets are defined relative to the analyzer frequency. Up to 20 offsets can be defined and enabled. The offset index 19 to 0 corresponds to the index used in manual control.

param frequency_offset Range: -4 MHz to 0 MHz Additional parameters: OFF | ON (disables | enables the offset)

set_upper(*frequency_offset: List[float]*) → None

```
# SCPI: CONFigure:EVDO:MEASurement<instance>
↪:MEValuation:ACP:EXTended:FOFFsets:UPPer
driver.configure.multiEval.acp.extended.foffsets.set_upper(frequency_offset =␣
↪[1.1, True, 2.2, False, 3.3])
```

Defines the positive (upper) frequency offsets 0 to 19, to be used for extended ACP measurements of the selected carrier (see method RsCmwEvdoMeas.Configure.MultiEval.Carrier.select) . The offsets are defined relative to the analyzer frequency. Up to 20 offsets can be defined and enabled. The offset index 0 to 19 corresponds to the index used in manual control.

param frequency_offset Range: 0 MHz to 4 MHz , Unit: Hz Additional parameters: OFF | ON (disables | enables the offset)

7.2.2.3.2.2 Rbw

SCPI Commands

```
CONFigure:EVDO:MEASurement<Instance>:MEValuation:ACP:EXTended:RBW:LOWer
CONFigure:EVDO:MEASurement<Instance>:MEValuation:ACP:EXTended:RBW:UPPer
```

class Rbw

Rbw commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

get_lower() → List[RsCmwEvdoMeas.enums.Rbw]

```
# SCPI: CONFigure:EVDO:MEASurement<instance>:MEValuation:ACP:EXTended:RBW:LOWer
value: List[enums.Rbw] = driver.configure.multiEval.acp.extended.rbw.get_lower()
```

Defines the resolution bandwidth to be used for lower frequency offset 19 to 0 of the selected carrier for extended ACP measurements (see method RsCmwEvdoMeas.Configure.MultiEval.Carrier.select) .

return rbw: F1K0 | F6K25 | F10K | F12K5 | F25K | F30K | F50K | F100k | F1M0 | F1M23 F1K0: 1 kHz F6K25: 6.25 kHz F10K: 10 kHz F12K5: 12.5 kHz F25K: 25 kHz F30K: 30 kHz F50K: 50 kHz F100k: 100 kHz F1M0: 1 MHz F1M23: 1.23 MHz Unit: Hz

get_upper() → List[RsCmwEvdoMeas.enums.Rbw]

```
# SCPI: CONFigure:EVDO:MEASurement<instance>:MEValuation:ACP:EXTended:RBW:UPPer
value: List[enums.Rbw] = driver.configure.multiEval.acp.extended.rbw.get_upper()
```

Defines the resolution bandwidth to be used for upper frequency offset 0 to 9 of the selected carrier for extended ACP measurements (see method RsCmwEvdoMeas.Configure.MultiEval.Carrier.select) .

return rbw: F1K0 | F6K25 | F10K | F12K5 | F25K | F30K | F50K | F100k | F1M0 | F1M23 F1K0: 1 kHz F6K25: 6.25 kHz F10K: 10 kHz F12K5: 12.5 kHz F25K: 25 kHz F30K: 30 kHz F50K: 50 kHz F100k: 100 kHz F1M0: 1 MHz F1M23: 1.23 MHz Unit: Hz

set_lower(*rbw: List[RsCmwEvdoMeas.enums.Rbw]*) → None

```
# SCPI: CONFigure:EVDO:MEASurement<instance>:MEValuation:ACP:EXTended:RBW:LOWer
driver.configure.multiEval.acp.extended.rbw.set_lower(rbw = [Rbw.F100k, Rbw.
↪F6K25])
```

Defines the resolution bandwidth to be used for lower frequency offset 19 to 0 of the selected carrier for extended ACP measurements (see method RsCmwEvdoMeas.Configure.MultiEval.Carrier.select) .

param rbw F1K0 | F6K25 | F10K | F12K5 | F25K | F30K | F50K | F100k | F1M0 | F1M23 F1K0: 1 kHz F6K25: 6.25 kHz F10K: 10 kHz F12K5: 12.5 kHz F25K: 25 kHz F30K: 30 kHz F50K: 50 kHz F100k: 100 kHz F1M0: 1 MHz F1M23: 1.23 MHz Unit: Hz

set_upper(*rbw: List[RsCmwEvdoMeas.enums.Rbw]*) → None

```
# SCPI: CONFigure:EVDO:MEASurement<instance>:MEValuation:ACP:EXTended:RBW:UPPer
driver.configure.multiEval.acp.extended.rbw.set_upper(rbw = [Rbw.F100k, Rbw.
↪F6K25])
```

Defines the resolution bandwidth to be used for upper frequency offset 0 to 9 of the selected carrier for extended ACP measurements (see method RsCmwEvdoMeas.Configure.MultiEval.Carrier.select) .

param rbw F1K0 | F6K25 | F10K | F12K5 | F25K | F30K | F50K | F100k | F1M0 | F1M23 F1K0: 1 kHz F6K25: 6.25 kHz F10K: 10 kHz F12K5: 12.5 kHz F25K: 25 kHz F30K: 30 kHz F50K: 50 kHz F100k: 100 kHz F1M0: 1 MHz F1M23: 1.23 MHz Unit: Hz

7.2.2.3.3 Rbw

SCPI Commands

```
CONFigure:EVDO:MEASurement<Instance>:MEValuation:ACP:RBW:LOWer
CONFigure:EVDO:MEASurement<Instance>:MEValuation:ACP:RBW:UPPer
```

class Rbw

Rbw commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

get_lower() → List[RsCmwEvdoMeas.enums.Rbw]

```
# SCPI: CONFigure:EVDO:MEASurement<instance>:MEValuation:ACP:RBW:LOWer
value: List[enums.Rbw] = driver.configure.multiEval.acp.rbw.get_lower()
```

Defines the resolution bandwidth to be used for lower frequency offset 9 to 0 of the selected carrier for ACP measurements (see method RsCmwEvdoMeas.Configure.MultiEval.Carrier.select) .

return rbw: F1K0 | F6K25 | F10K | F12K5 | F25K | F30K | F50K | F100k | F1M0 | F1M23 F1K0: 1 kHz F6K25: 6.25 kHz F10K: 10 kHz F12K5: 12.5 kHz F25K: 25 kHz F30K: 30 kHz F50K: 50 kHz F100k: 100 kHz F1M0: 1 MHz F1M23: 1.23 MHz Unit: Hz

get_upper() → List[RsCmwEvdoMeas.enums.Rbw]

```
# SCPI: CONFigure:EVDO:MEASurement<instance>:MEValuation:ACP:RBW:UPPer
value: List[enums.Rbw] = driver.configure.multiEval.acp.rbw.get_upper()
```

Defines the resolution bandwidth to be used for upper frequency offset 0 to 9 of the selected carrier for ACP measurements (see method RsCmwEvdoMeas.Configure.MultiEval.Carrier.select) .

return rbw: F1K0 | F6K25 | F10K | F12K5 | F25K | F30K | F50K | F100k | F1M0 | F1M23 F1K0: 1 kHz F6K25: 6.25 kHz F10K: 10 kHz F12K5: 12.5 kHz F25K: 25 kHz F30K: 30 kHz F50K: 50 kHz F100k: 100 kHz F1M0: 1 MHz F1M23: 1.23 MHz Unit: Hz

set_lower(*rbw: List[RsCmwEvdoMeas.enums.Rbw]*) → None

```
# SCPI: CONFigure:EVDO:MEASurement<instance>:MEValuation:ACP:RBW:LOWer
driver.configure.multiEval.acp.rbw.set_lower(rbw = [Rbw.F100k, Rbw.F6K25])
```

Defines the resolution bandwidth to be used for lower frequency offset 9 to 0 of the selected carrier for ACP measurements (see method RsCmwEvdoMeas.Configure.MultiEval.Carrier.select) .

param rbw F1K0 | F6K25 | F10K | F12K5 | F25K | F30K | F50K | F100k | F1M0 | F1M23 F1K0: 1 kHz F6K25: 6.25 kHz F10K: 10 kHz F12K5: 12.5 kHz F25K: 25 kHz F30K: 30 kHz F50K: 50 kHz F100k: 100 kHz F1M0: 1 MHz F1M23: 1.23 MHz Unit: Hz

set_upper(*rbw: List[RsCmwEvdoMeas.enums.Rbw]*) → None

```
# SCPI: CONFigure:EVDO:MEASurement<instance>:MEValuation:ACP:RBW:UPPer
driver.configure.multiEval.acp.rbw.set_upper(rbw = [Rbw.F100k, Rbw.F6K25])
```

Defines the resolution bandwidth to be used for upper frequency offset 0 to 9 of the selected carrier for ACP measurements (see method RsCmwEvdoMeas.Configure.MultiEval.Carrier.select) .

param rbw F1K0 | F6K25 | F10K | F12K5 | F25K | F30K | F50K | F100k | F1M0 | F1M23 F1K0: 1 kHz F6K25: 6.25 kHz F10K: 10 kHz F12K5: 12.5 kHz F25K: 25 kHz F30K: 30 kHz F50K: 50 kHz F100k: 100 kHz F1M0: 1 MHz F1M23: 1.23 MHz Unit: Hz

7.2.2.4 Result

SCPI Commands

```
CONFigure:EVDO:MEASurement<Instance>:MEValuation:RESult:ALL
CONFigure:EVDO:MEASurement<Instance>:MEValuation:RESult:EVMagnitude
CONFigure:EVDO:MEASurement<Instance>:MEValuation:RESult:MERRor
CONFigure:EVDO:MEASurement<Instance>:MEValuation:RESult:PERRor
CONFigure:EVDO:MEASurement<Instance>:MEValuation:RESult:CDP
CONFigure:EVDO:MEASurement<Instance>:MEValuation:RESult:CDE
CONFigure:EVDO:MEASurement<Instance>:MEValuation:RESult:ACP
CONFigure:EVDO:MEASurement<Instance>:MEValuation:RESult:POWer
CONFigure:EVDO:MEASurement<Instance>:MEValuation:RESult:TXM
CONFigure:EVDO:MEASurement<Instance>:MEValuation:RESult:CP
CONFigure:EVDO:MEASurement<Instance>:MEValuation:RESult:OBW
CONFigure:EVDO:MEASurement<Instance>:MEValuation:RESult:IQ
```

class Result

Result commands group definition. 12 total commands, 0 Sub-groups, 12 group commands

class AllStruct

Structure for reading output parameters. Fields:

- Evm: bool: OFF | ON Error vector magnitude ON: Evaluate results and show the view OFF: Do not evaluate results, hide the view
- Magnitude_Error: bool: OFF | ON Magnitude error
- Phase_Error: bool: OFF | ON Phase error
- Acp: bool: OFF | ON Adjacent channel power
- Cdp: bool: OFF | ON Code domain power
- Cde: bool: OFF | ON Code domain error
- Power: bool: OFF | ON Power

- Tx_Measurements: bool: OFF | ON TX measurement
- Channel_Power: bool: OFF | ON Channel power
- Obw: bool: OFF | ON Occupied bandwidth
- Iq: bool: OFF | ON IQ

get_acp() → bool

```
# SCPI: CONFigure:EVDO:MEASurement<instance>:MEValuation:RESult:ACP
value: bool = driver.configure.multiEval.result.get_acp()
```

Enables or disables the evaluation of results and shows or hides the views in the multi-evaluation measurement. The mnemonic after 'RESult' denotes the view type: Error vector magnitude, magnitude error, phase error, adjacent channel power, occupied bandwidth, code domain power, code domain error, channel power, IQ, power and TX measurement.

return enable: OFF | ON ON: Evaluate results and show view OFF: Do not evaluate results, hide view

get_all() → AllStruct

```
# SCPI: CONFigure:EVDO:MEASurement<instance>:MEValuation:RESult[:ALL]
value: AllStruct = driver.configure.multiEval.result.get_all()
```

Enables or disables the evaluation of results and shows or hides the views in the multi-evaluation measurement. This command combines all other CONFigure:EVDO:MEAS<i>:MEValuation:RESult... commands.

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

get_cde() → bool

```
# SCPI: CONFigure:EVDO:MEASurement<instance>:MEValuation:RESult:CDE
value: bool = driver.configure.multiEval.result.get_cde()
```

Enables or disables the evaluation of results and shows or hides the views in the multi-evaluation measurement. The mnemonic after 'RESult' denotes the view type: Error vector magnitude, magnitude error, phase error, adjacent channel power, occupied bandwidth, code domain power, code domain error, channel power, IQ, power and TX measurement.

return enable: OFF | ON ON: Evaluate results and show view OFF: Do not evaluate results, hide view

get_cdp() → bool

```
# SCPI: CONFigure:EVDO:MEASurement<instance>:MEValuation:RESult:CDP
value: bool = driver.configure.multiEval.result.get_cdp()
```

Enables or disables the evaluation of results and shows or hides the views in the multi-evaluation measurement. The mnemonic after 'RESult' denotes the view type: Error vector magnitude, magnitude error, phase error, adjacent channel power, occupied bandwidth, code domain power, code domain error, channel power, IQ, power and TX measurement.

return enable: OFF | ON ON: Evaluate results and show view OFF: Do not evaluate results, hide view

get_cp() → bool

```
# SCPI: CONFigure:EVDO:MEASurement<instance>:MEValuation:RESult:CP
value: bool = driver.configure.multiEval.result.get_cp()
```

Enables or disables the evaluation of results and shows or hides the views in the multi-evaluation measurement. The mnemonic after 'RESult' denotes the view type: Error vector magnitude, magnitude error, phase error, adjacent channel power, occupied bandwidth, code domain power, code domain error, channel power, IQ, power and TX measurement.

return enable: OFF | ON ON: Evaluate results and show view OFF: Do not evaluate results, hide view

get_ev_magnitude() → bool

```
# SCPI: CONFigure:EVDO:MEASurement<instance>:MEValuation:RESult:EVMagnitude
value: bool = driver.configure.multiEval.result.get_ev_magnitude()
```

Enables or disables the evaluation of results and shows or hides the views in the multi-evaluation measurement. The mnemonic after 'RESult' denotes the view type: Error vector magnitude, magnitude error, phase error, adjacent channel power, occupied bandwidth, code domain power, code domain error, channel power, IQ, power and TX measurement.

return enable: OFF | ON ON: Evaluate results and show view OFF: Do not evaluate results, hide view

get_iq() → bool

```
# SCPI: CONFigure:EVDO:MEASurement<instance>:MEValuation:RESult:IQ
value: bool = driver.configure.multiEval.result.get_iq()
```

Enables or disables the evaluation of results and shows or hides the views in the multi-evaluation measurement. The mnemonic after 'RESult' denotes the view type: Error vector magnitude, magnitude error, phase error, adjacent channel power, occupied bandwidth, code domain power, code domain error, channel power, IQ, power and TX measurement.

return enable: OFF | ON ON: Evaluate results and show view OFF: Do not evaluate results, hide view

get_merror() → bool

```
# SCPI: CONFigure:EVDO:MEASurement<instance>:MEValuation:RESult:MERRor
value: bool = driver.configure.multiEval.result.get_merror()
```

Enables or disables the evaluation of results and shows or hides the views in the multi-evaluation measurement. The mnemonic after 'RESult' denotes the view type: Error vector magnitude, magnitude error, phase error, adjacent channel power, occupied bandwidth, code domain power, code domain error, channel power, IQ, power and TX measurement.

return enable: OFF | ON ON: Evaluate results and show view OFF: Do not evaluate results, hide view

get_obw() → bool

```
# SCPI: CONFigure:EVDO:MEASurement<instance>:MEValuation:RESult:OBW
value: bool = driver.configure.multiEval.result.get_obw()
```

Enables or disables the evaluation of results and shows or hides the views in the multi-evaluation measurement. The mnemonic after 'RESult' denotes the view type: Error vector magnitude, magnitude error, phase error, adjacent channel power, occupied bandwidth, code domain power, code domain error, channel power, IQ, power and TX measurement.

return enable: OFF | ON ON: Evaluate results and show view OFF: Do not evaluate results, hide view

get_perror() → bool

```
# SCPI: CONFigure:EVDO:MEASurement<instance>:MEValuation:RESult:PERRor
value: bool = driver.configure.multiEval.result.get_perror()
```

Enables or disables the evaluation of results and shows or hides the views in the multi-evaluation measurement. The mnemonic after 'RESult' denotes the view type: Error vector magnitude, magnitude error, phase error, adjacent channel power, occupied bandwidth, code domain power, code domain error, channel power, IQ, power and TX measurement.

return enable: OFF | ON ON: Evaluate results and show view OFF: Do not evaluate results, hide view

get_power() → bool

```
# SCPI: CONFigure:EVDO:MEASurement<instance>:MEValuation:RESult:POWer
value: bool = driver.configure.multiEval.result.get_power()
```

Enables or disables the evaluation of results and shows or hides the views in the multi-evaluation measurement. The mnemonic after 'RESult' denotes the view type: Error vector magnitude, magnitude error, phase error, adjacent channel power, occupied bandwidth, code domain power, code domain error, channel power, IQ, power and TX measurement.

return enable: OFF | ON ON: Evaluate results and show view OFF: Do not evaluate results, hide view

get_txm() → bool

```
# SCPI: CONFigure:EVDO:MEASurement<instance>:MEValuation:RESult:TXM
value: bool = driver.configure.multiEval.result.get_txm()
```

Enables or disables the evaluation of results and shows or hides the views in the multi-evaluation measurement. The mnemonic after 'RESult' denotes the view type: Error vector magnitude, magnitude error, phase error, adjacent channel power, occupied bandwidth, code domain power, code domain error, channel power, IQ, power and TX measurement.

return enable: OFF | ON ON: Evaluate results and show view OFF: Do not evaluate results, hide view

set_acp(*enable: bool*) → None

```
# SCPI: CONFigure:EVDO:MEASurement<instance>:MEValuation:RESult:ACP
driver.configure.multiEval.result.set_acp(enable = False)
```

Enables or disables the evaluation of results and shows or hides the views in the multi-evaluation measurement. The mnemonic after 'RESult' denotes the view type: Error vector magnitude, magnitude error, phase error, adjacent channel power, occupied bandwidth, code domain power, code domain error, channel power, IQ, power and TX measurement.

param enable OFF | ON ON: Evaluate results and show view OFF: Do not evaluate results, hide view

set_all(*value: RsCmwEvdoMeas.Implementations.Configure_.MultiEval_.Result.Result.AllStruct*) → None

```
# SCPI: CONFigure:EVDO:MEASurement<instance>:MEValuation:RESult[:ALL]
driver.configure.multiEval.result.set_all(value = AllStruct())
```

Enables or disables the evaluation of results and shows or hides the views in the multi-evaluation measurement. This command combines all other CONFigure:EVDO:MEAS<i>:MEValuation:RESult... commands.

param value see the help for AllStruct structure arguments.

set_cde(*enable: bool*) → None

```
# SCPI: CONFigure:EVDO:MEASurement<instance>:MEValuation:RESult:CDE
driver.configure.multiEval.result.set_cde(enable = False)
```

Enables or disables the evaluation of results and shows or hides the views in the multi-evaluation measurement. The mnemonic after 'RESult' denotes the view type: Error vector magnitude, magnitude error, phase error, adjacent channel power, occupied bandwidth, code domain power, code domain error, channel power, IQ, power and TX measurement.

param enable OFF | ON ON: Evaluate results and show view OFF: Do not evaluate results, hide view

set_cdp(*enable: bool*) → None

```
# SCPI: CONFigure:EVDO:MEASurement<instance>:MEValuation:RESult:CDP
driver.configure.multiEval.result.set_cdp(enable = False)
```

Enables or disables the evaluation of results and shows or hides the views in the multi-evaluation measurement. The mnemonic after 'RESult' denotes the view type: Error vector magnitude, magnitude error, phase error, adjacent channel power, occupied bandwidth, code domain power, code domain error, channel power, IQ, power and TX measurement.

param enable OFF | ON ON: Evaluate results and show view OFF: Do not evaluate results, hide view

set_cp(*enable: bool*) → None

```
# SCPI: CONFigure:EVDO:MEASurement<instance>:MEValuation:RESult:CP
driver.configure.multiEval.result.set_cp(enable = False)
```

Enables or disables the evaluation of results and shows or hides the views in the multi-evaluation measurement. The mnemonic after 'RESult' denotes the view type: Error vector magnitude, magnitude error, phase error, adjacent channel power, occupied bandwidth, code domain power, code domain error, channel power, IQ, power and TX measurement.

param enable OFF | ON ON: Evaluate results and show view OFF: Do not evaluate results, hide view

set_ev_magnitude(*enable: bool*) → None

```
# SCPI: CONFigure:EVDO:MEASurement<instance>:MEValuation:RESult:EVMagnitude
driver.configure.multiEval.result.set_ev_magnitude(enable = False)
```

Enables or disables the evaluation of results and shows or hides the views in the multi-evaluation measurement. The mnemonic after 'RESult' denotes the view type: Error vector magnitude, magnitude error, phase error, adjacent channel power, occupied bandwidth, code domain power, code domain error, channel power, IQ, power and TX measurement.

param enable OFF | ON ON: Evaluate results and show view OFF: Do not evaluate results, hide view

set_iq(*enable: bool*) → None

```
# SCPI: CONFigure:EVDO:MEASurement<instance>:MEValuation:RESult:IQ
driver.configure.multiEval.result.set_iq(enable = False)
```

Enables or disables the evaluation of results and shows or hides the views in the multi-evaluation measurement. The mnemonic after 'RESult' denotes the view type: Error vector magnitude, magnitude error, phase error, adjacent channel power, occupied bandwidth, code domain power, code domain error, channel power, IQ, power and TX measurement.

param enable OFF | ON ON: Evaluate results and show view OFF: Do not evaluate results, hide view

set_merror(*enable: bool*) → None

```
# SCPI: CONFigure:EVDO:MEASurement<instance>:MEValuation:RESult:MERRor
driver.configure.multiEval.result.set_merror(enable = False)
```

Enables or disables the evaluation of results and shows or hides the views in the multi-evaluation measurement. The mnemonic after 'RESult' denotes the view type: Error vector magnitude, magnitude error, phase error, adjacent channel power, occupied bandwidth, code domain power, code domain error, channel power, IQ, power and TX measurement.

param enable OFF | ON ON: Evaluate results and show view OFF: Do not evaluate results, hide view

set_obw(*enable: bool*) → None

```
# SCPI: CONFigure:EVDO:MEASurement<instance>:MEValuation:RESult:OBW
driver.configure.multiEval.result.set_obw(enable = False)
```

Enables or disables the evaluation of results and shows or hides the views in the multi-evaluation measurement. The mnemonic after 'RESult' denotes the view type: Error vector magnitude, magnitude error,

phase error, adjacent channel power, occupied bandwidth, code domain power, code domain error, channel power, IQ, power and TX measurement.

param enable OFF | ON ON: Evaluate results and show view OFF: Do not evaluate results, hide view

set_perror(*enable: bool*) → None

```
# SCPI: CONFigure:EVDO:MEASurement<instance>:MEValuation:RESult:PERRor
driver.configure.multiEval.result.set_perror(enable = False)
```

Enables or disables the evaluation of results and shows or hides the views in the multi-evaluation measurement. The mnemonic after 'RESult' denotes the view type: Error vector magnitude, magnitude error, phase error, adjacent channel power, occupied bandwidth, code domain power, code domain error, channel power, IQ, power and TX measurement.

param enable OFF | ON ON: Evaluate results and show view OFF: Do not evaluate results, hide view

set_power(*enable: bool*) → None

```
# SCPI: CONFigure:EVDO:MEASurement<instance>:MEValuation:RESult:POWer
driver.configure.multiEval.result.set_power(enable = False)
```

Enables or disables the evaluation of results and shows or hides the views in the multi-evaluation measurement. The mnemonic after 'RESult' denotes the view type: Error vector magnitude, magnitude error, phase error, adjacent channel power, occupied bandwidth, code domain power, code domain error, channel power, IQ, power and TX measurement.

param enable OFF | ON ON: Evaluate results and show view OFF: Do not evaluate results, hide view

set_txm(*enable: bool*) → None

```
# SCPI: CONFigure:EVDO:MEASurement<instance>:MEValuation:RESult:TXM
driver.configure.multiEval.result.set_txm(enable = False)
```

Enables or disables the evaluation of results and shows or hides the views in the multi-evaluation measurement. The mnemonic after 'RESult' denotes the view type: Error vector magnitude, magnitude error, phase error, adjacent channel power, occupied bandwidth, code domain power, code domain error, channel power, IQ, power and TX measurement.

param enable OFF | ON ON: Evaluate results and show view OFF: Do not evaluate results, hide view

7.2.2.5 Limit

SCPI Commands

```
CONFigure:EVDO:MEASurement<Instance>:MEValuation:LIMit:IQOFfset
CONFigure:EVDO:MEASurement<Instance>:MEValuation:LIMit:IQIMbalance
CONFigure:EVDO:MEASurement<Instance>:MEValuation:LIMit:CFERror
CONFigure:EVDO:MEASurement<Instance>:MEValuation:LIMit:TTERror
CONFigure:EVDO:MEASurement<Instance>:MEValuation:LIMit:WQUality
CONFigure:EVDO:MEASurement<Instance>:MEValuation:LIMit:MAXPower
CONFigure:EVDO:MEASurement<Instance>:MEValuation:LIMit:MINPower
CONFigure:EVDO:MEASurement<Instance>:MEValuation:LIMit:CDP
CONFigure:EVDO:MEASurement<Instance>:MEValuation:LIMit:CDE
CONFigure:EVDO:MEASurement<Instance>:MEValuation:LIMit:CP
CONFigure:EVDO:MEASurement<Instance>:MEValuation:LIMit:DWCP
```

class Limit

Limit commands group definition. 30 total commands, 5 Sub-groups, 11 group commands

get_cde() → float

```
# SCPI: CONFigure:EVDO:MEASurement<instance>:MEValuation:LIMit:CDE
value: float or bool = driver.configure.multiEval.limit.get_cde()
```

Defines an upper limit for the code domain error.

return cde_user_limit: Range: -70 dB to 0 dB, Unit: dB

get_cdp() → float

```
# SCPI: CONFigure:EVDO:MEASurement<instance>:MEValuation:LIMit:CDP
value: float or bool = driver.configure.multiEval.limit.get_cdp()
```

Defines an upper limit for the code domain power of inactive channels.

return cdp_user_limit: Range: -70 dB to 0 dB, Unit: dB

get_cf_error() → float

```
# SCPI: CONFigure:EVDO:MEASurement<instance>:MEValuation:LIMit:CFERror
value: float or bool = driver.configure.multiEval.limit.get_cf_error()
```

Defines an upper limit for the carrier frequency error.

return cfreq_error: Range: 0 Hz to 1000 Hz, Unit: Hz Additional parameters: OFF | ON (disables the limit check | enables the limit check using the previous/default limit values)

get_cp() → float

```
# SCPI: CONFigure:EVDO:MEASurement<instance>:MEValuation:LIMit:CP
value: float or bool = driver.configure.multiEval.limit.get_cp()
```

Defines an upper limit for the channel power.

return cp_user_limit: Range: -60 dB to 0 dB

get_dwcp() → float

```
# SCPI: CONFigure:EVDO:MEASurement<instance>:MEValuation:LIMit:DWCP
value: float or bool = driver.configure.multiEval.limit.get_dwcp()
```

Specifies the data Walsh code channel power limit.

return dwcp_user_limit: Range: -60 dB to 0 dB, Unit: dB

get_iq_imbalance() → float

```
# SCPI: CONFigure:EVDO:MEASurement<instance>:MEValuation:LIMit:IQIMbalance
value: float or bool = driver.configure.multiEval.limit.get_iq_imbalance()
```

Defines an upper limit for the I/Q imbalance.

return iq_imbalance: Range: -120 dB to -20 dB, Unit: dB Additional parameters: OFF | ON (disables the limit check | enables the limit check using the previous/default limit values)

get_iq_offset() → float

```
# SCPI: CONFigure:EVDO:MEASurement<instance>:MEValuation:LIMit:IQOFfset
value: float or bool = driver.configure.multiEval.limit.get_iq_offset()
```

Defines an upper limit for the I/Q origin offset.

return iq_offset: Range: -120 dB to -20 dB, Unit: dB Additional parameters: OFF | ON (disables the limit check | enables the limit check using the previous/default limit values)

get_max_power() → float

```
# SCPI: CONFigure:EVDO:MEASurement<instance>:MEValuation:LIMit:MAXPower
value: float or bool = driver.configure.multiEval.limit.get_max_power()
```

Defines an upper limit for the AT power.

return abs_max_power: Range: -127.9 dBm to 0 dBm, Unit: dBm Additional parameters: OFF | ON (disables the limit check | enables the limit check using the previous/default limit values)

get_min_power() → float

```
# SCPI: CONFigure:EVDO:MEASurement<instance>:MEValuation:LIMit:MINPower
value: float or bool = driver.configure.multiEval.limit.get_min_power()
```

Defines a lower limit for the AT power.

return abs_min_power: Range: -128 dBm to -0.1 dBm, Unit: dBm Additional parameters: OFF | ON (disables the limit check | enables the limit check using the previous/default limit values)

get_tt_error() → float

```
# SCPI: CONFigure:EVDO:MEASurement<instance>:MEValuation:LIMit:TTERror
value: float or bool = driver.configure.multiEval.limit.get_tt_error()
```

Defines an upper limit for the transmit time error.

return trans_time_err: Range: 0 µs to 10 µs, Unit: µs Additional parameters: OFF | ON (disables the limit check | enables the limit check using the previous/default limit values)

get_wquality() → float

```
# SCPI: CONFigure:EVDO:MEASurement<instance>:MEValuation:LIMit:WQUality
value: float or bool = driver.configure.multiEval.limit.get_wquality()
```

Defines a lower limit for the waveform quality. For an ideal transmitter, the waveform quality equals 1.

return wav_quality: Range: 0 to 1 Additional parameters: OFF | ON (disables the limit check | enables the limit check using the previous/default limit values)

set_cde(*cde_user_limit: float*) → None

```
# SCPI: CONFigure:EVDO:MEASurement<instance>:MEValuation:LIMit:CDE
driver.configure.multiEval.limit.set_cde(cde_user_limit = 1.0)
```

Defines an upper limit for the code domain error.

param cde_user_limit Range: -70 dB to 0 dB, Unit: dB

set_cdp(*cdp_user_limit: float*) → None

```
# SCPI: CONFigure:EVDO:MEASurement<instance>:MEValuation:LIMit:CDP
driver.configure.multiEval.limit.set_cdp(cdp_user_limit = 1.0)
```

Defines an upper limit for the code domain power of inactive channels.

param cdp_user_limit Range: -70 dB to 0 dB, Unit: dB

set_cf_error(*cfreq_error: float*) → None

```
# SCPI: CONFigure:EVDO:MEASurement<instance>:MEValuation:LIMit:CFERror
driver.configure.multiEval.limit.set_cf_error(cfreq_error = 1.0)
```

Defines an upper limit for the carrier frequency error.

param cfreq_error Range: 0 Hz to 1000 Hz, Unit: Hz Additional parameters: OFF | ON (disables the limit check | enables the limit check using the previous/default limit values)

set_cp(*cp_user_limit: float*) → None

```
# SCPI: CONFigure:EVDO:MEASurement<instance>:MEValuation:LIMit:CP
driver.configure.multiEval.limit.set_cp(cp_user_limit = 1.0)
```

Defines an upper limit for the channel power.

param cp_user_limit Range: -60 dB to 0 dB

set_dwcp(*dwcp_user_limit: float*) → None

```
# SCPI: CONFigure:EVDO:MEASurement<instance>:MEValuation:LIMit:DWCP
driver.configure.multiEval.limit.set_dwcp(dwcp_user_limit = 1.0)
```

Specifies the data Walsh code channel power limit.

param dwcp_user_limit Range: -60 dB to 0 dB, Unit: dB

set_iq_imbalance(*iq_imbalance: float*) → None

```
# SCPI: CONFigure:EVDO:MEASurement<instance>:MEValuation:LIMit:IQIMbalance
driver.configure.multiEval.limit.set_iq_imbalance(iq_imbalance = 1.0)
```

Defines an upper limit for the I/Q imbalance.

param iq_imbalance Range: -120 dB to -20 dB, Unit: dB Additional parameters: OFF | ON (disables the limit check | enables the limit check using the previous/default limit values)

set_iq_offset(*iq_offset: float*) → None

```
# SCPI: CONFigure:EVDO:MEASurement<instance>:MEValuation:LIMit:IQOFfset
driver.configure.multiEval.limit.set_iq_offset(iq_offset = 1.0)
```

Defines an upper limit for the I/Q origin offset.

param iq_offset Range: -120 dB to -20 dB, Unit: dB Additional parameters: OFF | ON (disables the limit check | enables the limit check using the previous/default limit values)

set_max_power(*abs_max_power: float*) → None

```
# SCPI: CONFigure:EVDO:MEASurement<instance>:MEValuation:LIMit:MAXPower
driver.configure.multiEval.limit.set_max_power(abs_max_power = 1.0)
```

Defines an upper limit for the AT power.

param abs_max_power Range: -127.9 dBm to 0 dBm, Unit: dBm Additional parameters: OFF | ON (disables the limit check | enables the limit check using the previous/default limit values)

set_min_power(*abs_min_power: float*) → None

```
# SCPI: CONFigure:EVDO:MEASurement<instance>:MEValuation:LIMit:MINPower
driver.configure.multiEval.limit.set_min_power(abs_min_power = 1.0)
```

Defines a lower limit for the AT power.

param abs_min_power Range: -128 dBm to -0.1 dBm, Unit: dBm Additional parameters: OFF | ON (disables the limit check | enables the limit check using the previous/default limit values)

set_tt_error(*trans_time_err: float*) → None

```
# SCPI: CONFigure:EVDO:MEASurement<instance>:MEValuation:LIMit:TTERror
driver.configure.multiEval.limit.set_tt_error(trans_time_err = 1.0)
```

Defines an upper limit for the transmit time error.

param trans_time_err Range: 0 µs to 10 µs, Unit: µs Additional parameters: OFF | ON (disables the limit check | enables the limit check using the previous/default limit values)

set_wquality(*wav_quality: float*) → None

```
# SCPI: CONFigure:EVDO:MEASurement<instance>:MEValuation:LIMit:WQUality
driver.configure.multiEval.limit.set_wquality(wav_quality = 1.0)
```

Defines a lower limit for the waveform quality. For an ideal transmitter, the waveform quality equals 1.

param wav_quality Range: 0 to 1 Additional parameters: OFF | ON (disables the limit check | enables the limit check using the previous/default limit values)

Cloning the Group

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

Subgroups

7.2.2.5.1 Evm

SCPI Commands

```
CONFigure:EVDO:MEASurement<Instance>:MEValuation:LIMit:EVM:PEAK
CONFigure:EVDO:MEASurement<Instance>:MEValuation:LIMit:EVM:RMS
```

class Evm

Evm commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

get_peak() → float

```
# SCPI: CONFigure:EVDO:MEASurement<instance>:MEValuation:LIMit:EVM:PEAK
value: float or bool = driver.configure.multiEval.limit.evm.get_peak()
```

Defines an upper limit for the peak values of the error vector magnitude (EVM) .

return evm_peak: Range: 0 % to 100 %, Unit: % Additional parameters: OFF | ON (disables the limit check | enables the limit check using the previous/default limit values)

get_rms() → float

```
# SCPI: CONFigure:EVDO:MEASurement<instance>:MEValuation:LIMit:EVM:RMS
value: float or bool = driver.configure.multiEval.limit.evm.get_rms()
```

Defines an upper limit for the RMS values of the error vector magnitude (EVM) .

return evm_rms: Range: 0 % to 100 %, Unit: % Additional parameters: OFF | ON (disables the limit check | enables the limit check using the previous/default limit values)

set_peak(*evm_peak: float*) → None

```
# SCPI: CONFigure:EVDO:MEASurement<instance>:MEValuation:LIMit:EVM:PEAK
driver.configure.multiEval.limit.evm.set_peak(evm_peak = 1.0)
```

Defines an upper limit for the peak values of the error vector magnitude (EVM) .

param evm_peak Range: 0 % to 100 %, Unit: % Additional parameters: OFF | ON (disables the limit check | enables the limit check using the previous/default limit values)

set_rms(*evm_rms: float*) → None

```
# SCPI: CONFigure:EVDO:MEASurement<instance>:MEValuation:LIMit:EVM:RMS
driver.configure.multiEval.limit.evm.set_rms(evm_rms = 1.0)
```

Defines an upper limit for the RMS values of the error vector magnitude (EVM) .

param evm_rms Range: 0 % to 100 %, Unit: % Additional parameters: OFF | ON (disables the limit check | enables the limit check using the previous/default limit values)

7.2.2.5.2 Merr

SCPI Commands

```
CONFigure:EVDO:MEASurement<Instance>:MEValuation:LIMit:MERR:PEAK
CONFigure:EVDO:MEASurement<Instance>:MEValuation:LIMit:MERR:RMS
```

class Merr

Merr commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

get_peak() → float

```
# SCPI: CONFigure:EVDO:MEASurement<instance>:MEValuation:LIMit:MERR:PEAK
value: float or bool = driver.configure.multiEval.limit.merr.get_peak()
```

Defines an upper limit for the peak values of the magnitude error.

return merr_peak: Range: 0 % to 100 %, Unit: % Additional parameters: OFF | ON (disables the limit check | enables the limit check using the previous/default limit values)

get_rms() → float

```
# SCPI: CONFigure:EVDO:MEASurement<instance>:MEValuation:LIMit:MERR:RMS
value: float or bool = driver.configure.multiEval.limit.merr.get_rms()
```

Defines an upper limit for the RMS values of the magnitude error.

return merr_rms: Range: 0 % to 100 %, Unit: % Additional parameters: OFF | ON (disables the limit check | enables the limit check using the previous/default limit values)

set_peak(*merr_peak: float*) → None

```
# SCPI: CONFigure:EVDO:MEASurement<instance>:MEValuation:LIMit:MERR:PEAK
driver.configure.multiEval.limit.merr.set_peak(merr_peak = 1.0)
```

Defines an upper limit for the peak values of the magnitude error.

param merr_peak Range: 0 % to 100 %, Unit: % Additional parameters: OFF | ON (disables the limit check | enables the limit check using the previous/default limit values)

set_rms(*merr_rms: float*) → None

```
# SCPI: CONFigure:EVDO:MEASurement<instance>:MEValuation:LIMit:MERR:RMS
driver.configure.multiEval.limit.merr.set_rms(merr_rms = 1.0)
```

Defines an upper limit for the RMS values of the magnitude error.

param merr_rms Range: 0 % to 100 %, Unit: % Additional parameters: OFF | ON (disables the limit check | enables the limit check using the previous/default limit values)

7.2.2.5.3 Perr

SCPI Commands

```
CONFigure:EVDO:MEASurement<Instance>:MEValuation:LIMit:PERR:PEAK
CONFigure:EVDO:MEASurement<Instance>:MEValuation:LIMit:PERR:RMS
```

class Perr

Perr commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

get_peak() → float

```
# SCPI: CONFigure:EVDO:MEASurement<instance>:MEValuation:LIMit:PERR:PEAK
value: float or bool = driver.configure.multiEval.limit.perr.get_peak()
```

Defines a symmetric limit for the peak values of the phase error. The limit check fails if the absolute value of the measured phase error exceeds the specified value.

return perr_peak: Range: 0 deg to 180 deg Additional parameters: OFF | ON (disables the limit check | enables the limit check using the previous/default limit values)

get_rms() → float

```
# SCPI: CONFigure:EVDO:MEASurement<instance>:MEValuation:LIMit:PERR:RMS
value: float or bool = driver.configure.multiEval.limit.perr.get_rms()
```

Defines a symmetric limit for the RMS values of the phase error. The limit check fails if the absolute value of the measured phase error exceeds the specified value.

return perr_rms: Range: 0 deg to 180 deg, Unit: deg Additional parameters: OFF | ON (disables the limit check | enables the limit check using the previous/default limit values)

set_peak(*perr_peak: float*) → None

```
# SCPI: CONFigure:EVDO:MEASurement<instance>:MEValuation:LIMit:PERR:PEAK
driver.configure.multiEval.limit.perr.set_peak(perr_peak = 1.0)
```

Defines a symmetric limit for the peak values of the phase error. The limit check fails if the absolute value of the measured phase error exceeds the specified value.

param perr_peak Range: 0 deg to 180 deg Additional parameters: OFF | ON (disables the limit check | enables the limit check using the previous/default limit values)

set_rms(*perr_rms: float*) → None

```
# SCPI: CONFigure:EVDO:MEASurement<instance>:MEValuation:LIMit:PERR:RMS
driver.configure.multiEval.limit.perr.set_rms(perr_rms = 1.0)
```

Defines a symmetric limit for the RMS values of the phase error. The limit check fails if the absolute value of the measured phase error exceeds the specified value.

param perr_rms Range: 0 deg to 180 deg, Unit: deg Additional parameters: OFF | ON (disables the limit check | enables the limit check using the previous/default limit values)

7.2.2.5.4 Acp

class Acp

Acp commands group definition. 8 total commands, 3 Sub-groups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.configure.multiEval.limit.acp.clone()
```

Subgroups

7.2.2.5.4.1 Lower

SCPI Commands

```
CONFigure:EVDO:MEASurement<Instance>:MEValuation:LIMit:ACP:LOWer:RELative
CONFigure:EVDO:MEASurement<Instance>:MEValuation:LIMit:ACP:LOWer:ABSolute
```

class Lower

Lower commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

get_absolute() → List[float]

```
# SCPI: CONFigure:EVDO:MEASurement<instance>
↪:MEValuation:LIMit:ACP:LOWer:ABSolute
value: List[float or bool] = driver.configure.multiEval.limit.acp.lower.get_
↪absolute()
```

Defines lower limits 9 to 0 for the ACP measurement in dBm of the selected carrier (see method RsCmwEvdoMeas.Configure. MultiEval.Carrier.select) at the individual negative offset frequencies (set via method RsCmwEvdoMeas.Configure.MultiEval. Acp.Foffsets.lower) . The limit index 9 to 0 corresponds to the index used in manual control.

return acp_setting: Range: -80 dBm to 10 dBm , Unit: dBm Additional parameters: OFF | ON (disables | enables the limit check)

get_relative() → List[float]

```
# SCPI: CONFigure:EVDO:MEASurement<instance>
↪:MEValuation:LIMit:ACP:LOWer[:RELative]
value: List[float or bool] = driver.configure.multiEval.limit.acp.lower.get_
↪relative()
```

Defines limits for the relative ACP in dBc of the selected carrier (see method RsCmwEvdoMeas.Configure.MultiEval.Carrier. select) at the individual negative offset frequencies (set via method RsCmwEvdoMeas.Configure.MultiEval.Acp.Foffsets. lower) . The limit index 0 to 9 corresponds to the index used in manual control.

return acp_setting: No help available

set_absolute(*acp_setting: List[float]*) → None

```
# SCPI: CONFigure:EVDO:MEASurement<instance>
↪:MEValuation:LIMit:ACP:LOWer:ABSolute
driver.configure.multiEval.limit.acp.lower.set_absolute(acp_setting = [1.1,␣
↪True, 2.2, False, 3.3])
```

Defines lower limits 9 to 0 for the ACP measurement in dBm of the selected carrier (see method RsCmwEvdoMeas.Configure. MultiEval.Carrier.select) at the individual negative offset frequencies (set via method RsCmwEvdoMeas.Configure.MultiEval. Acp.Foffsets.lower) . The limit index 9 to 0 corresponds to the index used in manual control.

param acp_setting Range: -80 dBm to 10 dBm , Unit: dBm Additional parameters: OFF | ON (disables | enables the limit check)

set_relative(*acp_setting: List[float]*) → None

```
# SCPI: CONFigure:EVDO:MEASurement<instance>
↪:MEValuation:LIMit:ACP:LOWer[:RELative]
driver.configure.multiEval.limit.acp.lower.set_relative(acp_setting = [1.1,␣
↪True, 2.2, False, 3.3])
```

Defines limits for the relative ACP in dBc of the selected carrier (see method RsCmwEvdoMeas.Configure.MultiEval.Carrier. select) at the individual negative offset frequencies (set via method RsCmwEvdoMeas.Configure.MultiEval.Acp.Foffsets. lower) . The limit index 0 to 9 corresponds to the index used in manual control.

param acp_setting numeric | ON | OFF Range: -80 dBc to 10 dBc, Unit: dBc Additional parameters: OFF | ON (disables the limit check | enables the limit check using the previous/default limit values)

7.2.2.5.4.2 Extended

class Extended

Extended commands group definition. 4 total commands, 2 Sub-groups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.configure.multiEval.limit.acp.extended.clone()
```

Subgroups

7.2.2.5.4.3 Lower

SCPI Commands

```
CONFigure:EVDO:MEASurement<Instance>:MEValuation:LIMit:ACP:EXTended:LOWer:RELative
CONFigure:EVDO:MEASurement<Instance>:MEValuation:LIMit:ACP:EXTended:LOWer:ABSolute
```

class Lower

Lower commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

get_absolute() → List[float]

```
# SCPI: CONFigure:EVDO:MEASurement<instance>
↪:MEValuation:LIMit:ACP:EXTended:LOWer:ABSolute
value: List[float or bool] = driver.configure.multiEval.limit.acp.extended.
↪lower.get_absolute()
```

Defines lower limits 19 to 0 for the extended ACP measurement in dBm of the selected carrier (see method RsCmwEvdoMeas. Configure.MultiEval.Carrier.select) at the individual negative offset frequencies (set

via method RsCmwEvdoMeas.Configure. MultiEval.Acp.Foffsets.lower) . The limit index 19 to 0 corresponds to the index used in manual control.

return acp_setting: Range: -80 dBm to 10 dBm , Unit: dBm Additional parameters: OFF | ON (disables | enables the limit check)

get_relative() → List[float]

```
# SCPI: CONFigure:EVDO:MEASurement<instance>
↪:MEValuation:LIMit:ACP:EXTended:LOWer[:RELative]
value: List[float or bool] = driver.configure.multiEval.limit.acp.extended.
↪lower.get_relative()
```

Defines lower limits 19 to 0 for the extended ACP measurement in dBc of the selected carrier (see method RsCmwEvdoMeas. Configure.MultiEval.Carrier.select) at the individual negative offset frequencies (set via method RsCmwEvdoMeas.Configure. MultiEval.Acp.Foffsets.lower) . The limit index 19 to 0 corresponds to the index used in manual control.

return acp_setting: Range: -80 dBc to 10 dBc , Unit: dBc Additional parameters: ON | OFF (enables | disables the limit check)

set_absolute(*acp_setting: List[float]*) → None

```
# SCPI: CONFigure:EVDO:MEASurement<instance>
↪:MEValuation:LIMit:ACP:EXTended:LOWer:ABSolute
driver.configure.multiEval.limit.acp.extended.lower.set_absolute(acp_setting =␣
↪[1.1, True, 2.2, False, 3.3])
```

Defines lower limits 19 to 0 for the extended ACP measurement in dBm of the selected carrier (see method RsCmwEvdoMeas. Configure.MultiEval.Carrier.select) at the individual negative offset frequencies (set via method RsCmwEvdoMeas.Configure. MultiEval.Acp.Foffsets.lower) . The limit index 19 to 0 corresponds to the index used in manual control.

param acp_setting Range: -80 dBm to 10 dBm , Unit: dBm Additional parameters: OFF | ON (disables | enables the limit check)

set_relative(*acp_setting: List[float]*) → None

```
# SCPI: CONFigure:EVDO:MEASurement<instance>
↪:MEValuation:LIMit:ACP:EXTended:LOWer[:RELative]
driver.configure.multiEval.limit.acp.extended.lower.set_relative(acp_setting =␣
↪[1.1, True, 2.2, False, 3.3])
```

Defines lower limits 19 to 0 for the extended ACP measurement in dBc of the selected carrier (see method RsCmwEvdoMeas. Configure.MultiEval.Carrier.select) at the individual negative offset frequencies (set via method RsCmwEvdoMeas.Configure. MultiEval.Acp.Foffsets.lower) . The limit index 19 to 0 corresponds to the index used in manual control.

param acp_setting Range: -80 dBc to 10 dBc , Unit: dBc Additional parameters: ON | OFF (enables | disables the limit check)

7.2.2.5.4.4 Upper

SCPI Commands

```
CONFigure:EVDO:MEASurement<Instance>:MEValuation:LIMit:ACP:EXTended:UPPer:RELative
CONFigure:EVDO:MEASurement<Instance>:MEValuation:LIMit:ACP:EXTended:UPPer:ABSolute
```

class Upper

Upper commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

get_absolute() → List[float]

```
# SCPI: CONFigure:EVDO:MEASurement<instance>
↪:MEValuation:LIMit:ACP:EXTended:UPPer:ABSolute
value: List[float or bool] = driver.configure.multiEval.limit.acp.extended.
↪upper.get_absolute()
```

Defines upper limits 0 to 19 for the extended ACP measurement in dBm of the selected carrier (see method RsCmwEvdoMeas. Configure.MultiEval.Carrier.select) at the individual positive offset frequencies (set via method RsCmwEvdoMeas.Configure. MultiEval.Acp.Foffsets.upper) . The limit index 0 to 19 corresponds to the index used in manual control.

return acp_setting: Range: -80 dBm to 10 dBm , Unit: dBm Additional parameters: ON | OFF (enables | disables the limit check)

get_relative() → List[float]

```
# SCPI: CONFigure:EVDO:MEASurement<instance>
↪:MEValuation:LIMit:ACP:EXTended:UPPer[:RELative]
value: List[float or bool] = driver.configure.multiEval.limit.acp.extended.
↪upper.get_relative()
```

Defines upper limits 0 to 19 for the extended ACP measurement in dBc of the selected carrier (see method RsCmwEvdoMeas. Configure.MultiEval.Carrier.select) at the individual negative offset frequencies (set via method RsCmwEvdoMeas.Configure. MultiEval.Acp.Foffsets.lower) . The limit index 0 to 19 corresponds to the index used in manual control.

return acp_setting: Range: -80 dBc to 10 dBc , Unit: dBc Additional parameters: OFF | ON (enables | disables the limit check)

set_absolute(*acp_setting: List[float]*) → None

```
# SCPI: CONFigure:EVDO:MEASurement<instance>
↪:MEValuation:LIMit:ACP:EXTended:UPPer:ABSolute
driver.configure.multiEval.limit.acp.extended.upper.set_absolute(acp_setting =␣
↪[1.1, True, 2.2, False, 3.3])
```

Defines upper limits 0 to 19 for the extended ACP measurement in dBm of the selected carrier (see method RsCmwEvdoMeas. Configure.MultiEval.Carrier.select) at the individual positive offset frequencies (set via method RsCmwEvdoMeas.Configure. MultiEval.Acp.Foffsets.upper) . The limit index 0 to 19 corresponds to the index used in manual control.

param acp_setting Range: -80 dBm to 10 dBm , Unit: dBm Additional parameters: ON | OFF (enables | disables the limit check)

set_relative(*acp_setting: List[float]*) → None

```
# SCPI: CONFigure:EVDO:MEASurement<instance>
↪:MEValuation:LIMit:ACP:EXTended:UPPer[:RELative]
driver.configure.multiEval.limit.acp.extended.upper.set_relative(acp_setting =␣
↪[1.1, True, 2.2, False, 3.3])
```

Defines upper limits 0 to 19 for the extended ACP measurement in dBc of the selected carrier (see method RsCmwEvdoMeas. Configure.MultiEval.Carrier.select) at the individual negative offset frequencies (set via method RsCmwEvdoMeas.Configure. MultiEval.Acp.Foffsets.lower) . The limit index 0 to 19 corresponds to the index used in manual control.

param acp_setting Range: -80 dBc to 10 dBc , Unit: dBc Additional parameters: OFF | ON (enables | disables the limit check)

7.2.2.5.4.5 Upper

SCPI Commands

```
CONFigure:EVDO:MEASurement<Instance>:MEValuation:LIMit:ACP:UPPer:RELative
CONFigure:EVDO:MEASurement<Instance>:MEValuation:LIMit:ACP:UPPer:ABSolute
```

class Upper

Upper commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

get_absolute() → List[float]

```
# SCPI: CONFigure:EVDO:MEASurement<instance>
↪:MEValuation:LIMit:ACP:UPPer:ABSolute
value: List[float or bool] = driver.configure.multiEval.limit.acp.upper.get_
↪absolute()
```

Defines upper limits 0 to 9 for the extended ACP measurement in dBm of the selected carrier (see method RsCmwEvdoMeas. Configure.MultiEval.Carrier.select) at the individual positive offset frequencies (set via method RsCmwEvdoMeas.Configure. MultiEval.Acp.Foffsets.upper) . The limit index 0 to 9 corresponds to the index used in manual control.

return acp_setting: Range: -80 dBm to 10 dBm , Unit: dBm Additional parameters: ON | OFF (enables | disables the limit check)

get_relative() → List[float]

```
# SCPI: CONFigure:EVDO:MEASurement<instance>
↪:MEValuation:LIMit:ACP:UPPer[:RELative]
value: List[float or bool] = driver.configure.multiEval.limit.acp.upper.get_
↪relative()
```

Defines limits for the relative ACP in dBc of the selected carrier (see method RsCmwEvdoMeas.Configure.MultiEval.Carrier. select) at the individual positive offset frequencies (set via method RsCmwEvdoMeas.Configure.MultiEval.Acp.Foffsets. upper) . The limit index 0 to 9 corresponds to the index used in manual control.

return acp_setting: No help available

set_absolute(*acp_setting: List[float]*) → None

```
# SCPI: CONFigure:EVDO:MEASurement<instance>
↪:MEValuation:LIMit:ACP:UPPer:ABSolute
driver.configure.multiEval.limit.acp.upper.set_absolute(acp_setting = [1.1,␣
↪True, 2.2, False, 3.3])
```

Defines upper limits 0 to 9 for the extended ACP measurement in dBm of the selected carrier (see method RsCmwEvdoMeas. Configure.MultiEval.Carrier.select) at the individual positive offset frequencies (set via method RsCmwEvdoMeas.Configure. MultiEval.Acp.Foffsets.upper) . The limit index 0 to 9 corresponds to the index used in manual control.

param acp_setting Range: -80 dBm to 10 dBm , Unit: dBm Additional parameters: ON | OFF (enables | disables the limit check)

set_relative(*acp_setting: List[float]*) → None

```
# SCPI: CONFigure:EVDO:MEASurement<instance>
↪:MEValuation:LIMit:ACP:UPPer[:RELative]
driver.configure.multiEval.limit.acp.upper.set_relative(acp_setting = [1.1,␣
↪True, 2.2, False, 3.3])
```

Defines limits for the relative ACP in dBc of the selected carrier (see method RsCmwEvdoMeas.Configure.MultiEval.Carrier. select) at the individual positive offset frequencies (set via method RsCmwEvdoMeas.Configure.MultiEval.Acp.Foffsets. upper) . The limit index 0 to 9 corresponds to the index used in manual control.

param acp_setting numeric | ON | OFF Range: -80 dBc to 10 dBc, Unit: dBc Additional parameters: OFF | ON (disables the limit check | enables the limit check using the previous/default limit values)

7.2.2.5.5 Obw

SCPI Commands

```
CONFigure:EVDO:MEASurement<Instance>:MEValuation:LIMit:OBW:MULTi
CONFigure:EVDO:MEASurement<Instance>:MEValuation:LIMit:OBW:SETA
CONFigure:EVDO:MEASurement<Instance>:MEValuation:LIMit:OBW:SETB
```

class Obw

Obw commands group definition. 5 total commands, 1 Sub-groups, 3 group commands

get_multi() → float

```
# SCPI: CONFigure:EVDO:MEASurement<instance>:MEValuation:LIMit:OBW:MULTi
value: float or bool = driver.configure.multiEval.limit.obw.get_multi()
```

Gets/sets the overall occupied bandwidth limit and the state of the limit check for multi-carrier configurations.

return obw_limit: Range: 0 MHz to 16 MHz Additional parameter values: OFF | ON (disables | enables the limit check)

get_seta() → List[float]

```
# SCPI: CONFigure:EVDO:MEASurement<instance>:MEValuation:LIMit:OBW:SETA
value: List[float] = driver.configure.multiEval.limit.obw.get_seta()
```

Gets/sets the OBW limits for limit set A. This limit set is dedicated to band class 0.

return obw_limit_set_a: No help available

get_setb() → List[float]

```
# SCPI: CONFigure:EVDO:MEASurement<instance>:MEValuation:LIMit:OBW:SETB
value: List[float] = driver.configure.multiEval.limit.obw.get_setb()
```

Gets/sets the OBW limits for limit set B. This limit set is used if the currently selected band class is different from 0.

return obw_limit_set_b: No help available

set_multi(*obw_limit: float*) → None

```
# SCPI: CONFigure:EVDO:MEASurement<instance>:MEValuation:LIMit:OBW:MULTi
driver.configure.multiEval.limit.obw.set_multi(obw_limit = 1.0)
```

Gets/sets the overall occupied bandwidth limit and the state of the limit check for multi-carrier configurations.

param obw_limit Range: 0 MHz to 16 MHz Additional parameter values: OFF | ON (disables | enables the limit check)

set_seta(*obw_limit_set_a: List[float]*) → None

```
# SCPI: CONFigure:EVDO:MEASurement<instance>:MEValuation:LIMit:OBW:SETA
driver.configure.multiEval.limit.obw.set_seta(obw_limit_set_a = [1.1, 2.2, 3.3])
```

Gets/sets the OBW limits for limit set A. This limit set is dedicated to band class 0.

param obw_limit_set_a No help available

set_setb(*obw_limit_set_b: List[float]*) → None

```
# SCPI: CONFigure:EVDO:MEASurement<instance>:MEValuation:LIMit:OBW:SETB
driver.configure.multiEval.limit.obw.set_setb(obw_limit_set_b = [1.1, 2.2, 3.3])
```

Gets/sets the OBW limits for limit set B. This limit set is used if the currently selected band class is different from 0.

param obw_limit_set_b No help available

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.configure.multiEval.limit.obw.clone()
```

Subgroups

7.2.2.5.5.1 Check

SCPI Commands

```
CONFigure:EVDO:MEASurement<Instance>:MEValuation:LIMit:OBW:CHECk:USED
CONFigure:EVDO:MEASurement<Instance>:MEValuation:LIMit:OBW:CHECk
```

class Check

Check commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

get_used() → RsCmwEvdoMeas.enums.ObwUsedLimitSet

```
# SCPI: CONFigure:EVDO:MEASurement<instance>:MEValuation:LIMit:OBW:CHECk:USED
value: enums.ObwUsedLimitSet = driver.configure.multiEval.limit.obw.check.get_
↪used()
```

Returns the currently used OBW limit set. This limit is ultimately determined by the currently selected band class: limit set A is dedicated to band class 0, limit set B is used for all other band classes.

return obw_used_limit_set: SETA | SETB The currently used limit set.

get_value() → bool

```
# SCPI: CONFigure:EVDO:MEASurement<instance>:MEValuation:LIMit:OBW:CHECk
value: bool = driver.configure.multiEval.limit.obw.check.get_value()
```

Gets/sets the enabled state of OBW limit checks. Depending on the current band class, either limit set A or B applies.

return obw_limit_check: OFF | ON Disable/enable OBW limit checks.

set_value(*obw_limit_check: bool*) → None

```
# SCPI: CONFigure:EVDO:MEASurement<instance>:MEValuation:LIMit:OBW:CHECk
driver.configure.multiEval.limit.obw.check.set_value(obw_limit_check = False)
```

Gets/sets the enabled state of OBW limit checks. Depending on the current band class, either limit set A or B applies.

param obw_limit_check OFF | ON Disable/enable OBW limit checks.

7.2.2.6 ListPy

SCPI Commands

```
CONFigure:EVDO:MEASurement<Instance>:MEValuation:LIST:COUNt
CONFigure:EVDO:MEASurement<Instance>:MEValuation:LIST
```

class ListPy

ListPy commands group definition. 7 total commands, 2 Sub-groups, 2 group commands

get_count() → int

```
# SCPI: CONFigure:EVDO:MEASurement<instance>:MEValuation:LIST:COUNt
value: int = driver.configure.multiEval.listPy.get_count()
```

Defines the number of segments in the entire measurement interval.

return segments: numeric Range: 1 to 200

get_value() → bool

```
# SCPI: CONFigure:EVDO:MEASurement<instance>:MEValuation:LIST
value: bool = driver.configure.multiEval.listPy.get_value()
```

Enables or disables the list mode.

return enable: OFF | ON ON: Enable list mode OFF: Disable list mode

set_count(*segments: int*) → None

```
# SCPI: CONFigure:EVDO:MEASurement<instance>:MEValuation:LIST:COUNt
driver.configure.multiEval.listPy.set_count(segments = 1)
```

Defines the number of segments in the entire measurement interval.

param segments numeric Range: 1 to 200

set_value(*enable: bool*) → None

```
# SCPI: CONFigure:EVDO:MEASurement<instance>:MEValuation:LIST
driver.configure.multiEval.listPy.set_value(enable = False)
```

Enables or disables the list mode.

param enable OFF | ON ON: Enable list mode OFF: Disable list mode

Cloning the Group

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

Subgroups

7.2.2.6.1 SingleCmw

SCPI Commands

```
CONFigure:EVDO:MEASurement<Instance>:MEValuation:LIST:CMWS:CMODe
```

class SingleCmw

SingleCmw commands group definition. 1 total commands, 0 Sub-groups, 1 group commands

get_cmode() → RsCmwEvdoMeas.enums.ParameterSetMode

```
# SCPI: CONFigure:EVDO:MEASurement<instance>:MEValuation:LIST:CMWS:CMODe
value: enums.ParameterSetMode = driver.configure.multiEval.listPy.singleCmw.get_
↪cmode()
```

Specifies how the input connector is selected for 1xEV-DO list mode measurements with the R&S CMWS.

return cmws_connector_mode: No help available

set_cmode(*cmws_connector_mode: RsCmwEvdoMeas.enums.ParameterSetMode*) → None

```
# SCPI: CONFigure:EVDO:MEASurement<instance>:MEValuation:LIST:CMWS:CMODe
driver.configure.multiEval.listPy.singleCmw.set_cmode(cmws_connector_mode =␣
↪enums.ParameterSetMode.GLOBal)
```

Specifies how the input connector is selected for 1xEV-DO list mode measurements with the R&S CMWS.

param cmws_connector_mode GLOBal | LIST GLOBal: The same input connector is used for all segments. It is selected in the same way as without list mode, for example via ROUTe:EVDO:MEASi:SCENario:SALone. LIST: The input connector is configured individually for each segment. See method RsCmwEvdoMeas.Configure.MultiEval.ListPy.Segment.SingleCmw.Connector.set or method RsCmwEvdoMeas.Configure.MultiEval.ListPy.Segment.Setup.set.

7.2.2.6.2 Segment

RepCap Settings

```
# Range: Nr1 .. Nr200
rc = driver.configure.multiEval.listPy.segment.repcap_segment_get()
driver.configure.multiEval.listPy.segment.repcap_segment_set(repcap.Segment.Nr1)
```

class Segment

Segment commands group definition. 4 total commands, 4 Sub-groups, 0 group commands Repeated Capability: Segment, default value after init: Segment.Nr1

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.configure.multiEval.listPy.segment.clone()
```

Subgroups

7.2.2.6.2.1 Setup

SCPI Commands

```
CONFigure:EVDO:MEASurement<Instance>:MEValuation:LIST:SEGMent<Segment>:SETup
```

class Setup

Setup commands group definition. 1 total commands, 0 Sub-groups, 1 group commands

class SetupStruct

Structure for setting input parameters. Contains optional setting parameters. Fields:

- Segment_Length: int: integer Number of measured slots in the segment. The sum of the segment lengths must not exceed 8192 slots; limit violation is indicated by a reliability indicator of 2 (capture buffer overflow) . Range: 1 to 1000
- Level: float: numeric 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 data sheet.
- Frequency: float: numeric Center frequency of the RF analyzer. Range: 100 MHz to 6 GHz , Unit: Hz
- Retrigger_Option: enums.RetriggerOption: OFF | ON | IFPower | IFPSync Enables or disables the trigger system to start segment measurement. Note: For the first segment (no = 1) the setting of this parameter is ignored, since the general MELM measurement starts with the measurement of the first segment. That means always with the first trigger event the first segment is measured. OFF: Disabled retrigger. The segment measurement is started by the first trigger event. ON: Enables the retrigger. The list mode measurement continues only if a new event for this segment is triggered (retrigger) . IFPower: Waits for the power ramp of the received bursts before measuring the segment. IFPSync: Before measuring the next segment, the R&S CMW waits for the power ramp of the received bursts and tries to synchronize to a slot boundary after the trigger event.
- Pl_Subtype: enums.PlSubtype: ST01 | ST2 | ST3 Selects the physical layer protocol subtype. ST01: subtype 0 or 1 ST2: subtype 2 ST3: subtype 3
- Half_Slot: enums.HalfSlot: FHSLot | SHSLot | BHSLots Specifies which half-slots of the code channel is/are evaluated for measurements using physical layer subtype 2 or 3 (see parameter before) . Consider that the DSC channel and the ACK channel are transmitted time-multiplexed on Walsh channel W1232. The ACK is transmitted on the first half-slot and the DSC on the second half-slot. FHSLot: evaluate the first half-slot SHSLot: evaluate the second half-slot BHSLots: evaluate both half-slots
- Long_Code_Mask_I: float: numeric Specifies the long code mask for the I branch. Range: #H0 to #H3FFFFFFFFF

- Long_Code_Mask_Q: float: numeric Specifies the long code mask for the Q branch. Range: #H0 to #H3FFFFFFFFF
- Cmws_Connector: enums.CmwsConnector: Optional setting parameter. Optional parameter, as alternative to the command [CMDLINK: CONFigure:EVDO:MEASi:MEValuation:LIST:SEGMentno:CMWS:CONNector CMDLINK]

get(*segment*=<*Segment.Default: -1*>) → SetupStruct

```
# SCPI: CONFigure:EVDO:MEASurement<instance>:MEValuation:LIST:SEGMent<nr>:SETup
value: SetupStruct = driver.configure.multiEval.listPy.segment.setup.
↪get(segment = repcap.Segment.Default)
```

Defines the length of segment <no> and the analyzer settings. In general, this command must be sent for all segments measured.

param segment optional repeated capability selector. Default value: Nr1 (settable in the interface 'Segment')

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

set(*structure: RsCmwEvdoMeas.Implementations.Configure_.MultiEval_.ListPy_.Segment_.Setup.Setup.SetupStruct*, *segment*=<*Segment.Default: -1*>) → None

```
# SCPI: CONFigure:EVDO:MEASurement<instance>:MEValuation:LIST:SEGMent<nr>:SETup
driver.configure.multiEval.listPy.segment.setup.set(value = [PROPERTY_STRUCT_
↪NAME](), segment = repcap.Segment.Default)
```

Defines the length of segment <no> and the analyzer settings. In general, this command must be sent for all segments measured.

param structure for set value, see the help for SetupStruct structure arguments.

param segment optional repeated capability selector. Default value: Nr1 (settable in the interface 'Segment')

7.2.2.6.2.2 SingleCmw

class SingleCmw

SingleCmw commands group definition. 1 total commands, 1 Sub-groups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.configure.multiEval.listPy.segment.singleCmw.clone()
```

Subgroups

7.2.2.6.2.3 Connector

SCPI Commands

```
CONFigure:EVDO:MEASurement<Instance>:MEValuation:LIST:SEGMent<Segment>:CMWS:CONNector
```

class Connector

Connector commands group definition. 1 total commands, 0 Sub-groups, 1 group commands

get(*segment*=<*Segment.Default: -1*>) → RsCmwEvdoMeas.enums.CmwsConnector

```
# SCPI: CONFigure:EVDO:MEASurement<instance>:MEValuation:LIST:SEGMent<nr>
↪:CMWS:CONNector
value: enums.CmwsConnector = driver.configure.multiEval.listPy.segment.
↪singleCmw.connector.get(segment = repcap.Segment.Default)
```

Selects the RF input connector for segment <no> for 1xEV-DO list mode measurements with the R&S CMWS. This setting is only relevant for connector mode LIST, see method RsCmwEvdoMeas.Configure.MultiEval.ListPy.SingleCmw.cmode. All segments of a list mode measurement must use connectors of the same bench. For possible connector values, see 'Values for RF Path Selection'.

param segment optional repeated capability selector. Default value: Nr1 (settable in the interface 'Segment')

return cmws_connector: Selects the input connector of the R&S CMWS

set(*cmws_connector: RsCmwEvdoMeas.enums.CmwsConnector*, *segment*=<*Segment.Default: -1*>) → None

```
# SCPI: CONFigure:EVDO:MEASurement<instance>:MEValuation:LIST:SEGMent<nr>
↪:CMWS:CONNector
driver.configure.multiEval.listPy.segment.singleCmw.connector.set(cmws_
↪connector = enums.CmwsConnector.R11, segment = repcap.Segment.Default)
```

Selects the RF input connector for segment <no> for 1xEV-DO list mode measurements with the R&S CMWS. This setting is only relevant for connector mode LIST, see method RsCmwEvdoMeas.Configure.MultiEval.ListPy.SingleCmw.cmode. All segments of a list mode measurement must use connectors of the same bench. For possible connector values, see 'Values for RF Path Selection'.

param cmws_connector Selects the input connector of the R&S CMWS

param segment optional repeated capability selector. Default value: Nr1 (settable in the interface 'Segment')

7.2.2.6.2.4 Modulation

SCPI Commands

```
CONFigure:EVDO:MEASurement<Instance>:MEValuation:LIST:SEGMent<Segment>:MODulation
```

class Modulation

Modulation commands group definition. 1 total commands, 0 Sub-groups, 1 group commands

class ModulationStruct

Structure for setting input parameters. Contains optional setting parameters. Fields:

- Statistic_Length: int: integer The statistical length is limited by the number of steps in the segment which is defined by [CMDLINK: CONFigure:EVDO:MEASi:MEValuation:LIST:SEGMentno:SETup CMDLINK].
- Enable_Evm: bool: OFF | ON OFF: Disable measurement. ON: Enable measurement of EVM.
- Enable_Mag_Error: bool: OFF | ON Disable or enable measurement of magnitude error.
- Enable_Phase_Err: bool: OFF | ON Disable or enable measurement of phase error.
- Enable_Wave_Qual: bool: OFF | ON Disable or enable measurement of waveform quality.
- Enable_Iq_Error: bool: OFF | ON Disable or enable measurement of I/Q origin offset and imbalance.
- Enable_Ch_Pow: bool: OFF | ON Disable or enable measurement of channel power.
- Enable_Dwcp: bool: OFF | ON Disable or enable measurement of data Walsh code channel power.
- Enable_Wbnb_Pow: bool: OFF | ON Disable or enable measurement of wideband and narrowband power.
- Enable_Freq_Err: bool: OFF | ON Disable or enable measurement of carrier frequency error.
- Enable_Melm_Tte: bool: Optional setting parameter. OFF | ON Disable or enable measurement of transmit time error.

get(*segment=<Segment.Default: -1>*) → ModulationStruct

```
# SCPI: CONFigure:EVDO:MEASurement<instance>:MEValuation:LIST:SEGMent<nr>
↪:MODulation
value: ModulationStruct = driver.configure.multiEval.listPy.segment.modulation.
↪get(segment = repcap.Segment.Default)
```

Defines the statistical length for AVERage, MAXimum, MINimum and SDEViation calculation and enables the calculation of the different modulation results in segment <no>; see 'Multi-Evaluation List Mode'.

param segment optional repeated capability selector. Default value: Nr1 (settable in the interface 'Segment')

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

set(*structure: RsCmwEvdoMeas.Implementations.Configure_.MultiEval_.ListPy_.Segment_.Modulation.Modulation.ModulationStruct*, *segment=<Segment.Default: -1>*) → None

```
# SCPI: CONFigure:EVDO:MEASurement<instance>:MEValuation:LIST:SEGMent<nr>
↪:MODulation
driver.configure.multiEval.listPy.segment.modulation.set(value = [PROPERTY_
↪STRUCT_NAME](), segment = repcap.Segment.Default)
```

Defines the statistical length for AVERage, MAXimum, MINimum and SDEViation calculation and enables the calculation of the different modulation results in segment <no>; see 'Multi-Evaluation List Mode'.

param structure for set value, see the help for ModulationStruct structure arguments.

param segment optional repeated capability selector. Default value: Nr1 (settable in the interface 'Segment')

7.2.2.6.2.5 Spectrum

SCPI Commands

```
CONFigure:EVDO:MEASurement<Instance>:MEValuation:LIST:SEGMent<Segment>:SPECtrum
```

class Spectrum

Spectrum commands group definition. 1 total commands, 0 Sub-groups, 1 group commands

class SpectrumStruct

Structure for setting input parameters. Fields:

- Statistic_Length: int: integer The statistical length is limited by the number of steps in the segment which is defined by [CMDLINK: CONFigure:EVDO:MEASi:MEValuation:LIST:SEGMentno:SETup CMDLINK].
- Enable_Acp_Rms: bool: OFF | ON OFF: Disable measurement ON: Enable measurement of adjacent channel power (RMS) .
- Enable_Obw: bool: OFF | ON Disable or enable measurement of the occupied bandwidth.

get(*segment=<Segment.Default: -1>*) → SpectrumStruct

```
# SCPI: CONFigure:EVDO:MEASurement<instance>:MEValuation:LIST:SEGMent<nr>
↪:SPECtrum
value: SpectrumStruct = driver.configure.multiEval.listPy.segment.spectrum.
↪get(segment = repcap.Segment.Default)
```

Defines the statistical length for AVERage, MAXimum, MINimum and SDEViation calculation and enables the calculation of the different spectrum results in segment <no>; see 'Multi-Evaluation List Mode'.

param segment optional repeated capability selector. Default value: Nr1 (settable in the interface 'Segment')

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

set(*structure: RsCmwEvdoMeas.Implementations.Configure_.MultiEval_.ListPy_.Segment_.Spectrum.Spectrum.SpectrumStruct*, *segment=<Segment.Default: -1>*) → None

```
# SCPI: CONFigure:EVDO:MEASurement<instance>:MEValuation:LIST:SEGMent<nr>
↪:SPECtrum
driver.configure.multiEval.listPy.segment.spectrum.set(value = [PROPERTY_STRUCT_
↪NAME](), segment = repcap.Segment.Default)
```

Defines the statistical length for AVERage, MAXimum, MINimum and SDEViation calculation and enables the calculation of the different spectrum results in segment <no>; see 'Multi-Evaluation List Mode'.

param structure for set value, see the help for SpectrumStruct structure arguments.

param segment optional repeated capability selector. Default value: Nr1 (settable in the interface 'Segment')

7.2.3 Oltr

SCPI Commands

```
CONFigure:EVDO:MEASurement<Instance>:OLTR:TOUT
CONFigure:EVDO:MEASurement<Instance>:OLTR:REPetition
CONFigure:EVDO:MEASurement<Instance>:OLTR:MOEXception
CONFigure:EVDO:MEASurement<Instance>:OLTR:SEQuence
```

class Oltr

Oltr commands group definition. 11 total commands, 4 Sub-groups, 4 group commands

get_mo_exception() → bool

```
# SCPI: CONFigure:EVDO:MEASurement<instance>:OLTR:MOEXception
value: bool = driver.configure.oltr.get_mo_exception()
```

Specifies whether measurement results that the R&S CMW identifies as faulty or inaccurate are rejected.

return meas_on_exception: OFF | ON ON: Results are never rejected OFF: Faulty results are rejected

get_repetition() → RsCmwEvdoMeas.enums.Repeat

```
# SCPI: CONFigure:EVDO:MEASurement<instance>:OLTR:REPetition
value: enums.Repeat = driver.configure.oltr.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 | CONTinuous SINGleshot: Single-shot measurement CONTinuous: Continuous measurement

get_sequence() → int

```
# SCPI: CONFigure:EVDO:MEASurement<instance>:OLTR:SEQuence
value: int = driver.configure.oltr.get_sequence()
```

Sets/gets the number of measurement sequences within a single OLTR measurement. Each sequence consists of power UP or power DOWN step, followed by a power step in the opposite direction (see method RsCmwEvdoMeas.Configure.Oltr.Pstep. direction.

return no_of_meas_seq: numeric Range: 1 to 5

get_timeout() → float

```
# SCPI: CONFigure:EVDO:MEASurement<instance>:OLTR:TOUT
value: float = driver.configure.oltr.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 ([ON | OFF] key or [RESTART | STOP] key) . 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: numeric Unit: s

set_mo_exception(*meas_on_exception: bool*) → None

```
# SCPI: CONFigure:EVDO:MEASurement<instance>:OLTR:MOEXception
driver.configure.oltr.set_mo_exception(meas_on_exception = False)
```

Specifies whether measurement results that the R&S CMW identifies as faulty or inaccurate are rejected.

param meas_on_exception OFF | ON ON: Results are never rejected OFF: Faulty results are rejected

set_sequence(*no_of_meas_seq: int*) → None

```
# SCPI: CONFigure:EVDO:MEASurement<instance>:OLTR:SEQuence
driver.configure.oltr.set_sequence(no_of_meas_seq = 1)
```

Sets/gets the number of measurement sequences within a single OLTR measurement. Each sequence consists of power UP or power DOWN step, followed by a power step in the opposite direction (see method RsCmwEvdoMeas.Configure.Oltr.Pstep. direction.

param no_of_meas_seq numeric Range: 1 to 5

set_timeout(*timeout: float*) → None

```
# SCPI: CONFigure:EVDO:MEASurement<instance>:OLTR:TOUT
driver.configure.oltr.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 ([ON | OFF] key or [RESTART | STOP] key) . 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 numeric Unit: s

Cloning the Group

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

Subgroups

7.2.3.1 Pstep

SCPI Commands

```
CONFigure:EVDO:MEASurement<Instance>:OLTR:PSTep:DIRection
CONFigure:EVDO:MEASurement<Instance>:OLTR:PSTep
```

class Pstep

Pstep commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

get_direction() → RsCmwEvdoMeas.enums.UpDownDirection

```
# SCPI: CONFigure:EVDO:MEASurement<instance>:OLTR:PSTep:DIRection
value: enums.UpDownDirection = driver.configure.oltr.pstep.get_direction()
```

Defines the direction of the first power step within an OLTR measurement. For each subsequent power step, the direction is toggled.

return pstep_direction: DOWN | UP

get_value() → float

```
# SCPI: CONFigure:EVDO:MEASurement<instance>:OLTR:PSTep
value: float = driver.configure.oltr.pstep.get_value()
```

Defines the size of the power steps, i.e. the increases and decreases in the total BSS power during the OLTR measurement.

return power_step: numeric The power step is relative to the measured reference power.
Range: 0 dB to 40 dB, Unit: dB

set_direction(*pstep_direction: RsCmwEvdoMeas.enums.UpDownDirection*) → None

```
# SCPI: CONFigure:EVDO:MEASurement<instance>:OLTR:PSTep:DIRection
driver.configure.oltr.pstep.set_direction(pstep_direction = enums.
↪UpDownDirection.DOWN)
```

Defines the direction of the first power step within an OLTR measurement. For each subsequent power step, the direction is toggled.

param pstep_direction DOWN | UP

set_value(*power_step: float*) → None

```
# SCPI: CONFigure:EVDO:MEASurement<instance>:OLTR:PSTep
driver.configure.oltr.pstep.set_value(power_step = 1.0)
```

Defines the size of the power steps, i.e. the increases and decreases in the total BSS power during the OLTR measurement.

param power_step numeric The power step is relative to the measured reference power. Range: 0 dB to 40 dB, Unit: dB

7.2.3.2 RpInterval

SCPI Commands

```
CONFigure:EVDO:MEASurement<Instance>:OLTR:RPINterval:TIME
CONFigure:EVDO:MEASurement<Instance>:OLTR:RPINterval
```

class RpInterval

RpInterval commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

get_time() → float

```
# SCPI: CONFigure:EVDO:MEASurement<instance>:OLTR:RPINterval:TIME
value: float = driver.configure.oltr.rpInterval.get_time()
```

Gets the duration of the reference power interval, i.e. the interval that is used to calculate the AT reference power for the subsequent power step.

return ref_pow_interval: float Range: 5 ms to 40 ms , Unit: ms

get_value() → int

```
# SCPI: CONFigure:EVDO:MEASurement<instance>:OLTR:RPINterval
value: int = driver.configure.oltr.rpInterval.get_value()
```

Gets the duration of the reference power interval, i.e. the interval that is used to calculate the AT reference power for the subsequent power step.

return ref_pow_interval: integer The time as the number of slots: from 3 (= 5 ms) to 24 (=40 ms) Range: 3 to 24

set_value(*ref_pow_interval: int*) → None

```
# SCPI: CONFigure:EVDO:MEASurement<instance>:OLTR:RPINterval
driver.configure.oltr.rpInterval.set_value(ref_pow_interval = 1)
```

Gets the duration of the reference power interval, i.e. the interval that is used to calculate the AT reference power for the subsequent power step.

param ref_pow_interval integer The time as the number of slots: from 3 (= 5 ms) to 24 (=40 ms) Range: 3 to 24

7.2.3.3 Ginterval

SCPI Commands

```
CONFigure:EVDO:MEASurement<Instance>:OLTR:GINTerval:TIME
CONFigure:EVDO:MEASurement<Instance>:OLTR:GINTerval
```

class Ginterval

Ginterval commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

get_time() → float

```
# SCPI: CONFigure:EVDO:MEASurement<instance>:OLTR:GINTerval:TIME
value: float = driver.configure.oltr.ginterval.get_time()
```

Gets the duration of the guard intervals, i.e. the intervals succeeding the OLTR evaluation intervals and preceding the reference power intervals.

return guard_interval: float Range: 5 ms to 100 ms

get_value() → int

```
# SCPI: CONFigure:EVDO:MEASurement<instance>:OLTR:GINTerval
value: int = driver.configure.oltr.ginterval.get_value()
```

Defines the duration of the guard intervals, i.e. the intervals succeeding the OLTR evaluation intervals and preceding the reference power intervals.

return guard_interval: integer The duration of the guard interval as number of power control groups (1.25 ms) . Range: 3 to 60

set_value(*guard_interval: int*) → None

```
# SCPI: CONFigure:EVDO:MEASurement<instance>:OLTR:GINTerval
driver.configure.oltr.ginterval.set_value(guard_interval = 1)
```

Defines the duration of the guard intervals, i.e. the intervals succeeding the OLTR evaluation intervals and preceding the reference power intervals.

param guard_interval integer The duration of the guard interval as number of power control groups (1.25 ms) . Range: 3 to 60

7.2.3.4 Limit

SCPI Commands

```
CONFigure:EVDO:MEASurement<Instance>:OLTR:LIMit:ILOWer
```

class Limit

Limit commands group definition. 1 total commands, 0 Sub-groups, 1 group commands

get_ilower() → int

```
# SCPI: CONFigure:EVDO:MEASurement<instance>:OLTR:LIMit:ILOWer
value: int = driver.configure.oltr.limit.get_ilower()
```

Sets initial lower limit for open loop power control step response (3GPP2 C.S0033) .

return initial_lower: numeric Range: -2 dB to -1 dB, Unit: dB

set_ilower(*initial_lower: int*) → None

```
# SCPI: CONFigure:EVDO:MEASurement<instance>:OLTR:LIMit:ILOWer
driver.configure.oltr.limit.set_ilower(initial_lower = 1)
```

Sets initial lower limit for open loop power control step response (3GPP2 C.S0033) .

param initial_lower numeric Range: -2 dB to -1 dB, Unit: dB

7.3 Trigger

class Trigger

Trigger commands group definition. 9 total commands, 1 Sub-groups, 0 group commands

Cloning the Group

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

Subgroups

7.3.1 MultiEval

SCPI Commands

```
TRIGger:EVDO:MEASurement<Instance>:MEValuation:SOURce
TRIGger:EVDO:MEASurement<Instance>:MEValuation:SLOPe
TRIGger:EVDO:MEASurement<Instance>:MEValuation:THReshold
TRIGger:EVDO:MEASurement<Instance>:MEValuation:DELay
TRIGger:EVDO:MEASurement<Instance>:MEValuation:TOUT
TRIGger:EVDO:MEASurement<Instance>:MEValuation:MGAP
TRIGger:EVDO:MEASurement<Instance>:MEValuation:EOFFset
```

class MultiEval

MultiEval commands group definition. 9 total commands, 2 Sub-groups, 7 group commands

get_delay() → float

```
# SCPI: TRIGger:EVDO:MEASurement<instance>:MEValuation:DELay
value: float = driver.trigger.multiEval.get_delay()
```

Defines a time delaying the start of the measurement relative to the trigger event. This setting has no influence on 'Free Run' measurements.

return delay: numeric Range: -1.25E-3 s to 0.08 s, Unit: s

get_eoffset() → int

```
# SCPI: TRIGger:EVDO:MEASurement<instance>:MEValuation:EOFFset
value: int = driver.trigger.multiEval.get_eoffset()
```

Defines a delay time for the measurement relative to the 'IF Power' or external trigger events (see method RsCmwEvdoMeas. Trigger.MultiEval.source) . The range is entered as an integer number of slots. Each slot has a duration of 1.25 ms.

return eval_offset: integer Range: 0 to 64, Unit: slot

get_mgap() → float

```
# SCPI: TRIGger:EVDO:MEASurement<instance>:MEValuation:MGAP
value: float = driver.trigger.multiEval.get_mgap()
```

Sets a minimum time during which the IF signal must be below the trigger threshold before the trigger is armed so that an IF power trigger event can be generated.

return min_trigger_gap: numeric Range: 0 s to 0.01 s, Unit: s

get_slope() → RsCmwEvdoMeas.enums.TriggerSlope

```
# SCPI: TRIGger:EVDO:MEASurement<instance>:MEValuation:SLOPe
value: enums.TriggerSlope = driver.trigger.multiEval.get_slope()
```

Qualifies whether the trigger event is generated at the rising or at the falling edge of the trigger pulse (valid for external and power trigger sources) .

return trigger_slope: REDGe | FEDGe REDGe: Rising edge FEDGe: Falling edge

get_source() → str

```
# SCPI: TRIGger:EVDO:MEASurement<instance>:MEValuation:SOURce
value: str = driver.trigger.multiEval.get_source()
```

Selects the source of the trigger events. Some values are always available. They are listed below. Depending on the installed options, additional values are available. You can query a list of all supported values via TRIGger:... :CATalog:SOURce?.

return trigger_source: string 'Free Run': Free run (untriggered) 'IF Power': Power trigger (received RF power) 'IF Auto Sync': Power trigger auto synchronized

get_threshold() → float

```
# SCPI: TRIGger:EVDO:MEASurement<instance>:MEValuation:THReshold
value: float = driver.trigger.multiEval.get_threshold()
```

Defines the trigger threshold for power trigger sources.

return threshold: numeric Range: -50 dB to 0 dB, Unit: dB

get_timeout() → float

```
# SCPI: TRIGger:EVDO:MEASurement<instance>:MEValuation:TOUT
value: float or bool = driver.trigger.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. This setting has no influence on 'Free Run' measurements.

return trigger_time: Range: 0 s to 83.88607E+3 s, Unit: s Additional parameters: OFF | ON (disables | enables the timeout)

set_delay(*delay: float*) → None

```
# SCPI: TRIGger:EVDO:MEASurement<instance>:MEValuation:DELay
driver.trigger.multiEval.set_delay(delay = 1.0)
```

Defines a time delaying the start of the measurement relative to the trigger event. This setting has no influence on 'Free Run' measurements.

param delay numeric Range: -1.25E-3 s to 0.08 s, Unit: s

set_eoffset(*eval_offset: int*) → None

```
# SCPI: TRIGger:EVDO:MEASurement<instance>:MEValuation:EOFFset
driver.trigger.multiEval.set_eoffset(eval_offset = 1)
```

Defines a delay time for the measurement relative to the 'IF Power' or external trigger events (see method RsCmwEvdoMeas. Trigger.MultiEval.source) . The range is entered as an integer number of slots. Each slot has a duration of 1.25 ms.

param eval_offset integer Range: 0 to 64, Unit: slot

set_mgap(*min_trigger_gap: float*) → None

```
# SCPI: TRIGger:EVDO:MEASurement<instance>:MEValuation:MGAP
driver.trigger.multiEval.set_mgap(min_trigger_gap = 1.0)
```

Sets a minimum time during which the IF signal must be below the trigger threshold before the trigger is armed so that an IF power trigger event can be generated.

param min_trigger_gap numeric Range: 0 s to 0.01 s, Unit: s

set_slope(*trigger_slope: RsCmwEvdoMeas.enums.TriggerSlope*) → None

```
# SCPI: TRIGger:EVDO:MEASurement<instance>:MEValuation:SLOPe
driver.trigger.multiEval.set_slope(trigger_slope = enums.TriggerSlope.FEDGe)
```

Qualifies whether the trigger event is generated at the rising or at the falling edge of the trigger pulse (valid for external and power trigger sources) .

param trigger_slope REDGe | FEDGe REDGe: Rising edge FEDGe: Falling edge

set_source(*trigger_source: str*) → None

```
# SCPI: TRIGger:EVDO:MEASurement<instance>:MEValuation:SOURce
driver.trigger.multiEval.set_source(trigger_source = '1')
```

Selects the source of the trigger events. Some values are always available. They are listed below. Depending on the installed options, additional values are available. You can query a list of all supported values via TRIGger:... :CATalog:SOURce?.

param trigger_source string 'Free Run': Free run (untriggered) 'IF Power': Power trigger (received RF power) 'IF Auto Sync': Power trigger auto synchronized

set_threshold(*threshold: float*) → None

```
# SCPI: TRIGger:EVDO:MEASurement<instance>:MEValuation:THReshold
driver.trigger.multiEval.set_threshold(threshold = 1.0)
```

Defines the trigger threshold for power trigger sources.

param threshold numeric Range: -50 dB to 0 dB, Unit: dB

set_timeout(*trigger_time: float*) → None

```
# SCPI: TRIGger:EVDO:MEASurement<instance>:MEValuation:TOUT
driver.trigger.multiEval.set_timeout(trigger_time = 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. This setting has no influence on 'Free Run' measurements.

param trigger_time Range: 0 s to 83.88607E+3 s, Unit: s Additional parameters: OFF | ON (disables | enables the timeout)

Cloning the Group

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

Subgroups

7.3.1.1 Catalog

SCPI Commands

```
TRIGger:EVDO:MEASurement<Instance>:MEValuation:CATalog:SOURce
```

class Catalog

Catalog commands group definition. 1 total commands, 0 Sub-groups, 1 group commands

get_source() → List[str]

```
# SCPI: TRIGger:EVDO:MEASurement<instance>:MEValuation:CATalog:SOURce
value: List[str] = driver.trigger.multiEval.catalog.get_source()
```

Lists all trigger source values that can be set using method RsCmwEvdoMeas.Trigger.MultiEval.source.

return trigger_list: string Comma-separated list of all supported values. Each value is represented as a string.

7.3.1.2 ListPy

SCPI Commands

```
TRIGger:EVDO:MEASurement<Instance>:MEValuation:LIST:MODE
```

class ListPy

ListPy commands group definition. 1 total commands, 0 Sub-groups, 1 group commands

get_mode() → RsCmwEvdoMeas.enums.RetriggerMode

```
# SCPI: TRIGger:EVDO:MEASurement<instance>:MEValuation:LIST:MODE
value: enums.RetriggerMode = driver.trigger.multiEval.listPy.get_mode()
```

Specifies whether a trigger event initiates a measurement of the entire measurement interval (comprising the number of segments defined via method RsCmwEvdoMeas.Configure.MultiEval.ListPy.count) or the retrigger information from the segments is used.

return retrigger_mode: ONCE | SEGMent ONCE: Trigger only once. Every segment is measured irrespective the setting of the parameter RetriggerOption for the segment (method RsCmwEvdoMeas.Configure.MultiEval.ListPy.Segment.Setup.set) . The trigger is rearmed only after the measurement is stopped and restarted. SEGM: The measurement starts after the first trigger event and continues as long as no segment is reached that requires a retrigger (method RsCmwEvdoMeas.Configure.MultiEval.ListPy.Segment.Setup.set) . This mode is recommended for statistic counts where retriggering can compensate a possible time drift of the AT.

set_mode(*retrigger_mode: RsCmwEvdoMeas.enums.RetriggerMode*) → None

```
# SCPI: TRIGger:EVDO:MEASurement<instance>:MEValuation:LIST:MODE
driver.trigger.multiEval.listPy.set_mode(retrigger_mode = enums.RetriggerMode.
↪ONCE)
```

Specifies whether a trigger event initiates a measurement of the entire measurement interval (comprising the number of segments defined via method RsCmwEvdoMeas.Configure.MultiEval.ListPy.count) or the retrigger information from the segments is used.

param retrigger_mode ONCE | SEGMent ONCE: Trigger only once. Every segment is measured irrespective the setting of the parameter RetriggerOption for the segment (method RsCmwEvdoMeas.Configure.MultiEval.ListPy.Segment.Setup.set) . The trigger is rearmed only after the measurement is stopped and restarted. SEGM: The measurement starts after the first trigger event and continues as long as no segment is reached that requires a retrigger (method RsCmwEvdoMeas.Configure.MultiEval.ListPy.Segment.Setup.set) . This mode is recommended for statistic counts where retriggering can compensate a possible time drift of the AT.

7.4 MultiEval

SCPI Commands

```
INITiate:EVDO:MEASurement<Instance>:MEValuation
ABORt:EVDO:MEASurement<Instance>:MEValuation
STOP:EVDO:MEASurement<Instance>:MEValuation
```

class MultiEval

MultiEval commands group definition. 592 total commands, 7 Sub-groups, 3 group commands

abort() → None

```
# SCPI: ABORt:EVDO:MEASurement<instance>:MEValuation
driver.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.

abort_with_opc() → None

```
# SCPI: ABORt:EVDO:MEASurement<instance>:MEValuation
driver.multiEval.abort_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 abort, but waits for the operation to complete before continuing further. Use the RsCmwEvdoMeas.utilities.opc_timeout_set() to set the timeout value.

initiate() → None

```
# SCPI: INITiate:EVDO:MEASurement<instance>:MEValuation
driver.multiEval.initiate()


    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.

initiate_with_opc() → None

```
# SCPI: INITiate:EVDO:MEASurement<instance>:MEValuation
driver.multiEval.initiate_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 initiate, but waits for the operation to complete before continuing further. Use the RsCmwEvdoMeas.utilities.opc_timeout_set() to set the timeout value.

stop() → None

```
# SCPI: STOP:EVDO:MEASurement<instance>:MEValuation
driver.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.
```

(continues on next page)

(continued from previous page)

```
    - 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() → None

```
# SCPI: STOP:EVDO:MEASurement<instance>:MEValuation
driver.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 RsCmwEvdoMeas.utilities.opc_timeout_set() to set the timeout value.

Cloning the Group

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

Subgroups

7.4.1 State

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:STATe
```

class State

State commands group definition. 2 total commands, 1 Sub-groups, 1 group commands

fetch() → RsCmwEvdoMeas.enums.ResourceState

```
# SCPI: FETCh:EVDO:MEASurement<instance>:MEValuation:STATe
value: enums.ResourceState = driver.multiEval.state.fetch()
```

Queries the main measurement state. Use FETCh:...:STATe:ALL? to query the measurement state including the substates. Use INITiate..., STOP..., ABORt... to change the measurement state.

return meas_state: OFF | RUN | RDY OFF: measurement switched off, no resources allocated, no results available (when entered after ABORt...) RUN: measurement running (after INITiate..., READ...) , synchronization pending or adjusted, resources active or queued RDY: measurement has been terminated, valid results are available

Cloning the Group

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

Subgroups

7.4.1.1 All

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:STATe:ALL
```

class All

All commands group definition. 1 total commands, 0 Sub-groups, 1 group commands

class FetchStruct

Response structure. Fields:

- Main_State: enums.ResourceState: OFF | RDY | RUN OFF: measurement switched off, no resources allocated, no results available (when entered after STOP...) RDY: measurement has been terminated, valid results are available RUN: measurement running (after INITiate..., READ...) , synchronization pending or adjusted, resources active or queued
- Sync_State: enums.ResourceState: PEND | ADJ | INV PEND: waiting for resource allocation, adjustment, hardware switching ('pending') ADJ: all necessary adjustments finished, measurement running ('adjusted') INV: not applicable because MainState: OFF or RDY ('invalid')
- Resource_State: enums.ResourceState: QUE | ACT | INV QUE: measurement without resources, no results available ('queued') ACT: resources allocated, acquisition of results in progress but not complete ('active') INV: not applicable because MainState: OFF or RDY ('invalid')

fetch() → FetchStruct

```
# SCPI: FETCh:EVDO:MEASurement<instance>:MEValuation:STATe:ALL
value: FetchStruct = driver.multiEval.state.all.fetch()
```

Queries the main measurement state and the measurement substates. Both measurement substates are relevant for running measurements only. Use FETCh:...:STATe? to query the main measurement state only. Use INITiate..., STOP..., ABORt... to change the measurement state.

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

7.4.2 Trace

class Trace
Trace commands group definition. 174 total commands, 10 Sub-groups, 0 group commands

Cloning the Group

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

Subgroups

7.4.2.1 EvMagnitude

class EvMagnitude
EvMagnitude commands group definition. 6 total commands, 3 Sub-groups, 0 group commands

Cloning the Group

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

Subgroups

7.4.2.1.1 Current

SCPI Commands

```
READ:EVDO:MEASurement<Instance>:MEValuation:TRACe:EVMagnitude:CURRent
FETCh:EVDO:MEASurement<Instance>:MEValuation:TRACe:EVMagnitude:CURRent
```

class Current
Current commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

fetch() → List[float]

```
# SCPI: FETCh:EVDO:MEASurement<instance>:MEValuation:TRACe:EVMagnitude:CURRent
value: List[float] = driver.multiEval.trace.evMagnitude.current.fetch()
```

Returns the values of the RMS EVM traces. A trace covers a time interval of 833 s (one half-slot) and contains one value per chip. The results of the current, average and maximum traces can be retrieved.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return curr_evm: float Range: 0 % to 100 %, Unit: %

read() → List[float]

```
# SCPI: READ:EVDO:MEASurement<instance>:MEValuation:TRACe:EVMagnitude:CURRent
value: List[float] = driver.multiEval.trace.evMagnitude.current.read()
```

Returns the values of the RMS EVM traces. A trace covers a time interval of 833 s (one half-slot) and contains one value per chip. The results of the current, average and maximum traces can be retrieved.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return curr_evm: float Range: 0 % to 100 %, Unit: %

7.4.2.1.2 Average

SCPI Commands

```
READ:EVDO:MEASurement<Instance>:MEValuation:TRACe:EVMagnitude:AVERage
FETCh:EVDO:MEASurement<Instance>:MEValuation:TRACe:EVMagnitude:AVERage
```

class Average

Average commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

fetch() → List[float]

```
# SCPI: FETCh:EVDO:MEASurement<instance>:MEValuation:TRACe:EVMagnitude:AVERage
value: List[float] = driver.multiEval.trace.evMagnitude.average.fetch()
```

Returns the values of the RMS EVM traces. A trace covers a time interval of 833 s (one half-slot) and contains one value per chip. The results of the current, average and maximum traces can be retrieved.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return aver_evm: float Range: 0 % to 100 %, Unit: %

read() → List[float]

```
# SCPI: READ:EVDO:MEASurement<instance>:MEValuation:TRACe:EVMagnitude:AVERage
value: List[float] = driver.multiEval.trace.evMagnitude.average.read()
```

Returns the values of the RMS EVM traces. A trace covers a time interval of 833 s (one half-slot) and contains one value per chip. The results of the current, average and maximum traces can be retrieved.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return aver_evm: float Range: 0 % to 100 %, Unit: %

7.4.2.1.3 Maximum

SCPI Commands

```
READ:EVDO:MEASurement<Instance>:MEValuation:TRACe:EVMagnitude:MAXimum
FETCh:EVDO:MEASurement<Instance>:MEValuation:TRACe:EVMagnitude:MAXimum
```

class Maximum

Maximum commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

fetch() → List[float]

```
# SCPI: FETCh:EVDO:MEASurement<instance>:MEValuation:TRACe:EVMagnitude:MAXimum
value: List[float] = driver.multiEval.trace.evMagnitude.maximum.fetch()
```

Returns the values of the RMS EVM traces. A trace covers a time interval of 833 s (one half-slot) and contains one value per chip. The results of the current, average and maximum traces can be retrieved.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return max_evm: float Range: 0 % to 100 %, Unit: %

read() → List[float]

```
# SCPI: READ:EVDO:MEASurement<instance>:MEValuation:TRACe:EVMagnitude:MAXimum
value: List[float] = driver.multiEval.trace.evMagnitude.maximum.read()
```

Returns the values of the RMS EVM traces. A trace covers a time interval of 833 s (one half-slot) and contains one value per chip. The results of the current, average and maximum traces can be retrieved.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return max_evm: float Range: 0 % to 100 %, Unit: %

7.4.2.2 Merror

class Merror

Merror commands group definition. 6 total commands, 3 Sub-groups, 0 group commands

Cloning the Group

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

Subgroups

7.4.2.2.1 Current

SCPI Commands

```
READ:EVDO:MEASurement<Instance>:MEValuation:TRACe:MERRor:CURRent
FETCh:EVDO:MEASurement<Instance>:MEValuation:TRACe:MERRor:CURRent
```

class Current

Current commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

fetch() → List[float]

```
# SCPI: FETCh:EVDO:MEASurement<instance>:MEValuation:TRACe:MERRor:CURRent
value: List[float] = driver.multiEval.trace.merror.current.fetch()
```

Returns the values of the RMS magnitude error traces. A trace covers a time interval of 833 s (one half-slot) and contains one value per chip. The results of the current, average and maximum traces can be retrieved.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return curr_merr: float Range: -100 % to 100 %, Unit: %

read() → List[float]

```
# SCPI: READ:EVDO:MEASurement<instance>:MEValuation:TRACe:MERRor:CURRent
value: List[float] = driver.multiEval.trace.merror.current.read()
```

Returns the values of the RMS magnitude error traces. A trace covers a time interval of 833 s (one half-slot) and contains one value per chip. The results of the current, average and maximum traces can be retrieved.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return curr_merr: float Range: -100 % to 100 %, Unit: %

7.4.2.2.2 Average

SCPI Commands

```
READ:EVDO:MEASurement<Instance>:MEValuation:TRACe:MERRor:AVERage
FETCh:EVDO:MEASurement<Instance>:MEValuation:TRACe:MERRor:AVERage
```

class Average

Average commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

fetch() → List[float]

```
# SCPI: FETCh:EVDO:MEASurement<instance>:MEValuation:TRACe:MERRor:AVERage
value: List[float] = driver.multiEval.trace.merror.average.fetch()
```

Returns the values of the RMS magnitude error traces. A trace covers a time interval of 833 s (one half-slot) and contains one value per chip. The results of the current, average and maximum traces can be retrieved.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return aver_merr: float Range: -100 % to 100 %, Unit: %

read() → List[float]

```
# SCPI: READ:EVDO:MEASurement<instance>:MEValuation:TRACe:MERRor:AVERage
value: List[float] = driver.multiEval.trace.merror.average.read()
```

Returns the values of the RMS magnitude error traces. A trace covers a time interval of 833 s (one half-slot) and contains one value per chip. The results of the current, average and maximum traces can be retrieved.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return aver_merr: float Range: -100 % to 100 %, Unit: %

7.4.2.2.3 Maximum

SCPI Commands

```
READ:EVDO:MEASurement<Instance>:MEValuation:TRACe:MERRor:MAXimum
FETCh:EVDO:MEASurement<Instance>:MEValuation:TRACe:MERRor:MAXimum
```

class Maximum

Maximum commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

fetch() → List[float]

```
# SCPI: FETCh:EVDO:MEASurement<instance>:MEValuation:TRACe:MERRor:MAXimum
value: List[float] = driver.multiEval.trace.merror.maximum.fetch()
```

Returns the values of the RMS magnitude error traces. A trace covers a time interval of 833 s (one half-slot) and contains one value per chip. The results of the current, average and maximum traces can be retrieved.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return max_merr: float Range: -100 % to 100 %, Unit: %

read() → List[float]

```
# SCPI: READ:EVDO:MEASurement<instance>:MEValuation:TRACe:MERRor:MAXimum
value: List[float] = driver.multiEval.trace.merror.maximum.read()
```

Returns the values of the RMS magnitude error traces. A trace covers a time interval of 833 s (one half-slot) and contains one value per chip. The results of the current, average and maximum traces can be retrieved.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return max_merr: float Range: -100 % to 100 %, Unit: %

7.4.2.3 Perror

class Perror

Perror commands group definition. 6 total commands, 3 Sub-groups, 0 group commands

Cloning the Group

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

Subgroups

7.4.2.3.1 Current

SCPI Commands

```
READ:EVDO:MEASurement<Instance>:MEValuation:TRACe:PERRor:CURRent
FETCh:EVDO:MEASurement<Instance>:MEValuation:TRACe:PERRor:CURRent
```

class Current

Current commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

fetch() → List[float]

```
# SCPI: FETCh:EVDO:MEASurement<instance>:MEValuation:TRACe:PERRor:CURRent
value: List[float] = driver.multiEval.trace.perror.current.fetch()
```

Returns the values of the phase error traces. A trace covers a time interval of 833 s (one half-slot) and contains one value per chip. The results of the current, average and maximum traces can be retrieved.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return curr_perr: float Range: -180 deg to 180 deg, Unit: deg

read() → List[float]

```
# SCPI: READ:EVDO:MEASurement<instance>:MEValuation:TRACe:PERRor:CURRent
value: List[float] = driver.multiEval.trace.perror.current.read()
```

Returns the values of the phase error traces. A trace covers a time interval of 833 s (one half-slot) and contains one value per chip. The results of the current, average and maximum traces can be retrieved.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return curr_perr: float Range: -180 deg to 180 deg, Unit: deg

7.4.2.3.2 Average

SCPI Commands

```
READ:EVDO:MEASurement<Instance>:MEValuation:TRACe:PERRor:AVERage
FETCh:EVDO:MEASurement<Instance>:MEValuation:TRACe:PERRor:AVERage
```

class Average

Average commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

fetch() → List[float]

```
# SCPI: FETCh:EVDO:MEASurement<instance>:MEValuation:TRACe:PERRor:AVERage
value: List[float] = driver.multiEval.trace.perror.average.fetch()
```

Returns the values of the phase error traces. A trace covers a time interval of 833 s (one half-slot) and contains one value per chip. The results of the current, average and maximum traces can be retrieved.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return aver_perr: float Range: -180 deg to 180 deg, Unit: deg

read() → List[float]

```
# SCPI: READ:EVDO:MEASurement<instance>:MEValuation:TRACe:PERRor:AVERage
value: List[float] = driver.multiEval.trace.perror.average.read()
```

Returns the values of the phase error traces. A trace covers a time interval of 833 s (one half-slot) and contains one value per chip. The results of the current, average and maximum traces can be retrieved.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return aver_perr: float Range: -180 deg to 180 deg, Unit: deg

7.4.2.3.3 Maximum

SCPI Commands

```
READ:EVDO:MEASurement<Instance>:MEValuation:TRACe:PERRor:MAXimum
FETCh:EVDO:MEASurement<Instance>:MEValuation:TRACe:PERRor:MAXimum
```

class Maximum

Maximum commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

fetch() → List[float]

```
# SCPI: FETCh:EVDO:MEASurement<instance>:MEValuation:TRACe:PERRor:MAXimum
value: List[float] = driver.multiEval.trace.perror.maximum.fetch()
```

Returns the values of the phase error traces. A trace covers a time interval of 833 s (one half-slot) and contains one value per chip. The results of the current, average and maximum traces can be retrieved.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return max_perr: float Range: -180 deg to 180 deg, Unit: deg

read() → List[float]

```
# SCPI: READ:EVDO:MEASurement<instance>:MEValuation:TRACe:PERRor:MAXimum
value: List[float] = driver.multiEval.trace.perror.maximum.read()
```

Returns the values of the phase error traces. A trace covers a time interval of 833 s (one half-slot) and contains one value per chip. The results of the current, average and maximum traces can be retrieved.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return max_perr: float Range: -180 deg to 180 deg, Unit: deg

7.4.2.4 Cdp

class Cdp
Cdp commands group definition. 56 total commands, 2 Sub-groups, 0 group commands

Cloning the Group

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

Subgroups

7.4.2.4.1 Isignal

class Isignal
Isignal commands group definition. 28 total commands, 2 Sub-groups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.multiEval.trace.cdp.isignal.clone()
```

Subgroups

7.4.2.4.1.1 Rri

class Rri
Rri commands group definition. 14 total commands, 6 Sub-groups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.multiEval.trace.cdp.isignal.rri.clone()
```

Subgroups

7.4.2.4.1.2 Current

SCPI Commands

```
READ:EVDO:MEASurement<Instance>:MEValuation:TRACe:CDP:ISIGnal:RRI:CURRent
FETCh:EVDO:MEASurement<Instance>:MEValuation:TRACe:CDP:ISIGnal:RRI:CURRent
CALCulate:EVDO:MEASurement<Instance>:MEValuation:TRACe:CDP:ISIGnal:RRI:CURRent
```

class Current
Current commands group definition. 3 total commands, 0 Sub-groups, 3 group commands

calculate() → List[RsCmwEvdoMeas.enums.ResultStatus2]

```
# SCPI: CALCulate:EVDO:MEASurement<instance>
↪:MEValuation:TRACe:CDP:ISIGnal:RRI:CURRent
value: List[enums.ResultStatus2] = driver.multiEval.trace.cdp.isignal.rri.
↪current.calculate()
```

Return values of the code domain power (CDP) I-Signal and Q-Signal bar graphs. The results of the current, average, maximum and minimum bar graphs can be retrieved. For a physical layer subtype 2 or 3 measurement, the bar graphs contain only RRI results. For a physical layer subtype 0/1 measurement the bar graphs contain also pilot results, see method RsCmwEvdoMeas.MultiEval.Trace.Cdp.Isignal.Pilot.Current.fetch etc. 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.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return isig_rri_curr_cdp: float The number of results depends on the physical layer subtype (see method RsCmwEvdoMeas.Configure.MultiEval.plSubtype) : SF=15 for subtype 0/1 and SF=31 for subtypes 2 and 3. Range: - 70 dB to 0 dB, Unit: dB

fetch() → List[float]

```
# SCPI: FETCh:EVDO:MEASurement<instance>
↪:MEValuation:TRACe:CDP:ISIGnal:RRI:CURRent
value: List[float] = driver.multiEval.trace.cdp.isignal.rri.current.fetch()
```

Return values of the code domain power (CDP) I-Signal and Q-Signal bar graphs. The results of the current, average, maximum and minimum bar graphs can be retrieved. For a physical layer subtype 2 or 3 measurement, the bar graphs contain only RRI results. For a physical layer subtype 0/1 measurement the bar graphs contain also pilot results, see method RsCmwEvdoMeas.MultiEval.Trace.Cdp.Isignal.Pilot.Current.fetch etc. 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.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return isig_rri_curr_cdp: float The number of results depends on the physical layer subtype (see method RsCmwEvdoMeas.Configure.MultiEval.plSubtype) : SF=15 for subtype 0/1 and SF=31 for subtypes 2 and 3. Range: - 70 dB to 0 dB, Unit: dB

read() → List[float]

```
# SCPI: READ:EVDO:MEASurement<instance>
↪:MEValuation:TRACe:CDP:ISIGnal:RRI:CURRent
value: List[float] = driver.multiEval.trace.cdp.isignal.rri.current.read()
```

Return values of the code domain power (CDP) I-Signal and Q-Signal bar graphs. The results of the current, average, maximum and minimum bar graphs can be retrieved. For a physical layer subtype 2 or 3 measurement, the bar graphs contain only RRI results. For a physical layer subtype 0/1 measurement the bar graphs contain also pilot results, see method RsCmwEvdoMeas.MultiEval.Trace.Cdp.Isignal.Pilot.Current.fetch etc. 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.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return isig_rri_curr_cdp: float The number of results depends on the physical layer subtype (see method RsCmwEvdoMeas.Configure.MultiEval.plSubtype) : SF=15 for subtype 0/1 and SF=31 for subtypes 2 and 3. Range: - 70 dB to 0 dB, Unit: dB

7.4.2.4.1.3 Average

SCPI Commands

```
READ:EVDO:MEASurement<Instance>:MEValuation:TRACe:CDP:ISIGnal:RRI:AVERage
FETCh:EVDO:MEASurement<Instance>:MEValuation:TRACe:CDP:ISIGnal:RRI:AVERage
CALCulate:EVDO:MEASurement<Instance>:MEValuation:TRACe:CDP:ISIGnal:RRI:AVERage
```

class Average

Average commands group definition. 3 total commands, 0 Sub-groups, 3 group commands

calculate() → List[RsCmwEvdoMeas.enums.ResultStatus2]

```
# SCPI: CALCulate:EVDO:MEASurement<instance>
↪:MEValuation:TRACe:CDP:ISIGnal:RRI:AVERage
value: List[enums.ResultStatus2] = driver.multiEval.trace.cdp.isignal.rri.
↪average.calculate()
```

Return values of the code domain power (CDP) I-Signal and Q-Signal bar graphs. The results of the current, average, maximum and minimum bar graphs can be retrieved. For a physical layer subtype 2 or 3 measurement, the bar graphs contain only RRI results. For a physical layer subtype 0/1 measurement the bar graphs contain also pilot results, see method RsCmwEvdoMeas.MultiEval.Trace.Cdp.Isignal.Pilot.Current.fetch etc. 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.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return isig_rri_aver_cdp: float The number of results depends on the physical layer subtype (see method RsCmwEvdoMeas.Configure.MultiEval.plSubtype) : SF=15 for subtype 0/1 and SF=31 for subtypes 2 and 3. Range: - 70 dB to 0 dB, Unit: dB

fetch() → List[float]

```
# SCPI: FETCh:EVDO:MEASurement<instance>
↪:MEValuation:TRACe:CDP:ISIGnal:RRI:AVERage
value: List[float] = driver.multiEval.trace.cdp.isignal.rri.average.fetch()
```

Return values of the code domain power (CDP) I-Signal and Q-Signal bar graphs. The results of the current, average, maximum and minimum bar graphs can be retrieved. For a physical layer subtype 2 or 3 measurement, the bar graphs contain only RRI results. For a physical layer subtype 0/1 measurement the bar graphs contain also pilot results, see method RsCmwEvdoMeas.MultiEval.Trace.Cdp.Isignal.Pilot.Current.fetch etc. 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.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return isig_rri_aver_cdp: float The number of results depends on the physical layer subtype (see method RsCmwEvdoMeas.Configure.MultiEval.plSubtype) : SF=15 for subtype 0/1 and SF=31 for subtypes 2 and 3. Range: - 70 dB to 0 dB, Unit: dB

read() → List[float]

```
# SCPI: READ:EVDO:MEASurement<instance>
↪:MEValuation:TRACe:CDP:ISIGnal:RRI:AVERage
value: List[float] = driver.multiEval.trace.cdp.isignal.rri.average.read()
```

Return values of the code domain power (CDP) I-Signal and Q-Signal bar graphs. The results of the current, average, maximum and minimum bar graphs can be retrieved. For a physical layer subtype 2 or 3 measurement, the bar graphs contain only RRI results. For a physical layer subtype 0/1 measurement the bar graphs contain also pilot results, see method RsCmwEvdoMeas.MultiEval.Trace.Cdp.Isignal.Pilot.Current.fetch etc. 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.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return isig_rri_aver_cdp: float The number of results depends on the physical layer subtype (see method RsCmwEvdoMeas.Configure.MultiEval.plSubtype) : SF=15 for subtype 0/1 and SF=31 for subtypes 2 and 3. Range: - 70 dB to 0 dB, Unit: dB

7.4.2.4.1.4 Maximum

SCPI Commands

```
READ:EVDO:MEASurement<Instance>:MEValuation:TRACe:CDP:ISIGnal:RRI:MAXimum
FETCh:EVDO:MEASurement<Instance>:MEValuation:TRACe:CDP:ISIGnal:RRI:MAXimum
CALCulate:EVDO:MEASurement<Instance>:MEValuation:TRACe:CDP:ISIGnal:RRI:MAXimum
```

class Maximum

Maximum commands group definition. 3 total commands, 0 Sub-groups, 3 group commands

calculate() → List[RsCmwEvdoMeas.enums.ResultStatus2]

```
# SCPI: CALCulate:EVDO:MEASurement<instance>
↪:MEValuation:TRACe:CDP:ISIGnal:RRI:MAXimum
value: List[enums.ResultStatus2] = driver.multiEval.trace.cdp.isignal.rri.
↪maximum.calculate()
```

Return values of the code domain power (CDP) I-Signal and Q-Signal bar graphs. The results of the current, average, maximum and minimum bar graphs can be retrieved. For a physical layer subtype 2 or 3 measurement, the bar graphs contain only RRI results. For a physical layer subtype 0/1 measurement the bar graphs contain also pilot results, see method RsCmwEvdoMeas.MultiEval.Trace.Cdp.Isignal.Pilot.Current.fetch etc. 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.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return isig_rri_max_cdp: float The number of results depends on the physical layer subtype (see method RsCmwEvdoMeas.Configure.MultiEval.plSubtype) : SF=15 for subtype 0/1 and SF=31 for subtypes 2 and 3. Range: - 70 dB to 0 dB, Unit: dB

fetch() → List[float]

```
# SCPI: FETCh:EVDO:MEASurement<instance>
↪:MEValuation:TRACe:CDP:ISIGnal:RRI:MAXimum
value: List[float] = driver.multiEval.trace.cdp.isignal.rri.maximum.fetch()
```

Return values of the code domain power (CDP) I-Signal and Q-Signal bar graphs. The results of the current, average, maximum and minimum bar graphs can be retrieved. For a physical layer subtype 2 or 3 measurement, the bar graphs contain only RRI results. For a physical layer subtype 0/1 measurement the bar graphs contain also pilot results, see method RsCmwEvdoMeas.MultiEval.Trace.Cdp.Isignal.Pilot.Current.fetch etc. 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.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return isig_rri_max_cdp: float The number of results depends on the physical layer subtype (see method RsCmwEvdoMeas.Configure.MultiEval.plSubtype) : SF=15 for subtype 0/1 and SF=31 for subtypes 2 and 3. Range: - 70 dB to 0 dB, Unit: dB

read() → List[float]

```
# SCPI: READ:EVDO:MEASurement<instance>
↪:MEValuation:TRACe:CDP:ISIGnal:RRI:MAXimum
value: List[float] = driver.multiEval.trace.cdp.isignal.rri.maximum.read()
```

Return values of the code domain power (CDP) I-Signal and Q-Signal bar graphs. The results of the current, average, maximum and minimum bar graphs can be retrieved. For a physical layer subtype 2 or 3 measurement, the bar graphs contain only RRI results. For a physical layer subtype 0/1 measurement the bar graphs contain also pilot results, see method RsCmwEvdoMeas.MultiEval.Trace.Cdp.Isignal.Pilot.Current.fetch etc. 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.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return isig_rri_max_cdp: float The number of results depends on the physical layer subtype (see method RsCmwEvdoMeas.Configure.MultiEval.plSubtype) : SF=15 for subtype 0/1 and SF=31 for subtypes 2 and 3. Range: - 70 dB to 0 dB, Unit: dB

7.4.2.4.1.5 Minimum

SCPI Commands

```
READ:EVDO:MEASurement<Instance>:MEValuation:TRACe:CDP:ISIGnal:RRI:MINimum
FETCh:EVDO:MEASurement<Instance>:MEValuation:TRACe:CDP:ISIGnal:RRI:MINimum
CALCulate:EVDO:MEASurement<Instance>:MEValuation:TRACe:CDP:ISIGnal:RRI:MINimum
```

class Minimum

Minimum commands group definition. 3 total commands, 0 Sub-groups, 3 group commands

calculate() → List[RsCmwEvdoMeas.enums.ResultStatus2]

```
# SCPI: CALCulate:EVDO:MEASurement<instance>
↪:MEValuation:TRACe:CDP:ISIGnal:RRI:MINimum
value: List[enums.ResultStatus2] = driver.multiEval.trace.cdp.isignal.rri.
↪minimum.calculate()
```

Return values of the code domain power (CDP) I-Signal and Q-Signal bar graphs. The results of the current, average, maximum and minimum bar graphs can be retrieved. For a physical layer subtype 2 or 3 measurement, the bar graphs contain only RRI results. For a physical layer subtype 0/1 measurement the bar graphs contain also pilot results, see method RsCmwEvdoMeas.MultiEval.Trace.Cdp.Isignal.Pilot.Current.fetch etc. 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.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return isig_rri_min_cdp: float The number of results depends on the physical layer subtype (see method RsCmwEvdoMeas.Configure.MultiEval.plSubtype) : SF=15 for subtype 0/1 and SF=31 for subtypes 2 and 3. Range: - 70 dB to 0 dB, Unit: dB

fetch() → List[float]

```
# SCPI: FETCh:EVDO:MEASurement<instance>
↪:MEValuation:TRACe:CDP:ISIGnal:RRI:MINimum
value: List[float] = driver.multiEval.trace.cdp.isignal.rri.minimum.fetch()
```

Return values of the code domain power (CDP) I-Signal and Q-Signal bar graphs. The results of the current, average, maximum and minimum bar graphs can be retrieved. For a physical layer subtype 2 or 3 measurement, the bar graphs contain only RRI results. For a physical layer subtype 0/1 measurement the bar graphs contain also pilot results, see method RsCmwEvdoMeas.MultiEval.Trace.Cdp.Isignal.Pilot.Current.fetch etc. 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.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return isig_rri_min_cdp: float The number of results depends on the physical layer subtype (see method RsCmwEvdoMeas.Configure.MultiEval.plSubtype) : SF=15 for subtype 0/1 and SF=31 for subtypes 2 and 3. Range: - 70 dB to 0 dB, Unit: dB

read() → List[float]

```
# SCPI: READ:EVDO:MEASurement<instance>
↪:MEValuation:TRACe:CDP:ISIGnal:RRI:MINimum
value: List[float] = driver.multiEval.trace.cdp.isignal.rri.minimum.read()
```

Return values of the code domain power (CDP) I-Signal and Q-Signal bar graphs. The results of the current, average, maximum and minimum bar graphs can be retrieved. For a physical layer subtype 2 or 3 measurement, the bar graphs contain only RRI results. For a physical layer subtype 0/1 measurement the bar graphs contain also pilot results, see method RsCmwEvdoMeas.MultiEval.Trace.Cdp.Isignal.Pilot.Current.fetch etc. 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.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return isig_rri_min_cdp: float The number of results depends on the physical layer subtype (see method RsCmwEvdoMeas.Configure.MultiEval.plSubtype) : SF=15 for subtype 0/1 and SF=31 for subtypes 2 and 3. Range: - 70 dB to 0 dB, Unit: dB

7.4.2.4.1.6 State

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:TRACe:CDP:ISIGnal:RRI:STATe
```

class State

State commands group definition. 1 total commands, 0 Sub-groups, 1 group commands

fetch() → List[RsCmwEvdoMeas.enums.ResultStateB]

```
# SCPI: FETCh:EVDO:MEASurement<instance>:MEValuation:TRACe:CDP:ISIGnal:RRI:STATe
value: List[enums.ResultStateB] = driver.multiEval.trace.cdp.isignal.rri.state.
↪fetch()
```

Return the states of the code domain power (CDP) I-signal and Q-signal bar graphs. For a physical layer subtype 2 or 3 measurement, the bar graphs contain only RRI results. For a physical layer subtype 0/1 measurement the bar graphs contain also pilot results, see method RsCmwEvdoMeas.MultiEval.Trace.Cdp.Qsignal.Pilot.State.fetch.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return isig_rri_state: NAV | ACTive | INACtive The number of results depends on the physical layer subtype (see method RsCmwEvdoMeas.Configure.MultiEval.plSubtype) : SF=15 for subtype 0/1 and SF=31 for subtypes 2 and 3 (only NAV values) . NAV: No channel available ACTive: Active code channel INACtive: Inactive code channel

7.4.2.4.1.7 Limit

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:TRACe:CDP:ISIGnal:RRI:LIMit
```

class Limit

Limit commands group definition. 1 total commands, 0 Sub-groups, 1 group commands

fetch() → List[float]

```
# SCPI: FETCh:EVDO:MEASurement<instance>:MEValuation:TRACe:CDP:ISIGnal:RRI:LIMit
value: List[float] = driver.multiEval.trace.cdp.isignal.rri.limit.fetch()
```

Return limit check results for the code domain power (CDP) I-signal and Q-signal bar graphs. For a physical layer subtype 2 or 3 measurement, the bar graphs contain only RRI results. For a physical layer subtype 0/1 measurement the bar graphs contain also pilot results, see method RsCmwEvdoMeas.MultiEval.Trace.Cdp.Qsignal.Pilot.Limit.fetch.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return isig_rri_limit: float Return the exceeded limits as float values. The number of results depends on the physical layer subtype (see method RsCmwEvdoMeas.Configure.MultiEval.plSubtype) : SF=15 for subtype 0/1 and SF=31 for subtypes 2 and 3. Range: -70 dB to 0 dB , Unit: dB

7.4.2.4.1.8 Pilot

class Pilot

Pilot commands group definition. 14 total commands, 6 Sub-groups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.multiEval.trace.cdp.isignal.pilot.clone()
```

Subgroups

7.4.2.4.1.9 Current

SCPI Commands

```
READ:EVDO:MEASurement<Instance>:MEValuation:TRACe:CDP:ISIGnal:PILot:CURRent
FETCh:EVDO:MEASurement<Instance>:MEValuation:TRACe:CDP:ISIGnal:PILot:CURRent
CALCulate:EVDO:MEASurement<Instance>:MEValuation:TRACe:CDP:ISIGnal:PILot:CURRent
```

class Current

Current commands group definition. 3 total commands, 0 Sub-groups, 3 group commands

calculate() → List[RsCmwEvdoMeas.enums.ResultStatus2]

```
# SCPI: CALCulate:EVDO:MEASurement<instance>
↪:MEValuation:TRACe:CDP:ISIGnal:PILot:CURRent
value: List[enums.ResultStatus2] = driver.multiEval.trace.cdp.isignal.pilot.
↪current.calculate()
```

Return values of the code domain power (CDP) I-signal and Q-signal bar graphs. The results of the current, average, maximum and minimum bar graphs can be retrieved. Only the pilot part of a physical layer subtype 0/1 measurement is retrieved. For RRI part and subtype 2 or 3 measurements, see method RsCmwEvdoMeas.MultiEval.Trace.Cdp.Isignal.Rri.Current. fetch etc. 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.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return isig_pilo_curr_cdp: float The number of results depends on the physical layer subtype (see method RsCmwEvdoMeas.Configure.MultiEval.plSubtype) : SF=15 for subtype 0/1 and SF=31 for subtypes 2 and 3 (only NAV values) . Range: -70 dB to 0 dB, Unit: dB

fetch() → List[float]

```
# SCPI: FETCh:EVDO:MEASurement<instance>
↪:MEValuation:TRACe:CDP:ISIGnal:PILot:CURRent
value: List[float] = driver.multiEval.trace.cdp.isignal.pilot.current.fetch()
```

Return values of the code domain power (CDP) I-signal and Q-signal bar graphs. The results of the current, average, maximum and minimum bar graphs can be retrieved. Only the pilot part of a physical layer subtype 0/1 measurement is retrieved. For RRI part and subtype 2 or 3 measurements, see method RsCmwEvdoMeas.MultiEval.Trace.Cdp.Isignal.Rri.Current. fetch etc. 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.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return isig_pilo_curr_cdp: float The number of results depends on the physical layer subtype (see method RsCmwEvdoMeas.Configure.MultiEval.plSubtype) : SF=15 for subtype 0/1 and SF=31 for subtypes 2 and 3 (only NAV values) . Range: -70 dB to 0 dB, Unit: dB

read() → List[float]

```
# SCPI: READ:EVDO:MEASurement<instance>
↪:MEValuation:TRACe:CDP:ISIGnal:PILot:CURRent
value: List[float] = driver.multiEval.trace.cdp.isignal.pilot.current.read()
```

Return values of the code domain power (CDP) I-signal and Q-signal bar graphs. The results of the current, average, maximum and minimum bar graphs can be retrieved. Only the pilot part of a physical layer subtype 0/1 measurement is retrieved. For RRI part and subtype 2 or 3 measurements, see method RsCmwEvdoMeas.MultiEval.Trace.Cdp.Isignal.Rri.Current. fetch etc. 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.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return isig_pil_curr_cdp: float The number of results depends on the physical layer subtype (see method RsCmwEvdoMeas.Configure.MultiEval.plSubtype) : SF=15 for subtype 0/1 and SF=31 for subtypes 2 and 3 (only NAV values) . Range: -70 dB to 0 dB, Unit: dB

7.4.2.4.1.10 Average

SCPI Commands

```
READ:EVDO:MEASurement<Instance>:MEValuation:TRACe:CDP:ISIGnal:PILot:AVERage
FETCh:EVDO:MEASurement<Instance>:MEValuation:TRACe:CDP:ISIGnal:PILot:AVERage
CALCulate:EVDO:MEASurement<Instance>:MEValuation:TRACe:CDP:ISIGnal:PILot:AVERage
```

class Average

Average commands group definition. 3 total commands, 0 Sub-groups, 3 group commands

calculate() → List[RsCmwEvdoMeas.enums.ResultStatus2]

```
# SCPI: CALCulate:EVDO:MEASurement<instance>
↪:MEValuation:TRACe:CDP:ISIGnal:PILot:AVERage
value: List[enums.ResultStatus2] = driver.multiEval.trace.cdp.isignal.pilot.
↪average.calculate()
```

Return values of the code domain power (CDP) I-signal and Q-signal bar graphs. The results of the current, average, maximum and minimum bar graphs can be retrieved. Only the pilot part of a physical layer

subtype 0/1 measurement is retrieved. For RRI part and subtype 2 or 3 measurements, see method RsCmwEvdoMeas.MultiEval.Trace.Cdp.Isignal.Rri.Current. fetch etc. 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.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return isig_pilo_aver_cdp: float The number of results depends on the physical layer subtype (see method RsCmwEvdoMeas.Configure.MultiEval.plSubtype) : SF=15 for subtype 0/1 and SF=31 for subtypes 2 and 3 (only NAV values) . Range: -70 dB to 0 dB, Unit: dB

fetch() → List[float]

```
# SCPI: FETCh:EVDO:MEASurement<instance>
↪:MEValuation:TRACe:CDP:ISIGnal:PILot:AVERage
value: List[float] = driver.multiEval.trace.cdp.isignal.pilot.average.fetch()
```

Return values of the code domain power (CDP) I-signal and Q-signal bar graphs. The results of the current, average, maximum and minimum bar graphs can be retrieved. Only the pilot part of a physical layer subtype 0/1 measurement is retrieved. For RRI part and subtype 2 or 3 measurements, see method RsCmwEvdoMeas.MultiEval.Trace.Cdp.Isignal.Rri.Current. fetch etc. 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.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return isig_pilo_aver_cdp: float The number of results depends on the physical layer subtype (see method RsCmwEvdoMeas.Configure.MultiEval.plSubtype) : SF=15 for subtype 0/1 and SF=31 for subtypes 2 and 3 (only NAV values) . Range: -70 dB to 0 dB, Unit: dB

read() → List[float]

```
# SCPI: READ:EVDO:MEASurement<instance>
↪:MEValuation:TRACe:CDP:ISIGnal:PILot:AVERage
value: List[float] = driver.multiEval.trace.cdp.isignal.pilot.average.read()
```

Return values of the code domain power (CDP) I-signal and Q-signal bar graphs. The results of the current, average, maximum and minimum bar graphs can be retrieved. Only the pilot part of a physical layer subtype 0/1 measurement is retrieved. For RRI part and subtype 2 or 3 measurements, see method RsCmwEvdoMeas.MultiEval.Trace.Cdp.Isignal.Rri.Current. fetch etc. 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.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return isig_pil_aver_cdp: float The number of results depends on the physical layer subtype (see method RsCmwEvdoMeas.Configure.MultiEval.plSubtype) : SF=15 for subtype 0/1 and SF=31 for subtypes 2 and 3 (only NAV values) . Range: -70 dB to 0 dB, Unit: dB

7.4.2.4.1.11 Maximum

SCPI Commands

```
READ:EVDO:MEASurement<Instance>:MEValuation:TRACe:CDP:ISIGnal:PILot:MAXimum
FETCh:EVDO:MEASurement<Instance>:MEValuation:TRACe:CDP:ISIGnal:PILot:MAXimum
CALCulate:EVDO:MEASurement<Instance>:MEValuation:TRACe:CDP:ISIGnal:PILot:MAXimum
```

class Maximum

Maximum commands group definition. 3 total commands, 0 Sub-groups, 3 group commands

calculate() → List[RsCmwEvdoMeas.enums.ResultStatus2]

```
# SCPI: CALCulate:EVDO:MEASurement<instance>
↪:MEValuation:TRACe:CDP:ISIGnal:PILot:MAXimum
value: List[enums.ResultStatus2] = driver.multiEval.trace.cdp.isignal.pilot.
↪maximum.calculate()
```

Return values of the code domain power (CDP) I-signal and Q-signal bar graphs. The results of the current, average, maximum and minimum bar graphs can be retrieved. Only the pilot part of a physical layer subtype 0/1 measurement is retrieved. For RRI part and subtype 2 or 3 measurements, see method RsCmwEvdoMeas.MultiEval.Trace.Cdp.Isignal.Rri.Current. fetch etc. 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.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return isig_pilo_max_cdp: float The number of results depends on the physical layer subtype (see method RsCmwEvdoMeas.Configure.MultiEval.plSubtype) : SF=15 for subtype 0/1 and SF=31 for subtypes 2 and 3 (only NAV values) . Range: -70 dB to 0 dB, Unit: dB

fetch() → List[float]

```
# SCPI: FETCh:EVDO:MEASurement<instance>
↪:MEValuation:TRACe:CDP:ISIGnal:PILot:MAXimum
value: List[float] = driver.multiEval.trace.cdp.isignal.pilot.maximum.fetch()
```

Return values of the code domain power (CDP) I-signal and Q-signal bar graphs. The results of the current, average, maximum and minimum bar graphs can be retrieved. Only the pilot part of a physical layer subtype 0/1 measurement is retrieved. For RRI part and subtype 2 or 3 measurements, see method RsCmwEvdoMeas.MultiEval.Trace.Cdp.Isignal.Rri.Current. fetch etc. 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.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return isig_pilo_max_cdp: float The number of results depends on the physical layer subtype (see method RsCmwEvdoMeas.Configure.MultiEval.plSubtype) : SF=15 for subtype 0/1 and SF=31 for subtypes 2 and 3 (only NAV values) . Range: -70 dB to 0 dB, Unit: dB

read() → List[float]

```
# SCPI: READ:EVDO:MEASurement<instance>
↪:MEValuation:TRACe:CDP:ISIGnal:PILot:MAXimum
value: List[float] = driver.multiEval.trace.cdp.isignal.pilot.maximum.read()
```

Return values of the code domain power (CDP) I-signal and Q-signal bar graphs. The results of the current, average, maximum and minimum bar graphs can be retrieved. Only the pilot part of a physical layer subtype 0/1 measurement is retrieved. For RRI part and subtype 2 or 3 measurements, see method RsCmwEvdoMeas.MultiEval.Trace.Cdp.Isignal.Rri.Current. fetch etc. 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.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return isig_pil_max_cdp: float The number of results depends on the physical layer subtype (see method RsCmwEvdoMeas.Configure.MultiEval.plSubtype) : SF=15 for subtype 0/1 and SF=31 for subtypes 2 and 3 (only NAV values) . Range: -70 dB to 0 dB, Unit: dB

7.4.2.4.1.12 Minimum

SCPI Commands

```
READ:EVDO:MEASurement<Instance>:MEValuation:TRACe:CDP:ISIGnal:PILot:MINimum
FETCh:EVDO:MEASurement<Instance>:MEValuation:TRACe:CDP:ISIGnal:PILot:MINimum
CALCulate:EVDO:MEASurement<Instance>:MEValuation:TRACe:CDP:ISIGnal:PILot:MINimum
```

class Minimum

Minimum commands group definition. 3 total commands, 0 Sub-groups, 3 group commands

calculate() → List[RsCmwEvdoMeas.enums.ResultStatus2]

```
# SCPI: CALCulate:EVDO:MEASurement<instance>
↪:MEValuation:TRACe:CDP:ISIGnal:PILot:MINimum
value: List[enums.ResultStatus2] = driver.multiEval.trace.cdp.isignal.pilot.
↪minimum.calculate()
```

Return values of the code domain power (CDP) I-signal and Q-signal bar graphs. The results of the current, average, maximum and minimum bar graphs can be retrieved. Only the pilot part of a physical layer subtype 0/1 measurement is retrieved. For RRI part and subtype 2 or 3 measurements, see method RsCmwEvdoMeas.MultiEval.Trace.Cdp.Isignal.Rri.Current. fetch etc. 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.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return isig_pilo_min_cdp: float The number of results depends on the physical layer subtype (see method RsCmwEvdoMeas.Configure.MultiEval.plSubtype) : SF=15 for subtype 0/1 and SF=31 for subtypes 2 and 3 (only NAV values) . Range: -70 dB to 0 dB, Unit: dB

fetch() → List[float]

```
# SCPI: FETCh:EVDO:MEASurement<instance>
↪:MEValuation:TRACe:CDP:ISIGnal:PILot:MINimum
value: List[float] = driver.multiEval.trace.cdp.isignal.pilot.minimum.fetch()
```

Return values of the code domain power (CDP) I-signal and Q-signal bar graphs. The results of the current, average, maximum and minimum bar graphs can be retrieved. Only the pilot part of a physical layer subtype 0/1 measurement is retrieved. For RRI part and subtype 2 or 3 measurements, see method RsCmwEvdoMeas.MultiEval.Trace.Cdp.Isignal.Rri.Current. fetch etc. 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.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return isig_pilo_min_cdp: float The number of results depends on the physical layer subtype (see method RsCmwEvdoMeas.Configure.MultiEval.plSubtype) : SF=15 for subtype 0/1 and SF=31 for subtypes 2 and 3 (only NAV values) . Range: -70 dB to 0 dB, Unit: dB

read() → List[float]

```
# SCPI: READ:EVDO:MEASurement<instance>
↪:MEValuation:TRACe:CDP:ISIGnal:PILot:MINimum
value: List[float] = driver.multiEval.trace.cdp.isignal.pilot.minimum.read()
```

Return values of the code domain power (CDP) I-signal and Q-signal bar graphs. The results of the current, average, maximum and minimum bar graphs can be retrieved. Only the pilot part of a physical layer subtype 0/1 measurement is retrieved. For RRI part and subtype 2 or 3 measurements, see method RsCmwEvdoMeas.MultiEval.Trace.Cdp.Isignal.Rri.Current. fetch etc. 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.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return isig_pil_min_cdp: float The number of results depends on the physical layer subtype (see method RsCmwEvdoMeas.Configure.MultiEval.plSubtype) : SF=15 for subtype 0/1 and SF=31 for subtypes 2 and 3 (only NAV values) . Range: -70 dB to 0 dB, Unit: dB

7.4.2.4.1.13 State

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:TRACe:CDP:ISIGnal:PILot:STATe
```

class State

State commands group definition. 1 total commands, 0 Sub-groups, 1 group commands

fetch() → List[RsCmwEvdoMeas.enums.ResultStateB]

```
# SCPI: FETCh:EVDO:MEASurement<instance>
↪:MEValuation:TRACe:CDP:ISIGnal:PILot:STATe
value: List[enums.ResultStateB] = driver.multiEval.trace.cdp.isignal.pilot.
↪state.fetch()
```

Return the states of the code domain power (CDP) I-signal and Q-signal bar graphs. Only the pilot part of a physical layer subtype 0/1 measurement is retrieved. For RRI part and subtype 2 or 3 measurements, see method RsCmwEvdoMeas. MultiEval.Trace.Cdp.Qsignal.Rri.State.fetch.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return isig_pilot_state: NAV | ACTive | INACtive The number of results depends on the physical layer subtype (see method RsCmwEvdoMeas.Configure.MultiEval.plSubtype) : SF=15 for subtype 0/1 and SF=31 for subtypes 2 and 3 (only NAV values) . NAV: No channel available ACTive: Active code channel INACtive: Inactive code channel

7.4.2.4.1.14 Limit

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:TRACe:CDP:ISIGnal:PILot:LIMit
```

class Limit

Limit commands group definition. 1 total commands, 0 Sub-groups, 1 group commands

fetch() → List[float]

```
# SCPI: FETCh:EVDO:MEASurement<instance>
↪:MEValuation:TRACe:CDP:ISIGnal:PILot:LIMit
value: List[float] = driver.multiEval.trace.cdp.isignal.pilot.limit.fetch()
```

Return limit check results for the code domain power (CDP) I-signal and Q-signal bar graphs. Only the pilot part of a physical layer subtype 0/1 measurement is retrieved. For RRI part and subtype 2 or 3 measurements, see method RsCmwEvdoMeas.MultiEval.Trace.Cdp.Qsignal.Rri.State.fetch.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return isig_pilot_limit: float Return the exceeded limits as float values. The number of results depends on the physical layer subtype (see method RsCmwEvdoMeas.Configure.MultiEval.plSubtype) : SF=15 for subtype 0/1 and SF=31 for subtypes 2 and 3. Range: -70 dB to 0 dB , Unit: dB

7.4.2.4.2 Qsignal

class Qsignal

Qsignal commands group definition. 28 total commands, 2 Sub-groups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.multiEval.trace.cdp.qsignal.clone()
```

Subgroups

7.4.2.4.2.1 Rri

class Rri
Rri commands group definition. 14 total commands, 6 Sub-groups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.multiEval.trace.cdp.qsignal.rri.clone()
```

Subgroups

7.4.2.4.2.2 Current

SCPI Commands

```
READ:EVDO:MEASurement<Instance>:MEValuation:TRACe:CDP:QSIGnal:RRI:CURRent
FETCh:EVDO:MEASurement<Instance>:MEValuation:TRACe:CDP:QSIGnal:RRI:CURRent
CALCulate:EVDO:MEASurement<Instance>:MEValuation:TRACe:CDP:QSIGnal:RRI:CURRent
```

class Current
Current commands group definition. 3 total commands, 0 Sub-groups, 3 group commands

calculate() → List[RsCmwEvdoMeas.enums.ResultStatus2]

```
# SCPI: CALCulate:EVDO:MEASurement<instance>
↪:MEValuation:TRACe:CDP:QSIGnal:RRI:CURRent
value: List[enums.ResultStatus2] = driver.multiEval.trace.cdp.qsignal.rri.
↪current.calculate()
```

Return values of the code domain power (CDP) I-Signal and Q-Signal bar graphs. The results of the current, average, maximum and minimum bar graphs can be retrieved. For a physical layer subtype 2 or 3 measurement, the bar graphs contain only RRI results. For a physical layer subtype 0/1 measurement the bar graphs contain also pilot results, see method RsCmwEvdoMeas.MultiEval.Trace.Cdp.Isignal.Pilot.Current.fetch etc. 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.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return qsig_rri_curr_cdp: float The number of results depends on the physical layer subtype (see method RsCmwEvdoMeas.Configure.MultiEval.plSubtype) : SF=15 for subtype 0/1 and SF=31 for subtypes 2 and 3. Range: - 70 dB to 0 dB, Unit: dB

fetch() → List[float]

```
# SCPI: FETCh:EVDO:MEASurement<instance>
↪:MEValuation:TRACe:CDP:QSIGnal:RRI:CURRent
value: List[float] = driver.multiEval.trace.cdp.qsignal.rri.current.fetch()
```

Return values of the code domain power (CDP) I-Signal and Q-Signal bar graphs. The results of the current, average, maximum and minimum bar graphs can be retrieved. For a physical layer subtype 2 or 3 measurement, the bar graphs contain only RRI results. For a physical layer subtype 0/1 measurement the bar graphs contain also pilot results, see method RsCmwEvdoMeas.MultiEval.Trace.Cdp.Isignal.Pilot.Current.fetch etc. 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.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return qsig_rri_curr_cdp: float The number of results depends on the physical layer subtype (see method RsCmwEvdoMeas.Configure.MultiEval.plSubtype) : SF=15 for subtype 0/1 and SF=31 for subtypes 2 and 3. Range: - 70 dB to 0 dB, Unit: dB

read() → List[float]

```
# SCPI: READ:EVDO:MEASurement<instance>
↪:MEValuation:TRACe:CDP:QSIGnal:RRI:CURRent
value: List[float] = driver.multiEval.trace.cdp.qsignal.rri.current.read()
```

Return values of the code domain power (CDP) I-Signal and Q-Signal bar graphs. The results of the current, average, maximum and minimum bar graphs can be retrieved. For a physical layer subtype 2 or 3 measurement, the bar graphs contain only RRI results. For a physical layer subtype 0/1 measurement the bar graphs contain also pilot results, see method RsCmwEvdoMeas.MultiEval.Trace.Cdp.Isignal.Pilot.Current.fetch etc. 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.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return qsig_rri_curr_cdp: float The number of results depends on the physical layer subtype (see method RsCmwEvdoMeas.Configure.MultiEval.plSubtype) : SF=15 for subtype 0/1 and SF=31 for subtypes 2 and 3. Range: - 70 dB to 0 dB, Unit: dB

7.4.2.4.2.3 Average

SCPI Commands

```
READ:EVDO:MEASurement<Instance>:MEValuation:TRACe:CDP:QSIGnal:RRI:AVERage
FETCh:EVDO:MEASurement<Instance>:MEValuation:TRACe:CDP:QSIGnal:RRI:AVERage
CALCulate:EVDO:MEASurement<Instance>:MEValuation:TRACe:CDP:QSIGnal:RRI:AVERage
```

class Average

Average commands group definition. 3 total commands, 0 Sub-groups, 3 group commands

calculate() → List[RsCmwEvdoMeas.enums.ResultStatus2]

```
# SCPI: CALCulate:EVDO:MEASurement<instance>
↪:MEValuation:TRACe:CDP:QSIGnal:RRI:AVERage
value: List[enums.ResultStatus2] = driver.multiEval.trace.cdp.qsignal.rri.
↪average.calculate()
```

Return values of the code domain power (CDP) I-Signal and Q-Signal bar graphs. The results of the current, average, maximum and minimum bar graphs can be retrieved. For a physical layer subtype 2 or 3 measurement, the bar graphs contain only RRI results. For a physical layer subtype 0/1 measurement the bar graphs contain also pilot results, see method RsCmwEvdoMeas.MultiEval.Trace.Cdp.Isignal.Pilot.Current.fetch etc. 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.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return qsig_rri_aver_cdp: float The number of results depends on the physical layer subtype (see method RsCmwEvdoMeas.Configure.MultiEval.plSubtype) : SF=15 for subtype 0/1 and SF=31 for subtypes 2 and 3. Range: - 70 dB to 0 dB, Unit: dB

fetch() → List[float]

```
# SCPI: FETCh:EVDO:MEASurement<instance>
↪:MEValuation:TRACe:CDP:QSIGnal:RRI:AVERage
value: List[float] = driver.multiEval.trace.cdp.qsignal.rri.average.fetch()
```

Return values of the code domain power (CDP) I-Signal and Q-Signal bar graphs. The results of the current, average, maximum and minimum bar graphs can be retrieved. For a physical layer subtype 2 or 3 measurement, the bar graphs contain only RRI results. For a physical layer subtype 0/1 measurement the bar graphs contain also pilot results, see method RsCmwEvdoMeas.MultiEval.Trace.Cdp.Isignal.Pilot.Current.fetch etc. 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.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return qsig_rri_aver_cdp: float The number of results depends on the physical layer subtype (see method RsCmwEvdoMeas.Configure.MultiEval.plSubtype) : SF=15 for subtype 0/1 and SF=31 for subtypes 2 and 3. Range: - 70 dB to 0 dB, Unit: dB

read() → List[float]

```
# SCPI: READ:EVDO:MEASurement<instance>
↪:MEValuation:TRACe:CDP:QSIGnal:RRI:AVERage
value: List[float] = driver.multiEval.trace.cdp.qsignal.rri.average.read()
```

Return values of the code domain power (CDP) I-Signal and Q-Signal bar graphs. The results of the current, average, maximum and minimum bar graphs can be retrieved. For a physical layer subtype 2 or 3 measurement, the bar graphs contain only RRI results. For a physical layer subtype 0/1 measurement the bar graphs contain also pilot results, see method RsCmwEvdoMeas.MultiEval.Trace.Cdp.Isignal.Pilot.Current.fetch etc. 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.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return qsig_rri_aver_cdp: float The number of results depends on the physical layer subtype (see method RsCmwEvdoMeas.Configure.MultiEval.plSubtype) : SF=15 for subtype 0/1 and SF=31 for subtypes 2 and 3. Range: - 70 dB to 0 dB, Unit: dB

7.4.2.4.2.4 Maximum

SCPI Commands

```
READ:EVDO:MEASurement<Instance>:MEValuation:TRACe:CDP:QSIGnal:RRI:MAXimum
FETCh:EVDO:MEASurement<Instance>:MEValuation:TRACe:CDP:QSIGnal:RRI:MAXimum
CALCulate:EVDO:MEASurement<Instance>:MEValuation:TRACe:CDP:QSIGnal:RRI:MAXimum
```

class Maximum

Maximum commands group definition. 3 total commands, 0 Sub-groups, 3 group commands

calculate() → List[RsCmwEvdoMeas.enums.ResultStatus2]

```
# SCPI: CALCulate:EVDO:MEASurement<instance>
↪:MEValuation:TRACe:CDP:QSIGnal:RRI:MAXimum
value: List[enums.ResultStatus2] = driver.multiEval.trace.cdp.qsignal.rri.
↪maximum.calculate()
```

Return values of the code domain power (CDP) I-Signal and Q-Signal bar graphs. The results of the current, average, maximum and minimum bar graphs can be retrieved. For a physical layer subtype 2 or 3 measurement, the bar graphs contain only RRI results. For a physical layer subtype 0/1 measurement the bar graphs contain also pilot results, see method RsCmwEvdoMeas.MultiEval.Trace.Cdp.Isignal.Pilot.Current.fetch etc. 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.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return qsig_rri_max_cdp: float The number of results depends on the physical layer subtype (see method RsCmwEvdoMeas.Configure.MultiEval.plSubtype) : SF=15 for subtype 0/1 and SF=31 for subtypes 2 and 3. Range: - 70 dB to 0 dB, Unit: dB

fetch() → List[float]

```
# SCPI: FETCh:EVDO:MEASurement<instance>
↪:MEValuation:TRACe:CDP:QSIGnal:RRI:MAXimum
value: List[float] = driver.multiEval.trace.cdp.qsignal.rri.maximum.fetch()
```

Return values of the code domain power (CDP) I-Signal and Q-Signal bar graphs. The results of the current, average, maximum and minimum bar graphs can be retrieved. For a physical layer subtype 2 or 3 measurement, the bar graphs contain only RRI results. For a physical layer subtype 0/1 measurement the bar graphs contain also pilot results, see method RsCmwEvdoMeas.MultiEval.Trace.Cdp.Isignal.Pilot.Current.fetch etc. 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.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return qsig_rri_max_cdp: float The number of results depends on the physical layer subtype (see method RsCmwEvdoMeas.Configure.MultiEval.plSubtype) : SF=15 for subtype 0/1 and SF=31 for subtypes 2 and 3. Range: - 70 dB to 0 dB, Unit: dB

read() → List[float]

```
# SCPI: READ:EVDO:MEASurement<instance>
↪:MEValuation:TRACe:CDP:QSIGnal:RRI:MAXimum
value: List[float] = driver.multiEval.trace.cdp.qsignal.rri.maximum.read()
```

Return values of the code domain power (CDP) I-Signal and Q-Signal bar graphs. The results of the current, average, maximum and minimum bar graphs can be retrieved. For a physical layer subtype 2 or 3 measurement, the bar graphs contain only RRI results. For a physical layer subtype 0/1 measurement the bar graphs contain also pilot results, see method RsCmwEvdoMeas.MultiEval.Trace.Cdp.Isignal.Pilot.Current.fetch etc. 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.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return qsig_rri_max_cdp: float The number of results depends on the physical layer subtype (see method RsCmwEvdoMeas.Configure.MultiEval.plSubtype) : SF=15 for subtype 0/1 and SF=31 for subtypes 2 and 3. Range: - 70 dB to 0 dB, Unit: dB

7.4.2.4.2.5 Minimum

SCPI Commands

```
READ:EVDO:MEASurement<Instance>:MEValuation:TRACe:CDP:QSIGnal:RRI:MINimum
FETCh:EVDO:MEASurement<Instance>:MEValuation:TRACe:CDP:QSIGnal:RRI:MINimum
CALCulate:EVDO:MEASurement<Instance>:MEValuation:TRACe:CDP:QSIGnal:RRI:MINimum
```

class Minimum

Minimum commands group definition. 3 total commands, 0 Sub-groups, 3 group commands

calculate() → List[RsCmwEvdoMeas.enums.ResultStatus2]

```
# SCPI: CALCulate:EVDO:MEASurement<instance>
↪:MEValuation:TRACe:CDP:QSIGnal:RRI:MINimum
value: List[enums.ResultStatus2] = driver.multiEval.trace.cdp.qsignal.rri.
↪minimum.calculate()
```

Return values of the code domain power (CDP) I-Signal and Q-Signal bar graphs. The results of the current, average, maximum and minimum bar graphs can be retrieved. For a physical layer subtype 2 or 3 measurement, the bar graphs contain only RRI results. For a physical layer subtype 0/1 measurement the bar graphs contain also pilot results, see method RsCmwEvdoMeas.MultiEval.Trace.Cdp.Isignal.Pilot.Current.fetch etc. 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.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return qsig_rri_min_cdp: float The number of results depends on the physical layer subtype (see method RsCmwEvdoMeas.Configure.MultiEval.plSubtype) : SF=15 for subtype 0/1 and SF=31 for subtypes 2 and 3. Range: - 70 dB to 0 dB, Unit: dB

fetch() → List[float]

```
# SCPI: FETCh:EVDO:MEASurement<instance>
↪:MEValuation:TRACe:CDP:QSIGnal:RRI:MINimum
value: List[float] = driver.multiEval.trace.cdp.qsignal.rri.minimum.fetch()
```

Return values of the code domain power (CDP) I-Signal and Q-Signal bar graphs. The results of the current, average, maximum and minimum bar graphs can be retrieved. For a physical layer subtype 2 or 3 measurement, the bar graphs contain only RRI results. For a physical layer subtype 0/1 measurement the bar graphs contain also pilot results, see method RsCmwEvdoMeas.MultiEval.Trace.Cdp.Isignal.Pilot.Current.fetch etc. 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.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return qsig_rri_min_cdp: float The number of results depends on the physical layer subtype (see method RsCmwEvdoMeas.Configure.MultiEval.plSubtype) : SF=15 for subtype 0/1 and SF=31 for subtypes 2 and 3. Range: - 70 dB to 0 dB, Unit: dB

read() → List[float]

```
# SCPI: READ:EVDO:MEASurement<instance>
↪:MEValuation:TRACe:CDP:QSIGnal:RRI:MINimum
value: List[float] = driver.multiEval.trace.cdp.qsignal.rri.minimum.read()
```

Return values of the code domain power (CDP) I-Signal and Q-Signal bar graphs. The results of the current, average, maximum and minimum bar graphs can be retrieved. For a physical layer subtype 2 or 3 measurement, the bar graphs contain only RRI results. For a physical layer subtype 0/1 measurement the bar graphs contain also pilot results, see method RsCmwEvdoMeas.MultiEval.Trace.Cdp.Isignal.Pilot.Current.fetch etc. 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.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return qsig_rri_min_cdp: float The number of results depends on the physical layer subtype (see method RsCmwEvdoMeas.Configure.MultiEval.plSubtype) : SF=15 for subtype 0/1 and SF=31 for subtypes 2 and 3. Range: - 70 dB to 0 dB, Unit: dB

7.4.2.4.2.6 State

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:TRACe:CDP:QSIGnal:RRI:STATe
```

class State

State commands group definition. 1 total commands, 0 Sub-groups, 1 group commands

fetch() → List[RsCmwEvdoMeas.enums.ResultStateB]

```
# SCPI: FETCh:EVDO:MEASurement<instance>:MEValuation:TRACe:CDP:QSIGnal:RRI:STATe
value: List[enums.ResultStateB] = driver.multiEval.trace.cdp.qsignal.rri.state.
↪fetch()
```

Return the states of the code domain power (CDP) I-signal and Q-signal bar graphs. For a physical layer subtype 2 or 3 measurement, the bar graphs contain only RRI results. For a physical layer subtype 0/1 measurement the bar graphs contain also pilot results, see method RsCmwEvdoMeas.MultiEval.Trace.Cdp.Qsignal.Pilot.State.fetch.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return qsig_rri_state: NAV | ACTive | INACtive The number of results depends on the physical layer subtype (see method RsCmwEvdoMeas.Configure.MultiEval.plSubtype) : SF=15 for subtype 0/1 and SF=31 for subtypes 2 and 3 (only NAV values) . NAV: No channel available ACTive: Active code channel INACtive: Inactive code channel

7.4.2.4.2.7 Limit

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:TRACe:CDP:QSIGnal:RRI:LIMit
```

class Limit

Limit commands group definition. 1 total commands, 0 Sub-groups, 1 group commands

fetch() → List[float]

```
# SCPI: FETCh:EVDO:MEASurement<instance>:MEValuation:TRACe:CDP:QSIGnal:RRI:LIMit
value: List[float] = driver.multiEval.trace.cdp.qsignal.rri.limit.fetch()
```

Return limit check results for the code domain power (CDP) I-signal and Q-signal bar graphs. For a physical layer subtype 2 or 3 measurement, the bar graphs contain only RRI results. For a physical layer subtype 0/1 measurement the bar graphs contain also pilot results, see method RsCmwEvdoMeas.MultiEval.Trace.Cdp.Qsignal.Pilot.Limit.fetch.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return qsig_rri_limit: float Return the exceeded limits as float values. The number of results depends on the physical layer subtype (see method RsCmwEvdoMeas.Configure.MultiEval.plSubtype) : SF=15 for subtype 0/1 and SF=31 for subtypes 2 and 3. Range: -70 dB to 0 dB , Unit: dB

7.4.2.4.2.8 Pilot

class Pilot

Pilot commands group definition. 14 total commands, 6 Sub-groups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.multiEval.trace.cdp.qsignal.pilot.clone()
```

Subgroups

7.4.2.4.2.9 Current

SCPI Commands

```
READ:EVDO:MEASurement<Instance>:MEValuation:TRACe:CDP:QSIGnal:PILot:CURRent
FETCh:EVDO:MEASurement<Instance>:MEValuation:TRACe:CDP:QSIGnal:PILot:CURRent
CALCulate:EVDO:MEASurement<Instance>:MEValuation:TRACe:CDP:QSIGnal:PILot:CURRent
```

class Current

Current commands group definition. 3 total commands, 0 Sub-groups, 3 group commands

calculate() → List[RsCmwEvdoMeas.enums.ResultStatus2]

```
# SCPI: CALCulate:EVDO:MEASurement<instance>
↪:MEValuation:TRACe:CDP:QSIGnal:PILot:CURRent
value: List[enums.ResultStatus2] = driver.multiEval.trace.cdp.qsignal.pilot.
↪current.calculate()
```

Return values of the code domain power (CDP) I-signal and Q-signal bar graphs. The results of the current, average, maximum and minimum bar graphs can be retrieved. Only the pilot part of a physical layer subtype 0/1 measurement is retrieved. For RRI part and subtype 2 or 3 measurements, see method RsCmwEvdoMeas.MultiEval.Trace.Cdp.Isignal.Rri.Current. fetch etc. 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.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return qsig_pil_curr_cdp: float The number of results depends on the physical layer subtype (see method RsCmwEvdoMeas.Configure.MultiEval.plSubtype) : SF=15 for subtype 0/1 and SF=31 for subtypes 2 and 3 (only NAV values) . Range: -70 dB to 0 dB, Unit: dB

fetch() → List[float]

```
# SCPI: FETCh:EVDO:MEASurement<instance>
↪:MEValuation:TRACe:CDP:QSIGnal:PILot:CURRent
value: List[float] = driver.multiEval.trace.cdp.qsignal.pilot.current.fetch()
```

Return values of the code domain power (CDP) I-signal and Q-signal bar graphs. The results of the current, average, maximum and minimum bar graphs can be retrieved. Only the pilot part of a physical layer subtype 0/1 measurement is retrieved. For RRI part and subtype 2 or 3 measurements, see method RsCmwEvdoMeas.MultiEval.Trace.Cdp.Isignal.Rri.Current. fetch etc. 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.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return qsig_pil_curr_cdp: float The number of results depends on the physical layer subtype (see method RsCmwEvdoMeas.Configure.MultiEval.plSubtype) : SF=15 for subtype 0/1 and SF=31 for subtypes 2 and 3 (only NAV values) . Range: -70 dB to 0 dB, Unit: dB

read() → List[float]

```
# SCPI: READ:EVDO:MEASurement<instance>
↪:MEValuation:TRACe:CDP:QSIGnal:PILot:CURRent
value: List[float] = driver.multiEval.trace.cdp.qsignal.pilot.current.read()
```

Return values of the code domain power (CDP) I-signal and Q-signal bar graphs. The results of the current, average, maximum and minimum bar graphs can be retrieved. Only the pilot part of a physical layer subtype 0/1 measurement is retrieved. For RRI part and subtype 2 or 3 measurements, see method RsCmwEvdoMeas.MultiEval.Trace.Cdp.Isignal.Rri.Current. fetch etc. 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.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return qsig_pil_curr_cdp: float The number of results depends on the physical layer subtype (see method RsCmwEvdoMeas.Configure.MultiEval.plSubtype) : SF=15 for subtype 0/1 and SF=31 for subtypes 2 and 3 (only NAV values) . Range: -70 dB to 0 dB, Unit: dB

7.4.2.4.2.10 Average

SCPI Commands

```
READ:EVDO:MEASurement<Instance>:MEValuation:TRACe:CDP:QSIGnal:PILot:AVERage
FETCh:EVDO:MEASurement<Instance>:MEValuation:TRACe:CDP:QSIGnal:PILot:AVERage
CALCulate:EVDO:MEASurement<Instance>:MEValuation:TRACe:CDP:QSIGnal:PILot:AVERage
```

class Average

Average commands group definition. 3 total commands, 0 Sub-groups, 3 group commands

calculate() → List[RsCmwEvdoMeas.enums.ResultStatus2]

```
# SCPI: CALCulate:EVDO:MEASurement<instance>
↪:MEValuation:TRACe:CDP:QSIGnal:PILot:AVERage
value: List[enums.ResultStatus2] = driver.multiEval.trace.cdp.qsignal.pilot.
↪average.calculate()
```

Return values of the code domain power (CDP) I-signal and Q-signal bar graphs. The results of the current, average, maximum and minimum bar graphs can be retrieved. Only the pilot part of a physical layer subtype 0/1 measurement is retrieved. For RRI part and subtype 2 or 3 measurements, see method RsCmwEvdoMeas.MultiEval.Trace.Cdp.Isignal.Rri.Current. fetch etc. 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.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return qsig_pil_aver_cdp: float The number of results depends on the physical layer subtype (see method RsCmwEvdoMeas.Configure.MultiEval.plSubtype) : SF=15 for subtype 0/1 and SF=31 for subtypes 2 and 3 (only NAV values) . Range: -70 dB to 0 dB, Unit: dB

fetch() → List[float]

```
# SCPI: FETCh:EVDO:MEASurement<instance>
↪:MEValuation:TRACe:CDP:QSIGnal:PILot:AVERage
value: List[float] = driver.multiEval.trace.cdp.qsignal.pilot.average.fetch()
```

Return values of the code domain power (CDP) I-signal and Q-signal bar graphs. The results of the current, average, maximum and minimum bar graphs can be retrieved. Only the pilot part of a physical layer subtype 0/1 measurement is retrieved. For RRI part and subtype 2 or 3 measurements, see method RsCmwEvdoMeas.MultiEval.Trace.Cdp.Isignal.Rri.Current. fetch etc. 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.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return qsig_pil_aver_cdp: float The number of results depends on the physical layer subtype (see method RsCmwEvdoMeas.Configure.MultiEval.plSubtype) : SF=15 for subtype 0/1 and SF=31 for subtypes 2 and 3 (only NAV values) . Range: -70 dB to 0 dB, Unit: dB

read() → List[float]

```
# SCPI: READ:EVDO:MEASurement<instance>
↪:MEValuation:TRACe:CDP:QSIGnal:PILot:AVERage
value: List[float] = driver.multiEval.trace.cdp.qsignal.pilot.average.read()
```

Return values of the code domain power (CDP) I-signal and Q-signal bar graphs. The results of the current, average, maximum and minimum bar graphs can be retrieved. Only the pilot part of a physical layer subtype 0/1 measurement is retrieved. For RRI part and subtype 2 or 3 measurements, see method RsCmwEvdoMeas.MultiEval.Trace.Cdp.Isignal.Rri.Current. fetch etc. 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.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return qsig_pil_aver_cdp: float The number of results depends on the physical layer subtype (see method RsCmwEvdoMeas.Configure.MultiEval.plSubtype) : SF=15 for subtype 0/1 and SF=31 for subtypes 2 and 3 (only NAV values) . Range: -70 dB to 0 dB, Unit: dB

7.4.2.4.2.11 Maximum

SCPI Commands

```
READ:EVDO:MEASurement<Instance>:MEValuation:TRACe:CDP:QSIGnal:PILot:MAXimum
FETCh:EVDO:MEASurement<Instance>:MEValuation:TRACe:CDP:QSIGnal:PILot:MAXimum
CALCulate:EVDO:MEASurement<Instance>:MEValuation:TRACe:CDP:QSIGnal:PILot:MAXimum
```

class Maximum

Maximum commands group definition. 3 total commands, 0 Sub-groups, 3 group commands

calculate() → List[RsCmwEvdoMeas.enums.ResultStatus2]

```
# SCPI: CALCulate:EVDO:MEASurement<instance>
↪:MEValuation:TRACe:CDP:QSIGnal:PILot:MAXimum
value: List[enums.ResultStatus2] = driver.multiEval.trace.cdp.qsignal.pilot.
↪maximum.calculate()
```

Return values of the code domain power (CDP) I-signal and Q-signal bar graphs. The results of the current, average, maximum and minimum bar graphs can be retrieved. Only the pilot part of a physical layer subtype 0/1 measurement is retrieved. For RRI part and subtype 2 or 3 measurements, see method RsCmwEvdoMeas.MultiEval.Trace.Cdp.Isignal.Rri.Current. fetch etc. 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.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return qsig_pil_max_cdp: float The number of results depends on the physical layer subtype (see method RsCmwEvdoMeas.Configure.MultiEval.plSubtype) : SF=15 for subtype 0/1 and SF=31 for subtypes 2 and 3 (only NAV values) . Range: -70 dB to 0 dB, Unit: dB

fetch() → List[float]

```
# SCPI: FETCh:EVDO:MEASurement<instance>
↪:MEValuation:TRACe:CDP:QSIGnal:PILot:MAXimum
value: List[float] = driver.multiEval.trace.cdp.qsignal.pilot.maximum.fetch()
```

Return values of the code domain power (CDP) I-signal and Q-signal bar graphs. The results of the current, average, maximum and minimum bar graphs can be retrieved. Only the pilot part of a physical layer subtype 0/1 measurement is retrieved. For RRI part and subtype 2 or 3 measurements, see method RsCmwEvdoMeas.MultiEval.Trace.Cdp.Isignal.Rri.Current. fetch etc. 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.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return qsig_pil_max_cdp: float The number of results depends on the physical layer subtype (see method RsCmwEvdoMeas.Configure.MultiEval.plSubtype) : SF=15 for subtype 0/1 and SF=31 for subtypes 2 and 3 (only NAV values) . Range: -70 dB to 0 dB, Unit: dB

read() → List[float]

```
# SCPI: READ:EVDO:MEASurement<instance>
↪:MEValuation:TRACe:CDP:QSIGnal:PILot:MAXimum
value: List[float] = driver.multiEval.trace.cdp.qsignal.pilot.maximum.read()
```

Return values of the code domain power (CDP) I-signal and Q-signal bar graphs. The results of the current, average, maximum and minimum bar graphs can be retrieved. Only the pilot part of a physical layer subtype 0/1 measurement is retrieved. For RRI part and subtype 2 or 3 measurements, see method RsCmwEvdoMeas.MultiEval.Trace.Cdp.Isignal.Rri.Current. fetch etc. 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.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return qsig_pil_max_cdp: float The number of results depends on the physical layer subtype (see method RsCmwEvdoMeas.Configure.MultiEval.plSubtype) : SF=15 for

subtype 0/1 and SF=31 for subtypes 2 and 3 (only NAV values) . Range: -70 dB to 0 dB, Unit: dB

7.4.2.4.2.12 Minimum

SCPI Commands

```
READ:EVDO:MEASurement<Instance>:MEValuation:TRACe:CDP:QSIGnal:PILot:MINimum
FETCh:EVDO:MEASurement<Instance>:MEValuation:TRACe:CDP:QSIGnal:PILot:MINimum
CALCulate:EVDO:MEASurement<Instance>:MEValuation:TRACe:CDP:QSIGnal:PILot:MINimum
```

class Minimum

Minimum commands group definition. 3 total commands, 0 Sub-groups, 3 group commands

calculate() → List[RsCmwEvdoMeas.enums.ResultStatus2]

```
# SCPI: CALCulate:EVDO:MEASurement<instance>
↪:MEValuation:TRACe:CDP:QSIGnal:PILot:MINimum
value: List[enums.ResultStatus2] = driver.multiEval.trace.cdp.qsignal.pilot.
↪minimum.calculate()
```

Return values of the code domain power (CDP) I-signal and Q-signal bar graphs. The results of the current, average, maximum and minimum bar graphs can be retrieved. Only the pilot part of a physical layer subtype 0/1 measurement is retrieved. For RRI part and subtype 2 or 3 measurements, see method RsCmwEvdoMeas.MultiEval.Trace.Cdp.Isignal.Rri.Current. fetch etc. 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.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return qsig_pil_min_cdp: float The number of results depends on the physical layer subtype (see method RsCmwEvdoMeas.Configure.MultiEval.plSubtype) : SF=15 for subtype 0/1 and SF=31 for subtypes 2 and 3 (only NAV values) . Range: -70 dB to 0 dB, Unit: dB

fetch() → List[float]

```
# SCPI: FETCh:EVDO:MEASurement<instance>
↪:MEValuation:TRACe:CDP:QSIGnal:PILot:MINimum
value: List[float] = driver.multiEval.trace.cdp.qsignal.pilot.minimum.fetch()
```

Return values of the code domain power (CDP) I-signal and Q-signal bar graphs. The results of the current, average, maximum and minimum bar graphs can be retrieved. Only the pilot part of a physical layer subtype 0/1 measurement is retrieved. For RRI part and subtype 2 or 3 measurements, see method RsCmwEvdoMeas.MultiEval.Trace.Cdp.Isignal.Rri.Current. fetch etc. 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.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return qsig_pil_min_cdp: float The number of results depends on the physical layer subtype (see method RsCmwEvdoMeas.Configure.MultiEval.plSubtype) : SF=15 for subtype 0/1 and SF=31 for subtypes 2 and 3 (only NAV values) . Range: -70 dB to 0 dB, Unit: dB

read() → List[float]

```
# SCPI: READ:EVDO:MEASurement<instance>
↪:MEValuation:TRACe:CDP:QSIGnal:PILot:MINimum
value: List[float] = driver.multiEval.trace.cdp.qsignal.pilot.minimum.read()
```

Return values of the code domain power (CDP) I-signal and Q-signal bar graphs. The results of the current, average, maximum and minimum bar graphs can be retrieved. Only the pilot part of a physical layer subtype 0/1 measurement is retrieved. For RRI part and subtype 2 or 3 measurements, see method RsCmwEvdoMeas.MultiEval.Trace.Cdp.Isignal.Rri.Current. fetch etc. 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.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return qsig_pil_min_cdp: float The number of results depends on the physical layer subtype (see method RsCmwEvdoMeas.Configure.MultiEval.plSubtype) : SF=15 for subtype 0/1 and SF=31 for subtypes 2 and 3 (only NAV values) . Range: -70 dB to 0 dB, Unit: dB

7.4.2.4.2.13 State

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:TRACe:CDP:QSIGnal:PILot:STATe
```

class State

State commands group definition. 1 total commands, 0 Sub-groups, 1 group commands

fetch() → List[RsCmwEvdoMeas.enums.ResultStateB]

```
# SCPI: FETCh:EVDO:MEASurement<instance>
↪:MEValuation:TRACe:CDP:QSIGnal:PILot:STATe
value: List[enums.ResultStateB] = driver.multiEval.trace.cdp.qsignal.pilot.
↪state.fetch()
```

Return the states of the code domain power (CDP) I-signal and Q-signal bar graphs. Only the pilot part of a physical layer subtype 0/1 measurement is retrieved. For RRI part and subtype 2 or 3 measurements, see method RsCmwEvdoMeas. MultiEval.Trace.Cdp.Qsignal.Rri.State.fetch.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return qsig_pilot_state: NAV | ACTive | INACtive The number of results depends on the physical layer subtype (see method RsCmwEvdoMeas.Configure.MultiEval.plSubtype) : SF=15 for subtype 0/1 and SF=31 for subtypes 2 and 3 (only NAV values) . NAV: No channel available ACTive: Active code channel INACtive: Inactive code channel

7.4.2.4.2.14 Limit

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:TRACe:CDP:QSIGnal:PILot:LIMit
```

class Limit

Limit commands group definition. 1 total commands, 0 Sub-groups, 1 group commands

fetch() → List[float]

```
# SCPI: FETCh:EVDO:MEASurement<instance>
↪:MEValuation:TRACe:CDP:QSIGnal:PILot:LIMit
value: List[float] = driver.multiEval.trace.cdp.qsignal.pilot.limit.fetch()
```

Return limit check results for the code domain power (CDP) I-signal and Q-signal bar graphs. Only the pilot part of a physical layer subtype 0/1 measurement is retrieved. For RRI part and subtype 2 or 3 measurements, see method RsCmwEvdoMeas.MultiEval.Trace.Cdp.Qsignal.Rri.State.fetch.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return qsig_pilot_limit: float Return the exceeded limits as float values. The number of results depends on the physical layer subtype (see method RsCmwEvdoMeas.Configure.MultiEval.plSubtype) : SF=15 for subtype 0/1 and SF=31 for subtypes 2 and 3. Range: -70 dB to 0 dB , Unit: dB

7.4.2.5 Cde

class Cde

Cde commands group definition. 44 total commands, 2 Sub-groups, 0 group commands

Cloning the Group

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

Subgroups

7.4.2.5.1 Isignal

class Isignal

Isignal commands group definition. 22 total commands, 2 Sub-groups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.multiEval.trace.cde.isignal.clone()
```

Subgroups

7.4.2.5.1.1 Rri

class Rri
Rri commands group definition. 11 total commands, 5 Sub-groups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.multiEval.trace.cde.isignal.rri.clone()
```

Subgroups

7.4.2.5.1.2 Current

SCPI Commands

```
READ:EVDO:MEASurement<Instance>:MEValuation:TRACe:CDE:ISIGnal:RRI:CURRent
FETCh:EVDO:MEASurement<Instance>:MEValuation:TRACe:CDE:ISIGnal:RRI:CURRent
CALCulate:EVDO:MEASurement<Instance>:MEValuation:TRACe:CDE:ISIGnal:RRI:CURRent
```

class Current
Current commands group definition. 3 total commands, 0 Sub-groups, 3 group commands

calculate() → List[RsCmwEvdoMeas.enums.ResultStatus2]

```
# SCPI: CALCulate:EVDO:MEASurement<instance>
↪:MEValuation:TRACe:CDE:ISIGnal:RRI:CURRent
value: List[enums.ResultStatus2] = driver.multiEval.trace.cde.isignal.rri.
↪current.calculate()
```

Return values of the code domain error (CDE) I-Signal and Q-Signal bar graphs. Current, average and maximum results can be retrieved. For a physical layer subtype 2 or 3 measurement, the bar graphs contain only RRI results. For a physical layer subtype 0/1 measurement the bar graphs contain also pilot results, see method RsCmwEvdoMeas.MultiEval.Trace.Cde. Isignal.Pilot.Current.fetch etc. 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.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return isig_rri_curr_cde: float The number of results depends on the physical layer subtype (see method RsCmwEvdoMeas.Configure.MultiEval.plSubtype) : SF=15 for subtype 0/1 and SF=31 for subtypes 2 and 3. Range: -70 dB to 0 dB, Unit: dB

fetch() → List[float]

```
# SCPI: FETCh:EVDO:MEASurement<instance>
↪:MEValuation:TRACe:CDE:ISIGnal:RRI:CURRent
value: List[float] = driver.multiEval.trace.cde.isignal.rri.current.fetch()
```

Return values of the code domain error (CDE) I-Signal and Q-Signal bar graphs. Current, average and maximum results can be retrieved. For a physical layer subtype 2 or 3 measurement, the bar graphs contain only RRI results. For a physical layer subtype 0/1 measurement the bar graphs contain also pilot results, see method RsCmwEvdoMeas.MultiEval.Trace.Cde. Isignal.Pilot.Current.fetch etc. 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.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return isig_rri_curr_cde: float The number of results depends on the physical layer subtype (see method RsCmwEvdoMeas.Configure.MultiEval.plSubtype) : SF=15 for subtype 0/1 and SF=31 for subtypes 2 and 3. Range: -70 dB to 0 dB, Unit: dB

read() → List[float]

```
# SCPI: READ:EVDO:MEASurement<instance>
↪:MEValuation:TRACe:CDE:ISIGnal:RRI:CURRent
value: List[float] = driver.multiEval.trace.cde.isignal.rri.current.read()
```

Return values of the code domain error (CDE) I-Signal and Q-Signal bar graphs. Current, average and maximum results can be retrieved. For a physical layer subtype 2 or 3 measurement, the bar graphs contain only RRI results. For a physical layer subtype 0/1 measurement the bar graphs contain also pilot results, see method RsCmwEvdoMeas.MultiEval.Trace.Cde. Isignal.Pilot.Current.fetch etc. 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.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return isig_rri_curr_cde: float The number of results depends on the physical layer subtype (see method RsCmwEvdoMeas.Configure.MultiEval.plSubtype) : SF=15 for subtype 0/1 and SF=31 for subtypes 2 and 3. Range: -70 dB to 0 dB, Unit: dB

7.4.2.5.1.3 Average

SCPI Commands

```
READ:EVDO:MEASurement<Instance>:MEValuation:TRACe:CDE:ISIGnal:RRI:AVERage
FETCh:EVDO:MEASurement<Instance>:MEValuation:TRACe:CDE:ISIGnal:RRI:AVERage
CALCulate:EVDO:MEASurement<Instance>:MEValuation:TRACe:CDE:ISIGnal:RRI:AVERage
```

class Average

Average commands group definition. 3 total commands, 0 Sub-groups, 3 group commands

calculate() → List[RsCmwEvdoMeas.enums.ResultStatus2]

```
# SCPI: CALCulate:EVDO:MEASurement<instance>
↪:MEValuation:TRACe:CDE:ISIGnal:RRI:AVERage
value: List[enums.ResultStatus2] = driver.multiEval.trace.cde.isignal.rri.
↪average.calculate()
```

Return values of the code domain error (CDE) I-Signal and Q-Signal bar graphs. Current, average and maximum results can be retrieved. For a physical layer subtype 2 or 3 measurement, the bar graphs contain only RRI results. For a physical layer subtype 0/1 measurement the bar graphs contain also pilot results, see method RsCmwEvdoMeas.MultiEval.Trace.Cde. Isignal.Pilot.Current.fetch etc. 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.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return isig_rri_aver_cde: float The number of results depends on the physical layer subtype (see method RsCmwEvdoMeas.Configure.MultiEval.plSubtype) : SF=15 for subtype 0/1 and SF=31 for subtypes 2 and 3. Range: -70 dB to 0 dB, Unit: dB

fetch() → List[float]

```
# SCPI: FETCh:EVDO:MEASurement<instance>
↪:MEValuation:TRACe:CDE:ISIGnal:RRI:AVERage
value: List[float] = driver.multiEval.trace.cde.isignal.rri.average.fetch()
```

Return values of the code domain error (CDE) I-Signal and Q-Signal bar graphs. Current, average and maximum results can be retrieved. For a physical layer subtype 2 or 3 measurement, the bar graphs contain only RRI results. For a physical layer subtype 0/1 measurement the bar graphs contain also pilot results, see method RsCmwEvdoMeas.MultiEval.Trace.Cde. Isignal.Pilot.Current.fetch etc. 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.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return isig_rri_aver_cde: float The number of results depends on the physical layer subtype (see method RsCmwEvdoMeas.Configure.MultiEval.plSubtype) : SF=15 for subtype 0/1 and SF=31 for subtypes 2 and 3. Range: -70 dB to 0 dB, Unit: dB

read() → List[float]

```
# SCPI: READ:EVDO:MEASurement<instance>
↪:MEValuation:TRACe:CDE:ISIGnal:RRI:AVERage
value: List[float] = driver.multiEval.trace.cde.isignal.rri.average.read()
```

Return values of the code domain error (CDE) I-Signal and Q-Signal bar graphs. Current, average and maximum results can be retrieved. For a physical layer subtype 2 or 3 measurement, the bar graphs contain only RRI results. For a physical layer subtype 0/1 measurement the bar graphs contain also pilot results, see method RsCmwEvdoMeas.MultiEval.Trace.Cde. Isignal.Pilot.Current.fetch etc. 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.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return isig_rri_aver_cde: float The number of results depends on the physical layer subtype (see method RsCmwEvdoMeas.Configure.MultiEval.plSubtype) : SF=15 for subtype 0/1 and SF=31 for subtypes 2 and 3. Range: -70 dB to 0 dB, Unit: dB

7.4.2.5.1.4 Maximum

SCPI Commands

```
READ:EVDO:MEASurement<Instance>:MEValuation:TRACe:CDE:ISIGnal:RRI:MAXimum
FETCh:EVDO:MEASurement<Instance>:MEValuation:TRACe:CDE:ISIGnal:RRI:MAXimum
CALCulate:EVDO:MEASurement<Instance>:MEValuation:TRACe:CDE:ISIGnal:RRI:MAXimum
```

class Maximum

Maximum commands group definition. 3 total commands, 0 Sub-groups, 3 group commands

calculate() → List[RsCmwEvdoMeas.enums.ResultStatus2]

```
# SCPI: CALCulate:EVDO:MEASurement<instance>
↪:MEValuation:TRACe:CDE:ISIGnal:RRI:MAXimum
value: List[enums.ResultStatus2] = driver.multiEval.trace.cde.isignal.rri.
↪maximum.calculate()
```

Return values of the code domain error (CDE) I-Signal and Q-Signal bar graphs. Current, average and maximum results can be retrieved. For a physical layer subtype 2 or 3 measurement, the bar graphs contain only RRI results. For a physical layer subtype 0/1 measurement the bar graphs contain also pilot results, see method RsCmwEvdoMeas.MultiEval.Trace.Cde. Isignal.Pilot.Current.fetch etc. 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.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return isig_rri_max_cde: float The number of results depends on the physical layer subtype (see method RsCmwEvdoMeas.Configure.MultiEval.plSubtype) : SF=15 for subtype 0/1 and SF=31 for subtypes 2 and 3. Range: -70 dB to 0 dB, Unit: dB

fetch() → List[float]

```
# SCPI: FETCh:EVDO:MEASurement<instance>
↪:MEValuation:TRACe:CDE:ISIGnal:RRI:MAXimum
value: List[float] = driver.multiEval.trace.cde.isignal.rri.maximum.fetch()
```

Return values of the code domain error (CDE) I-Signal and Q-Signal bar graphs. Current, average and maximum results can be retrieved. For a physical layer subtype 2 or 3 measurement, the bar graphs contain only RRI results. For a physical layer subtype 0/1 measurement the bar graphs contain also pilot results, see method RsCmwEvdoMeas.MultiEval.Trace.Cde. Isignal.Pilot.Current.fetch etc. 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.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return isig_rri_max_cde: float The number of results depends on the physical layer subtype (see method RsCmwEvdoMeas.Configure.MultiEval.plSubtype) : SF=15 for subtype 0/1 and SF=31 for subtypes 2 and 3. Range: -70 dB to 0 dB, Unit: dB

read() → List[float]

```
# SCPI: READ:EVDO:MEASurement<instance>
↪:MEValuation:TRACe:CDE:ISIGnal:RRI:MAXimum
value: List[float] = driver.multiEval.trace.cde.isignal.rri.maximum.read()
```

Return values of the code domain error (CDE) I-Signal and Q-Signal bar graphs. Current, average and maximum results can be retrieved. For a physical layer subtype 2 or 3 measurement, the bar graphs contain only RRI results. For a physical layer subtype 0/1 measurement the bar graphs contain also pilot results, see method RsCmwEvdoMeas.MultiEval.Trace.Cde. Isignal.Pilot.Current.fetch etc. 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.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return isig_rri_max_cde: float The number of results depends on the physical layer subtype (see method RsCmwEvdoMeas.Configure.MultiEval.plSubtype) : SF=15 for subtype 0/1 and SF=31 for subtypes 2 and 3. Range: -70 dB to 0 dB, Unit: dB

7.4.2.5.1.5 State

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:TRACe:CDE:ISIGnal:RRI:STATe
```

class State

State commands group definition. 1 total commands, 0 Sub-groups, 1 group commands

fetch() → List[RsCmwEvdoMeas.enums.ResultStateB]

```
# SCPI: FETCh:EVDO:MEASurement<instance>:MEValuation:TRACe:CDE:ISIGnal:RRI:STATe
value: List[enums.ResultStateB] = driver.multiEval.trace.cde.isignal.rri.state.
↪fetch()
```

Return the states of the code domain error (CDE) I-Signal and Q-Signal bar graphs. For a physical layer subtype 2 or measurement, the bar graphs contain only RRI results. For a physical layer subtype 0/1 measurement the bar graphs contain also pilot results, see method RsCmwEvdoMeas.MultiEval.Trace.Cde.Qsignal.Pilot.State.fetch.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return isig_rri_state: NAV | ACTive | INACtive The number of results depends on the physical layer subtype (see method RsCmwEvdoMeas.Configure.MultiEval.plSubtype) : SF=15 for subtype 0/1 and SF=31 for subtypes 2 and 3 (only NAV values) . NAV: No channel available ACTive: Active code channel INACtive: Inactive code channel

7.4.2.5.1.6 Limit

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:TRACe:CDE:ISIGnal:RRI:LIMit
```

class Limit

Limit commands group definition. 1 total commands, 0 Sub-groups, 1 group commands

fetch() → List[float]

```
# SCPI: FETCh:EVDO:MEASurement<instance>:MEValuation:TRACe:CDE:ISIGnal:RRI:LIMit
value: List[float] = driver.multiEval.trace.cde.isignal.rri.limit.fetch()
```

Return limit check results for the code domain error (CDE) I-Signal and Q-Signal bar graphs. For a physical layer subtype 2 or 3 measurement, the bar graphs contain only RRI results. For a physical layer subtype 0/1 measurement the bar graphs contain also pilot results, see method RsCmwEvdoMeas.MultiEval.Trace.Cde.Qsignal.Pilot.Limit.fetch.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return isig_rri_limit: float The number of results depends on the physical layer subtype (see method RsCmwEvdoMeas.Configure.MultiEval.plSubtype) : SF=15 for subtype 0/1 and SF=31 for subtypes 2 and 3. OK: The result is located within the limits or no limit has been defined/enabled for this result. ULEU (user limit exceeded upper) : An upper limit is violated. The result is located above the limit. ULEL (user limit exceeded lower) : A lower limit is violated. The result is located below the limit. Range: OK | ULEU | ULEL

7.4.2.5.1.7 Pilot

class Pilot

Pilot commands group definition. 11 total commands, 5 Sub-groups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.multiEval.trace.cde.isignal.pilot.clone()
```

Subgroups

7.4.2.5.1.8 Current

SCPI Commands

```
READ:EVDO:MEASurement<Instance>:MEValuation:TRACe:CDE:ISIGnal:PILot:CURRent
FETCh:EVDO:MEASurement<Instance>:MEValuation:TRACe:CDE:ISIGnal:PILot:CURRent
CALCulate:EVDO:MEASurement<Instance>:MEValuation:TRACe:CDE:ISIGnal:PILot:CURRent
```

class Current

Current commands group definition. 3 total commands, 0 Sub-groups, 3 group commands

calculate() → List[RsCmwEvdoMeas.enums.ResultStatus2]

```
# SCPI: CALCulate:EVDO:MEASurement<instance>
↪:MEValuation:TRACe:CDE:ISIGnal:PILot:CURRent
value: List[enums.ResultStatus2] = driver.multiEval.trace.cde.isignal.pilot.
↪current.calculate()
```

Return values of the code domain error (CDE) I-Signal and Q-Signal bar graphs. The results of the current, average and maximum bar graphs can be retrieved. Only the pilot part of a physical layer subtype 0/1 measurement is retrieved. For RRI part and subtype 2 or 3 measurements, see method RsCmwEvdoMeas.MultiEval.Trace.Cde.Isignal.Rri.Current.fetch etc. 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.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return isig_pil_curr_cde: float The number of results depends on the physical layer subtype (see method RsCmwEvdoMeas.Configure.MultiEval.plSubtype) : SF=15 for subtype 0/1 and SF=31 for subtypes 2 and 3 (only NAV values) . Range: -70 to 0 dB, Unit: dB

fetch() → List[float]

```
# SCPI: FETCh:EVDO:MEASurement<instance>
↪:MEValuation:TRACe:CDE:ISIGnal:PILot:CURRent
value: List[float] = driver.multiEval.trace.cde.isignal.pilot.current.fetch()
```

Return values of the code domain error (CDE) I-Signal and Q-Signal bar graphs. The results of the current, average and maximum bar graphs can be retrieved. Only the pilot part of a physical layer subtype 0/1 measurement is retrieved. For RRI part and subtype 2 or 3 measurements, see method RsCmwEvdoMeas.MultiEval.Trace.Cde.Isignal.Rri.Current.fetch etc. 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.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return isig_pil_curr_cde: float The number of results depends on the physical layer subtype (see method RsCmwEvdoMeas.Configure.MultiEval.plSubtype) : SF=15 for subtype 0/1 and SF=31 for subtypes 2 and 3 (only NAV values) . Range: -70 to 0 dB, Unit: dB

read() → List[float]

```
# SCPI: READ:EVDO:MEASurement<instance>
↪:MEValuation:TRACe:CDE:ISIGnal:PILot:CURRent
value: List[float] = driver.multiEval.trace.cde.isignal.pilot.current.read()
```

Return values of the code domain error (CDE) I-Signal and Q-Signal bar graphs. The results of the current, average and maximum bar graphs can be retrieved. Only the pilot part of a physical layer subtype 0/1 measurement is retrieved. For RRI part and subtype 2 or 3 measurements, see method RsCmwEvdoMeas.MultiEval.Trace.Cde.Isignal.Rri.Current.fetch etc. 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.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return isig_pil_curr_cde: float The number of results depends on the physical layer subtype (see method RsCmwEvdoMeas.Configure.MultiEval.plSubtype) : SF=15 for subtype 0/1 and SF=31 for subtypes 2 and 3 (only NAV values) . Range: -70 to 0 dB, Unit: dB

7.4.2.5.1.9 Average

SCPI Commands

```
READ:EVDO:MEASurement<Instance>:MEValuation:TRACe:CDE:ISIGnal:PILot:AVERage
FETCh:EVDO:MEASurement<Instance>:MEValuation:TRACe:CDE:ISIGnal:PILot:AVERage
CALCulate:EVDO:MEASurement<Instance>:MEValuation:TRACe:CDE:ISIGnal:PILot:AVERage
```

class Average

Average commands group definition. 3 total commands, 0 Sub-groups, 3 group commands

calculate() → List[RsCmwEvdoMeas.enums.ResultStatus2]

```
# SCPI: CALCulate:EVDO:MEASurement<instance>
↪:MEValuation:TRACe:CDE:ISIGnal:PILot:AVERage
value: List[enums.ResultStatus2] = driver.multiEval.trace.cde.isignal.pilot.
↪average.calculate()
```

Return values of the code domain error (CDE) I-Signal and Q-Signal bar graphs. The results of the current, average and maximum bar graphs can be retrieved. Only the pilot part of a physical layer subtype 0/1 measurement is retrieved. For RRI part and subtype 2 or 3 measurements, see method RsCmwEvdoMeas.MultiEval.Trace.Cde.Isignal.Rri.Current.fetch etc. 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.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return isig_pil_aver_cde: float The number of results depends on the physical layer subtype (see method RsCmwEvdoMeas.Configure.MultiEval.plSubtype) : SF=15 for subtype 0/1 and SF=31 for subtypes 2 and 3 (only NAV values) . Range: -70 to 0 dB, Unit: dB

fetch() → List[float]

```
# SCPI: FETCh:EVDO:MEASurement<instance>
↪:MEValuation:TRACe:CDE:ISIGnal:PILot:AVERage
value: List[float] = driver.multiEval.trace.cde.isignal.pilot.average.fetch()
```

Return values of the code domain error (CDE) I-Signal and Q-Signal bar graphs. The results of the current, average and maximum bar graphs can be retrieved. Only the pilot part of a physical layer subtype 0/1 measurement is retrieved. For RRI part and subtype 2 or 3 measurements, see method RsCmwEvdoMeas.MultiEval.Trace.Cde.Isignal.Rri.Current.fetch etc. 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.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return isig_pil_aver_cde: float The number of results depends on the physical layer subtype (see method RsCmwEvdoMeas.Configure.MultiEval.plSubtype) : SF=15 for subtype 0/1 and SF=31 for subtypes 2 and 3 (only NAV values) . Range: -70 to 0 dB, Unit: dB

read() → List[float]

```
# SCPI: READ:EVDO:MEASurement<instance>
↪:MEValuation:TRACe:CDE:ISIGnal:PILot:AVERage
value: List[float] = driver.multiEval.trace.cde.isignal.pilot.average.read()
```

Return values of the code domain error (CDE) I-Signal and Q-Signal bar graphs. The results of the current, average and maximum bar graphs can be retrieved. Only the pilot part of a physical layer subtype 0/1 measurement is retrieved. For RRI part and subtype 2 or 3 measurements, see method RsCmwEvdoMeas.MultiEval.Trace.Cde.Isignal.Rri.Current.fetch etc. 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.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return isig_pil_aver_cde: float The number of results depends on the physical layer subtype (see method RsCmwEvdoMeas.Configure.MultiEval.plSubtype) : SF=15 for subtype 0/1 and SF=31 for subtypes 2 and 3 (only NAV values) . Range: -70 to 0 dB, Unit: dB

7.4.2.5.1.10 Maximum

SCPI Commands

```
READ:EVDO:MEASurement<Instance>:MEValuation:TRACe:CDE:ISIGnal:PILot:MAXimum
FETCh:EVDO:MEASurement<Instance>:MEValuation:TRACe:CDE:ISIGnal:PILot:MAXimum
CALCulate:EVDO:MEASurement<Instance>:MEValuation:TRACe:CDE:ISIGnal:PILot:MAXimum
```

class Maximum

Maximum commands group definition. 3 total commands, 0 Sub-groups, 3 group commands

calculate() → List[RsCmwEvdoMeas.enums.ResultStatus2]

```
# SCPI: CALCulate:EVDO:MEASurement<instance>
↪:MEValuation:TRACe:CDE:ISIGnal:PILot:MAXimum
value: List[enums.ResultStatus2] = driver.multiEval.trace.cde.isignal.pilot.
↪maximum.calculate()
```

Return values of the code domain error (CDE) I-Signal and Q-Signal bar graphs. The results of the current, average and maximum bar graphs can be retrieved. Only the pilot part of a physical layer subtype 0/1 measurement is retrieved. For RRI part and subtype 2 or 3 measurements, see method RsCmwEvdoMeas.MultiEval.Trace.Cde.Isignal.Rri.Current.fetch etc. 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.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return isig_pil_max_cde: float The number of results depends on the physical layer subtype (see method RsCmwEvdoMeas.Configure.MultiEval.plSubtype) : SF=15 for subtype 0/1 and SF=31 for subtypes 2 and 3 (only NAV values) . Range: -70 to 0 dB, Unit: dB

fetch() → List[float]

```
# SCPI: FETCh:EVDO:MEASurement<instance>
↪:MEValuation:TRACe:CDE:ISIGnal:PILot:MAXimum
value: List[float] = driver.multiEval.trace.cde.isignal.pilot.maximum.fetch()
```

Return values of the code domain error (CDE) I-Signal and Q-Signal bar graphs. The results of the current, average and maximum bar graphs can be retrieved. Only the pilot part of a physical layer subtype 0/1 measurement is retrieved. For RRI part and subtype 2 or 3 measurements, see method RsCmwEvdoMeas.MultiEval.Trace.Cde.Isignal.Rri.Current.fetch etc. 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.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return isig_pil_max_cde: float The number of results depends on the physical layer subtype (see method RsCmwEvdoMeas.Configure.MultiEval.plSubtype) : SF=15 for subtype 0/1 and SF=31 for subtypes 2 and 3 (only NAV values) . Range: -70 to 0 dB, Unit: dB

read() → List[float]

```
# SCPI: READ:EVDO:MEASurement<instance>
↪:MEValuation:TRACe:CDE:ISIGnal:PILot:MAXimum
value: List[float] = driver.multiEval.trace.cde.isignal.pilot.maximum.read()
```

Return values of the code domain error (CDE) I-Signal and Q-Signal bar graphs. The results of the current, average and maximum bar graphs can be retrieved. Only the pilot part of a physical layer subtype 0/1 measurement is retrieved. For RRI part and subtype 2 or 3 measurements, see method RsCmwEvdoMeas.MultiEval.Trace.Cde.Isignal.Rri.Current.fetch etc. 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.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return isig_pil_max_cde: float The number of results depends on the physical layer subtype (see method RsCmwEvdoMeas.Configure.MultiEval.plSubtype) : SF=15 for subtype 0/1 and SF=31 for subtypes 2 and 3 (only NAV values) . Range: -70 to 0 dB, Unit: dB

7.4.2.5.1.11 State

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:TRACe:CDE:ISIGnal:PILot:STATe
```

class State

State commands group definition. 1 total commands, 0 Sub-groups, 1 group commands

fetch() → List[RsCmwEvdoMeas.enums.ResultStateB]

```
# SCPI: FETCh:EVDO:MEASurement<instance>
↪:MEValuation:TRACe:CDE:ISIGnal:PILot:STATe
value: List[enums.ResultStateB] = driver.multiEval.trace.cde.isignal.pilot.
↪state.fetch()
```

Return the states of the code domain error (CDE) I-Signal and Q-Signal bar graphs. Only the pilot part of a physical layer subtype 0/1 measurement is retrieved. For RRI part and subtype 2 or 3 measurements, see method RsCmwEvdoMeas. MultiEval.Trace.Cde.Qsignal.Rri.State.fetch.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return isig_pilot_state: NAV | ACTive | INACtive The number of results depends on the physical layer subtype (see method RsCmwEvdoMeas.Configure.MultiEval.plSubtype) : SF=15 for subtype 0/1 and SF=31 for subtypes 2 and 3 (only NAV values) . NAV: No channel available ACTive: Active code channel INACtive: Inactive code channel

7.4.2.5.1.12 Limit

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:TRACe:CDE:ISIGnal:PILot:LIMit
```

class Limit

Limit commands group definition. 1 total commands, 0 Sub-groups, 1 group commands

fetch() → List[float]

```
# SCPI: FETCh:EVDO:MEASurement<instance>
↪:MEValuation:TRACe:CDE:ISIGnal:PILot:LIMit
value: List[float] = driver.multiEval.trace.cde.isignal.pilot.limit.fetch()
```

Return limit check results for the code domain error (CDE) I-Signal and Q-Signal bar graphs. Only the pilot part of a physical layer subtype 0/1 measurement is retrieved. For RRI part and subtype 2 or 3 measurements, see method RsCmwEvdoMeas.MultiEval.Trace.Cde.Qsignal.Rri.Limit.fetch.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return isig_pilot_limit: float Return the exceeded limits as float values. The number of results depends on the physical layer subtype (see method RsCmwEvdoMeas.Configure.MultiEval.plSubtype) : SF=15 for subtype 0/1 and SF=31 for subtypes 2 and 3 (only INV values) . Range: -70 dB to 0 dB , Unit: dB

7.4.2.5.2 Qsignal

class Qsignal

Qsignal commands group definition. 22 total commands, 2 Sub-groups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.multiEval.trace.cde.qsignal.clone()
```

Subgroups

7.4.2.5.2.1 Rri

class Rri

Rri commands group definition. 11 total commands, 5 Sub-groups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.multiEval.trace.cde.qsignal.rri.clone()
```

Subgroups

7.4.2.5.2.2 Current

SCPI Commands

```
READ:EVDO:MEASurement<Instance>:MEValuation:TRACe:CDE:QSIGnal:RRI:CURRent
FETCh:EVDO:MEASurement<Instance>:MEValuation:TRACe:CDE:QSIGnal:RRI:CURRent
CALCulate:EVDO:MEASurement<Instance>:MEValuation:TRACe:CDE:QSIGnal:RRI:CURRent
```

class Current

Current commands group definition. 3 total commands, 0 Sub-groups, 3 group commands

calculate() → List[RsCmwEvdoMeas.enums.ResultStatus2]

```
# SCPI: CALCulate:EVDO:MEASurement<instance>
↪:MEValuation:TRACe:CDE:QSIGnal:RRI:CURRent
value: List[enums.ResultStatus2] = driver.multiEval.trace.cde.qsignal.rri.
↪current.calculate()
```

Return values of the code domain error (CDE) I-Signal and Q-Signal bar graphs. Current, average and maximum results can be retrieved. For a physical layer subtype 2 or 3 measurement, the bar graphs contain only RRI results. For a physical layer subtype 0/1 measurement the bar graphs contain also pilot results, see method RsCmwEvdoMeas.MultiEval.Trace.Cde. Isignal.Pilot.Current.fetch etc. 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.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return qsig_rri_curr_cde: float The number of results depends on the physical layer subtype (see method RsCmwEvdoMeas.Configure.MultiEval.plSubtype) : SF=15 for subtype 0/1 and SF=31 for subtypes 2 and 3. Range: -70 dB to 0 dB, Unit: dB

fetch() → List[float]

```
# SCPI: FETCh:EVDO:MEASurement<instance>
↪:MEValuation:TRACe:CDE:QSIGnal:RRI:CURRent
value: List[float] = driver.multiEval.trace.cde.qsignal.rri.current.fetch()
```

Return values of the code domain error (CDE) I-Signal and Q-Signal bar graphs. Current, average and maximum results can be retrieved. For a physical layer subtype 2 or 3 measurement, the bar graphs contain only RRI results. For a physical layer subtype 0/1 measurement the bar graphs contain also pilot results, see method RsCmwEvdoMeas.MultiEval.Trace.Cde. Isignal.Pilot.Current.fetch etc. 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.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return qsig_rri_curr_cde: float The number of results depends on the physical layer subtype (see method RsCmwEvdoMeas.Configure.MultiEval.plSubtype) : SF=15 for subtype 0/1 and SF=31 for subtypes 2 and 3. Range: -70 dB to 0 dB, Unit: dB

read() → List[float]

```
# SCPI: READ:EVDO:MEASurement<instance>
↪:MEValuation:TRACe:CDE:QSIGnal:RRI:CURRent
value: List[float] = driver.multiEval.trace.cde.qsignal.rri.current.read()
```

Return values of the code domain error (CDE) I-Signal and Q-Signal bar graphs. Current, average and maximum results can be retrieved. For a physical layer subtype 2 or 3 measurement, the bar graphs contain only RRI results. For a physical layer subtype 0/1 measurement the bar graphs contain also pilot results, see method RsCmwEvdoMeas.MultiEval.Trace.Cde. Isignal.Pilot.Current.fetch etc. 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.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return qsig_rri_curr_cde: float The number of results depends on the physical layer subtype (see method RsCmwEvdoMeas.Configure.MultiEval.plSubtype) : SF=15 for subtype 0/1 and SF=31 for subtypes 2 and 3. Range: -70 dB to 0 dB, Unit: dB

7.4.2.5.2.3 Average

SCPI Commands

```
READ:EVDO:MEASurement<Instance>:MEValuation:TRACe:CDE:QSIGnal:RRI:AVERage
FETCh:EVDO:MEASurement<Instance>:MEValuation:TRACe:CDE:QSIGnal:RRI:AVERage
CALCulate:EVDO:MEASurement<Instance>:MEValuation:TRACe:CDE:QSIGnal:RRI:AVERage
```

class Average

Average commands group definition. 3 total commands, 0 Sub-groups, 3 group commands

calculate() → List[RsCmwEvdoMeas.enums.ResultStatus2]

```
# SCPI: CALCulate:EVDO:MEASurement<instance>
↪:MEValuation:TRACe:CDE:QSIGnal:RRI:AVERage
value: List[enums.ResultStatus2] = driver.multiEval.trace.cde.qsignal.rri.
↪average.calculate()
```

Return values of the code domain error (CDE) I-Signal and Q-Signal bar graphs. Current, average and maximum results can be retrieved. For a physical layer subtype 2 or 3 measurement, the bar graphs contain only RRI results. For a physical layer subtype 0/1 measurement the bar graphs contain also pilot results, see method RsCmwEvdoMeas.MultiEval.Trace.Cde. Isignal.Pilot.Current.fetch etc. 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.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return qsig_rri_aver_cde: float The number of results depends on the physical layer subtype (see method RsCmwEvdoMeas.Configure.MultiEval.plSubtype) : SF=15 for subtype 0/1 and SF=31 for subtypes 2 and 3. Range: -70 dB to 0 dB, Unit: dB

fetch() → List[float]

```
# SCPI: FETCh:EVDO:MEASurement<instance>
↪:MEValuation:TRACe:CDE:QSIGnal:RRI:AVERage
value: List[float] = driver.multiEval.trace.cde.qsignal.rri.average.fetch()
```

Return values of the code domain error (CDE) I-Signal and Q-Signal bar graphs. Current, average and maximum results can be retrieved. For a physical layer subtype 2 or 3 measurement, the bar graphs contain only RRI results. For a physical layer subtype 0/1 measurement the bar graphs contain also pilot results, see method RsCmwEvdoMeas.MultiEval.Trace.Cde. Isignal.Pilot.Current.fetch etc. 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.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return qsig_rri_aver_cde: float The number of results depends on the physical layer subtype (see method RsCmwEvdoMeas.Configure.MultiEval.plSubtype) : SF=15 for subtype 0/1 and SF=31 for subtypes 2 and 3. Range: -70 dB to 0 dB, Unit: dB

read() → List[float]

```
# SCPI: READ:EVDO:MEASurement<instance>
↪:MEValuation:TRACe:CDE:QSIGnal:RRI:AVERage
value: List[float] = driver.multiEval.trace.cde.qsignal.rri.average.read()
```

Return values of the code domain error (CDE) I-Signal and Q-Signal bar graphs. Current, average and maximum results can be retrieved. For a physical layer subtype 2 or 3 measurement, the bar graphs contain only RRI results. For a physical layer subtype 0/1 measurement the bar graphs contain also pilot results, see method RsCmwEvdoMeas.MultiEval.Trace.Cde. Isignal.Pilot.Current.fetch etc. 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.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return qsig_rri_aver_cde: float The number of results depends on the physical layer subtype (see method RsCmwEvdoMeas.Configure.MultiEval.plSubtype) : SF=15 for subtype 0/1 and SF=31 for subtypes 2 and 3. Range: -70 dB to 0 dB, Unit: dB

7.4.2.5.2.4 Maximum

SCPI Commands

```
READ:EVDO:MEASurement<Instance>:MEValuation:TRACe:CDE:QSIGnal:RRI:MAXimum
FETCh:EVDO:MEASurement<Instance>:MEValuation:TRACe:CDE:QSIGnal:RRI:MAXimum
CALCulate:EVDO:MEASurement<Instance>:MEValuation:TRACe:CDE:QSIGnal:RRI:MAXimum
```

class Maximum

Maximum commands group definition. 3 total commands, 0 Sub-groups, 3 group commands

calculate() → List[RsCmwEvdoMeas.enums.ResultStatus2]

```
# SCPI: CALCulate:EVDO:MEASurement<instance>
↪:MEValuation:TRACe:CDE:QSIGnal:RRI:MAXimum
value: List[enums.ResultStatus2] = driver.multiEval.trace.cde.qsignal.rri.
↪maximum.calculate()
```

Return values of the code domain error (CDE) I-Signal and Q-Signal bar graphs. Current, average and maximum results can be retrieved. For a physical layer subtype 2 or 3 measurement, the bar graphs contain only RRI results. For a physical layer subtype 0/1 measurement the bar graphs contain also pilot results, see method RsCmwEvdoMeas.MultiEval.Trace.Cde. Isignal.Pilot.Current.fetch etc. 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.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return qsig_rri_max_cde: float The number of results depends on the physical layer subtype (see method RsCmwEvdoMeas.Configure.MultiEval.plSubtype) : SF=15 for subtype 0/1 and SF=31 for subtypes 2 and 3. Range: -70 dB to 0 dB, Unit: dB

fetch() → List[float]

```
# SCPI: FETCh:EVDO:MEASurement<instance>
↪:MEValuation:TRACe:CDE:QSIGnal:RRI:MAXimum
value: List[float] = driver.multiEval.trace.cde.qsignal.rri.maximum.fetch()
```

Return values of the code domain error (CDE) I-Signal and Q-Signal bar graphs. Current, average and maximum results can be retrieved. For a physical layer subtype 2 or 3 measurement, the bar graphs contain only RRI results. For a physical layer subtype 0/1 measurement the bar graphs contain also pilot results, see method RsCmwEvdoMeas.MultiEval.Trace.Cde. Isignal.Pilot.Current.fetch etc. 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.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return qsig_rri_max_cde: float The number of results depends on the physical layer subtype (see method RsCmwEvdoMeas.Configure.MultiEval.plSubtype) : SF=15 for subtype 0/1 and SF=31 for subtypes 2 and 3. Range: -70 dB to 0 dB, Unit: dB

read() → List[float]

```
# SCPI: READ:EVDO:MEASurement<instance>
↪:MEValuation:TRACe:CDE:QSIGnal:RRI:MAXimum
value: List[float] = driver.multiEval.trace.cde.qsignal.rri.maximum.read()
```

Return values of the code domain error (CDE) I-Signal and Q-Signal bar graphs. Current, average and maximum results can be retrieved. For a physical layer subtype 2 or 3 measurement, the bar graphs contain only RRI results. For a physical layer subtype 0/1 measurement the bar graphs contain also pilot results, see method RsCmwEvdoMeas.MultiEval.Trace.Cde. Isignal.Pilot.Current.fetch etc. 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.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return qsig_rri_max_cde: float The number of results depends on the physical layer subtype (see method RsCmwEvdoMeas.Configure.MultiEval.plSubtype) : SF=15 for subtype 0/1 and SF=31 for subtypes 2 and 3. Range: -70 dB to 0 dB, Unit: dB

7.4.2.5.2.5 State

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:TRACe:CDE:QSIGnal:RRI:STATe
```

class State

State commands group definition. 1 total commands, 0 Sub-groups, 1 group commands

fetch() → List[RsCmwEvdoMeas.enums.ResultStateB]

```
# SCPI: FETCh:EVDO:MEASurement<instance>:MEValuation:TRACe:CDE:QSIGnal:RRI:STATe
value: List[enums.ResultStateB] = driver.multiEval.trace.cde.qsignal.rri.state.
↪fetch()
```

Return the states of the code domain error (CDE) I-Signal and Q-Signal bar graphs. For a physical layer subtype 2 or measurement, the bar graphs contain only RRI results. For a physical layer subtype 0/1 measurement the bar graphs contain also pilot results, see method RsCmwEvdoMeas.MultiEval.Trace.Cde.Qsignal.Pilot.State.fetch.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return qsig_rri_state: NAV | ACTive | INACtive The number of results depends on the physical layer subtype (see method RsCmwEvdoMeas.Configure.MultiEval.plSubtype) : SF=15 for subtype 0/1 and SF=31 for subtypes 2 and 3 (only NAV values) . NAV: No channel available ACTive: Active code channel INACtive: Inactive code channel

7.4.2.5.2.6 Limit

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:TRACe:CDE:QSIGnal:RRI:LIMit
```

class Limit

Limit commands group definition. 1 total commands, 0 Sub-groups, 1 group commands

fetch() → List[float]

```
# SCPI: FETCh:EVDO:MEASurement<instance>:MEValuation:TRACe:CDE:QSIGnal:RRI:LIMit
value: List[float] = driver.multiEval.trace.cde.qsignal.rri.limit.fetch()
```

Return limit check results for the code domain error (CDE) I-Signal and Q-Signal bar graphs. For a physical layer subtype 2 or 3 measurement, the bar graphs contain only RRI results. For a physical layer subtype 0/1 measurement the bar graphs contain also pilot results, see method RsCmwEvdoMeas.MultiEval.Trace.Cde.Qsignal.Pilot.Limit.fetch.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return qsig_rri_limit: float The number of results depends on the physical layer subtype (see method RsCmwEvdoMeas.Configure.MultiEval.plSubtype) : SF=15 for subtype 0/1 and SF=31 for subtypes 2 and 3. OK: The result is located within the limits or no limit has been defined/enabled for this result. ULEU (user limit exceeded upper) : An upper limit is violated. The result is located above the limit. ULEL (user limit exceeded lower) : A lower limit is violated. The result is located below the limit. Range: OK | ULEU | ULEL

7.4.2.5.2.7 Pilot

class Pilot

Pilot commands group definition. 11 total commands, 5 Sub-groups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.multiEval.trace.cde.qsignal.pilot.clone()
```

Subgroups

7.4.2.5.2.8 Current

SCPI Commands

```
READ:EVDO:MEASurement<Instance>:MEValuation:TRACe:CDE:QSIGnal:PILot:CURRent
FETCh:EVDO:MEASurement<Instance>:MEValuation:TRACe:CDE:QSIGnal:PILot:CURRent
CALCulate:EVDO:MEASurement<Instance>:MEValuation:TRACe:CDE:QSIGnal:PILot:CURRent
```

class Current

Current commands group definition. 3 total commands, 0 Sub-groups, 3 group commands

calculate() → List[RsCmwEvdoMeas.enums.ResultStatus2]

```
# SCPI: CALCulate:EVDO:MEASurement<instance>
↪:MEValuation:TRACe:CDE:QSIGnal:PILot:CURRent
value: List[enums.ResultStatus2] = driver.multiEval.trace.cde.qsignal.pilot.
↪current.calculate()
```

Return values of the code domain error (CDE) I-Signal and Q-Signal bar graphs. The results of the current, average and maximum bar graphs can be retrieved. Only the pilot part of a physical layer subtype 0/1 measurement is retrieved. For RRI part and subtype 2 or 3 measurements, see method RsCmwEvdoMeas.MultiEval.Trace.Cde.Isignal.Rri.Current.fetch etc. 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.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return qsig_pil_curr_cde: float The number of results depends on the physical layer subtype (see method RsCmwEvdoMeas.Configure.MultiEval.plSubtype) : SF=15 for subtype 0/1 and SF=31 for subtypes 2 and 3 (only NAV values) . Range: -70 to 0 dB, Unit: dB

fetch() → List[float]

```
# SCPI: FETCh:EVDO:MEASurement<instance>
↪:MEValuation:TRACe:CDE:QSIGnal:PILot:CURRent
value: List[float] = driver.multiEval.trace.cde.qsignal.pilot.current.fetch()
```

Return values of the code domain error (CDE) I-Signal and Q-Signal bar graphs. The results of the current, average and maximum bar graphs can be retrieved. Only the pilot part of a physical layer subtype 0/1 measurement is retrieved. For RRI part and subtype 2 or 3 measurements, see method RsCmwEvdoMeas.MultiEval.Trace.Cde.Isignal.Rri.Current.fetch etc. 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.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return qsig_pil_curr_cde: float The number of results depends on the physical layer subtype (see method RsCmwEvdoMeas.Configure.MultiEval.plSubtype) : SF=15 for subtype 0/1 and SF=31 for subtypes 2 and 3 (only NAV values) . Range: -70 to 0 dB, Unit: dB

read() → List[float]

```
# SCPI: READ:EVDO:MEASurement<instance>
↪:MEValuation:TRACe:CDE:QSIGnal:PILot:CURRent
value: List[float] = driver.multiEval.trace.cde.qsignal.pilot.current.read()
```

Return values of the code domain error (CDE) I-Signal and Q-Signal bar graphs. The results of the current, average and maximum bar graphs can be retrieved. Only the pilot part of a physical layer subtype 0/1 measurement is retrieved. For RRI part and subtype 2 or 3 measurements, see method RsCmwEvdoMeas.MultiEval.Trace.Cde.Isignal.Rri.Current.fetch etc. 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.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return qsig_pil_curr_cde: float The number of results depends on the physical layer subtype (see method RsCmwEvdoMeas.Configure.MultiEval.plSubtype) : SF=15 for subtype 0/1 and SF=31 for subtypes 2 and 3 (only NAV values) . Range: -70 to 0 dB, Unit: dB

7.4.2.5.2.9 Average

SCPI Commands

```
READ:EVDO:MEASurement<Instance>:MEValuation:TRACe:CDE:QSIGnal:PILot:AVERage
FETCh:EVDO:MEASurement<Instance>:MEValuation:TRACe:CDE:QSIGnal:PILot:AVERage
CALCulate:EVDO:MEASurement<Instance>:MEValuation:TRACe:CDE:QSIGnal:PILot:AVERage
```

class Average

Average commands group definition. 3 total commands, 0 Sub-groups, 3 group commands

calculate() → List[RsCmwEvdoMeas.enums.ResultStatus2]

```
# SCPI: CALCulate:EVDO:MEASurement<instance>
↪:MEValuation:TRACe:CDE:QSIGnal:PILot:AVERage
value: List[enums.ResultStatus2] = driver.multiEval.trace.cde.qsignal.pilot.
↪average.calculate()
```

Return values of the code domain error (CDE) I-Signal and Q-Signal bar graphs. The results of the current, average and maximum bar graphs can be retrieved. Only the pilot part of a physical layer subtype 0/1 measurement is retrieved. For RRI part and subtype 2 or 3 measurements, see method RsCmwEvdoMeas.MultiEval.Trace.Cde.Isignal.Rri.Current.fetch etc. 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.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return qsig_pil_aver_cde: float The number of results depends on the physical layer subtype (see method RsCmwEvdoMeas.Configure.MultiEval.plSubtype) : SF=15 for subtype 0/1 and SF=31 for subtypes 2 and 3 (only NAV values) . Range: -70 to 0 dB, Unit: dB

fetch() → List[float]

```
# SCPI: FETCh:EVDO:MEASurement<instance>
↪:MEValuation:TRACe:CDE:QSIGnal:PILot:AVERage
value: List[float] = driver.multiEval.trace.cde.qsignal.pilot.average.fetch()
```

Return values of the code domain error (CDE) I-Signal and Q-Signal bar graphs. The results of the current, average and maximum bar graphs can be retrieved. Only the pilot part of a physical layer subtype 0/1 measurement is retrieved. For RRI part and subtype 2 or 3 measurements, see method RsCmwEvdoMeas.MultiEval.Trace.Cde.Isignal.Rri.Current.fetch etc. 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.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return qsig_pil_aver_cde: float The number of results depends on the physical layer subtype (see method RsCmwEvdoMeas.Configure.MultiEval.plSubtype) : SF=15 for subtype 0/1 and SF=31 for subtypes 2 and 3 (only NAV values) . Range: -70 to 0 dB, Unit: dB

read() → List[float]

```
# SCPI: READ:EVDO:MEASurement<instance>
↪:MEValuation:TRACe:CDE:QSIGnal:PILot:AVERage
value: List[float] = driver.multiEval.trace.cde.qsignal.pilot.average.read()
```

Return values of the code domain error (CDE) I-Signal and Q-Signal bar graphs. The results of the current, average and maximum bar graphs can be retrieved. Only the pilot part of a physical layer subtype 0/1 measurement is retrieved. For RRI part and subtype 2 or 3 measurements, see method RsCmwEvdoMeas.MultiEval.Trace.Cde.Isignal.Rri.Current.fetch etc. 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.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return qsig_pil_aver_cde: float The number of results depends on the physical layer subtype (see method RsCmwEvdoMeas.Configure.MultiEval.plSubtype) : SF=15 for subtype 0/1 and SF=31 for subtypes 2 and 3 (only NAV values) . Range: -70 to 0 dB, Unit: dB

7.4.2.5.2.10 Maximum

SCPI Commands

```
READ:EVDO:MEASurement<Instance>:MEValuation:TRACe:CDE:QSIGnal:PILot:MAXimum
FETCh:EVDO:MEASurement<Instance>:MEValuation:TRACe:CDE:QSIGnal:PILot:MAXimum
CALCulate:EVDO:MEASurement<Instance>:MEValuation:TRACe:CDE:QSIGnal:PILot:MAXimum
```

class Maximum

Maximum commands group definition. 3 total commands, 0 Sub-groups, 3 group commands

calculate() → List[RsCmwEvdoMeas.enums.ResultStatus2]

```
# SCPI: CALCulate:EVDO:MEASurement<instance>
↪:MEValuation:TRACe:CDE:QSIGnal:PILot:MAXimum
value: List[enums.ResultStatus2] = driver.multiEval.trace.cde.qsignal.pilot.
↪maximum.calculate()
```

Return values of the code domain error (CDE) I-Signal and Q-Signal bar graphs. The results of the current, average and maximum bar graphs can be retrieved. Only the pilot part of a physical layer subtype 0/1 measurement is retrieved. For RRI part and subtype 2 or 3 measurements, see method RsCmwEvdoMeas.MultiEval.Trace.Cde.Isignal.Rri.Current.fetch etc. 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.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return qsig_pil_max_cde: float The number of results depends on the physical layer subtype (see method RsCmwEvdoMeas.Configure.MultiEval.plSubtype) : SF=15 for subtype 0/1 and SF=31 for subtypes 2 and 3 (only NAV values) . Range: -70 to 0 dB, Unit: dB

fetch() → List[float]

```
# SCPI: FETCh:EVDO:MEASurement<instance>
↪:MEValuation:TRACe:CDE:QSIGnal:PILot:MAXimum
value: List[float] = driver.multiEval.trace.cde.qsignal.pilot.maximum.fetch()
```

Return values of the code domain error (CDE) I-Signal and Q-Signal bar graphs. The results of the current, average and maximum bar graphs can be retrieved. Only the pilot part of a physical layer subtype 0/1 measurement is retrieved. For RRI part and subtype 2 or 3 measurements, see method RsCmwEvdoMeas.MultiEval.Trace.Cde.Isignal.Rri.Current.fetch etc. 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.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return qsig_pil_max_cde: float The number of results depends on the physical layer subtype (see method RsCmwEvdoMeas.Configure.MultiEval.plSubtype) : SF=15 for subtype 0/1 and SF=31 for subtypes 2 and 3 (only NAV values) . Range: -70 to 0 dB, Unit: dB

read() → List[float]

```
# SCPI: READ:EVDO:MEASurement<instance>
↪:MEValuation:TRACe:CDE:QSIGnal:PILot:MAXimum
value: List[float] = driver.multiEval.trace.cde.qsignal.pilot.maximum.read()
```

Return values of the code domain error (CDE) I-Signal and Q-Signal bar graphs. The results of the current, average and maximum bar graphs can be retrieved. Only the pilot part of a physical layer subtype 0/1 measurement is retrieved. For RRI part and subtype 2 or 3 measurements, see method RsCmwEvdoMeas.MultiEval.Trace.Cde.Isignal.Rri.Current.fetch etc. 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.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return qsig_pil_max_cde: float The number of results depends on the physical layer subtype (see method RsCmwEvdoMeas.Configure.MultiEval.plSubtype) : SF=15 for subtype 0/1 and SF=31 for subtypes 2 and 3 (only NAV values) . Range: -70 to 0 dB, Unit: dB

7.4.2.5.2.11 State

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:TRACe:CDE:QSIGnal:PILot:STATe
```

class State

State commands group definition. 1 total commands, 0 Sub-groups, 1 group commands

fetch() → List[RsCmwEvdoMeas.enums.ResultStateB]

```
# SCPI: FETCh:EVDO:MEASurement<instance>
↪:MEValuation:TRACe:CDE:QSIGnal:PILot:STATe
value: List[enums.ResultStateB] = driver.multiEval.trace.cde.qsignal.pilot.
↪state.fetch()
```

Return the states of the code domain error (CDE) I-Signal and Q-Signal bar graphs. Only the pilot part of a physical layer subtype 0/1 measurement is retrieved. For RRI part and subtype 2 or 3 measurements, see method RsCmwEvdoMeas. MultiEval.Trace.Cde.Qsignal.Rri.State.fetch.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return qsig_pilot_state: NAV | ACTive | INACtive The number of results depends on the physical layer subtype (see method RsCmwEvdoMeas.Configure.MultiEval.plSubtype) : SF=15 for subtype 0/1 and SF=31 for subtypes 2 and 3 (only NAV values) . NAV: No channel available ACTive: Active code channel INACtive: Inactive code channel

7.4.2.5.2.12 Limit

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:TRACe:CDE:QSIGnal:PILot:LIMit
```

class Limit

Limit commands group definition. 1 total commands, 0 Sub-groups, 1 group commands

fetch() → List[float]

```
# SCPI: FETCh:EVDO:MEASurement<instance>
↪:MEValuation:TRACe:CDE:QSIGnal:PILot:LIMit
value: List[float] = driver.multiEval.trace.cde.qsignal.pilot.limit.fetch()
```

Return limit check results for the code domain error (CDE) I-Signal and Q-Signal bar graphs. Only the pilot part of a physical layer subtype 0/1 measurement is retrieved. For RRI part and subtype 2 or 3 measurements, see method RsCmwEvdoMeas.MultiEval.Trace.Cde.Qsignal.Rri.Limit.fetch.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return qsig_pilot_limit: float Return the exceeded limits as float values. The number of results depends on the physical layer subtype (see method RsCmwEvdoMeas.Configure.MultiEval.plSubtype) : SF=15 for subtype 0/1 and SF=31 for subtypes 2 and 3 (only INV values) . Range: -70 dB to 0 dB , Unit: dB

7.4.2.6 Cp

class Cp
Cp commands group definition. 13 total commands, 5 Sub-groups, 0 group commands

Cloning the Group

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

Subgroups

7.4.2.6.1 Current

SCPI Commands

```
READ:EVDO:MEASurement<Instance>:MEValuation:TRACe:CP:CURRent
FETCh:EVDO:MEASurement<Instance>:MEValuation:TRACe:CP:CURRent
CALCulate:EVDO:MEASurement<Instance>:MEValuation:TRACe:CP:CURRent
```

class Current
Current commands group definition. 3 total commands, 0 Sub-groups, 3 group commands

class CalculateStruct
Response structure. Fields:

- Reliability: int: decimal 'Reliability Indicator'
- Rri: enums.ResultStatus2: float RMS channel power values for the physical channels. Depending on the subtype 0/1 or 2, some channels exist or not. If not available, no power measurement is possible therefore NAV is returned. Unit: dB
- Pilot: enums.ResultStatus2: float RMS channel power values for the physical channels. Depending on the subtype 0/1 or 2, some channels exist or not. If not available, no power measurement is possible therefore NAV is returned. Unit: dB
- Ack_Dsc: enums.ResultStatus2: float RMS channel power values for the physical channels. Depending on the subtype 0/1 or 2, some channels exist or not. If not available, no power measurement is possible therefore NAV is returned. Unit: dB
- Aux_Pilot: enums.ResultStatus2: float RMS channel power values for the physical channels. Depending on the subtype 0/1 or 2, some channels exist or not. If not available, no power measurement is possible therefore NAV is returned. Unit: dB
- Drc: enums.ResultStatus2: float RMS channel power values for the physical channels. Depending on the subtype 0/1 or 2, some channels exist or not. If not available, no power measurement is possible therefore NAV is returned. Unit: dB
- Data: enums.ResultStatus2: float RMS channel power values for the physical channels. Depending on the subtype 0/1 or 2, some channels exist or not. If not available, no power measurement is possible therefore NAV is returned. Unit: dB

class ResultData
Response structure. Fields:

- Reliability: int: decimal 'Reliability Indicator'

- Rri: float: float RMS channel power values for the physical channels. Depending on the subtype 0/1 or 2, some channels exist or not. If not available, no power measurement is possible therefore NAV is returned. Unit: dB
- Pilot: float: float RMS channel power values for the physical channels. Depending on the subtype 0/1 or 2, some channels exist or not. If not available, no power measurement is possible therefore NAV is returned. Unit: dB
- Ack_Dsc: float: float RMS channel power values for the physical channels. Depending on the subtype 0/1 or 2, some channels exist or not. If not available, no power measurement is possible therefore NAV is returned. Unit: dB
- Aux_Pilot: float: float RMS channel power values for the physical channels. Depending on the subtype 0/1 or 2, some channels exist or not. If not available, no power measurement is possible therefore NAV is returned. Unit: dB
- Drc: float: float RMS channel power values for the physical channels. Depending on the subtype 0/1 or 2, some channels exist or not. If not available, no power measurement is possible therefore NAV is returned. Unit: dB
- Data: float: float RMS channel power values for the physical channels. Depending on the subtype 0/1 or 2, some channels exist or not. If not available, no power measurement is possible therefore NAV is returned. Unit: dB

calculate() → CalculateStruct

```
# SCPI: CALCulate:EVDO:MEASurement<instance>:MEValuation:TRACe:CP:CURRent
value: CalculateStruct = driver.multiEval.trace.cp.current.calculate()
```

Returns the channel power of the reverse link physical channels of both the I and Q signal. The slots for the pilot and the RRI channel are evaluated within the same measurement slot. 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:EVDO:MEASurement<instance>:MEValuation:TRACe:CP:CURRent
value: ResultData = driver.multiEval.trace.cp.current.fetch()
```

Returns the channel power of the reverse link physical channels of both the I and Q signal. The slots for the pilot and the RRI channel are evaluated within the same measurement slot. 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:EVDO:MEASurement<instance>:MEValuation:TRACe:CP:CURRent
value: ResultData = driver.multiEval.trace.cp.current.read()
```

Returns the channel power of the reverse link physical channels of both the I and Q signal. The slots for the pilot and the RRI channel are evaluated within the same measurement slot. 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.

7.4.2.6.2 Average

SCPI Commands

```
READ:EVDO:MEASurement<Instance>:MEValuation:TRACe:CP:AVERage
FETCh:EVDO:MEASurement<Instance>:MEValuation:TRACe:CP:AVERage
CALCulate:EVDO:MEASurement<Instance>:MEValuation:TRACe:CP:AVERage
```

class Average

Average commands group definition. 3 total commands, 0 Sub-groups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: decimal 'Reliability Indicator'
- Rri: enums.ResultStatus2: float RMS channel power values for the physical channels. Depending on the subtype 0/1 or 2, some channels exist or not. If not available, no power measurement is possible therefore NAV is returned. Unit: dB
- Pilot: enums.ResultStatus2: float RMS channel power values for the physical channels. Depending on the subtype 0/1 or 2, some channels exist or not. If not available, no power measurement is possible therefore NAV is returned. Unit: dB
- Ack_Dsc: enums.ResultStatus2: float RMS channel power values for the physical channels. Depending on the subtype 0/1 or 2, some channels exist or not. If not available, no power measurement is possible therefore NAV is returned. Unit: dB
- Aux_Pilot: enums.ResultStatus2: float RMS channel power values for the physical channels. Depending on the subtype 0/1 or 2, some channels exist or not. If not available, no power measurement is possible therefore NAV is returned. Unit: dB
- Drc: enums.ResultStatus2: float RMS channel power values for the physical channels. Depending on the subtype 0/1 or 2, some channels exist or not. If not available, no power measurement is possible therefore NAV is returned. Unit: dB
- Data: enums.ResultStatus2: float RMS channel power values for the physical channels. Depending on the subtype 0/1 or 2, some channels exist or not. If not available, no power measurement is possible therefore NAV is returned. Unit: dB

class ResultData

Response structure. Fields:

- Reliability: int: decimal 'Reliability Indicator'
- Rri: float: float RMS channel power values for the physical channels. Depending on the subtype 0/1 or 2, some channels exist or not. If not available, no power measurement is possible therefore NAV is returned. Unit: dB
- Pilot: float: float RMS channel power values for the physical channels. Depending on the subtype 0/1 or 2, some channels exist or not. If not available, no power measurement is possible therefore NAV is returned. Unit: dB
- Ack_Dsc: float: float RMS channel power values for the physical channels. Depending on the subtype 0/1 or 2, some channels exist or not. If not available, no power measurement is possible therefore NAV is returned. Unit: dB

- Aux_Pilot: float: float RMS channel power values for the physical channels. Depending on the subtype 0/1 or 2, some channels exist or not. If not available, no power measurement is possible therefore NAV is returned. Unit: dB
- Drc: float: float RMS channel power values for the physical channels. Depending on the subtype 0/1 or 2, some channels exist or not. If not available, no power measurement is possible therefore NAV is returned. Unit: dB
- Data: float: float RMS channel power values for the physical channels. Depending on the subtype 0/1 or 2, some channels exist or not. If not available, no power measurement is possible therefore NAV is returned. Unit: dB

calculate() → CalculateStruct

```
# SCPI: CALCulate:EVDO:MEASurement<instance>:MEValuation:TRACe:CP:AVERage
value: CalculateStruct = driver.multiEval.trace.cp.average.calculate()
```

Returns the channel power of the reverse link physical channels of both the I and Q signal. The slots for the pilot and the RRI channel are evaluated within the same measurement slot. 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:EVDO:MEASurement<instance>:MEValuation:TRACe:CP:AVERage
value: ResultData = driver.multiEval.trace.cp.average.fetch()
```

Returns the channel power of the reverse link physical channels of both the I and Q signal. The slots for the pilot and the RRI channel are evaluated within the same measurement slot. 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:EVDO:MEASurement<instance>:MEValuation:TRACe:CP:AVERage
value: ResultData = driver.multiEval.trace.cp.average.read()
```

Returns the channel power of the reverse link physical channels of both the I and Q signal. The slots for the pilot and the RRI channel are evaluated within the same measurement slot. 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.

7.4.2.6.3 Maximum

SCPI Commands

```
READ:EVDO:MEASurement<Instance>:MEValuation:TRACe:CP:MAXimum
FETCh:EVDO:MEASurement<Instance>:MEValuation:TRACe:CP:MAXimum
CALCulate:EVDO:MEASurement<Instance>:MEValuation:TRACe:CP:MAXimum
```

class Maximum

Maximum commands group definition. 3 total commands, 0 Sub-groups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: decimal 'Reliability Indicator'
- Rri: enums.ResultStatus2: float RMS channel power values for the physical channels. Depending on the subtype 0/1 or 2, some channels exist or not. If not available, no power measurement is possible therefore NAV is returned. Unit: dB
- Pilot: enums.ResultStatus2: float RMS channel power values for the physical channels. Depending on the subtype 0/1 or 2, some channels exist or not. If not available, no power measurement is possible therefore NAV is returned. Unit: dB
- Ack_Dsc: enums.ResultStatus2: float RMS channel power values for the physical channels. Depending on the subtype 0/1 or 2, some channels exist or not. If not available, no power measurement is possible therefore NAV is returned. Unit: dB
- Aux_Pilot: enums.ResultStatus2: float RMS channel power values for the physical channels. Depending on the subtype 0/1 or 2, some channels exist or not. If not available, no power measurement is possible therefore NAV is returned. Unit: dB
- Drc: enums.ResultStatus2: float RMS channel power values for the physical channels. Depending on the subtype 0/1 or 2, some channels exist or not. If not available, no power measurement is possible therefore NAV is returned. Unit: dB
- Data: enums.ResultStatus2: float RMS channel power values for the physical channels. Depending on the subtype 0/1 or 2, some channels exist or not. If not available, no power measurement is possible therefore NAV is returned. Unit: dB

class ResultData

Response structure. Fields:

- Reliability: int: decimal 'Reliability Indicator'
- Rri: float: float RMS channel power values for the physical channels. Depending on the subtype 0/1 or 2, some channels exist or not. If not available, no power measurement is possible therefore NAV is returned. Unit: dB
- Pilot: float: float RMS channel power values for the physical channels. Depending on the subtype 0/1 or 2, some channels exist or not. If not available, no power measurement is possible therefore NAV is returned. Unit: dB
- Ack_Dsc: float: float RMS channel power values for the physical channels. Depending on the subtype 0/1 or 2, some channels exist or not. If not available, no power measurement is possible therefore NAV is returned. Unit: dB
- Aux_Pilot: float: float RMS channel power values for the physical channels. Depending on the subtype 0/1 or 2, some channels exist or not. If not available, no power measurement is possible therefore NAV is returned. Unit: dB

- Drc: float: float RMS channel power values for the physical channels. Depending on the subtype 0/1 or 2, some channels exist or not. If not available, no power measurement is possible therefore NAV is returned. Unit: dB
- Data: float: float RMS channel power values for the physical channels. Depending on the subtype 0/1 or 2, some channels exist or not. If not available, no power measurement is possible therefore NAV is returned. Unit: dB

calculate() → CalculateStruct

```
# SCPI: CALCulate:EVDO:MEASurement<instance>:MEValuation:TRACe:CP:MAXimum
value: CalculateStruct = driver.multiEval.trace.cp.maximum.calculate()
```

Returns the channel power of the reverse link physical channels of both the I and Q signal. The slots for the pilot and the RRI channel are evaluated within the same measurement slot. 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:EVDO:MEASurement<instance>:MEValuation:TRACe:CP:MAXimum
value: ResultData = driver.multiEval.trace.cp.maximum.fetch()
```

Returns the channel power of the reverse link physical channels of both the I and Q signal. The slots for the pilot and the RRI channel are evaluated within the same measurement slot. 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:EVDO:MEASurement<instance>:MEValuation:TRACe:CP:MAXimum
value: ResultData = driver.multiEval.trace.cp.maximum.read()
```

Returns the channel power of the reverse link physical channels of both the I and Q signal. The slots for the pilot and the RRI channel are evaluated within the same measurement slot. 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.

7.4.2.6.4 Minimum

SCPI Commands

```
READ:EVDO:MEASurement<Instance>:MEValuation:TRACe:CP:MINimum
FETCh:EVDO:MEASurement<Instance>:MEValuation:TRACe:CP:MINimum
CALCulate:EVDO:MEASurement<Instance>:MEValuation:TRACe:CP:MINimum
```

class Minimum

Minimum commands group definition. 3 total commands, 0 Sub-groups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: decimal 'Reliability Indicator'
- Rri: enums.ResultStatus2: float RMS channel power values for the physical channels. Depending on the subtype 0/1 or 2, some channels exist or not. If not available, no power measurement is possible therefore NAV is returned. Unit: dB
- Pilot: enums.ResultStatus2: float RMS channel power values for the physical channels. Depending on the subtype 0/1 or 2, some channels exist or not. If not available, no power measurement is possible therefore NAV is returned. Unit: dB
- Ack_Dsc: enums.ResultStatus2: float RMS channel power values for the physical channels. Depending on the subtype 0/1 or 2, some channels exist or not. If not available, no power measurement is possible therefore NAV is returned. Unit: dB
- Aux_Pilot: enums.ResultStatus2: float RMS channel power values for the physical channels. Depending on the subtype 0/1 or 2, some channels exist or not. If not available, no power measurement is possible therefore NAV is returned. Unit: dB
- Drc: enums.ResultStatus2: float RMS channel power values for the physical channels. Depending on the subtype 0/1 or 2, some channels exist or not. If not available, no power measurement is possible therefore NAV is returned. Unit: dB
- Data: enums.ResultStatus2: float RMS channel power values for the physical channels. Depending on the subtype 0/1 or 2, some channels exist or not. If not available, no power measurement is possible therefore NAV is returned. Unit: dB

class ResultData

Response structure. Fields:

- Reliability: int: decimal 'Reliability Indicator'
- Rri: float: float RMS channel power values for the physical channels. Depending on the subtype 0/1 or 2, some channels exist or not. If not available, no power measurement is possible therefore NAV is returned. Unit: dB
- Pilot: float: float RMS channel power values for the physical channels. Depending on the subtype 0/1 or 2, some channels exist or not. If not available, no power measurement is possible therefore NAV is returned. Unit: dB
- Ack_Dsc: float: float RMS channel power values for the physical channels. Depending on the subtype 0/1 or 2, some channels exist or not. If not available, no power measurement is possible therefore NAV is returned. Unit: dB
- Aux_Pilot: float: float RMS channel power values for the physical channels. Depending on the subtype 0/1 or 2, some channels exist or not. If not available, no power measurement is possible therefore NAV is returned. Unit: dB
- Drc: float: float RMS channel power values for the physical channels. Depending on the subtype 0/1 or 2, some channels exist or not. If not available, no power measurement is possible therefore NAV is returned. Unit: dB
- Data: float: float RMS channel power values for the physical channels. Depending on the subtype 0/1 or 2, some channels exist or not. If not available, no power measurement is possible therefore NAV is returned. Unit: dB

calculate() → CalculateStruct

```
# SCPI: CALCulate:EVDO:MEASurement<instance>:MEValuation:TRACe:CP:MINimum
value: CalculateStruct = driver.multiEval.trace.cp.minimum.calculate()
```

Returns the channel power of the reverse link physical channels of both the I and Q signal. The slots for the pilot and the RRI channel are evaluated within the same measurement slot. 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:EVDO:MEASurement<instance>:MEValuation:TRACe:CP:MINimum
value: ResultData = driver.multiEval.trace.cp.minimum.fetch()
```

Returns the channel power of the reverse link physical channels of both the I and Q signal. The slots for the pilot and the RRI channel are evaluated within the same measurement slot. 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:EVDO:MEASurement<instance>:MEValuation:TRACe:CP:MINimum
value: ResultData = driver.multiEval.trace.cp.minimum.read()
```

Returns the channel power of the reverse link physical channels of both the I and Q signal. The slots for the pilot and the RRI channel are evaluated within the same measurement slot. 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.

7.4.2.6.5 State

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:TRACe:CP:STATe
```

class State

State commands group definition. 1 total commands, 0 Sub-groups, 1 group commands

class FetchStruct

Response structure. Fields:

- Reliability: int: decimal 'Reliability Indicator'
- Rri: enums.ResultStateA: INVisible | ACTive | IACTive The number of results depends on the physical layer subtype (see [CMDLINK: CONFigure:EVDO:MEASi:MEValuation:PLSubtype CMDLINK]) : SF=15 for subtype 0/1 and SF=31 for subtypes 2 and 3 (only NAV values) . INV: No channel power available ACTive: Active channel power INACtive: Inactive channel power
- Pilot: enums.ResultStateA: INVisible | ACTive | IACTive The number of results depends on the physical layer subtype (see [CMDLINK: CONFigure:EVDO:MEASi:MEValuation:PLSubtype

CMDLINK]) : SF=15 for subtype 0/1 and SF=31 for subtypes 2 and 3 (only NAV values) . INV: No channel power available ACTive: Active channel power INACtive: Inactive channel power

- Ack_Dsc: enums.ResultStateA: INVisible | ACTive | IACTive The number of results depends on the physical layer subtype (see [CMDLINK: CONFigure:EVDO:MEASi:MEValuation:PLSubtype CMDLINK]) : SF=15 for subtype 0/1 and SF=31 for subtypes 2 and 3 (only NAV values) . INV: No channel power available ACTive: Active channel power INACtive: Inactive channel power
- Aux_Pilot: enums.ResultStateA: INVisible | ACTive | IACTive The number of results depends on the physical layer subtype (see [CMDLINK: CONFigure:EVDO:MEASi:MEValuation:PLSubtype CMDLINK]) : SF=15 for subtype 0/1 and SF=31 for subtypes 2 and 3 (only NAV values) . INV: No channel power available ACTive: Active channel power INACtive: Inactive channel power
- Drc: enums.ResultStateA: INVisible | ACTive | IACTive The number of results depends on the physical layer subtype (see [CMDLINK: CONFigure:EVDO:MEASi:MEValuation:PLSubtype CMDLINK]) : SF=15 for subtype 0/1 and SF=31 for subtypes 2 and 3 (only NAV values) . INV: No channel power available ACTive: Active channel power INACtive: Inactive channel power
- Data: enums.ResultStateA: INVisible | ACTive | IACTive The number of results depends on the physical layer subtype (see [CMDLINK: CONFigure:EVDO:MEASi:MEValuation:PLSubtype CMDLINK]) : SF=15 for subtype 0/1 and SF=31 for subtypes 2 and 3 (only NAV values) . INV: No channel power available ACTive: Active channel power INACtive: Inactive channel power

fetch() → FetchStruct

```
# SCPI: FETCh:EVDO:MEASurement<instance>:MEValuation:TRACe:CP:STATe
value: FetchStruct = driver.multiEval.trace.cp.state.fetch()
```

No command help available

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

7.4.2.7 Acp

class Acp

Acp commands group definition. 36 total commands, 4 Sub-groups, 0 group commands

Cloning the Group

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

Subgroups

7.4.2.7.1 Current

class Current

Current commands group definition. 6 total commands, 2 Sub-groups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.multiEval.trace.acp.current.clone()
```

Subgroups

7.4.2.7.1.1 Relative

SCPI Commands

```
READ:EVDO:MEASurement<Instance>:MEValuation:TRACe:ACP:CURRent:RELative
FETCh:EVDO:MEASurement<Instance>:MEValuation:TRACe:ACP:CURRent:RELative
CALCulate:EVDO:MEASurement<Instance>:MEValuation:TRACe:ACP:CURRent:RELative
```

class Relative

Relative commands group definition. 3 total commands, 0 Sub-groups, 3 group commands

calculate() → List[float]

```
# SCPI: CALCulate:EVDO:MEASurement<instance>
↪:MEValuation:TRACe:ACP:CURRent[:RELative]
value: List[float] = driver.multiEval.trace.acp.current.relative.calculate()
```

Returns the adjacent channel power measured at a series of frequencies. The current, average and maximum traces can be retrieved. The frequencies are determined by the offset values defined via the commands method RsCmwEvdoMeas.Configure. MultiEval.Acp.Foffsets.lower and method RsCmwEvdoMeas.Configure.MultiEval.Acp.Foffsets.upper. All defined offset values are considered (irrespective of their activation status) . 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.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return curr_acp: No help available

fetch() → List[float]

```
# SCPI: FETCh:EVDO:MEASurement<instance>
↪:MEValuation:TRACe:ACP:CURRent[:RELative]
value: List[float] = driver.multiEval.trace.acp.current.relative.fetch()
```

Returns the adjacent channel power measured at a series of frequencies. The current, average and maximum traces can be retrieved. The frequencies are determined by the offset values defined via the commands method RsCmwEvdoMeas.Configure. MultiEval.Acp.Foffsets.lower and method RsCmwEvdoMeas.Configure.MultiEval.Acp.Foffsets.upper. All defined offset values are considered (irrespective of their activation status) . 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.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return curr_acp: No help available

read() → List[float]

```
# SCPI: READ:EVDO:MEASurement<instance>:MEValuation:TRACe:ACP:CURRent[:RELative]
value: List[float] = driver.multiEval.trace.acp.current.relative.read()
```

Returns the adjacent channel power measured at a series of frequencies. The current, average and maximum traces can be retrieved. The frequencies are determined by the offset values defined via the commands method RsCmwEvdoMeas.Configure. MultiEval.Acp.Foffsets.lower and method RsCmwEvdoMeas.Configure.MultiEval.Acp.Foffsets.upper. All defined offset values are considered (irrespective of their activation status) . 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.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return curr_acp: No help available

7.4.2.7.1.2 Absolute

SCPI Commands

```
READ:EVDO:MEASurement<Instance>:MEValuation:TRACe:ACP:CURRent:ABSolute
FETCh:EVDO:MEASurement<Instance>:MEValuation:TRACe:ACP:CURRent:ABSolute
CALCulate:EVDO:MEASurement<Instance>:MEValuation:TRACe:ACP:CURRent:ABSolute
```

class Absolute

Absolute commands group definition. 3 total commands, 0 Sub-groups, 3 group commands

calculate() → List[float]

```
# SCPI: CALCulate:EVDO:MEASurement<instance>
↪:MEValuation:TRACe:ACP:CURRent:ABSolute
value: List[float] = driver.multiEval.trace.acp.current.absolute.calculate()
```

Returns the adjacent channel absolute power measured in dBm at a series of frequencies. The frequencies are determined by the offset values defined via the commands method RsCmwEvdoMeas.Configure.MultiEval.Acp.Foffsets.lower and method RsCmwEvdoMeas.Configure.MultiEval.Acp.Foffsets.upper. All defined offset values are considered (irrespective of their activation status) . The current, average and maximum traces can be retrieved.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return curr_acp: No help available

fetch() → List[float]

```
# SCPI: FETCh:EVDO:MEASurement<instance>:MEValuation:TRACe:ACP:CURRent:ABSolute
value: List[float] = driver.multiEval.trace.acp.current.absolute.fetch()
```

Returns the adjacent channel absolute power measured in dBm at a series of frequencies. The frequencies are determined by the offset values defined via the commands method RsCmwEvdoMeas.Configure.MultiEval.Acp.Foffsets.lower and method RsCmwEvdoMeas.Configure.MultiEval.Acp.Foffsets.upper. All defined offset values are considered (irrespective of their activation status) . The current, average and maximum traces can be retrieved.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return curr_acp: No help available

read() → List[float]

```
# SCPI: READ:EVDO:MEASurement<instance>:MEValuation:TRACe:ACP:CURRent:ABSolute
value: List[float] = driver.multiEval.trace.acp.current.absolute.read()
```

Returns the adjacent channel absolute power measured in dBm at a series of frequencies. The frequencies are determined by the offset values defined via the commands method RsCmwEvdoMeas.Configure.MultiEval.Acp.Foffsets.lower and method RsCmwEvdoMeas.Configure.MultiEval.Acp.Foffsets.upper. All defined offset values are considered (irrespective of their activation status) . The current, average and maximum traces can be retrieved.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return curr_acp: No help available

7.4.2.7.2 Extended

class Extended

Extended commands group definition. 18 total commands, 3 Sub-groups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.multiEval.trace.acp.extended.clone()
```

Subgroups

7.4.2.7.2.1 Current

class Current

Current commands group definition. 6 total commands, 2 Sub-groups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.multiEval.trace.acp.extended.current.clone()
```

Subgroups

7.4.2.7.2.2 Relative

SCPI Commands

```
READ:EVDO:MEASurement<Instance>:MEValuation:TRACe:ACP:EXTended:CURRent:RELative
FETCh:EVDO:MEASurement<Instance>:MEValuation:TRACe:ACP:EXTended:CURRent:RELative
CALCulate:EVDO:MEASurement<Instance>:MEValuation:TRACe:ACP:EXTended:CURRent:RELative
```

class Relative

Relative commands group definition. 3 total commands, 0 Sub-groups, 3 group commands

calculate() → List[float]

```
# SCPI: CALCulate:EVDO:MEASurement<instance>
↪:MEValuation:TRACe:ACP:EXTended:CURRent[:RELative]
value: List[float] = driver.multiEval.trace.acp.extended.current.relative.
↪calculate()
```

Returns the adjacent channel relative power measured in dBc at a series of frequencies. The current, average and maximum traces can be retrieved. The frequencies are determined by the offset values defined via the commands method RsCmwEvdoMeas.Configure.MultiEval.Acp.Foffsets.lower and method RsCmwEvdoMeas.Configure.MultiEval.Acp.Foffsets.upper. All defined offset values are considered (irrespective of their activation status) . 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.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return curr_acp: No help available

fetch() → List[float]

```
# SCPI: FETCh:EVDO:MEASurement<instance>
↪:MEValuation:TRACe:ACP:EXTended:CURRent[:RELative]
value: List[float] = driver.multiEval.trace.acp.extended.current.relative.
↪fetch()
```

Returns the adjacent channel relative power measured in dBc at a series of frequencies. The current, average and maximum traces can be retrieved. The frequencies are determined by the offset values defined via the commands method RsCmwEvdoMeas.Configure.MultiEval.Acp.Foffsets.lower and method RsCmwEvdoMeas.Configure.MultiEval.Acp.Foffsets.upper. All defined offset values are considered (irrespective of their activation status) . 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.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return curr_acp: No help available

read() → List[float]

```
# SCPI: READ:EVDO:MEASurement<instance>
↪:MEValuation:TRACe:ACP:EXTended:CURRent[:RELative]
value: List[float] = driver.multiEval.trace.acp.extended.current.relative.read()
```

Returns the adjacent channel relative power measured in dBc at a series of frequencies. The current, average and maximum traces can be retrieved. The frequencies are determined by the offset values defined via the commands method RsCmwEvdoMeas.Configure.MultiEval.Acp.Foffsets.lower and method RsCmwEvdoMeas.Configure.MultiEval.Acp.Foffsets.upper. All defined offset values are considered (irrespective of their activation status) . 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.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return curr_acp: No help available

7.4.2.7.2.3 Absolute

SCPI Commands

```
READ:EVDO:MEASurement<Instance>:MEValuation:TRACe:ACP:EXTended:CURRent:ABSolute
FETCh:EVDO:MEASurement<Instance>:MEValuation:TRACe:ACP:EXTended:CURRent:ABSolute
CALCulate:EVDO:MEASurement<Instance>:MEValuation:TRACe:ACP:EXTended:CURRent:ABSolute
```

class Absolute

Absolute commands group definition. 3 total commands, 0 Sub-groups, 3 group commands

calculate() → List[float]

```
# SCPI: CALCulate:EVDO:MEASurement<instance>
↪:MEValuation:TRACe:ACP:EXTended:CURRent:ABSolute
value: List[float] = driver.multiEval.trace.acp.extended.current.absolute.
↪calculate()
```

Returns the adjacent channel absolute power measured in dBm at a series of frequencies. The current, average and maximum traces can be retrieved. The frequencies are determined by the offset values defined via the commands method RsCmwEvdoMeas.Configure.MultiEval.Acp.Foffsets.lower and method RsCmwEvdoMeas.Configure.MultiEval.Acp.Foffsets.upper. All defined offset values are considered (irrespective of their activation status) . 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.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return curr_acp: No help available

fetch() → List[float]

```
# SCPI: FETCh:EVDO:MEASurement<instance>
↪:MEValuation:TRACe:ACP:EXTended:CURRent:ABSolute
value: List[float] = driver.multiEval.trace.acp.extended.current.absolute.
↪fetch()
```

Returns the adjacent channel absolute power measured in dBm at a series of frequencies. The current, average and maximum traces can be retrieved. The frequencies are determined by the offset values defined via the commands method RsCmwEvdoMeas.Configure.MultiEval.Acp.Foffsets.lower and method RsCmwEvdoMeas.Configure.MultiEval.Acp.Foffsets.upper. All defined offset values are considered (irrespective of their activation status) . 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.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return curr_acp: No help available

read() → List[float]

```
# SCPI: READ:EVDO:MEASurement<instance>
↪:MEValuation:TRACe:ACP:EXTended:CURRent:ABSolute
value: List[float] = driver.multiEval.trace.acp.extended.current.absolute.read()
```

Returns the adjacent channel absolute power measured in dBm at a series of frequencies. The current, average and maximum traces can be retrieved. The frequencies are determined by the offset values defined via the commands method RsCmwEvdoMeas.Configure.MultiEval.Acp.Foffsets.lower and method RsCmwEvdoMeas.Configure.MultiEval.Acp.Foffsets.upper. All defined offset values are considered (irrespective of their activation status) . 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.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return curr_acp: No help available

7.4.2.7.2.4 Average

class Average

Average commands group definition. 6 total commands, 2 Sub-groups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.multiEval.trace.acp.extended.average.clone()
```

Subgroups

7.4.2.7.2.5 Relative

SCPI Commands

```
READ:EVDO:MEASurement<Instance>:MEValuation:TRACe:ACP:EXTended:AVERage:RELative
FETCh:EVDO:MEASurement<Instance>:MEValuation:TRACe:ACP:EXTended:AVERage:RELative
CALCulate:EVDO:MEASurement<Instance>:MEValuation:TRACe:ACP:EXTended:AVERage:RELative
```

class Relative

Relative commands group definition. 3 total commands, 0 Sub-groups, 3 group commands

calculate() → List[float]

```
# SCPI: CALCulate:EVDO:MEASurement<instance>
↪:MEValuation:TRACe:ACP:EXTended:AVERage[:RELative]
value: List[float] = driver.multiEval.trace.acp.extended.average.relative.
↪calculate()
```

Returns the adjacent channel relative power measured in dBc at a series of frequencies. The current, average and maximum traces can be retrieved. The frequencies are determined by the offset values defined via the commands method RsCmwEvdoMeas.Configure.MultiEval.Acp.Foffsets.lower and method RsCmwEvdoMeas.Configure.MultiEval.Acp.Foffsets.upper. All defined offset values are considered (irrespective

of their activation status) . 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.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return aver_acp: No help available

fetch() → List[float]

```
# SCPI: FETCh:EVDO:MEASurement<instance>
↪:MEValuation:TRACe:ACP:EXTended:AVERage[:RELative]
value: List[float] = driver.multiEval.trace.acp.extended.average.relative.
↪fetch()
```

Returns the adjacent channel relative power measured in dBc at a series of frequencies. The current, average and maximum traces can be retrieved. The frequencies are determined by the offset values defined via the commands method RsCmwEvdoMeas.Configure.MultiEval.Acp.Foffsets.lower and method RsCmwEvdoMeas.Configure.MultiEval.Acp.Foffsets.upper. All defined offset values are considered (irrespective of their activation status) . 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.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return aver_acp: No help available

read() → List[float]

```
# SCPI: READ:EVDO:MEASurement<instance>
↪:MEValuation:TRACe:ACP:EXTended:AVERage[:RELative]
value: List[float] = driver.multiEval.trace.acp.extended.average.relative.read()
```

Returns the adjacent channel relative power measured in dBc at a series of frequencies. The current, average and maximum traces can be retrieved. The frequencies are determined by the offset values defined via the commands method RsCmwEvdoMeas.Configure.MultiEval.Acp.Foffsets.lower and method RsCmwEvdoMeas.Configure.MultiEval.Acp.Foffsets.upper. All defined offset values are considered (irrespective of their activation status) . 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.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return aver_acp: No help available

7.4.2.7.2.6 Absolute

SCPI Commands

```
READ:EVDO:MEASurement<Instance>:MEValuation:TRACe:ACP:EXTended:AVERage:ABSolute
FETCh:EVDO:MEASurement<Instance>:MEValuation:TRACe:ACP:EXTended:AVERage:ABSolute
CALCulate:EVDO:MEASurement<Instance>:MEValuation:TRACe:ACP:EXTended:AVERage:ABSolute
```

class Absolute

Absolute commands group definition. 3 total commands, 0 Sub-groups, 3 group commands

calculate() → List[float]

```
# SCPI: CALCulate:EVDO:MEASurement<instance>
↪:MEValuation:TRACe:ACP:EXTended:AVERage:ABSolute
value: List[float] = driver.multiEval.trace.acp.extended.average.absolute.
↪calculate()
```

Returns the adjacent channel absolute power measured in dBm at a series of frequencies. The current, average and maximum traces can be retrieved. The frequencies are determined by the offset values defined via the commands method RsCmwEvdoMeas.Configure.MultiEval.Acp.Foffsets.lower and method RsCmwEvdoMeas.Configure.MultiEval.Acp.Foffsets.upper. All defined offset values are considered (irrespective of their activation status) . 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.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return aver_acp: No help available

fetch() → List[float]

```
# SCPI: FETCh:EVDO:MEASurement<instance>
↪:MEValuation:TRACe:ACP:EXTended:AVERage:ABSolute
value: List[float] = driver.multiEval.trace.acp.extended.average.absolute.
↪fetch()
```

Returns the adjacent channel absolute power measured in dBm at a series of frequencies. The current, average and maximum traces can be retrieved. The frequencies are determined by the offset values defined via the commands method RsCmwEvdoMeas.Configure.MultiEval.Acp.Foffsets.lower and method RsCmwEvdoMeas.Configure.MultiEval.Acp.Foffsets.upper. All defined offset values are considered (irrespective of their activation status) . 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.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return aver_acp: No help available

read() → List[float]

```
# SCPI: READ:EVDO:MEASurement<instance>
↪:MEValuation:TRACe:ACP:EXTended:AVERage:ABSolute
value: List[float] = driver.multiEval.trace.acp.extended.average.absolute.read()
```

Returns the adjacent channel absolute power measured in dBm at a series of frequencies. The current, average and maximum traces can be retrieved. The frequencies are determined by the offset values defined via the commands method RsCmwEvdoMeas.Configure.MultiEval.Acp.Foffsets.lower and method RsCmwEvdoMeas.Configure.MultiEval.Acp.Foffsets.upper. All defined offset values are considered (irrespective of their activation status) . 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.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return aver_acp: No help available

7.4.2.7.2.7 Maximum

class Maximum
Maximum commands group definition. 6 total commands, 2 Sub-groups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.multiEval.trace.acp.extended.maximum.clone()
```

Subgroups

7.4.2.7.2.8 Relative

SCPI Commands

```
READ:EVDO:MEASurement<Instance>:MEValuation:TRACe:ACP:EXTended:MAXimum:RELative
FETCh:EVDO:MEASurement<Instance>:MEValuation:TRACe:ACP:EXTended:MAXimum:RELative
CALCulate:EVDO:MEASurement<Instance>:MEValuation:TRACe:ACP:EXTended:MAXimum:RELative
```

class Relative
Relative commands group definition. 3 total commands, 0 Sub-groups, 3 group commands

calculate() → List[float]

```
# SCPI: CALCulate:EVDO:MEASurement<instance>
↪:MEValuation:TRACe:ACP:EXTended:MAXimum[:RELative]
value: List[float] = driver.multiEval.trace.acp.extended.maximum.relative.
↪calculate()
```

Returns the adjacent channel relative power measured in dBc at a series of frequencies. The current, average and maximum traces can be retrieved. The frequencies are determined by the offset values defined via the commands method RsCmwEvdoMeas.Configure.MultiEval.Acp.Foffsets.lower and method RsCmwEvdoMeas.Configure.MultiEval.Acp.Foffsets.upper. All defined offset values are considered (irrespective of their activation status) . 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.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return max_acp: No help available

fetch() → List[float]

```
# SCPI: FETCh:EVDO:MEASurement<instance>
↪:MEValuation:TRACe:ACP:EXTended:MAXimum[:RELative]
value: List[float] = driver.multiEval.trace.acp.extended.maximum.relative.
↪fetch()
```

Returns the adjacent channel relative power measured in dBc at a series of frequencies. The current, average and maximum traces can be retrieved. The frequencies are determined by the offset values defined via

the commands method RsCmwEvdoMeas.Configure.MultiEval.Acp.Foffsets.lower and method RsCmwEvdoMeas.Configure.MultiEval.Acp.Foffsets.upper. All defined offset values are considered (irrespective of their activation status) . 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.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return max_acp: No help available

read() → List[float]

```
# SCPI: READ:EVDO:MEASurement<instance>
↪:MEValuation:TRACe:ACP:EXTended:MAXimum[:RELative]
value: List[float] = driver.multiEval.trace.acp.extended.maximum.relative.read()
```

Returns the adjacent channel relative power measured in dBc at a series of frequencies. The current, average and maximum traces can be retrieved. The frequencies are determined by the offset values defined via the commands method RsCmwEvdoMeas.Configure.MultiEval.Acp.Foffsets.lower and method RsCmwEvdoMeas.Configure.MultiEval.Acp.Foffsets.upper. All defined offset values are considered (irrespective of their activation status) . 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.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return max_acp: No help available

7.4.2.7.2.9 Absolute

SCPI Commands

```
READ:EVDO:MEASurement<Instance>:MEValuation:TRACe:ACP:EXTended:MAXimum:ABSolute
FETCh:EVDO:MEASurement<Instance>:MEValuation:TRACe:ACP:EXTended:MAXimum:ABSolute
CALCulate:EVDO:MEASurement<Instance>:MEValuation:TRACe:ACP:EXTended:MAXimum:ABSolute
```

class Absolute

Absolute commands group definition. 3 total commands, 0 Sub-groups, 3 group commands

calculate() → List[float]

```
# SCPI: CALCulate:EVDO:MEASurement<instance>
↪:MEValuation:TRACe:ACP:EXTended:MAXimum:ABSolute
value: List[float] = driver.multiEval.trace.acp.extended.maximum.absolute.
↪calculate()
```

Returns the adjacent channel absolute power measured in dBm at a series of frequencies. The current, average and maximum traces can be retrieved. The frequencies are determined by the offset values defined via the commands method RsCmwEvdoMeas.Configure.MultiEval.Acp.Foffsets.lower and method RsCmwEvdoMeas.Configure.MultiEval.Acp.Foffsets.upper. All defined offset values are considered (irrespective of their activation status) . 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.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return max_acp: No help available

fetch() → List[float]

```
# SCPI: FETCh:EVDO:MEASurement<instance>
↪:MEValuation:TRACe:ACP:EXTended:MAXimum:ABSolute
value: List[float] = driver.multiEval.trace.acp.extended.maximum.absolute.
↪fetch()
```

Returns the adjacent channel absolute power measured in dBm at a series of frequencies. The current, average and maximum traces can be retrieved. The frequencies are determined by the offset values defined via the commands method RsCmwEvdoMeas.Configure.MultiEval.Acp.Foffsets.lower and method RsCmwEvdoMeas.Configure.MultiEval.Acp.Foffsets.upper. All defined offset values are considered (irrespective of their activation status) . 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.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return max_acp: No help available

read() → List[float]

```
# SCPI: READ:EVDO:MEASurement<instance>
↪:MEValuation:TRACe:ACP:EXTended:MAXimum:ABSolute
value: List[float] = driver.multiEval.trace.acp.extended.maximum.absolute.read()
```

Returns the adjacent channel absolute power measured in dBm at a series of frequencies. The current, average and maximum traces can be retrieved. The frequencies are determined by the offset values defined via the commands method RsCmwEvdoMeas.Configure.MultiEval.Acp.Foffsets.lower and method RsCmwEvdoMeas.Configure.MultiEval.Acp.Foffsets.upper. All defined offset values are considered (irrespective of their activation status) . 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.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return max_acp: No help available

7.4.2.7.3 Average

class Average

Average commands group definition. 6 total commands, 2 Sub-groups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.multiEval.trace.acp.average.clone()
```

Subgroups

7.4.2.7.3.1 Relative

SCPI Commands

```
READ:EVDO:MEASurement<Instance>:MEValuation:TRACe:ACP:AVERage:RELative
FETCh:EVDO:MEASurement<Instance>:MEValuation:TRACe:ACP:AVERage:RELative
CALCulate:EVDO:MEASurement<Instance>:MEValuation:TRACe:ACP:AVERage:RELative
```

class Relative

Relative commands group definition. 3 total commands, 0 Sub-groups, 3 group commands

calculate() → List[float]

```
# SCPI: CALCulate:EVDO:MEASurement<instance>
↪:MEValuation:TRACe:ACP:AVERage[:RELative]
value: List[float] = driver.multiEval.trace.acp.average.relative.calculate()
```

Returns the adjacent channel power measured at a series of frequencies. The current, average and maximum traces can be retrieved. The frequencies are determined by the offset values defined via the commands method RsCmwEvdoMeas.Configure. MultiEval.Acp.Foffsets.lower and method RsCmwEvdoMeas.Configure.MultiEval.Acp.Foffsets.upper. All defined offset values are considered (irrespective of their activation status) . 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.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return aver_acp: No help available

fetch() → List[float]

```
# SCPI: FETCh:EVDO:MEASurement<instance>
↪:MEValuation:TRACe:ACP:AVERage[:RELative]
value: List[float] = driver.multiEval.trace.acp.average.relative.fetch()
```

Returns the adjacent channel power measured at a series of frequencies. The current, average and maximum traces can be retrieved. The frequencies are determined by the offset values defined via the commands method RsCmwEvdoMeas.Configure. MultiEval.Acp.Foffsets.lower and method RsCmwEvdoMeas.Configure.MultiEval.Acp.Foffsets.upper. All defined offset values are considered (irrespective of their activation status) . 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.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return aver_acp: No help available

read() → List[float]

```
# SCPI: READ:EVDO:MEASurement<instance>:MEValuation:TRACe:ACP:AVERage[:RELative]
value: List[float] = driver.multiEval.trace.acp.average.relative.read()
```

Returns the adjacent channel power measured at a series of frequencies. The current, average and maximum traces can be retrieved. The frequencies are determined by the offset values defined via the

commands method RsCmwEvdoMeas.Configure. MultiEval.Acp.Foffsets.lower and method RsCmwEvdoMeas.Configure.MultiEval.Acp.Foffsets.upper. All defined offset values are considered (irrespective of their activation status) . 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.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return aver_acp: No help available

7.4.2.7.3.2 Absolute

SCPI Commands

```
READ:EVDO:MEASurement<Instance>:MEValuation:TRACe:ACP:AVERage:ABSolute
FETCh:EVDO:MEASurement<Instance>:MEValuation:TRACe:ACP:AVERage:ABSolute
CALCulate:EVDO:MEASurement<Instance>:MEValuation:TRACe:ACP:AVERage:ABSolute
```

class Absolute

Absolute commands group definition. 3 total commands, 0 Sub-groups, 3 group commands

calculate() → List[float]

```
# SCPI: CALCulate:EVDO:MEASurement<instance>
↪:MEValuation:TRACe:ACP:AVERage:ABSolute
value: List[float] = driver.multiEval.trace.acp.average.absolute.calculate()
```

Returns the adjacent channel absolute power measured in dBm at a series of frequencies. The frequencies are determined by the offset values defined via the commands method RsCmwEvdoMeas.Configure.MultiEval.Acp.Foffsets.lower and method RsCmwEvdoMeas.Configure.MultiEval.Acp.Foffsets.upper. All defined offset values are considered (irrespective of their activation status) . The current, average and maximum traces can be retrieved.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return aver_acp: No help available

fetch() → List[float]

```
# SCPI: FETCh:EVDO:MEASurement<instance>:MEValuation:TRACe:ACP:AVERage:ABSolute
value: List[float] = driver.multiEval.trace.acp.average.absolute.fetch()
```

Returns the adjacent channel absolute power measured in dBm at a series of frequencies. The frequencies are determined by the offset values defined via the commands method RsCmwEvdoMeas.Configure.MultiEval.Acp.Foffsets.lower and method RsCmwEvdoMeas.Configure.MultiEval.Acp.Foffsets.upper. All defined offset values are considered (irrespective of their activation status) . The current, average and maximum traces can be retrieved.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return aver_acp: No help available

read() → List[float]

```
# SCPI: READ:EVDO:MEASurement<instance>:MEValuation:TRACe:ACP:AVERage:ABSolute
value: List[float] = driver.multiEval.trace.acp.average.absolute.read()
```

Returns the adjacent channel absolute power measured in dBm at a series of frequencies. The frequencies are determined by the offset values defined via the commands method RsCmwEvdoMeas.Configure.MultiEval.Acp.Foffsets.lower and method RsCmwEvdoMeas.Configure.MultiEval.Acp.Foffsets.upper. All defined offset values are considered (irrespective of their activation status) . The current, average and maximum traces can be retrieved.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return aver_acp: No help available

7.4.2.7.4 Maximum

class Maximum

Maximum commands group definition. 6 total commands, 2 Sub-groups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.multiEval.trace.acp.maximum.clone()
```

Subgroups

7.4.2.7.4.1 Relative

SCPI Commands

```
READ:EVDO:MEASurement<Instance>:MEValuation:TRACe:ACP:MAXimum:RELative
FETCh:EVDO:MEASurement<Instance>:MEValuation:TRACe:ACP:MAXimum:RELative
CALCulate:EVDO:MEASurement<Instance>:MEValuation:TRACe:ACP:MAXimum:RELative
```

class Relative

Relative commands group definition. 3 total commands, 0 Sub-groups, 3 group commands

calculate() → List[float]

```
# SCPI: CALCulate:EVDO:MEASurement<instance>
↪:MEValuation:TRACe:ACP:MAXimum[:RELative]
value: List[float] = driver.multiEval.trace.acp.maximum.relative.calculate()
```

Returns the adjacent channel power measured at a series of frequencies. The current, average and maximum traces can be retrieved. The frequencies are determined by the offset values defined via the commands method RsCmwEvdoMeas.Configure. MultiEval.Acp.Foffsets.lower and method RsCmwEvdoMeas.Configure.MultiEval.Acp.Foffsets.upper. All defined offset values are considered (irrespective of their activation status) . 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.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return max_acp: No help available

fetch() → List[float]

```
# SCPI: FETCh:EVDO:MEASurement<instance>
↪:MEValuation:TRACe:ACP:MAXimum[:RELative]
value: List[float] = driver.multiEval.trace.acp.maximum.relative.fetch()
```

Returns the adjacent channel power measured at a series of frequencies. The current, average and maximum traces can be retrieved. The frequencies are determined by the offset values defined via the commands method RsCmwEvdoMeas.Configure. MultiEval.Acp.Foffsets.lower and method RsCmwEvdoMeas.Configure.MultiEval.Acp.Foffsets.upper. All defined offset values are considered (irrespective of their activation status) . 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.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return max_acp: No help available

read() → List[float]

```
# SCPI: READ:EVDO:MEASurement<instance>:MEValuation:TRACe:ACP:MAXimum[:RELative]
value: List[float] = driver.multiEval.trace.acp.maximum.relative.read()
```

Returns the adjacent channel power measured at a series of frequencies. The current, average and maximum traces can be retrieved. The frequencies are determined by the offset values defined via the commands method RsCmwEvdoMeas.Configure. MultiEval.Acp.Foffsets.lower and method RsCmwEvdoMeas.Configure.MultiEval.Acp.Foffsets.upper. All defined offset values are considered (irrespective of their activation status) . 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.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return max_acp: No help available

7.4.2.7.4.2 Absolute

SCPI Commands

```
READ:EVDO:MEASurement<Instance>:MEValuation:TRACe:ACP:MAXimum:ABSolute
FETCh:EVDO:MEASurement<Instance>:MEValuation:TRACe:ACP:MAXimum:ABSolute
CALCulate:EVDO:MEASurement<Instance>:MEValuation:TRACe:ACP:MAXimum:ABSolute
```

class Absolute

Absolute commands group definition. 3 total commands, 0 Sub-groups, 3 group commands

calculate() → List[float]

```
# SCPI: CALCulate:EVDO:MEASurement<instance>
↪:MEValuation:TRACe:ACP:MAXimum:ABSolute
value: List[float] = driver.multiEval.trace.acp.maximum.absolute.calculate()
```

Returns the adjacent channel absolute power measured in dBm at a series of frequencies. The frequencies are determined by the offset values defined via the commands

method RsCmwEvdoMeas.Configure.MultiEval.Acp.Foffsets.lower and method RsCmwEvdoMeas.Configure.MultiEval.Acp.Foffsets.upper. All defined offset values are considered (irrespective of their activation status) . The current, average and maximum traces can be retrieved.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return max_acp: No help available

fetch() → List[float]

```
# SCPI: FETCh:EVDO:MEASurement<instance>:MEValuation:TRACe:ACP:MAXimum:ABSolute
value: List[float] = driver.multiEval.trace.acp.maximum.absolute.fetch()
```

Returns the adjacent channel absolute power measured in dBm at a series of frequencies. The frequencies are determined by the offset values defined via the commands method RsCmwEvdoMeas.Configure.MultiEval.Acp.Foffsets.lower and method RsCmwEvdoMeas.Configure.MultiEval.Acp.Foffsets.upper. All defined offset values are considered (irrespective of their activation status) . The current, average and maximum traces can be retrieved.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return max_acp: No help available

read() → List[float]

```
# SCPI: READ:EVDO:MEASurement<instance>:MEValuation:TRACe:ACP:MAXimum:ABSolute
value: List[float] = driver.multiEval.trace.acp.maximum.absolute.read()
```

Returns the adjacent channel absolute power measured in dBm at a series of frequencies. The frequencies are determined by the offset values defined via the commands method RsCmwEvdoMeas.Configure.MultiEval.Acp.Foffsets.lower and method RsCmwEvdoMeas.Configure.MultiEval.Acp.Foffsets.upper. All defined offset values are considered (irrespective of their activation status) . The current, average and maximum traces can be retrieved.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return max_acp: No help available

7.4.2.8 Obw<Obw>

RepCap Settings

```
# Range: Nr1 .. Nr4
rc = driver.multiEval.trace.obw.repcap_obw_get()
driver.multiEval.trace.obw.repcap_obw_set(repcap.Obw.Nr1)
```

class Obw

Obw commands group definition. 3 total commands, 1 Sub-groups, 0 group commands Repeated Capability: Obw, default value after init: Obw.Nr1

Cloning the Group

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

Subgroups

7.4.2.8.1 Current

SCPI Commands

```
READ:EVDO:MEASurement<Instance>:MEValuation:TRACe:OBW<Obw>:CURRent
FETCh:EVDO:MEASurement<Instance>:MEValuation:TRACe:OBW<Obw>:CURRent
CALCulate:EVDO:MEASurement<Instance>:MEValuation:TRACe:OBW<Obw>:CURRent
```

class Current

Current commands group definition. 3 total commands, 0 Sub-groups, 3 group commands

calculate(*obw=<Obw.Default: -1>*) → List[RsCmwEvdoMeas.enums.ResultStatus2]

```
# SCPI: CALCulate:EVDO:MEASurement<instance>:MEValuation:TRACe:OBW<Number>
↪:CURRent
value: List[enums.ResultStatus2] = driver.multiEval.trace.obw.current.
↪calculate(obw = repcap.Obw.Default)
```

For the carrier or carrier group addressed by the <Number> suffix, returns the lower and upper edge of the occupied bandwidth (OBW) . For single-carrier configurations (i.e. only carrier 0 is active) the <Number> suffix can be omitted to obtain the OBW results. For multi-carrier configurations the <Number> suffixes are used as follows:

INTRO_CMD_HELP: Starts, stops, or aborts the measurement:

- For an active and isolated carrier i, use <Number> = i+1 to get its OBW results (i = 0, 1, 2)
- Use <Number> = 4 to display the OBW results of the 'Overall Carrier'
- If all active carriers are adjacent, use <Number>=4 to get the group (overall) OBW results
- If three carriers are active and exactly two carriers i,j with 0i<j2 are adjacent, use <Number> = i+1 for the joint OBW results of adjacent carriers.

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.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

param obw optional repeated capability selector. Default value: Nr1 (settable in the interface 'Obw')

return curr_obw: No help available

fetch(*obw=<Obw.Default: -1>*) → List[float]

```
# SCPI: FETCh:EVDO:MEASurement<instance>:MEValuation:TRACe:OBW<Number>:CURRent
value: List[float] = driver.multiEval.trace.obw.current.fetch(obw = repcap.Obw.
↪Default)
```

(continues on next page)

(continued from previous page)

For the carrier or carrier group addressed by the <Number> suffix, returns the lower and upper edge of the occupied bandwidth (OBW) . For single-carrier configurations (i.e. only carrier 0 is active) the <Number> suffix can be omitted to obtain the OBW results. For multi-carrier configurations the <Number> suffixes are used as follows:

INTRO_CMD_HELP: Starts, stops, or aborts the measurement:

- For an active and isolated carrier i, use <Number> = i+1 to get its OBW results (i = 0, 1, 2)
- Use <Number> = 4 to display the OBW results of the 'Overall Carrier'
- If all active carriers are adjacent, use <Number>=4 to get the group (overall) OBW results
- If three carriers are active and exactly two carriers i,j with 0i<j2 are adjacent, use <Number> = i+1 for the joint OBW results of adjacent carriers.

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.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

param obw optional repeated capability selector. Default value: Nr1 (settable in the interface 'Obw')

return curr_obw: No help available

read(*obw=<Obw.Default: -1>*) → List[float]

```
# SCPI: READ:EVDO:MEASurement<instance>:MEValuation:TRACe:OBW<Number>:CURRent
value: List[float] = driver.multiEval.trace.obw.current.read(obw = repcap.Obw.
↪Default)
```

For the carrier or carrier group addressed by the <Number> suffix, returns the lower and upper edge of the occupied bandwidth (OBW) . For single-carrier configurations (i.e. only carrier 0 is active) the <Number> suffix can be omitted to obtain the OBW results. For multi-carrier configurations the <Number> suffixes are used as follows:

INTRO_CMD_HELP: Starts, stops, or aborts the measurement:

- For an active and isolated carrier i, use <Number> = i+1 to get its OBW results (i = 0, 1, 2)
- Use <Number> = 4 to display the OBW results of the 'Overall Carrier'
- If all active carriers are adjacent, use <Number>=4 to get the group (overall) OBW results
- If three carriers are active and exactly two carriers i,j with 0i<j2 are adjacent, use <Number> = i+1 for the joint OBW results of adjacent carriers.

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.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

param obw optional repeated capability selector. Default value: Nr1 (settable in the interface 'Obw')

return curr_obw: No help available

7.4.2.9 Spectrum

class Spectrum
Spectrum commands group definition. 2 total commands, 1 Sub-groups, 0 group commands

Cloning the Group

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

Subgroups

7.4.2.9.1 Current

SCPI Commands

```
READ:EVDO:MEASurement<Instance>:MEValuation:TRACe:SPECtrum:CURRent
FETCh:EVDO:MEASurement<Instance>:MEValuation:TRACe:SPECtrum:CURRent
```

class Current
Current commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

fetch() → List[float]

```
# SCPI: FETCh:EVDO:MEASurement<instance>:MEValuation:TRACe:SPECtrum:CURRent
value: List[float] = driver.multiEval.trace.spectrum.current.fetch()
```

Returns the power spectrum traces across the frequency span (8 MHz or 16 MHz for single-carrier configurations, 16 MHz for multi-carrier configurations; see method RsCmwEvdoMeas.Configure.MultiEval.Carrier.wbFilter) .

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return curr_spectrum: float Frequency-dependent power values. Each of the traces contains 1667 values. Range: -100 dB to +57 dB , Unit: dB

read() → List[float]

```
# SCPI: READ:EVDO:MEASurement<instance>:MEValuation:TRACe:SPECtrum:CURRent
value: List[float] = driver.multiEval.trace.spectrum.current.read()
```

Returns the power spectrum traces across the frequency span (8 MHz or 16 MHz for single-carrier configurations, 16 MHz for multi-carrier configurations; see method RsCmwEvdoMeas.Configure.MultiEval.Carrier.wbFilter) .

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return curr_spectrum: float Frequency-dependent power values. Each of the traces contains 1667 values. Range: -100 dB to +57 dB , Unit: dB

7.4.2.10 Iq

class Iq
Iq commands group definition. 2 total commands, 1 Sub-groups, 0 group commands

Cloning the Group

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

Subgroups

7.4.2.10.1 Current

SCPI Commands

```
READ:EVDO:MEASurement<Instance>:MEValuation:TRACe:IQ:CURRent
FETCh:EVDO:MEASurement<Instance>:MEValuation:TRACe:IQ:CURRent
```

class Current
Current commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

class ResultData
Response structure. Fields:

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

fetch() → ResultData

```
# SCPI: FETCh:EVDO:MEASurement<instance>:MEValuation:TRACe:IQ:CURRent
value: ResultData = driver.multiEval.trace.iq.current.fetch()
```

Returns the results in the I/Q constellation diagram. Every fourth value corresponds to a constellation point. The other values are on the path between two constellation points.

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

read() → ResultData

```
# SCPI: READ:EVDO:MEASurement<instance>:MEValuation:TRACe:IQ:CURRent
value: ResultData = driver.multiEval.trace.iq.current.read()
```

Returns the results in the I/Q constellation diagram. Every fourth value corresponds to a constellation point. The other values are on the path between two constellation points.

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

7.4.3 Modulation

class Modulation
Modulation commands group definition. 15 total commands, 5 Sub-groups, 0 group commands

Cloning the Group

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

Subgroups

7.4.3.1 Current

SCPI Commands

```
READ:EVDO:MEASurement<Instance>:MEValuation:MODulation:CURRent
FETCh:EVDO:MEASurement<Instance>:MEValuation:MODulation:CURRent
CALCulate:EVDO:MEASurement<Instance>:MEValuation:MODulation:CURRent
```

class Current
Current commands group definition. 3 total commands, 0 Sub-groups, 3 group commands

class CalculateStruct
Response structure. Fields:

- Reliability: int: decimal 'Reliability Indicator'
- Evm_Rms: float: float Error vector magnitude RMS and peak value Range: 0 % to 100 %, Unit: %
- Evm_Peak: float: float Error vector magnitude RMS and peak value Range: 0 % to 100 %, Unit: %
- Merr_Rms: float: float Magnitude error RMS value Range: 0 % to 100 %, Unit: %
- Merr_Peak: float: float Magnitude error peak value Range: -100 % to 100 % (AVERage: 0% to 100 %, SDEViation: 0 % to 50 %) , Unit: %
- Perr_Rms: float: float Phase error RMS value Range: 0 deg to 180 deg, Unit: deg
- Perr_Peak: float: float Phase error peak value Range: -180 deg to 180 deg (AVERage: 0 deg to 180 deg, SDEViation: 0 deg to 90 deg) , Unit: deg
- Iq_Offset: float: float I/Q origin offset Range: -100 dB to 0 dB, Unit: dB
- Iq_Imbalance: float: float I/Q imbalance Range: -100 dB to 0 dB, Unit: dB
- Cfreq_Error: float: float Carrier frequency error Range: -5000 Hz to 5000 Hz, Unit: Hz
- Trans_Time_Err: float: float Transmit time error Range: -100 µs to 100 µs, Unit: µs
- At_Power_1_M_23: float: No parameter help available
- At_Power_Wideband: float: No parameter help available
- Wav_Quality: float: float Waveform quality Range: 0 to 1
- Wav_Qual_At_Max_Pow: float: No parameter help available
- Wav_Qual_At_Min_Pow: float: No parameter help available

- Out_Of_Tol_Count: float: float Out of tolerance result, i.e. percentage of measurement intervals of the statistic count (see [CMDLINK: CONFigure:EVDO:MEASi:MEValuation:SCOunt:MODulation CMDLINK]) exceeding the specified limits, see 'Limits (Modulation) '. Range: 0 % to 100 %, Unit: %
- Co_Ch_Filt_Mat_Rat: float: No parameter help available

class ResultData

Response structure. Fields:

- Reliability: int: decimal 'Reliability Indicator'
- Evm_Rms: float: float Error vector magnitude RMS and peak value Range: 0 % to 100 %, Unit: %
- Evm_Peak: float: float Error vector magnitude RMS and peak value Range: 0 % to 100 %, Unit: %
- Merr_Rms: float: float Magnitude error RMS value Range: 0 % to 100 %, Unit: %
- Merr_Peak: float: float Magnitude error peak value Range: -100 % to 100 % (AVERage: 0% to 100 %, SDEViation: 0 % to 50 %) , Unit: %
- Perr_Rms: float: float Phase error RMS value Range: 0 deg to 180 deg, Unit: deg
- Perr_Peak: float: float Phase error peak value Range: -180 deg to 180 deg (AVERage: 0 deg to 180 deg, SDEViation: 0 deg to 90 deg) , Unit: deg
- Iq_Offset: float: float I/Q origin offset Range: -100 dB to 0 dB, Unit: dB
- Iq_Imbalance: float: float I/Q imbalance Range: -100 dB to 0 dB, Unit: dB
- Cfreq_Error: float: float Carrier frequency error Range: -5000 Hz to 5000 Hz, Unit: Hz
- Trans_Time_Err: float: float Transmit time error Range: -100 µs to 100 µs, Unit: µs
- At_Power_1_M_23: float: No parameter help available
- At_Power_Wideband: float: No parameter help available
- Wav_Quality: float: float Waveform quality Range: 0 to 1
- Wav_Qual_At_Max_Pow: float: No parameter help available
- Wav_Qual_At_Min_Pow: float: No parameter help available
- Out_Of_Tol_Count: float: float Out of tolerance result, i.e. percentage of measurement intervals of the statistic count (see [CMDLINK: CONFigure:EVDO:MEASi:MEValuation:SCOunt:MODulation CMDLINK]) exceeding the specified limits, see 'Limits (Modulation) '. Range: 0 % to 100 %, Unit: %
- Co_Ch_Filt_Mat_Rat: float: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:EVDO:MEASurement<instance>:MEValuation:MODulation:CURRent
value: CalculateStruct = driver.multiEval.modulation.current.calculate()
```

Return the current, average, minimum, maximum and standard deviation modulation single value 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. The ranges indicated below apply to all results except standard deviation results. The minimum for standard deviation results equals 0. The maximum equals the width of the indicated range divided by two. Exceptions are explicitly stated.

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

fetch() → ResultData

```
# SCPI: FETCh:EVDO:MEASurement<instance>:MEValuation:MODulation:CURRent
value: ResultData = driver.multiEval.modulation.current.fetch()
```

Return the current, average, minimum, maximum and standard deviation modulation single value 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. The ranges indicated below apply to all results except standard deviation results. The minimum for standard deviation results equals 0. The maximum equals the width of the indicated range divided by two. Exceptions are explicitly stated.

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

read() → ResultData

```
# SCPI: READ:EVDO:MEASurement<instance>:MEValuation:MODulation:CURRent
value: ResultData = driver.multiEval.modulation.current.read()
```

Return the current, average, minimum, maximum and standard deviation modulation single value 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. The ranges indicated below apply to all results except standard deviation results. The minimum for standard deviation results equals 0. The maximum equals the width of the indicated range divided by two. Exceptions are explicitly stated.

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

7.4.3.2 Average

SCPI Commands

```
READ:EVDO:MEASurement<Instance>:MEValuation:MODulation:AVERage
FETCh:EVDO:MEASurement<Instance>:MEValuation:MODulation:AVERage
CALCulate:EVDO:MEASurement<Instance>:MEValuation:MODulation:AVERage
```

class Average

Average commands group definition. 3 total commands, 0 Sub-groups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: decimal 'Reliability Indicator'
- Evm_Rms: float: float Error vector magnitude RMS and peak value Range: 0 % to 100 %, Unit: %
- Evm_Peak: float: float Error vector magnitude RMS and peak value Range: 0 % to 100 %, Unit: %
- Merr_Rms: float: float Magnitude error RMS value Range: 0 % to 100 %, Unit: %
- Merr_Peak: float: float Magnitude error peak value Range: -100 % to 100 % (AVERage: 0% to 100 %, SDEViation: 0 % to 50 %) , Unit: %
- Perr_Rms: float: float Phase error RMS value Range: 0 deg to 180 deg, Unit: deg
- Perr_Peak: float: float Phase error peak value Range: -180 deg to 180 deg (AVERage: 0 deg to 180 deg, SDEViation: 0 deg to 90 deg) , Unit: deg
- Iq_Offset: float: float I/Q origin offset Range: -100 dB to 0 dB, Unit: dB

- Iq_Imbalance: float: float I/Q imbalance Range: -100 dB to 0 dB, Unit: dB
- Cfreq_Error: float: float Carrier frequency error Range: -5000 Hz to 5000 Hz, Unit: Hz
- Trans_Time_Err: float: float Transmit time error Range: -100 µs to 100 µs, Unit: µs
- At_Power_1_M_23: float: No parameter help available
- At_Power_Wideband: float: No parameter help available
- Wav_Quality: float: float Waveform quality Range: 0 to 1
- Wav_Qual_At_Max_Pow: float: No parameter help available
- Wav_Qual_At_Min_Pow: float: No parameter help available
- Out_Of_Tol_Count: float: float Out of tolerance result, i.e. percentage of measurement intervals of the statistic count (see [CMDLINK: CONFigure:EVDO:MEASi:MEValuation:SCOunt:MODulation CMDLINK]) exceeding the specified limits, see 'Limits (Modulation) '. Range: 0 % to 100 %, Unit: %
- Co_Ch_Filt_Mat_Rat: float: No parameter help available

class ResultData

Response structure. Fields:

- Reliability: int: decimal 'Reliability Indicator'
- Evm_Rms: float: float Error vector magnitude RMS and peak value Range: 0 % to 100 %, Unit: %
- Evm_Peak: float: float Error vector magnitude RMS and peak value Range: 0 % to 100 %, Unit: %
- Merr_Rms: float: float Magnitude error RMS value Range: 0 % to 100 %, Unit: %
- Merr_Peak: float: float Magnitude error peak value Range: -100 % to 100 % (AVERage: 0% to 100 %, SDEViation: 0 % to 50 %) , Unit: %
- Perr_Rms: float: float Phase error RMS value Range: 0 deg to 180 deg, Unit: deg
- Perr_Peak: float: float Phase error peak value Range: -180 deg to 180 deg (AVERage: 0 deg to 180 deg, SDEViation: 0 deg to 90 deg) , Unit: deg
- Iq_Offset: float: float I/Q origin offset Range: -100 dB to 0 dB, Unit: dB
- Iq_Imbalance: float: float I/Q imbalance Range: -100 dB to 0 dB, Unit: dB
- Cfreq_Error: float: float Carrier frequency error Range: -5000 Hz to 5000 Hz, Unit: Hz
- Trans_Time_Err: float: float Transmit time error Range: -100 µs to 100 µs, Unit: µs
- At_Power_1_M_23: float: No parameter help available
- At_Power_Wideband: float: No parameter help available
- Wav_Quality: float: float Waveform quality Range: 0 to 1
- Wav_Qual_At_Max_Pow: float: No parameter help available
- Wav_Qual_At_Min_Pow: float: No parameter help available
- Out_Of_Tol_Count: float: float Out of tolerance result, i.e. percentage of measurement intervals of the statistic count (see [CMDLINK: CONFigure:EVDO:MEASi:MEValuation:SCOunt:MODulation CMDLINK]) exceeding the specified limits, see 'Limits (Modulation) '. Range: 0 % to 100 %, Unit: %
- Co_Ch_Filt_Mat_Rat: float: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:EVDO:MEASurement<instance>:MEValuation:MODulation:AVERage
value: CalculateStruct = driver.multiEval.modulation.average.calculate()
```

Return the current, average, minimum, maximum and standard deviation modulation single value 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. The ranges indicated below apply to all results except standard deviation results. The minimum for standard deviation results equals 0. The maximum equals the width of the indicated range divided by two. Exceptions are explicitly stated.

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

fetch() → ResultData

```
# SCPI: FETCh:EVDO:MEASurement<instance>:MEValuation:MODulation:AVERage
value: ResultData = driver.multiEval.modulation.average.fetch()
```

Return the current, average, minimum, maximum and standard deviation modulation single value 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. The ranges indicated below apply to all results except standard deviation results. The minimum for standard deviation results equals 0. The maximum equals the width of the indicated range divided by two. Exceptions are explicitly stated.

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

read() → ResultData

```
# SCPI: READ:EVDO:MEASurement<instance>:MEValuation:MODulation:AVERage
value: ResultData = driver.multiEval.modulation.average.read()
```

Return the current, average, minimum, maximum and standard deviation modulation single value 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. The ranges indicated below apply to all results except standard deviation results. The minimum for standard deviation results equals 0. The maximum equals the width of the indicated range divided by two. Exceptions are explicitly stated.

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

7.4.3.3 Maximum

SCPI Commands

```
READ:EVDO:MEASurement<Instance>:MEValuation:MODulation:MAXimum
FETCh:EVDO:MEASurement<Instance>:MEValuation:MODulation:MAXimum
CALCulate:EVDO:MEASurement<Instance>:MEValuation:MODulation:MAXimum
```

class Maximum

Maximum commands group definition. 3 total commands, 0 Sub-groups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: decimal 'Reliability Indicator'

- Evm_Rms: float: float Error vector magnitude RMS and peak value Range: 0 % to 100 %, Unit: %
- Evm_Peak: float: float Error vector magnitude RMS and peak value Range: 0 % to 100 %, Unit: %
- Merr_Rms: float: float Magnitude error RMS value Range: 0 % to 100 %, Unit: %
- Merr_Peak: float: float Magnitude error peak value Range: -100 % to 100 % (AVERage: 0% to 100 %, SDEViation: 0 % to 50 %) , Unit: %
- Perr_Rms: float: float Phase error RMS value Range: 0 deg to 180 deg, Unit: deg
- Perr_Peak: float: float Phase error peak value Range: -180 deg to 180 deg (AVERage: 0 deg to 180 deg, SDEViation: 0 deg to 90 deg) , Unit: deg
- Iq_Offset: float: float I/Q origin offset Range: -100 dB to 0 dB, Unit: dB
- Iq_Imbalance: float: float I/Q imbalance Range: -100 dB to 0 dB, Unit: dB
- Cfreq_Error: float: float Carrier frequency error Range: -5000 Hz to 5000 Hz, Unit: Hz
- Trans_Time_Err: float: float Transmit time error Range: -100 µs to 100 µs, Unit: µs
- At_Power_1_M_23: float: No parameter help available
- At_Power_Wideband: float: No parameter help available
- Wav_Quality: float: float Waveform quality Range: 0 to 1
- Wav_Qual_At_Max_Pow: float: No parameter help available
- Wav_Qual_At_Min_Pow: float: No parameter help available
- Out_Of_Tol_Count: float: float Out of tolerance result, i.e. percentage of measurement intervals of the statistic count (see [CMDLINK: CONFigure:EVDO:MEASi:MEValuation:SCOunt:MODulation CMDLINK]) exceeding the specified limits, see 'Limits (Modulation) '. Range: 0 % to 100 %, Unit: %
- Co_Ch_Filt_Mat_Rat: float: No parameter help available

class FetchStruct

Response structure. Fields:

- Reliability: int: decimal 'Reliability Indicator'
- Evm_Rms: float: float Error vector magnitude RMS and peak value Range: 0 % to 100 %, Unit: %
- Evm_Peak: float: float Error vector magnitude RMS and peak value Range: 0 % to 100 %, Unit: %
- Merr_Rms: float: float Magnitude error RMS value Range: 0 % to 100 %, Unit: %
- Merr_Peak: float: float Magnitude error peak value Range: -100 % to 100 % (AVERage: 0% to 100 %, SDEViation: 0 % to 50 %) , Unit: %
- Perr_Rms: float: float Phase error RMS value Range: 0 deg to 180 deg, Unit: deg
- Perr_Peak: float: float Phase error peak value Range: -180 deg to 180 deg (AVERage: 0 deg to 180 deg, SDEViation: 0 deg to 90 deg) , Unit: deg
- Iq_Offset: float: float I/Q origin offset Range: -100 dB to 0 dB, Unit: dB
- Iq_Imbalance: float: float I/Q imbalance Range: -100 dB to 0 dB, Unit: dB
- Cfreq_Error: float: float Carrier frequency error Range: -5000 Hz to 5000 Hz, Unit: Hz
- Trans_Time_Err: float: float Transmit time error Range: -100 µs to 100 µs, Unit: µs
- At_Power_1_M_23: float: No parameter help available
- At_Power_Wideband: float: No parameter help available

- Wav_Quality: float: float Waveform quality Range: 0 to 1
- Wav_Qual_At_Max_Pow: float: No parameter help available
- Wav_Qual_At_Min_Pow: float: No parameter help available
- Out_Of_Tol_Count: float: float Out of tolerance result, i.e. percentage of measurement intervals of the statistic count (see [CMDLINK: CONFigure:EVDO:MEASi:MEValuation:SCOunt:MODulation CMDLINK]) exceeding the specified limits, see 'Limits (Modulation) '. Range: 0 % to 100 %, Unit: %
- Co_Ch_Filt_Mat_Rat: float: No parameter help available

class ReadStruct

Response structure. Fields:

- Reliability: int: decimal 'Reliability Indicator'
- Evm_Rms: float: float Error vector magnitude RMS and peak value Range: 0 % to 100 %, Unit: %
- Evm_Peak: float: float Error vector magnitude RMS and peak value Range: 0 % to 100 %, Unit: %
- Merr_Rms: float: float Magnitude error RMS value Range: 0 % to 100 %, Unit: %
- Merr_Peak: float: float Magnitude error peak value Range: -100 % to 100 % (AVERage: 0% to 100 %, SDEViation: 0 % to 50 %) , Unit: %
- Perr_Rms: float: float Phase error RMS value Range: 0 deg to 180 deg, Unit: deg
- Perr_Peak: float: float Phase error peak value Range: -180 deg to 180 deg (AVERage: 0 deg to 180 deg, SDEViation: 0 deg to 90 deg) , Unit: deg
- Iq_Offset: float: float I/Q origin offset Range: -100 dB to 0 dB, Unit: dB
- Iq_Imbalance: float: float I/Q imbalance Range: -100 dB to 0 dB, Unit: dB
- Cfreq_Error: float: float Carrier frequency error Range: -5000 Hz to 5000 Hz, Unit: Hz
- Trans_Time_Err: float: float Transmit time error Range: -100 µs to 100 µs, Unit: µs
- At_Power_1_M_23: float: No parameter help available
- At_Power_Wideband: float: No parameter help available
- Wav_Quality: float: float Waveform quality Range: 0 to 1
- Wav_Qual_At_Max_Pow: float: No parameter help available
- Wav_Qual_Min_Pow: float: No parameter help available
- Out_Of_Tol_Count: float: float Out of tolerance result, i.e. percentage of measurement intervals of the statistic count (see [CMDLINK: CONFigure:EVDO:MEASi:MEValuation:SCOunt:MODulation CMDLINK]) exceeding the specified limits, see 'Limits (Modulation) '. Range: 0 % to 100 %, Unit: %
- Co_Ch_Filt_Mat_Rat: float: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:EVDO:MEASurement<instance>:MEValuation:MODulation:MAXimum
value: CalculateStruct = driver.multiEval.modulation.maximum.calculate()
```

Return the current, average, minimum, maximum and standard deviation modulation single value 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. The ranges indicated below apply to

all results except standard deviation results. The minimum for standard deviation results equals 0. The maximum equals the width of the indicated range divided by two. Exceptions are explicitly stated.

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

fetch() → FetchStruct

```
# SCPI: FETCh:EVDO:MEASurement<instance>:MEValuation:MODulation:MAXimum
value: FetchStruct = driver.multiEval.modulation.maximum.fetch()
```

Return the current, average, minimum, maximum and standard deviation modulation single value 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. The ranges indicated below apply to all results except standard deviation results. The minimum for standard deviation results equals 0. The maximum equals the width of the indicated range divided by two. Exceptions are explicitly stated.

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

read() → ReadStruct

```
# SCPI: READ:EVDO:MEASurement<instance>:MEValuation:MODulation:MAXimum
value: ReadStruct = driver.multiEval.modulation.maximum.read()
```

Return the current, average, minimum, maximum and standard deviation modulation single value 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. The ranges indicated below apply to all results except standard deviation results. The minimum for standard deviation results equals 0. The maximum equals the width of the indicated range divided by two. Exceptions are explicitly stated.

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

7.4.3.4 Minimum

SCPI Commands

```
READ:EVDO:MEASurement<Instance>:MEValuation:MODulation:MINimum
FETCh:EVDO:MEASurement<Instance>:MEValuation:MODulation:MINimum
CALCulate:EVDO:MEASurement<Instance>:MEValuation:MODulation:MINimum
```

class Minimum

Minimum commands group definition. 3 total commands, 0 Sub-groups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: decimal 'Reliability Indicator'
- Evm_Rms: float: float Error vector magnitude RMS and peak value Range: 0 % to 100 %, Unit: %
- Evm_Peak: float: float Error vector magnitude RMS and peak value Range: 0 % to 100 %, Unit: %
- Merr_Rms: float: float Magnitude error RMS value Range: 0 % to 100 %, Unit: %
- Merr_Peak: float: float Magnitude error peak value Range: -100 % to 100 % (AVERage: 0% to 100 %, SDEViation: 0 % to 50 %) , Unit: %
- Perr_Rms: float: float Phase error RMS value Range: 0 deg to 180 deg, Unit: deg

- Perr_Peak: float: float Phase error peak value Range: -180 deg to 180 deg (AVERage: 0 deg to 180 deg, SDEViation: 0 deg to 90 deg) , Unit: deg
- Iq_Offset: float: float I/Q origin offset Range: -100 dB to 0 dB, Unit: dB
- Iq_Imbalance: float: float I/Q imbalance Range: -100 dB to 0 dB, Unit: dB
- Cfreq_Error: float: float Carrier frequency error Range: -5000 Hz to 5000 Hz, Unit: Hz
- Trans_Time_Err: float: float Transmit time error Range: -100 µs to 100 µs, Unit: µs
- At_Power_1_M_23: float: No parameter help available
- At_Power_Wideband: float: No parameter help available
- Wav_Quality: float: float Waveform quality Range: 0 to 1
- Wav_Qual_At_Max_Pow: float: No parameter help available
- Wav_Qual_At_Min_Pow: float: No parameter help available
- Out_Of_Tol_Count: float: float Out of tolerance result, i.e. percentage of measurement intervals of the statistic count (see [CMDLINK: CONFigure:EVDO:MEASi:MEValuation:SCOunt:MODulation CMDLINK]) exceeding the specified limits, see 'Limits (Modulation) '. Range: 0 % to 100 %, Unit: %
- Co_Ch_Filt_Mat_Rat: float: No parameter help available

class ResultData

Response structure. Fields:

- Reliability: int: decimal 'Reliability Indicator'
- Evm_Rms: float: float Error vector magnitude RMS and peak value Range: 0 % to 100 %, Unit: %
- Evm_Peak: float: float Error vector magnitude RMS and peak value Range: 0 % to 100 %, Unit: %
- Merr_Rms: float: float Magnitude error RMS value Range: 0 % to 100 %, Unit: %
- Merr_Peak: float: float Magnitude error peak value Range: -100 % to 100 % (AVERage: 0% to 100 %, SDEViation: 0 % to 50 %) , Unit: %
- Perr_Rms: float: float Phase error RMS value Range: 0 deg to 180 deg, Unit: deg
- Perr_Peak: float: float Phase error peak value Range: -180 deg to 180 deg (AVERage: 0 deg to 180 deg, SDEViation: 0 deg to 90 deg) , Unit: deg
- Iq_Offset: float: float I/Q origin offset Range: -100 dB to 0 dB, Unit: dB
- Iq_Imbalance: float: float I/Q imbalance Range: -100 dB to 0 dB, Unit: dB
- Cfreq_Error: float: float Carrier frequency error Range: -5000 Hz to 5000 Hz, Unit: Hz
- Trans_Time_Err: float: float Transmit time error Range: -100 µs to 100 µs, Unit: µs
- At_Power_1_M_23: float: No parameter help available
- At_Power_Wideband: float: No parameter help available
- Wav_Quality: float: float Waveform quality Range: 0 to 1
- Wav_Qual_At_Max_Pow: float: No parameter help available
- Wav_Qual_At_Min_Pow: float: No parameter help available
- Out_Of_Tol_Count: float: float Out of tolerance result, i.e. percentage of measurement intervals of the statistic count (see [CMDLINK: CONFigure:EVDO:MEASi:MEValuation:SCOunt:MODulation

CMDLINK]) exceeding the specified limits, see 'Limits (Modulation) '. Range: 0 % to 100 %, Unit: %

- Co_Ch_Filt_Mat_Rat: float: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:EVDO:MEASurement<instance>:MEValuation:MODulation:MINimum
value: CalculateStruct = driver.multiEval.modulation.minimum.calculate()
```

Return the current, average, minimum, maximum and standard deviation modulation single value 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. The ranges indicated below apply to all results except standard deviation results. The minimum for standard deviation results equals 0. The maximum equals the width of the indicated range divided by two. Exceptions are explicitly stated.

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

fetch() → ResultData

```
# SCPI: FETCh:EVDO:MEASurement<instance>:MEValuation:MODulation:MINimum
value: ResultData = driver.multiEval.modulation.minimum.fetch()
```

Return the current, average, minimum, maximum and standard deviation modulation single value 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. The ranges indicated below apply to all results except standard deviation results. The minimum for standard deviation results equals 0. The maximum equals the width of the indicated range divided by two. Exceptions are explicitly stated.

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

read() → ResultData

```
# SCPI: READ:EVDO:MEASurement<instance>:MEValuation:MODulation:MINimum
value: ResultData = driver.multiEval.modulation.minimum.read()
```

Return the current, average, minimum, maximum and standard deviation modulation single value 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. The ranges indicated below apply to all results except standard deviation results. The minimum for standard deviation results equals 0. The maximum equals the width of the indicated range divided by two. Exceptions are explicitly stated.

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

7.4.3.5 StandardDev

SCPI Commands

```
READ:EVDO:MEASurement<Instance>:MEValuation:MODulation:SDEViation
FETCh:EVDO:MEASurement<Instance>:MEValuation:MODulation:SDEViation
CALCulate:EVDO:MEASurement<Instance>:MEValuation:MODulation:SDEViation
```

class StandardDev

StandardDev commands group definition. 3 total commands, 0 Sub-groups, 3 group commands

class CalculateStruct
Response structure. Fields:

- Reliability: int: decimal 'Reliability Indicator'
- Evm_Rms: float: float Error vector magnitude RMS and peak value Range: 0 % to 100 %, Unit: %
- Evm_Peak: float: float Error vector magnitude RMS and peak value Range: 0 % to 100 %, Unit: %
- Merr_Rms: float: float Magnitude error RMS value Range: 0 % to 100 %, Unit: %
- Merr_Peak: float: float Magnitude error peak value Range: -100 % to 100 % (AVERage: 0% to 100 %, SDEViation: 0 % to 50 %) , Unit: %
- Perr_Rms: float: float Phase error RMS value Range: 0 deg to 180 deg, Unit: deg
- Perr_Peak: float: float Phase error peak value Range: -180 deg to 180 deg (AVERage: 0 deg to 180 deg, SDEViation: 0 deg to 90 deg) , Unit: deg
- Iq_Offset: float: float I/Q origin offset Range: -100 dB to 0 dB, Unit: dB
- Iq_Imbalance: float: float I/Q imbalance Range: -100 dB to 0 dB, Unit: dB
- Cfreq_Error: float: float Carrier frequency error Range: -5000 Hz to 5000 Hz, Unit: Hz
- Trans_Time_Err: float: float Transmit time error Range: -100 µs to 100 µs, Unit: µs
- At_Power_1_M_23: float: No parameter help available
- At_Power_Wideband: float: No parameter help available
- Wav_Quality: float: float Waveform quality Range: 0 to 1
- Wav_Qual_At_Max_Pow: float: No parameter help available
- Wav_Qual_At_Min_Pow: float: No parameter help available
- Out_Of_Tol_Count: float: float Out of tolerance result, i.e. percentage of measurement intervals of the statistic count (see [CMDLINK: CONFigure:EVDO:MEASi:MEValuation:SCOunt:MODulation CMDLINK]) exceeding the specified limits, see 'Limits (Modulation) '. Range: 0 % to 100 %, Unit: %
- Co_Ch_Filt_Mat_Rat: float: No parameter help available

class ResultData
Response structure. Fields:

- Reliability: int: decimal 'Reliability Indicator'
- Evm_Rms: float: float Error vector magnitude RMS and peak value Range: 0 % to 100 %, Unit: %
- Evm_Peak: float: float Error vector magnitude RMS and peak value Range: 0 % to 100 %, Unit: %
- Merr_Rms: float: float Magnitude error RMS value Range: 0 % to 100 %, Unit: %
- Merr_Peak: float: float Magnitude error peak value Range: -100 % to 100 % (AVERage: 0% to 100 %, SDEViation: 0 % to 50 %) , Unit: %
- Perr_Rms: float: float Phase error RMS value Range: 0 deg to 180 deg, Unit: deg
- Perr_Peak: float: float Phase error peak value Range: -180 deg to 180 deg (AVERage: 0 deg to 180 deg, SDEViation: 0 deg to 90 deg) , Unit: deg
- Iq_Offset: float: float I/Q origin offset Range: -100 dB to 0 dB, Unit: dB
- Iq_Imbalance: float: float I/Q imbalance Range: -100 dB to 0 dB, Unit: dB
- Cfreq_Error: float: float Carrier frequency error Range: -5000 Hz to 5000 Hz, Unit: Hz

- Trans_Time_Err: float: float Transmit time error Range: -100 µs to 100 µs, Unit: µs
- At_Power_1_M_23: float: No parameter help available
- At_Power_Wideband: float: No parameter help available
- Wav_Quality: float: float Waveform quality Range: 0 to 1
- Wav_Qual_At_Max_Pow: float: No parameter help available
- Wav_Qual_At_Min_Pow: float: No parameter help available
- Out_Of_Tol_Count: float: float Out of tolerance result, i.e. percentage of measurement intervals of the statistic count (see [CMDLINK: CONFigure:EVDO:MEASi:MEValuation:SCOunt:MODulation CMDLINK]) exceeding the specified limits, see 'Limits (Modulation) '. Range: 0 % to 100 %, Unit: %
- Co_Ch_Filt_Mat_Rat: float: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:EVDO:MEASurement<instance>:MEValuation:MODulation:SDEViation
value: CalculateStruct = driver.multiEval.modulation.standardDev.calculate()
```

Return the current, average, minimum, maximum and standard deviation modulation single value 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. The ranges indicated below apply to all results except standard deviation results. The minimum for standard deviation results equals 0. The maximum equals the width of the indicated range divided by two. Exceptions are explicitly stated.

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

fetch() → ResultData

```
# SCPI: FETCh:EVDO:MEASurement<instance>:MEValuation:MODulation:SDEViation
value: ResultData = driver.multiEval.modulation.standardDev.fetch()
```

Return the current, average, minimum, maximum and standard deviation modulation single value 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. The ranges indicated below apply to all results except standard deviation results. The minimum for standard deviation results equals 0. The maximum equals the width of the indicated range divided by two. Exceptions are explicitly stated.

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

read() → ResultData

```
# SCPI: READ:EVDO:MEASurement<instance>:MEValuation:MODulation:SDEViation
value: ResultData = driver.multiEval.modulation.standardDev.read()
```

Return the current, average, minimum, maximum and standard deviation modulation single value 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. The ranges indicated below apply to all results except standard deviation results. The minimum for standard deviation results equals 0. The maximum equals the width of the indicated range divided by two. Exceptions are explicitly stated.

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

7.4.4 Cp

class Cp

Cp commands group definition. 12 total commands, 4 Sub-groups, 0 group commands

Cloning the Group

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

Subgroups

7.4.4.1 Current

SCPI Commands

```
READ:EVDO:MEASurement<Instance>:MEValuation:CP:CURRent
FETCh:EVDO:MEASurement<Instance>:MEValuation:CP:CURRent
CALCulate:EVDO:MEASurement<Instance>:MEValuation:CP:CURRent
```

class Current

Current commands group definition. 3 total commands, 0 Sub-groups, 3 group commands

calculate() → List[RsCmwEvdoMeas.enums.ResultStatus2]

```
# SCPI: CALCulate:EVDO:MEASurement<instance>:MEValuation:CP:CURRent
value: List[enums.ResultStatus2] = driver.multiEval.cp.current.calculate()
```

Returns the scalar channel power for the data channel. The result is extended to the W24 and W12 I/Q values of the data channel. Only available if subtype 2 or 3 is selected. Otherwise NAV is returned. 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.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return data: No help available

fetch() → List[float]

```
# SCPI: FETCh:EVDO:MEASurement<instance>:MEValuation:CP:CURRent
value: List[float] = driver.multiEval.cp.current.fetch()
```

Returns the scalar channel power for the data channel. The result is extended to the W24 and W12 I/Q values of the data channel. Only available if subtype 2 or 3 is selected. Otherwise NAV is returned. 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.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return data: No help available

read() → List[float]

```
# SCPI: READ:EVDO:MEASurement<instance>:MEValuation:CP:CURRent
value: List[float] = driver.multiEval.cp.current.read()
```

Returns the scalar channel power for the data channel. The result is extended to the W24 and W12 I/Q values of the data channel. Only available if subtype 2 or 3 is selected. Otherwise NAV is returned. 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.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return data: No help available

7.4.4.2 Average

SCPI Commands

```
READ:EVDO:MEASurement<Instance>:MEValuation:CP:AVERage
FETCh:EVDO:MEASurement<Instance>:MEValuation:CP:AVERage
CALCulate:EVDO:MEASurement<Instance>:MEValuation:CP:AVERage
```

class Average

Average commands group definition. 3 total commands, 0 Sub-groups, 3 group commands

calculate() → List[RsCmwEvdoMeas.enums.ResultStatus2]

```
# SCPI: CALCulate:EVDO:MEASurement<instance>:MEValuation:CP:AVERage
value: List[enums.ResultStatus2] = driver.multiEval.cp.average.calculate()
```

Returns the scalar channel power for the data channel. The result is extended to the W24 and W12 I/Q values of the data channel. Only available if subtype 2 or 3 is selected. Otherwise NAV is returned. 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.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return data: No help available

fetch() → List[float]

```
# SCPI: FETCh:EVDO:MEASurement<instance>:MEValuation:CP:AVERage
value: List[float] = driver.multiEval.cp.average.fetch()
```

Returns the scalar channel power for the data channel. The result is extended to the W24 and W12 I/Q values of the data channel. Only available if subtype 2 or 3 is selected. Otherwise NAV is returned. 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.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return data: No help available

read() → List[float]

```
# SCPI: READ:EVDO:MEASurement<instance>:MEValuation:CP:AVERage
value: List[float] = driver.multiEval.cp.average.read()
```

Returns the scalar channel power for the data channel. The result is extended to the W24 and W12 I/Q values of the data channel. Only available if subtype 2 or 3 is selected. Otherwise NAV is returned. 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.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return data: No help available

7.4.4.3 Maximum

SCPI Commands

```
READ:EVDO:MEASurement<Instance>:MEValuation:CP:MAXimum
FETCh:EVDO:MEASurement<Instance>:MEValuation:CP:MAXimum
CALCulate:EVDO:MEASurement<Instance>:MEValuation:CP:MAXimum
```

class Maximum

Maximum commands group definition. 3 total commands, 0 Sub-groups, 3 group commands

calculate() → List[RsCmwEvdoMeas.enums.ResultStatus2]

```
# SCPI: CALCulate:EVDO:MEASurement<instance>:MEValuation:CP:MAXimum
value: List[enums.ResultStatus2] = driver.multiEval.cp.maximum.calculate()
```

Returns the scalar channel power for the data channel. The result is extended to the W24 and W12 I/Q values of the data channel. Only available if subtype 2 or 3 is selected. Otherwise NAV is returned. 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.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return data: No help available

fetch() → List[float]

```
# SCPI: FETCh:EVDO:MEASurement<instance>:MEValuation:CP:MAXimum
value: List[float] = driver.multiEval.cp.maximum.fetch()
```

Returns the scalar channel power for the data channel. The result is extended to the W24 and W12 I/Q values of the data channel. Only available if subtype 2 or 3 is selected. Otherwise NAV is returned. 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.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return data: No help available

read() → List[float]

```
# SCPI: READ:EVDO:MEASurement<instance>:MEValuation:CP:MAXimum
value: List[float] = driver.multiEval.cp.maximum.read()
```

Returns the scalar channel power for the data channel. The result is extended to the W24 and W12 I/Q values of the data channel. Only available if subtype 2 or 3 is selected. Otherwise NAV is returned. 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.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return data: No help available

7.4.4.4 Minimum

SCPI Commands

```
READ:EVDO:MEASurement<Instance>:MEValuation:CP:MINimum
FETCh:EVDO:MEASurement<Instance>:MEValuation:CP:MINimum
CALCulate:EVDO:MEASurement<Instance>:MEValuation:CP:MINimum
```

class Minimum

Minimum commands group definition. 3 total commands, 0 Sub-groups, 3 group commands

calculate() → List[RsCmwEvdoMeas.enums.ResultStatus2]

```
# SCPI: CALCulate:EVDO:MEASurement<instance>:MEValuation:CP:MINimum
value: List[enums.ResultStatus2] = driver.multiEval.cp.minimum.calculate()
```

Returns the scalar channel power for the data channel. The result is extended to the W24 and W12 I/Q values of the data channel. Only available if subtype 2 or 3 is selected. Otherwise NAV is returned. 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.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return data: No help available

fetch() → List[float]

```
# SCPI: FETCh:EVDO:MEASurement<instance>:MEValuation:CP:MINimum
value: List[float] = driver.multiEval.cp.minimum.fetch()
```

Returns the scalar channel power for the data channel. The result is extended to the W24 and W12 I/Q values of the data channel. Only available if subtype 2 or 3 is selected. Otherwise NAV is returned. 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.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return data: No help available

read() → List[float]

```
# SCPI: READ:EVDO:MEASurement<instance>:MEValuation:CP:MINimum
value: List[float] = driver.multiEval.cp.minimum.read()
```

Returns the scalar channel power for the data channel. The result is extended to the W24 and W12 I/Q values of the data channel. Only available if subtype 2 or 3 is selected. Otherwise NAV is returned. 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.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return data: No help available

7.4.5 Acp

class Acp

Acp commands group definition. 9 total commands, 3 Sub-groups, 0 group commands

Cloning the Group

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

Subgroups

7.4.5.1 Current

SCPI Commands

```
READ:EVDO:MEASurement<Instance>:MEValuation:ACP:CURRent
FETCh:EVDO:MEASurement<Instance>:MEValuation:ACP:CURRent
CALCulate:EVDO:MEASurement<Instance>:MEValuation:ACP:CURRent
```

class Current

Current commands group definition. 3 total commands, 0 Sub-groups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: decimal 'Reliability Indicator'
- At_Power_Narrow: float: float Access terminal power, measured with a filter bandwidth of 1.23 MHz. Range: -100 dBm to 50 dBm, Unit: dBm
- At_Power_Wide: float: float Access terminal power, measured with the wideband filter. Range: -100 dBm to 50 dBm, Unit: dBm
- Out_Of_Tol_Count: float: float Out of tolerance result, i.e. percentage of measurement intervals of the statistic count (see [CMDLINK: CONFigure:EVDO:MEASi:MEValuation:SCOunt:SPECtrum CMDLINK]) exceeding the specified limits, see 'Limits (Spectrum) '. Range: 0 % to 100 %, Unit: %
- Code_Ch_Filter: float: float Code channel filter match ratio, i.e. percentage of measurement intervals matching the specified code channel filter, see 'Multi-Evaluation: Code Channel Filter'. Range: 0 % to 100 %, Unit: %

class ResultData

Response structure. Fields:

- Reliability: int: decimal 'Reliability Indicator'
- At_Power_Narrow: float: float Access terminal power, measured with a filter bandwidth of 1.23 MHz. Range: -100 dBm to 50 dBm, Unit: dBm
- At_Power_Wide: float: float Access terminal power, measured with the wideband filter. Range: -100 dBm to 50 dBm, Unit: dBm
- Out_Of_Tol_Count: float: float Out of tolerance result, i.e. percentage of measurement intervals of the statistic count (see [CMDLINK: CONFigure:EVDO:MEASi:MEValuation:SCOunt:SPECtrum CMDLINK]) exceeding the specified limits, see 'Limits (Spectrum) '. Range: 0 % to 100 %, Unit: %
- Code_Ch_Filter: float: float Code channel filter match ratio, i.e. percentage of measurement intervals matching the specified code channel filter, see 'Multi-Evaluation: Code Channel Filter'. Range: 0 % to 100 %, Unit: %

calculate() → CalculateStruct

```
# SCPI: CALCulate:EVDO:MEASurement<instance>:MEValuation:ACP:CURRent
value: CalculateStruct = driver.multiEval.acp.current.calculate()
```

Return the out of tolerance result, the code channel filter match ratio result and the AT power results. For the AT power results, the current, average and maximum values can be retrieved. The out of tolerance and code channel filter match ratio results retrieved via the CURRent, AVERage and MAXimum command are identical. 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:EVDO:MEASurement<instance>:MEValuation:ACP:CURRent
value: ResultData = driver.multiEval.acp.current.fetch()
```

Return the out of tolerance result, the code channel filter match ratio result and the AT power results. For the AT power results, the current, average and maximum values can be retrieved. The out of tolerance and code channel filter match ratio results retrieved via the CURRent, AVERage and MAXimum command are identical. 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:EVDO:MEASurement<instance>:MEValuation:ACP:CURRent
value: ResultData = driver.multiEval.acp.current.read()
```

Return the out of tolerance result, the code channel filter match ratio result and the AT power results. For the AT power results, the current, average and maximum values can be retrieved. The out of tolerance and code channel filter match ratio results retrieved via the CURRent, AVERage and MAXimum command are identical. 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.

7.4.5.2 Average

SCPI Commands

```
READ:EVDO:MEASurement<Instance>:MEValuation:ACP:AVERage
FETCh:EVDO:MEASurement<Instance>:MEValuation:ACP:AVERage
CALCulate:EVDO:MEASurement<Instance>:MEValuation:ACP:AVERage
```

class Average

Average commands group definition. 3 total commands, 0 Sub-groups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: decimal 'Reliability Indicator'
- At_Power_Narrow: float: float Access terminal power, measured with a filter bandwidth of 1.23 MHz. Range: -100 dBm to 50 dBm, Unit: dBm
- At_Power_Wide: float: float Access terminal power, measured with the wideband filter. Range: -100 dBm to 50 dBm, Unit: dBm
- Out_Of_Tol_Count: float: float Out of tolerance result, i.e. percentage of measurement intervals of the statistic count (see [CMDLINK: CONFigure:EVDO:MEASi:MEValuation:SCOunt:SPECtrum CMDLINK]) exceeding the specified limits, see 'Limits (Spectrum) '. Range: 0 % to 100 %, Unit: %
- Code_Ch_Filter: float: float Code channel filter match ratio, i.e. percentage of measurement intervals matching the specified code channel filter, see 'Multi-Evaluation: Code Channel Filter'. Range: 0 % to 100 %, Unit: %

class ResultData

Response structure. Fields:

- Reliability: int: decimal 'Reliability Indicator'
- At_Power_Narrow: float: float Access terminal power, measured with a filter bandwidth of 1.23 MHz. Range: -100 dBm to 50 dBm, Unit: dBm
- At_Power_Wide: float: float Access terminal power, measured with the wideband filter. Range: -100 dBm to 50 dBm, Unit: dBm
- Out_Of_Tol_Count: float: float Out of tolerance result, i.e. percentage of measurement intervals of the statistic count (see [CMDLINK: CONFigure:EVDO:MEASi:MEValuation:SCOunt:SPECtrum CMDLINK]) exceeding the specified limits, see 'Limits (Spectrum) '. Range: 0 % to 100 %, Unit: %
- Code_Ch_Filter: float: float Code channel filter match ratio, i.e. percentage of measurement intervals matching the specified code channel filter, see 'Multi-Evaluation: Code Channel Filter'. Range: 0 % to 100 %, Unit: %

calculate() → CalculateStruct

```
# SCPI: CALCulate:EVDO:MEASurement<instance>:MEValuation:ACP:AVERage
value: CalculateStruct = driver.multiEval.acp.average.calculate()
```

Return the out of tolerance result, the code channel filter match ratio result and the AT power results. For the AT power results, the current, average and maximum values can be retrieved. The out of tolerance and code channel filter match ratio results retrieved via the CURRent, AVERage and MAXimum command are identical. 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:EVDO:MEASurement<instance>:MEValuation:ACP:AVERage
value: ResultData = driver.multiEval.acp.average.fetch()
```

Return the out of tolerance result, the code channel filter match ratio result and the AT power results. For the AT power results, the current, average and maximum values can be retrieved. The out of tolerance and code channel filter match ratio results retrieved via the CURRent, AVERage and MAXimum command are identical. 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:EVDO:MEASurement<instance>:MEValuation:ACP:AVERage
value: ResultData = driver.multiEval.acp.average.read()
```

Return the out of tolerance result, the code channel filter match ratio result and the AT power results. For the AT power results, the current, average and maximum values can be retrieved. The out of tolerance and code channel filter match ratio results retrieved via the CURRent, AVERage and MAXimum command are identical. 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.

7.4.5.3 Maximum

SCPI Commands

```
READ:EVDO:MEASurement<Instance>:MEValuation:ACP:MAXimum
FETCh:EVDO:MEASurement<Instance>:MEValuation:ACP:MAXimum
CALCulate:EVDO:MEASurement<Instance>:MEValuation:ACP:MAXimum
```

class Maximum

Maximum commands group definition. 3 total commands, 0 Sub-groups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: decimal 'Reliability Indicator'
- At_Power_Narrow: float: float Access terminal power, measured with a filter bandwidth of 1.23 MHz. Range: -100 dBm to 50 dBm, Unit: dBm
- At_Power_Wide: float: float Access terminal power, measured with the wideband filter. Range: -100 dBm to 50 dBm, Unit: dBm
- Out_Of_Tol_Count: float: float Out of tolerance result, i.e. percentage of measurement intervals of the statistic count (see [CMDLINK: CONFigure:EVDO:MEASi:MEValuation:SCOunt:SPECtrum CMDLINK]) exceeding the specified limits, see 'Limits (Spectrum) '. Range: 0 % to 100 %, Unit: %

- Code_Ch_Filter: float: float Code channel filter match ratio, i.e. percentage of measurement intervals matching the specified code channel filter, see 'Multi-Evaluation: Code Channel Filter'. Range: 0 % to 100 %, Unit: %

class ResultData

Response structure. Fields:

- Reliability: int: decimal 'Reliability Indicator'
- At_Power_Narrow: float: float Access terminal power, measured with a filter bandwidth of 1.23 MHz. Range: -100 dBm to 50 dBm, Unit: dBm
- At_Power_Wide: float: float Access terminal power, measured with the wideband filter. Range: -100 dBm to 50 dBm, Unit: dBm
- Out_Of_Tol_Count: float: float Out of tolerance result, i.e. percentage of measurement intervals of the statistic count (see [CMDLINK: CONFigure:EVDO:MEASi:MEValuation:SCOunt:SPECtrum CMDLINK]) exceeding the specified limits, see 'Limits (Spectrum) '. Range: 0 % to 100 %, Unit: %
- Code_Ch_Filter: float: float Code channel filter match ratio, i.e. percentage of measurement intervals matching the specified code channel filter, see 'Multi-Evaluation: Code Channel Filter'. Range: 0 % to 100 %, Unit: %

calculate() → CalculateStruct

```
# SCPI: CALCulate:EVDO:MEASurement<instance>:MEValuation:ACP:MAXimum
value: CalculateStruct = driver.multiEval.acp.maximum.calculate()
```

Return the out of tolerance result, the code channel filter match ratio result and the AT power results. For the AT power results, the current, average and maximum values can be retrieved. The out of tolerance and code channel filter match ratio results retrieved via the CURRent, AVERage and MAXimum command are identical. 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:EVDO:MEASurement<instance>:MEValuation:ACP:MAXimum
value: ResultData = driver.multiEval.acp.maximum.fetch()
```

Return the out of tolerance result, the code channel filter match ratio result and the AT power results. For the AT power results, the current, average and maximum values can be retrieved. The out of tolerance and code channel filter match ratio results retrieved via the CURRent, AVERage and MAXimum command are identical. 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:EVDO:MEASurement<instance>:MEValuation:ACP:MAXimum
value: ResultData = driver.multiEval.acp.maximum.read()
```

Return the out of tolerance result, the code channel filter match ratio result and the AT power results. For the AT power results, the current, average and maximum values can be retrieved. The out of tolerance and code channel filter match ratio results retrieved via the CURRent, AVERage and MAXimum command

are identical. 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.

7.4.6 Obw<Obw>

RepCap Settings

```
# Range: Nr1 .. Nr4
rc = driver.multiEval.obw.repcap_obw_get()
driver.multiEval.obw.repcap_obw_set(repcap.Obw.Nr1)
```

class Obw

Obw commands group definition. 9 total commands, 3 Sub-groups, 0 group commands Repeated Capability: Obw, default value after init: Obw.Nr1

Cloning the Group

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

Subgroups

7.4.6.1 Current

SCPI Commands

```
READ:EVDO:MEASurement<Instance>:MEValuation:OBW<Obw>:CURRent
FETCh:EVDO:MEASurement<Instance>:MEValuation:OBW<Obw>:CURRent
CALCulate:EVDO:MEASurement<Instance>:MEValuation:OBW<Obw>:CURRent
```

class Current

Current commands group definition. 3 total commands, 0 Sub-groups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: decimal 'Reliability Indicator'
- Obw: float: float Occupied bandwidth Range: 0 MHz to 16 MHz , Unit: Hz
- Out_Of_Tol_Count: float: float Out of tolerance result, i.e. percentage of measurement intervals of the statistic count [CMDLINK: CONFigure:EVDO:MEASi:MEValuation:SCOunt:SPECtrum CMDLINK] exceeding the specified limit, see 'Limits (Spectrum) '. Range: 0 % to 100 %
- Code_Ch_Filter: float: float Code channel filter match ratio, i.e. percentage of measurement intervals matching the specified code channel filter, see 'Common Elements of Views'. Range: 0 % to 100 %, Unit: %

class ResultData

Response structure. Fields:

- Reliability: int: decimal 'Reliability Indicator'

- Obw: float: float Occupied bandwidth Range: 0 MHz to 16 MHz , Unit: Hz
- Out_Of_Tol_Count: float: float Out of tolerance result, i.e. percentage of measurement intervals of the statistic count [CMDLINK: CONFigure:EVDO:MEASi:MEValuation:SCOunt:SPECtrum CMDLINK] exceeding the specified limit, see 'Limits (Spectrum) '. Range: 0 % to 100 %
- Code_Ch_Filter: float: float Code channel filter match ratio, i.e. percentage of measurement intervals matching the specified code channel filter, see 'Common Elements of Views'. Range: 0 % to 100 %, Unit: %

calculate(*obw=<Obw.Default: -1>*) → CalculateStruct

```
# SCPI: CALCulate:EVDO:MEASurement<instance>:MEValuation:OBW<Number>:CURRent
value: CalculateStruct = driver.multiEval.obw.current.calculate(obw = repcap.
↪Obw.Default)
```

Return the current, average and maximum occupied bandwidth value result and the 'out of tolerance' result, the percentage of measurement intervals of the statistic counts exceeding the specified limits. 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. For single-carrier configurations (i.e. only carrier 0 is active) the <Number> suffix can be omitted to obtain the OBW results. For multi-carrier configurations the <Number> suffixes are used as follows:

INTRO_CMD_HELP: Starts, stops, or aborts the measurement:

- For an active and isolated carrier i, use <Number> = i+1 to get its OBW results (i = 0, 1, 2)
- Use <Number> = 4 to display the OBW results of the 'Overall Carrier'
- If all active carriers are adjacent, use <Number>=4 to get the group (overall) OBW results
- If three carriers are active and exactly two carriers i,j with 0i<j2 are adjacent, use <Number> = i+1 for the joint OBW results of adjacent carriers.

param obw optional repeated capability selector. Default value: Nr1 (settable in the interface 'Obw')

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

fetch(*obw=<Obw.Default: -1>*) → ResultData

```
# SCPI: FETCh:EVDO:MEASurement<instance>:MEValuation:OBW<Number>:CURRent
value: ResultData = driver.multiEval.obw.current.fetch(obw = repcap.Obw.Default)
```

Return the current, average and maximum occupied bandwidth value result and the 'out of tolerance' result, the percentage of measurement intervals of the statistic counts exceeding the specified limits. 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. For single-carrier configurations (i.e. only carrier 0 is active) the <Number> suffix can be omitted to obtain the OBW results. For multi-carrier configurations the <Number> suffixes are used as follows:

INTRO_CMD_HELP: Starts, stops, or aborts the measurement:

- For an active and isolated carrier i, use <Number> = i+1 to get its OBW results (i = 0, 1, 2)
- Use <Number> = 4 to display the OBW results of the 'Overall Carrier'
- If all active carriers are adjacent, use <Number>=4 to get the group (overall) OBW results

- If three carriers are active and exactly two carriers i,j with 0i<j2 are adjacent, use <Number> = i+1 for the joint OBW results of adjacent carriers.

param obw optional repeated capability selector. Default value: Nr1 (settable in the interface 'Obw')

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

read(*obw=<Obw.Default: -1>*) → ResultData

```
# SCPI: READ:EVDO:MEASurement<instance>:MEValuation:OBW<Number>:CURRent
value: ResultData = driver.multiEval.obw.current.read(obw = repcap.Obw.Default)
```

Return the current, average and maximum occupied bandwidth value result and the 'out of tolerance' result, the percentage of measurement intervals of the statistic counts exceeding the specified limits. 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. For single-carrier configurations (i.e. only carrier 0 is active) the <Number> suffix can be omitted to obtain the OBW results. For multi-carrier configurations the <Number> suffixes are used as follows:

INTRO_CMD_HELP: Starts, stops, or aborts the measurement:

- For an active and isolated carrier i, use <Number> = i+1 to get its OBW results (i = 0, 1, 2)
- Use <Number> = 4 to display the OBW results of the 'Overall Carrier'
- If all active carriers are adjacent, use <Number>=4 to get the group (overall) OBW results
- If three carriers are active and exactly two carriers i,j with 0i<j2 are adjacent, use <Number> = i+1 for the joint OBW results of adjacent carriers.

param obw optional repeated capability selector. Default value: Nr1 (settable in the interface 'Obw')

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

7.4.6.2 Average

SCPI Commands

```
READ:EVDO:MEASurement<Instance>:MEValuation:OBW<Obw>:AVERage
FETCh:EVDO:MEASurement<Instance>:MEValuation:OBW<Obw>:AVERage
CALCulate:EVDO:MEASurement<Instance>:MEValuation:OBW<Obw>:AVERage
```

class Average

Average commands group definition. 3 total commands, 0 Sub-groups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: decimal 'Reliability Indicator'
- Obw: float: float Occupied bandwidth Range: 0 MHz to 16 MHz , Unit: Hz
- Out_Of_Tol_Count: float: float Out of tolerance result, i.e. percentage of measurement intervals of the statistic count [CMDLINK: CONFigure:EVDO:MEASi:MEValuation:SCOunt:SPECtrum CMDLINK] exceeding the specified limit, see 'Limits (Spectrum) '. Range: 0 % to 100 %

- Code_Ch_Filter: float: float Code channel filter match ratio, i.e. percentage of measurement intervals matching the specified code channel filter, see 'Common Elements of Views'. Range: 0 % to 100 %, Unit: %

class ResultData

Response structure. Fields:

- Reliability: int: decimal 'Reliability Indicator'
- Obw: float: float Occupied bandwidth Range: 0 MHz to 16 MHz , Unit: Hz
- Out_Of_Tol_Count: float: float Out of tolerance result, i.e. percentage of measurement intervals of the statistic count [CMDLINK: CONFigure:EVDO:MEASi:MEValuation:SCOunt:SPECtrum CMDLINK] exceeding the specified limit, see 'Limits (Spectrum) '. Range: 0 % to 100 %
- Code_Ch_Filter: float: float Code channel filter match ratio, i.e. percentage of measurement intervals matching the specified code channel filter, see 'Common Elements of Views'. Range: 0 % to 100 %, Unit: %

calculate(*obw=<Obw.Default: -1>*) → CalculateStruct

```
# SCPI: CALCulate:EVDO:MEASurement<instance>:MEValuation:OBW<Number>:AVERage
value: CalculateStruct = driver.multiEval.obw.average.calculate(obw = repcap.
↪Obw.Default)
```

Return the current, average and maximum occupied bandwidth value result and the 'out of tolerance' result, the percentage of measurement intervals of the statistic counts exceeding the specified limits. 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. For single-carrier configurations (i.e. only carrier 0 is active) the <Number> suffix can be omitted to obtain the OBW results. For multi-carrier configurations the <Number> suffixes are used as follows:

INTRO_CMD_HELP: Starts, stops, or aborts the measurement:

- For an active and isolated carrier i, use <Number> = i+1 to get its OBW results (i = 0, 1, 2)
- Use <Number> = 4 to display the OBW results of the 'Overall Carrier'
- If all active carriers are adjacent, use <Number>=4 to get the group (overall) OBW results
- If three carriers are active and exactly two carriers i,j with 0i<j2 are adjacent, use <Number> = i+1 for the joint OBW results of adjacent carriers.

param obw optional repeated capability selector. Default value: Nr1 (settable in the interface 'Obw')

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

fetch(*obw=<Obw.Default: -1>*) → ResultData

```
# SCPI: FETCh:EVDO:MEASurement<instance>:MEValuation:OBW<Number>:AVERage
value: ResultData = driver.multiEval.obw.average.fetch(obw = repcap.Obw.Default)
```

Return the current, average and maximum occupied bandwidth value result and the 'out of tolerance' result, the percentage of measurement intervals of the statistic counts exceeding the specified limits. 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. For single-carrier configurations (i.e. only carrier 0 is active) the <Number> suffix can be omitted to obtain the OBW results. For multi-carrier configurations the <Number> suffixes are used as follows:

INTRO_CMD_HELP: Starts, stops, or aborts the measurement:

- For an active and isolated carrier i, use <Number> = i+1 to get its OBW results (i = 0, 1, 2)
- Use <Number> = 4 to display the OBW results of the 'Overall Carrier'
- If all active carriers are adjacent, use <Number>=4 to get the group (overall) OBW results
- If three carriers are active and exactly two carriers i,j with 0i<j2 are adjacent, use <Number> = i+1 for the joint OBW results of adjacent carriers.

param obw optional repeated capability selector. Default value: Nr1 (settable in the interface 'Obw')

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

read(*obw=<Obw.Default: -1>*) → ResultData

```
# SCPI: READ:EVDO:MEASurement<instance>:MEValuation:OBW<Number>:AVERage
value: ResultData = driver.multiEval.obw.average.read(obw = repcap.Obw.Default)
```

Return the current, average and maximum occupied bandwidth value result and the 'out of tolerance' result, the percentage of measurement intervals of the statistic counts exceeding the specified limits. 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. For single-carrier configurations (i.e. only carrier 0 is active) the <Number> suffix can be omitted to obtain the OBW results. For multi-carrier configurations the <Number> suffixes are used as follows:

INTRO_CMD_HELP: Starts, stops, or aborts the measurement:

- For an active and isolated carrier i, use <Number> = i+1 to get its OBW results (i = 0, 1, 2)
- Use <Number> = 4 to display the OBW results of the 'Overall Carrier'
- If all active carriers are adjacent, use <Number>=4 to get the group (overall) OBW results
- If three carriers are active and exactly two carriers i,j with 0i<j2 are adjacent, use <Number> = i+1 for the joint OBW results of adjacent carriers.

param obw optional repeated capability selector. Default value: Nr1 (settable in the interface 'Obw')

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

7.4.6.3 Maximum

SCPI Commands

```
READ:EVDO:MEASurement<Instance>:MEValuation:OBW<Obw>:MAXimum
FETCh:EVDO:MEASurement<Instance>:MEValuation:OBW<Obw>:MAXimum
CALCulate:EVDO:MEASurement<Instance>:MEValuation:OBW<Obw>:MAXimum
```

class Maximum

Maximum commands group definition. 3 total commands, 0 Sub-groups, 3 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: decimal 'Reliability Indicator'

- Obw: float: float Occupied bandwidth Range: 0 MHz to 16 MHz , Unit: Hz
- Out_Of_Tol_Count: float: float Out of tolerance result, i.e. percentage of measurement intervals of the statistic count [CMDLINK: CONFigure:EVDO:MEASi:MEValuation:SCOunt:SPECtrum CMDLINK] exceeding the specified limit, see 'Limits (Spectrum) '. Range: 0 % to 100 %
- Code_Ch_Filter: float: float Code channel filter match ratio, i.e. percentage of measurement intervals matching the specified code channel filter, see 'Common Elements of Views'. Range: 0 % to 100 %, Unit: %

class ResultData

Response structure. Fields:

- Reliability: int: decimal 'Reliability Indicator'
- Obw: float: float Occupied bandwidth Range: 0 MHz to 16 MHz , Unit: Hz
- Out_Of_Tol_Count: float: float Out of tolerance result, i.e. percentage of measurement intervals of the statistic count [CMDLINK: CONFigure:EVDO:MEASi:MEValuation:SCOunt:SPECtrum CMDLINK] exceeding the specified limit, see 'Limits (Spectrum) '. Range: 0 % to 100 %
- Code_Ch_Filter: float: float Code channel filter match ratio, i.e. percentage of measurement intervals matching the specified code channel filter, see 'Common Elements of Views'. Range: 0 % to 100 %, Unit: %

calculate(*obw=<Obw.Default: -1>*) → CalculateStruct

```
# SCPI: CALCulate:EVDO:MEASurement<instance>:MEValuation:OBW<Number>:MAXimum
value: CalculateStruct = driver.multiEval.obw.maximum.calculate(obw = repcap.
↪Obw.Default)
```

Return the current, average and maximum occupied bandwidth value result and the 'out of tolerance' result, the percentage of measurement intervals of the statistic counts exceeding the specified limits. 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. For single-carrier configurations (i.e. only carrier 0 is active) the <Number> suffix can be omitted to obtain the OBW results. For multi-carrier configurations the <Number> suffixes are used as follows:

INTRO_CMD_HELP: Starts, stops, or aborts the measurement:

- For an active and isolated carrier i, use <Number> = i+1 to get its OBW results (i = 0, 1, 2)
- Use <Number> = 4 to display the OBW results of the 'Overall Carrier'
- If all active carriers are adjacent, use <Number>=4 to get the group (overall) OBW results
- If three carriers are active and exactly two carriers i,j with 0i<j2 are adjacent, use <Number> = i+1 for the joint OBW results of adjacent carriers.

param obw optional repeated capability selector. Default value: Nr1 (settable in the interface 'Obw')

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

fetch(*obw=<Obw.Default: -1>*) → ResultData

```
# SCPI: FETCh:EVDO:MEASurement<instance>:MEValuation:OBW<Number>:MAXimum
value: ResultData = driver.multiEval.obw.maximum.fetch(obw = repcap.Obw.Default)
```

Return the current, average and maximum occupied bandwidth value result and the 'out of tolerance' result, the percentage of measurement intervals of the statistic counts exceeding the specified limits. 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. For single-carrier configurations (i.e. only carrier 0 is active) the <Number> suffix can be omitted to obtain the OBW results. For multi-carrier configurations the <Number> suffixes are used as follows:

INTRO_CMD_HELP: Starts, stops, or aborts the measurement:

- For an active and isolated carrier i, use <Number> = i+1 to get its OBW results (i = 0, 1, 2)
- Use <Number> = 4 to display the OBW results of the 'Overall Carrier'
- If all active carriers are adjacent, use <Number>=4 to get the group (overall) OBW results
- If three carriers are active and exactly two carriers i,j with 0i<j2 are adjacent, use <Number> = i+1 for the joint OBW results of adjacent carriers.

param obw optional repeated capability selector. Default value: Nr1 (settable in the interface 'Obw')

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

read(*obw=<Obw.Default: -1>*) → ResultData

```
# SCPI: READ:EVDO:MEASurement<instance>:MEValuation:OBW<Number>:MAXimum
value: ResultData = driver.multiEval.obw.maximum.read(obw = repcap.Obw.Default)
```

Return the current, average and maximum occupied bandwidth value result and the 'out of tolerance' result, the percentage of measurement intervals of the statistic counts exceeding the specified limits. 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. For single-carrier configurations (i.e. only carrier 0 is active) the <Number> suffix can be omitted to obtain the OBW results. For multi-carrier configurations the <Number> suffixes are used as follows:

INTRO_CMD_HELP: Starts, stops, or aborts the measurement:

- For an active and isolated carrier i, use <Number> = i+1 to get its OBW results (i = 0, 1, 2)
- Use <Number> = 4 to display the OBW results of the 'Overall Carrier'
- If all active carriers are adjacent, use <Number>=4 to get the group (overall) OBW results
- If three carriers are active and exactly two carriers i,j with 0i<j2 are adjacent, use <Number> = i+1 for the joint OBW results of adjacent carriers.

param obw optional repeated capability selector. Default value: Nr1 (settable in the interface 'Obw')

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

7.4.7 ListPy

class ListPy
ListPy commands group definition. 368 total commands, 7 Sub-groups, 0 group commands

Cloning the Group

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

Subgroups

7.4.7.1 Sreliability

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:SRELiability
CALCulate:EVDO:MEASurement<Instance>:MEValuation:LIST:SRELiability
```

class Sreliability
Sreliability commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

calculate() → List[RsCmwEvdoMeas.enums.ResultStatus2]

```
# SCPI: CALCulate:EVDO:MEASurement<instance>:MEValuation:LIST:SRELiability
value: List[enums.ResultStatus2] = driver.multiEval.listPy.sreliability.
↪calculate()
```

Returns the segment reliabilities for all active list mode segments. A common reliability indicator of zero indicates that the results in all measured segments are valid. A non-zero value indicates that an error occurred in at least one of the measured segments. If you get a non-zero common reliability indicator, you can use this command to retrieve the individual reliability values of all measured segments for further analysis. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return seg_reliability: decimal Comma-separated list of values, one per active segment. The segment reliability indicates whether one of the following exceptions occurred in this segment: 0: No error 3: Signal overflow 4: Signal low 8: Synchronization error If a combination of exceptions occurs, the most severe error is indicated.

fetch() → List[int]

```
# SCPI: FETCh:EVDO:MEASurement<instance>:MEValuation:LIST:SRELiability
value: List[int] = driver.multiEval.listPy.sreliability.fetch()
```

Returns the segment reliabilities for all active list mode segments. A common reliability indicator of zero indicates that the results in all measured segments are valid. A non-zero value indicates that an error occurred in at least one of the measured segments. If you get a non-zero common reliability indicator, you can use this command to retrieve the individual reliability values of all measured segments for further

analysis. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return seg_reliability: decimal Comma-separated list of values, one per active segment. The segment reliability indicates whether one of the following exceptions occurred in this segment: 0: No error 3: Signal overflow 4: Signal low 8: Synchronization error If a combination of exceptions occurs, the most severe error is indicated.

7.4.7.2 Modulation

class Modulation

Modulation commands group definition. 124 total commands, 17 Sub-groups, 0 group commands

Cloning the Group

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

Subgroups

7.4.7.2.1 Otolerance

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:MODulation:OTOLerance
```

class Otolerance

Otolerance commands group definition. 1 total commands, 0 Sub-groups, 1 group commands

fetch() → List[int]

```
# SCPI: FETCh:EVDO:MEASurement<instance>:MEValuation:LIST:MODulation:OTOLerance
value: List[int] = driver.multiEval.listPy.modulation.otolerance.fetch()
```

Returns the out of tolerance count of the modulation measurement for all active list mode segments.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return out_of_tol_count: decimal The percentage of measurement intervals of the statistic count exceeding the specified limits. Comma-separated list of values, one per active segment. Range: 0 % to 100 % , Unit: %

7.4.7.2.2 StCount

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:MODulation:STCount
```

class StCount

StCount commands group definition. 1 total commands, 0 Sub-groups, 1 group commands

fetch() → List[int]

```
# SCPI: FETCh:EVDO:MEASurement<instance>:MEValuation:LIST:MODulation:STCount
value: List[int] = driver.multiEval.listPy.modulation.stCount.fetch()
```

Returns the statistic count in the modulation measurement for all active list mode segments.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return statistic_count: decimal The number of evaluated valid slots. Comma-separated list of values, one per active segment. Range: 0 to 1000

7.4.7.2.3 Evm

class Evm

Evm commands group definition. 16 total commands, 2 Sub-groups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.multiEval.listPy.modulation.evm.clone()
```

Subgroups

7.4.7.2.3.1 Rms

class Rms

Rms commands group definition. 8 total commands, 4 Sub-groups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.multiEval.listPy.modulation.evm.rms.clone()
```

Subgroups

7.4.7.2.3.2 Current

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:MODulation:EVM:RMS:CURRent
CALCulate:EVDO:MEASurement<Instance>:MEValuation:LIST:MODulation:EVM:RMS:CURRent
```

class Current

Current commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

calculate() → List[float]

```
# SCPI: CALCulate:EVDO:MEASurement<instance>
↪:MEValuation:LIST:MODulation:EVM:RMS:CURRent
value: List[float] = driver.multiEval.listPy.modulation.evm.rms.current.
↪calculate()
```

Returns EVM peak and RMS (statistical) values for all active list mode segments. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return evm_rms: float Comma-separated list of values, one per active segment. Range: 0 % to 100 %, Unit: %

fetch() → List[float]

```
# SCPI: FETCh:EVDO:MEASurement<instance>
↪:MEValuation:LIST:MODulation:EVM:RMS:CURRent
value: List[float] = driver.multiEval.listPy.modulation.evm.rms.current.fetch()
```

Returns EVM peak and RMS (statistical) values for all active list mode segments. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return evm_rms: float Comma-separated list of values, one per active segment. Range: 0 % to 100 %, Unit: %

7.4.7.2.3.3 Average

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:MODulation:EVM:RMS:AVERage
CALCulate:EVDO:MEASurement<Instance>:MEValuation:LIST:MODulation:EVM:RMS:AVERage
```

class Average

Average commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

calculate() → List[float]

```
# SCPI: CALCulate:EVDO:MEASurement<instance>
↪:MEValuation:LIST:MODulation:EVM:RMS:AVERage
value: List[float] = driver.multiEval.listPy.modulation.evm.rms.average.
↪calculate()
```

Returns EVM peak and RMS (statistical) values for all active list mode segments. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return evm_rms: float Comma-separated list of values, one per active segment. Range: 0 % to 100 %, Unit: %

fetch() → List[float]

```
# SCPI: FETCh:EVDO:MEASurement<instance>
↪:MEValuation:LIST:MODulation:EVM:RMS:AVERage
value: List[float] = driver.multiEval.listPy.modulation.evm.rms.average.fetch()
```

Returns EVM peak and RMS (statistical) values for all active list mode segments. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return evm_rms: float Comma-separated list of values, one per active segment. Range: 0 % to 100 %, Unit: %

7.4.7.2.3.4 Maximum

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:MODulation:EVM:RMS:MAXimum
CALCulate:EVDO:MEASurement<Instance>:MEValuation:LIST:MODulation:EVM:RMS:MAXimum
```

class Maximum

Maximum commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

calculate() → List[float]

```
# SCPI: CALCulate:EVDO:MEASurement<instance>
↪:MEValuation:LIST:MODulation:EVM:RMS:MAXimum
value: List[float] = driver.multiEval.listPy.modulation.evm.rms.maximum.
↪calculate()
```

Returns EVM peak and RMS (statistical) values for all active list mode segments. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return evm_rms: float Comma-separated list of values, one per active segment. Range: 0 % to 100 %, Unit: %

fetch() → List[float]

```
# SCPI: FETCh:EVDO:MEASurement<instance>
↪:MEValuation:LIST:MODulation:EVM:RMS:MAXimum
value: List[float] = driver.multiEval.listPy.modulation.evm.rms.maximum.fetch()
```

Returns EVM peak and RMS (statistical) values for all active list mode segments. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return evm_rms: float Comma-separated list of values, one per active segment. Range: 0 % to 100 %, Unit: %

7.4.7.2.3.5 StandardDev

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:MODulation:EVM:RMS:SDEViation
CALCulate:EVDO:MEASurement<Instance>:MEValuation:LIST:MODulation:EVM:RMS:SDEViation
```

class StandardDev

StandardDev commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

calculate() → List[float]

```
# SCPI: CALCulate:EVDO:MEASurement<instance>
↪:MEValuation:LIST:MODulation:EVM:RMS:SDEViation
value: List[float] = driver.multiEval.listPy.modulation.evm.rms.standardDev.
↪calculate()
```

Returns EVM peak and RMS (statistical) values for all active list mode segments. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return evm_rms: float Comma-separated list of values, one per active segment. Range: 0 % to 100 %, Unit: %

fetch() → List[float]

```
# SCPI: FETCh:EVDO:MEASurement<instance>
↪:MEValuation:LIST:MODulation:EVM:RMS:SDEViation
value: List[float] = driver.multiEval.listPy.modulation.evm.rms.standardDev.
↪fetch()
```

Returns EVM peak and RMS (statistical) values for all active list mode segments. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return evm_rms: float Comma-separated list of values, one per active segment. Range: 0 % to 100 %, Unit: %

7.4.7.2.3.6 Peak

class Peak

Peak commands group definition. 8 total commands, 4 Sub-groups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.multiEval.listPy.modulation.evm.peak.clone()
```

Subgroups

7.4.7.2.3.7 Current

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:MODulation:EVM:PEAK:CURRent
CALCulate:EVDO:MEASurement<Instance>:MEValuation:LIST:MODulation:EVM:PEAK:CURRent
```

class Current

Current commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

calculate() → List[float]

```
# SCPI: CALCulate:EVDO:MEASurement<instance>
↪:MEValuation:LIST:MODulation:EVM:PEAK:CURRent
value: List[float] = driver.multiEval.listPy.modulation.evm.peak.current.
↪calculate()
```

Returns EVM peak and RMS (statistical) values for all active list mode segments. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return evm_peak: float Comma-separated list of values, one per active segment. Range: 0 % to 100 %, Unit: %

fetch() → List[float]

```
# SCPI: FETCh:EVDO:MEASurement<instance>
↪:MEValuation:LIST:MODulation:EVM:PEAK:CURRent
value: List[float] = driver.multiEval.listPy.modulation.evm.peak.current.fetch()
```

Returns EVM peak and RMS (statistical) values for all active list mode segments. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return evm_peak: float Comma-separated list of values, one per active segment. Range: 0 % to 100 %, Unit: %

7.4.7.2.3.8 Average

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:MODulation:EVM:PEAK:AVERage
CALCulate:EVDO:MEASurement<Instance>:MEValuation:LIST:MODulation:EVM:PEAK:AVERage
```

class Average

Average commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

calculate() → List[float]

```
# SCPI: CALCulate:EVDO:MEASurement<instance>
↪:MEValuation:LIST:MODulation:EVM:PEAK:AVERage
value: List[float] = driver.multiEval.listPy.modulation.evm.peak.average.
↪calculate()
```

Returns EVM peak and RMS (statistical) values for all active list mode segments. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return evm_peak: float Comma-separated list of values, one per active segment. Range: 0 % to 100 %, Unit: %

fetch() → List[float]

```
# SCPI: FETCh:EVDO:MEASurement<instance>
↪:MEValuation:LIST:MODulation:EVM:PEAK:AVERage
value: List[float] = driver.multiEval.listPy.modulation.evm.peak.average.fetch()
```

Returns EVM peak and RMS (statistical) values for all active list mode segments. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return evm_peak: float Comma-separated list of values, one per active segment. Range: 0 % to 100 %, Unit: %

7.4.7.2.3.9 Maximum

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:MODulation:EVM:PEAK:MAXimum
CALCulate:EVDO:MEASurement<Instance>:MEValuation:LIST:MODulation:EVM:PEAK:MAXimum
```

class Maximum

Maximum commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

calculate() → List[float]

```
# SCPI: CALCulate:EVDO:MEASurement<instance>
↪:MEValuation:LIST:MODulation:EVM:PEAK:MAXimum
value: List[float] = driver.multiEval.listPy.modulation.evm.peak.maximum.
↪calculate()
```

Returns EVM peak and RMS (statistical) values for all active list mode segments. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return evm_peak: float Comma-separated list of values, one per active segment. Range: 0 % to 100 %, Unit: %

fetch() → List[float]

```
# SCPI: FETCh:EVDO:MEASurement<instance>
↪:MEValuation:LIST:MODulation:EVM:PEAK:MAXimum
value: List[float] = driver.multiEval.listPy.modulation.evm.peak.maximum.fetch()
```

Returns EVM peak and RMS (statistical) values for all active list mode segments. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return evm_peak: float Comma-separated list of values, one per active segment. Range: 0 % to 100 %, Unit: %

7.4.7.2.3.10 StandardDev

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:MODulation:EVM:PEAK:SDEViation
CALCulate:EVDO:MEASurement<Instance>:MEValuation:LIST:MODulation:EVM:PEAK:SDEViation
```

class StandardDev

StandardDev commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

calculate() → List[float]

```
# SCPI: CALCulate:EVDO:MEASurement<instance>
↪:MEValuation:LIST:MODulation:EVM:PEAK:SDEViation
value: List[float] = driver.multiEval.listPy.modulation.evm.peak.standardDev.
↪calculate()
```

Returns EVM peak and RMS (statistical) values for all active list mode segments. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return evm_peak: float Comma-separated list of values, one per active segment. Range: 0 % to 100 %, Unit: %

fetch() → List[float]

```
# SCPI: FETCh:EVDO:MEASurement<instance>
↪:MEValuation:LIST:MODulation:EVM:PEAK:SDEViation
value: List[float] = driver.multiEval.listPy.modulation.evm.peak.standardDev.
↪fetch()
```

Returns EVM peak and RMS (statistical) values for all active list mode segments. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return evm_peak: float Comma-separated list of values, one per active segment. Range: 0 % to 100 %, Unit: %

7.4.7.2.4 Merror

class Merror

Merror commands group definition. 16 total commands, 2 Sub-groups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.multiEval.listPy.modulation.merror.clone()
```

Subgroups

7.4.7.2.4.1 Rms

class Rms

Rms commands group definition. 8 total commands, 4 Sub-groups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.multiEval.listPy.modulation.merror.rms.clone()
```

Subgroups

7.4.7.2.4.2 Current

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:MODulation:MERRor:RMS:CURRent
CALCulate:EVDO:MEASurement<Instance>:MEValuation:LIST:MODulation:MERRor:RMS:CURRent
```

class Current

Current commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

calculate() → List[float]

```
# SCPI: CALCulate:EVDO:MEASurement<instance>
↪:MEValuation:LIST:MODulation:MERRor:RMS:CURRent
value: List[float] = driver.multiEval.listPy.modulation.merror.rms.current.
↪calculate()
```

Returns peak and RMS magnitude error (statistical) values for all active list mode segments. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return merr_rms: float Comma-separated list of values, one per active segment. Range: -100 % to +100 % (AVERage: 0% to 100 %, SDEViation: 0 % to 50 %) , Unit: %

fetch() → List[float]

```
# SCPI: FETCh:EVDO:MEASurement<instance>
↪:MEValuation:LIST:MODulation:MERRor:RMS:CURRent
value: List[float] = driver.multiEval.listPy.modulation.merror.rms.current.
↪fetch()
```

Returns peak and RMS magnitude error (statistical) values for all active list mode segments. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return merr_rms: float Comma-separated list of values, one per active segment. Range: -100 % to +100 % (AVERage: 0% to 100 %, SDEViation: 0 % to 50 %) , Unit: %

7.4.7.2.4.3 Average

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:MODulation:MERRor:RMS:AVERage
CALCulate:EVDO:MEASurement<Instance>:MEValuation:LIST:MODulation:MERRor:RMS:AVERage
```

class Average

Average commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

calculate() → List[float]

```
# SCPI: CALCulate:EVDO:MEASurement<instance>
↪:MEValuation:LIST:MODulation:MERRor:RMS:AVERage
value: List[float] = driver.multiEval.listPy.modulation.merror.rms.average.
↪calculate()
```

Returns peak and RMS magnitude error (statistical) values for all active list mode segments. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return merr_rms: float Comma-separated list of values, one per active segment. Range: -100 % to +100 % (AVERage: 0% to 100 %, SDEViation: 0 % to 50 %) , Unit: %

fetch() → List[float]

```
# SCPI: FETCh:EVDO:MEASurement<instance>
↪:MEValuation:LIST:MODulation:MERRor:RMS:AVERage
value: List[float] = driver.multiEval.listPy.modulation.merror.rms.average.
↪fetch()
```

Returns peak and RMS magnitude error (statistical) values for all active list mode segments. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return merr_rms: float Comma-separated list of values, one per active segment. Range: -100 % to +100 % (AVERage: 0% to 100 %, SDEViation: 0 % to 50 %) , Unit: %

7.4.7.2.4.4 Maximum

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:MODulation:MERRor:RMS:MAXimum
CALCulate:EVDO:MEASurement<Instance>:MEValuation:LIST:MODulation:MERRor:RMS:MAXimum
```

class Maximum

Maximum commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

calculate() → List[float]

```
# SCPI: CALCulate:EVDO:MEASurement<instance>
↪:MEValuation:LIST:MODulation:MERRor:RMS:MAXimum
value: List[float] = driver.multiEval.listPy.modulation.merror.rms.maximum.
↪calculate()
```

Returns peak and RMS magnitude error (statistical) values for all active list mode segments. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return merr_rms: float Comma-separated list of values, one per active segment. Range: -100 % to +100 % (AVERage: 0% to 100 %, SDEViation: 0 % to 50 %) , Unit: %

fetch() → List[float]

```
# SCPI: FETCh:EVDO:MEASurement<instance>
↪:MEValuation:LIST:MODulation:MERRor:RMS:MAXimum
value: List[float] = driver.multiEval.listPy.modulation.merror.rms.maximum.
↪fetch()
```

Returns peak and RMS magnitude error (statistical) values for all active list mode segments. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return merr_rms: float Comma-separated list of values, one per active segment. Range: -100 % to +100 % (AVERage: 0% to 100 %, SDEViation: 0 % to 50 %) , Unit: %

7.4.7.2.4.5 StandardDev

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:MODulation:MERRor:RMS:SDEViation
CALCulate:EVDO:MEASurement<Instance>:MEValuation:LIST:MODulation:MERRor:RMS:SDEViation
```

class StandardDev

StandardDev commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

calculate() → List[float]

```
# SCPI: CALCulate:EVDO:MEASurement<instance>
↪:MEValuation:LIST:MODulation:MERRor:RMS:SDEViation
value: List[float] = driver.multiEval.listPy.modulation.merror.rms.standardDev.
↪calculate()
```

Returns peak and RMS magnitude error (statistical) values for all active list mode segments. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return merr_rms: float Comma-separated list of values, one per active segment. Range: -100 % to +100 % (AVERage: 0% to 100 %, SDEViation: 0 % to 50 %) , Unit: %

fetch() → List[float]

```
# SCPI: FETCh:EVDO:MEASurement<instance>
↪:MEValuation:LIST:MODulation:MERRor:RMS:SDEViation
value: List[float] = driver.multiEval.listPy.modulation.merror.rms.standardDev.
↪fetch()
```

Returns peak and RMS magnitude error (statistical) values for all active list mode segments. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return merr_rms: float Comma-separated list of values, one per active segment. Range: -100 % to +100 % (AVERage: 0% to 100 %, SDEViation: 0 % to 50 %) , Unit: %

7.4.7.2.4.6 Peak

class Peak

Peak commands group definition. 8 total commands, 4 Sub-groups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.multiEval.listPy.modulation.merror.peak.clone()
```

Subgroups

7.4.7.2.4.7 Current

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:MODulation:MERRor:PEAK:CURRent
CALCulate:EVDO:MEASurement<Instance>:MEValuation:LIST:MODulation:MERRor:PEAK:CURRent
```

class Current

Current commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

calculate() → List[float]

```
# SCPI: CALCulate:EVDO:MEASurement<instance>
↪:MEValuation:LIST:MODulation:MERRor:PEAK:CURRent
value: List[float] = driver.multiEval.listPy.modulation.merror.peak.current.
↪calculate()
```

Returns peak and RMS magnitude error (statistical) values for all active list mode segments. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return merr_peak: float Comma-separated list of values, one per active segment. Range: -100 % to +100 % (AVERage: 0% to 100 %, SDEViation: 0 % to 50 %) , Unit: %

fetch() → List[float]

```
# SCPI: FETCh:EVDO:MEASurement<instance>
↪:MEValuation:LIST:MODulation:MERRor:PEAK:CURRent
value: List[float] = driver.multiEval.listPy.modulation.merror.peak.current.
↪fetch()
```

Returns peak and RMS magnitude error (statistical) values for all active list mode segments. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return merr_peak: float Comma-separated list of values, one per active segment. Range: -100 % to +100 % (AVERage: 0% to 100 %, SDEViation: 0 % to 50 %) , Unit: %

7.4.7.2.4.8 Average

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:MODulation:MERRor:PEAK:AVERage
CALCulate:EVDO:MEASurement<Instance>:MEValuation:LIST:MODulation:MERRor:PEAK:AVERage
```

class Average

Average commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

calculate() → List[float]

```
# SCPI: CALCulate:EVDO:MEASurement<instance>
↪:MEValuation:LIST:MODulation:MERRor:PEAK:AVERage
value: List[float] = driver.multiEval.listPy.modulation.merror.peak.average.
↪calculate()
```

Returns peak and RMS magnitude error (statistical) values for all active list mode segments. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return merr_peak: float Comma-separated list of values, one per active segment. Range: -100 % to +100 % (AVERage: 0% to 100 %, SDEViation: 0 % to 50 %) , Unit: %

fetch() → List[float]

```
# SCPI: FETCh:EVDO:MEASurement<instance>
↪:MEValuation:LIST:MODulation:MERRor:PEAK:AVERage
value: List[float] = driver.multiEval.listPy.modulation.merror.peak.average.
↪fetch()
```

Returns peak and RMS magnitude error (statistical) values for all active list mode segments. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return merr_peak: float Comma-separated list of values, one per active segment. Range: -100 % to +100 % (AVERage: 0% to 100 %, SDEViation: 0 % to 50 %) , Unit: %

7.4.7.2.4.9 Maximum

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:MODulation:MERRor:PEAK:MAXimum
CALCulate:EVDO:MEASurement<Instance>:MEValuation:LIST:MODulation:MERRor:PEAK:MAXimum
```

class Maximum

Maximum commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

calculate() → List[float]

```
# SCPI: CALCulate:EVDO:MEASurement<instance>
↪:MEValuation:LIST:MODulation:MERRor:PEAK:MAXimum
value: List[float] = driver.multiEval.listPy.modulation.merror.peak.maximum.
↪calculate()
```

Returns peak and RMS magnitude error (statistical) values for all active list mode segments. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return merr_peak: float Comma-separated list of values, one per active segment. Range: -100 % to +100 % (AVERage: 0% to 100 %, SDEViation: 0 % to 50 %) , Unit: %

fetch() → List[float]

```
# SCPI: FETCh:EVDO:MEASurement<instance>
↪:MEValuation:LIST:MODulation:MERRor:PEAK:MAXimum
value: List[float] = driver.multiEval.listPy.modulation.merror.peak.maximum.
↪fetch()
```

Returns peak and RMS magnitude error (statistical) values for all active list mode segments. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return merr_peak: float Comma-separated list of values, one per active segment. Range: -100 % to +100 % (AVERage: 0% to 100 %, SDEViation: 0 % to 50 %) , Unit: %

7.4.7.2.4.10 StandardDev

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:MODulation:MERRor:PEAK:SDEViation
CALCulate:EVDO:MEASurement<Instance>:MEValuation:LIST:MODulation:MERRor:PEAK:SDEViation
```

class StandardDev

StandardDev commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

calculate() → List[float]

```
# SCPI: CALCulate:EVDO:MEASurement<instance>
↪:MEValuation:LIST:MODulation:MERRor:PEAK:SDEViation
value: List[float] = driver.multiEval.listPy.modulation.merror.peak.standardDev.
↪calculate()
```

Returns peak and RMS magnitude error (statistical) values for all active list mode segments. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return merr_peak: float Comma-separated list of values, one per active segment. Range: -100 % to +100 % (AVERage: 0% to 100 %, SDEViation: 0 % to 50 %) , Unit: %

fetch() → List[float]

```
# SCPI: FETCh:EVDO:MEASurement<instance>
↪:MEValuation:LIST:MODulation:MERRor:PEAK:SDEViation
value: List[float] = driver.multiEval.listPy.modulation.merror.peak.standardDev.
↪fetch()
```

Returns peak and RMS magnitude error (statistical) values for all active list mode segments. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return merr_peak: float Comma-separated list of values, one per active segment. Range: -100 % to +100 % (AVERage: 0% to 100 %, SDEViation: 0 % to 50 %) , Unit: %

7.4.7.2.5 Perror

class Perror

Perror commands group definition. 16 total commands, 2 Sub-groups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.multiEval.listPy.modulation.perror.clone()
```

Subgroups

7.4.7.2.5.1 Rms

class Rms
Rms commands group definition. 8 total commands, 4 Sub-groups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.multiEval.listPy.modulation.perror.rms.clone()
```

Subgroups

7.4.7.2.5.2 Current

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:MODulation:PERRor:RMS:CURRent
CALCulate:EVDO:MEASurement<Instance>:MEValuation:LIST:MODulation:PERRor:RMS:CURRent
```

class Current
Current commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

calculate() → List[float]

```
# SCPI: CALCulate:EVDO:MEASurement<instance>
↪:MEValuation:LIST:MODulation:PERRor:RMS:CURRent
value: List[float] = driver.multiEval.listPy.modulation.perror.rms.current.
↪calculate()
```

Returns peak and RMS phase error (statistical) values for all active list mode segments. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return perr_rms: float Comma-separated list of values, one per active segment. Range: 0 deg to 100 deg deg

fetch() → List[float]

```
# SCPI: FETCh:EVDO:MEASurement<instance>
↪:MEValuation:LIST:MODulation:PERRor:RMS:CURRent
value: List[float] = driver.multiEval.listPy.modulation.perror.rms.current.
↪fetch()
```

Returns peak and RMS phase error (statistical) values for all active list mode segments. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return perr_rms: float Comma-separated list of values, one per active segment. Range: 0 deg to 100 deg deg

7.4.7.2.5.3 Average

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:MODulation:PERRor:RMS:AVERage
CALCulate:EVDO:MEASurement<Instance>:MEValuation:LIST:MODulation:PERRor:RMS:AVERage
```

class Average

Average commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

calculate() → List[float]

```
# SCPI: CALCulate:EVDO:MEASurement<instance>
↪:MEValuation:LIST:MODulation:PERRor:RMS:AVERage
value: List[float] = driver.multiEval.listPy.modulation.perror.rms.average.
↪calculate()
```

Returns peak and RMS phase error (statistical) values for all active list mode segments. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return perr_rms: float Comma-separated list of values, one per active segment. Range: 0 deg to 100 deg deg

fetch() → List[float]

```
# SCPI: FETCh:EVDO:MEASurement<instance>
↪:MEValuation:LIST:MODulation:PERRor:RMS:AVERage
value: List[float] = driver.multiEval.listPy.modulation.perror.rms.average.
↪fetch()
```

Returns peak and RMS phase error (statistical) values for all active list mode segments. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return perr_rms: float Comma-separated list of values, one per active segment. Range: 0 deg to 100 deg deg

7.4.7.2.5.4 Maximum

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:MODulation:PERRor:RMS:MAXimum
CALCulate:EVDO:MEASurement<Instance>:MEValuation:LIST:MODulation:PERRor:RMS:MAXimum
```

class Maximum

Maximum commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

calculate() → List[float]

```
# SCPI: CALCulate:EVDO:MEASurement<instance>
↪:MEValuation:LIST:MODulation:PERRor:RMS:MAXimum
value: List[float] = driver.multiEval.listPy.modulation.perror.rms.maximum.
↪calculate()
```

Returns peak and RMS phase error (statistical) values for all active list mode segments. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return perr_rms: float Comma-separated list of values, one per active segment. Range: 0 deg to 100 deg deg

fetch() → List[float]

```
# SCPI: FETCh:EVDO:MEASurement<instance>
↪:MEValuation:LIST:MODulation:PERRor:RMS:MAXimum
value: List[float] = driver.multiEval.listPy.modulation.perror.rms.maximum.
↪fetch()
```

Returns peak and RMS phase error (statistical) values for all active list mode segments. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return perr_rms: float Comma-separated list of values, one per active segment. Range: 0 deg to 100 deg deg

7.4.7.2.5.5 StandardDev

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:MODulation:PERRor:RMS:SDEViation
CALCulate:EVDO:MEASurement<Instance>:MEValuation:LIST:MODulation:PERRor:RMS:SDEViation
```

class StandardDev

StandardDev commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

calculate() → List[float]

```
# SCPI: CALCulate:EVDO:MEASurement<instance>
↪:MEValuation:LIST:MODulation:PERRor:RMS:SDEViation
value: List[float] = driver.multiEval.listPy.modulation.perror.rms.standardDev.
↪calculate()
```

Returns peak and RMS phase error (statistical) values for all active list mode segments. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return perr_rms: float Comma-separated list of values, one per active segment. Range: 0 deg to 100 deg deg

fetch() → List[float]

```
# SCPI: FETCh:EVDO:MEASurement<instance>
↪:MEValuation:LIST:MODulation:PERRor:RMS:SDEViation
value: List[float] = driver.multiEval.listPy.modulation.perror.rms.standardDev.
↪fetch()
```

Returns peak and RMS phase error (statistical) values for all active list mode segments. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return perr_rms: float Comma-separated list of values, one per active segment. Range: 0 deg to 100 deg deg

7.4.7.2.5.6 Peak

class Peak

Peak commands group definition. 8 total commands, 4 Sub-groups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.multiEval.listPy.modulation.perror.peak.clone()
```

Subgroups

7.4.7.2.5.7 Current

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:MODulation:PERRor:PEAK:CURRent
CALCulate:EVDO:MEASurement<Instance>:MEValuation:LIST:MODulation:PERRor:PEAK:CURRent
```

class Current

Current commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

calculate() → List[float]

```
# SCPI: CALCulate:EVDO:MEASurement<instance>
↪:MEValuation:LIST:MODulation:PERRor:PEAK:CURRent
value: List[float] = driver.multiEval.listPy.modulation.perror.peak.current.
↪calculate()
```

Returns peak and RMS phase error (statistical) values for all active list mode segments. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return perr_peak: float Comma-separated list of values, one per active segment. Range: 0 deg to 100 deg deg

fetch() → List[float]

```
# SCPI: FETCh:EVDO:MEASurement<instance>
↪:MEValuation:LIST:MODulation:PERRor:PEAK:CURRent
value: List[float] = driver.multiEval.listPy.modulation.perror.peak.current.
↪fetch()
```

Returns peak and RMS phase error (statistical) values for all active list mode segments. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return perr_peak: float Comma-separated list of values, one per active segment. Range: 0 deg to 100 deg deg

7.4.7.2.5.8 Average

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:MODulation:PERRor:PEAK:AVERage
CALCulate:EVDO:MEASurement<Instance>:MEValuation:LIST:MODulation:PERRor:PEAK:AVERage
```

class Average

Average commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

calculate() → List[float]

```
# SCPI: CALCulate:EVDO:MEASurement<instance>
↪:MEValuation:LIST:MODulation:PERRor:PEAK:AVERage
value: List[float] = driver.multiEval.listPy.modulation.perror.peak.average.
↪calculate()
```

Returns peak and RMS phase error (statistical) values for all active list mode segments. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return perr_peak: float Comma-separated list of values, one per active segment. Range: 0 deg to 100 deg deg

fetch() → List[float]

```
# SCPI: FETCh:EVDO:MEASurement<instance>
↪:MEValuation:LIST:MODulation:PERRor:PEAK:AVERage
value: List[float] = driver.multiEval.listPy.modulation.perror.peak.average.
↪fetch()
```

Returns peak and RMS phase error (statistical) values for all active list mode segments. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return perr_peak: float Comma-separated list of values, one per active segment. Range: 0 deg to 100 deg deg

7.4.7.2.5.9 Maximum

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:MODulation:PERRor:PEAK:MAXimum
CALCulate:EVDO:MEASurement<Instance>:MEValuation:LIST:MODulation:PERRor:PEAK:MAXimum
```

class Maximum

Maximum commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

calculate() → List[float]

```
# SCPI: CALCulate:EVDO:MEASurement<instance>
↪:MEValuation:LIST:MODulation:PERRor:PEAK:MAXimum
value: List[float] = driver.multiEval.listPy.modulation.perror.peak.maximum.
↪calculate()
```

Returns peak and RMS phase error (statistical) values for all active list mode segments. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return perr_peak: float Comma-separated list of values, one per active segment. Range: 0 deg to 100 deg deg

fetch() → List[float]

```
# SCPI: FETCh:EVDO:MEASurement<instance>
↪:MEValuation:LIST:MODulation:PERRor:PEAK:MAXimum
value: List[float] = driver.multiEval.listPy.modulation.perror.peak.maximum.
↪fetch()
```

Returns peak and RMS phase error (statistical) values for all active list mode segments. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return perr_peak: float Comma-separated list of values, one per active segment. Range: 0 deg to 100 deg deg

7.4.7.2.5.10 StandardDev

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:MODulation:PERRor:PEAK:SDEViation
CALCulate:EVDO:MEASurement<Instance>:MEValuation:LIST:MODulation:PERRor:PEAK:SDEViation
```

class StandardDev

StandardDev commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

calculate() → List[float]

```
# SCPI: CALCulate:EVDO:MEASurement<instance>
↪:MEValuation:LIST:MODulation:PERRor:PEAK:SDEViation
value: List[float] = driver.multiEval.listPy.modulation.perror.peak.standardDev.
↪calculate()
```

Returns peak and RMS phase error (statistical) values for all active list mode segments. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return perr_peak: float Comma-separated list of values, one per active segment. Range: 0 deg to 100 deg deg

fetch() → List[float]

```
# SCPI: FETCh:EVDO:MEASurement<instance>
↪:MEValuation:LIST:MODulation:PERRor:PEAK:SDEViation
value: List[float] = driver.multiEval.listPy.modulation.perror.peak.standardDev.
↪fetch()
```

Returns peak and RMS phase error (statistical) values for all active list mode segments. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return perr_peak: float Comma-separated list of values, one per active segment. Range: 0 deg to 100 deg deg

7.4.7.2.6 IqOffset

class IqOffset

IqOffset commands group definition. 8 total commands, 4 Sub-groups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.multiEval.listPy.modulation.iqOffset.clone()
```

Subgroups

7.4.7.2.6.1 Current

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:MODulation:IQOFfset:CURRent
CALCulate:EVDO:MEASurement<Instance>:MEValuation:LIST:MODulation:IQOFfset:CURRent
```

class Current

Current commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

calculate() → List[float]

```
# SCPI: CALCulate:EVDO:MEASurement<instance>
↪:MEValuation:LIST:MODulation:IQOFfset:CURRent
value: List[float] = driver.multiEval.listPy.modulation.iqOffset.current.
↪calculate()
```

Returns IQ origin offset (statistical) values for all active list mode segments. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return iq_offset: float Comma-separated list of values, one per active segment. Range: -100 dB to 0 dB , Unit: dB

fetch() → List[float]

```
# SCPI: FETCh:EVDO:MEASurement<instance>
↪:MEValuation:LIST:MODulation:IQOFfset:CURRent
value: List[float] = driver.multiEval.listPy.modulation.iqOffset.current.fetch()
```

Returns IQ origin offset (statistical) values for all active list mode segments. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return iq_offset: float Comma-separated list of values, one per active segment. Range: -100 dB to 0 dB , Unit: dB

7.4.7.2.6.2 Average

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:MODulation:IQOFfset:AVERage
CALCulate:EVDO:MEASurement<Instance>:MEValuation:LIST:MODulation:IQOFfset:AVERage
```

class Average

Average commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

calculate() → List[float]

```
# SCPI: CALCulate:EVDO:MEASurement<instance>
↪:MEValuation:LIST:MODulation:IQOFfset:AVERage
value: List[float] = driver.multiEval.listPy.modulation.iqOffset.average.
↪calculate()
```

Returns IQ origin offset (statistical) values for all active list mode segments. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return iq_offset: float Comma-separated list of values, one per active segment. Range: -100 dB to 0 dB , Unit: dB

fetch() → List[float]

```
# SCPI: FETCh:EVDO:MEASurement<instance>
↪:MEValuation:LIST:MODulation:IQOFfset:AVERage
value: List[float] = driver.multiEval.listPy.modulation.iqOffset.average.fetch()
```

Returns IQ origin offset (statistical) values for all active list mode segments. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return iq_offset: float Comma-separated list of values, one per active segment. Range: -100 dB to 0 dB , Unit: dB

7.4.7.2.6.3 Maximum

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:MODulation:IQOFfset:MAXimum
CALCulate:EVDO:MEASurement<Instance>:MEValuation:LIST:MODulation:IQOFfset:MAXimum
```

class Maximum

Maximum commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

calculate() → List[float]

```
# SCPI: CALCulate:EVDO:MEASurement<instance>
↪:MEValuation:LIST:MODulation:IQOFfset:MAXimum
value: List[float] = driver.multiEval.listPy.modulation.iqOffset.maximum.
↪calculate()
```

Returns IQ origin offset (statistical) values for all active list mode segments. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return iq_offset: float Comma-separated list of values, one per active segment. Range: -100 dB to 0 dB , Unit: dB

fetch() → List[float]

```
# SCPI: FETCh:EVDO:MEASurement<instance>
↪:MEValuation:LIST:MODulation:IQOFfset:MAXimum
value: List[float] = driver.multiEval.listPy.modulation.iqOffset.maximum.fetch()
```

Returns IQ origin offset (statistical) values for all active list mode segments. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return iq_offset: float Comma-separated list of values, one per active segment. Range: -100 dB to 0 dB , Unit: dB

7.4.7.2.6.4 StandardDev

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:MODulation:IQOFfset:SDEViation
CALCulate:EVDO:MEASurement<Instance>:MEValuation:LIST:MODulation:IQOFfset:SDEViation
```

class StandardDev

StandardDev commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

calculate() → List[float]

```
# SCPI: CALCulate:EVDO:MEASurement<instance>
↪:MEValuation:LIST:MODulation:IQOFfset:SDEViation
value: List[float] = driver.multiEval.listPy.modulation.iqOffset.standardDev.
↪calculate()
```

Returns IQ origin offset (statistical) values for all active list mode segments. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return iq_offset: float Comma-separated list of values, one per active segment. Range: -100 dB to 0 dB , Unit: dB

fetch() → List[float]

```
# SCPI: FETCh:EVDO:MEASurement<instance>
↪:MEValuation:LIST:MODulation:IQOFfset:SDEViation
value: List[float] = driver.multiEval.listPy.modulation.iqOffset.standardDev.
↪fetch()
```

Returns IQ origin offset (statistical) values for all active list mode segments. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return iq_offset: float Comma-separated list of values, one per active segment. Range: -100 dB to 0 dB , Unit: dB

7.4.7.2.7 IqImbalance

class IqImbalance

IqImbalance commands group definition. 8 total commands, 4 Sub-groups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.multiEval.listPy.modulation.iqImbalance.clone()
```

Subgroups

7.4.7.2.7.1 Current

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:MODulation:IQIMbalance:CURRent
CALCulate:EVDO:MEASurement<Instance>:MEValuation:LIST:MODulation:IQIMbalance:CURRent
```

class Current

Current commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

calculate() → List[float]

```
# SCPI: CALCulate:EVDO:MEASurement<instance>
↪:MEValuation:LIST:MODulation:IQIMbalance:CURRent
value: List[float] = driver.multiEval.listPy.modulation.iqImbalance.current.
↪calculate()
```

Returns IQ imbalance (statistical) values for all active list mode segments. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return iq_imbalance: float Comma-separated list of values, one per active segment.
Range: -100 dB to 0 dB , Unit: dB

fetch() → List[float]

```
# SCPI: FETCh:EVDO:MEASurement<instance>
↪:MEValuation:LIST:MODulation:IQIMbalance:CURRent
value: List[float] = driver.multiEval.listPy.modulation.iqImbalance.current.
↪fetch()
```

Returns IQ imbalance (statistical) values for all active list mode segments. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return iq_imbalance: float Comma-separated list of values, one per active segment.
Range: -100 dB to 0 dB , Unit: dB

7.4.7.2.7.2 Average

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:MODulation:IQIMbalance:AVERage
CALCulate:EVDO:MEASurement<Instance>:MEValuation:LIST:MODulation:IQIMbalance:AVERage
```

class Average

Average commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

calculate() → List[float]

```
# SCPI: CALCulate:EVDO:MEASurement<instance>
↪:MEValuation:LIST:MODulation:IQIMbalance:AVERage
value: List[float] = driver.multiEval.listPy.modulation.iqImbalance.average.
↪calculate()
```

Returns IQ imbalance (statistical) values for all active list mode segments. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return iq_imbalance: float Comma-separated list of values, one per active segment.
Range: -100 dB to 0 dB , Unit: dB

fetch() → List[float]

```
# SCPI: FETCh:EVDO:MEASurement<instance>
↪:MEValuation:LIST:MODulation:IQIMbalance:AVERage
value: List[float] = driver.multiEval.listPy.modulation.iqImbalance.average.
↪fetch()
```

Returns IQ imbalance (statistical) values for all active list mode segments. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return iq_imbalance: float Comma-separated list of values, one per active segment. Range: -100 dB to 0 dB , Unit: dB

7.4.7.2.7.3 Maximum

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:MODulation:IQIMbalance:MAXimum
CALCulate:EVDO:MEASurement<Instance>:MEValuation:LIST:MODulation:IQIMbalance:MAXimum
```

class Maximum

Maximum commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

calculate() → List[float]

```
# SCPI: CALCulate:EVDO:MEASurement<instance>
↪:MEValuation:LIST:MODulation:IQIMbalance:MAXimum
value: List[float] = driver.multiEval.listPy.modulation.iqImbalance.maximum.
↪calculate()
```

Returns IQ imbalance (statistical) values for all active list mode segments. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return iq_imbalance: float Comma-separated list of values, one per active segment. Range: -100 dB to 0 dB , Unit: dB

fetch() → List[float]

```
# SCPI: FETCh:EVDO:MEASurement<instance>
↪:MEValuation:LIST:MODulation:IQIMbalance:MAXimum
value: List[float] = driver.multiEval.listPy.modulation.iqImbalance.maximum.
↪fetch()
```

Returns IQ imbalance (statistical) values for all active list mode segments. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return iq_imbalance: float Comma-separated list of values, one per active segment. Range: -100 dB to 0 dB , Unit: dB

7.4.7.2.7.4 StandardDev

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:MODulation:IQIMbalance:SDEViation
CALCulate:EVDO:MEASurement<Instance>:MEValuation:LIST:MODulation:IQIMbalance:SDEViation
```

class StandardDev

StandardDev commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

calculate() → List[float]

```
# SCPI: CALCulate:EVDO:MEASurement<instance>
↪:MEValuation:LIST:MODulation:IQIMbalance:SDEViation
value: List[float] = driver.multiEval.listPy.modulation.iqImbalance.standardDev.
↪calculate()
```

Returns IQ imbalance (statistical) values for all active list mode segments. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return iq_imbalance: float Comma-separated list of values, one per active segment. Range: -100 dB to 0 dB , Unit: dB

fetch() → List[float]

```
# SCPI: FETCh:EVDO:MEASurement<instance>
↪:MEValuation:LIST:MODulation:IQIMbalance:SDEViation
value: List[float] = driver.multiEval.listPy.modulation.iqImbalance.standardDev.
↪fetch()
```

Returns IQ imbalance (statistical) values for all active list mode segments. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return iq_imbalance: float Comma-separated list of values, one per active segment. Range: -100 dB to 0 dB , Unit: dB

7.4.7.2.8 FreqError

class FreqError

FreqError commands group definition. 8 total commands, 4 Sub-groups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.multiEval.listPy.modulation.freqError.clone()
```

Subgroups

7.4.7.2.8.1 Current

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:MODulation:FERRor:CURRent
CALCulate:EVDO:MEASurement<Instance>:MEValuation:LIST:MODulation:FERRor:CURRent
```

class Current

Current commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

calculate() → List[float]

```
# SCPI: CALCulate:EVDO:MEASurement<instance>
↪:MEValuation:LIST:MODulation:FERRor:CURRent
value: List[float] = driver.multiEval.listPy.modulation.freqError.current.
↪calculate()
```

Returns carrier frequency error values for all active list mode segments. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return cfreq_error: float Comma-separated list of values, one per active segment. Range: -5000 Hz to 5000 Hz , Unit: Hz

fetch() → List[float]

```
# SCPI: FETCh:EVDO:MEASurement<instance>
↪:MEValuation:LIST:MODulation:FERRor:CURRent
value: List[float] = driver.multiEval.listPy.modulation.freqError.current.
↪fetch()
```

Returns carrier frequency error values for all active list mode segments. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return cfreq_error: float Comma-separated list of values, one per active segment. Range: -5000 Hz to 5000 Hz , Unit: Hz

7.4.7.2.8.2 Average

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:MODulation:FERRor:AVERage
CALCulate:EVDO:MEASurement<Instance>:MEValuation:LIST:MODulation:FERRor:AVERage
```

class Average

Average commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

calculate() → List[float]

```
# SCPI: CALCulate:EVDO:MEASurement<instance>
↪:MEValuation:LIST:MODulation:FERRor:AVERage
value: List[float] = driver.multiEval.listPy.modulation.freqError.average.
↪calculate()
```

Returns carrier frequency error values for all active list mode segments. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return cfreq_error: float Comma-separated list of values, one per active segment. Range: -5000 Hz to 5000 Hz , Unit: Hz

fetch() → List[float]

```
# SCPI: FETCh:EVDO:MEASurement<instance>
↪:MEValuation:LIST:MODulation:FERRor:AVERage
value: List[float] = driver.multiEval.listPy.modulation.freqError.average.
↪fetch()
```

Returns carrier frequency error values for all active list mode segments. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return cfreq_error: float Comma-separated list of values, one per active segment. Range: -5000 Hz to 5000 Hz , Unit: Hz

7.4.7.2.8.3 Maximum

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:MODulation:FERRor:MAXimum
CALCulate:EVDO:MEASurement<Instance>:MEValuation:LIST:MODulation:FERRor:MAXimum
```

class Maximum

Maximum commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

calculate() → List[float]

```
# SCPI: CALCulate:EVDO:MEASurement<instance>
↪:MEValuation:LIST:MODulation:FERRor:MAXimum
value: List[float] = driver.multiEval.listPy.modulation.freqError.maximum.
↪calculate()
```

Returns carrier frequency error values for all active list mode segments. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return cfreq_error: float Comma-separated list of values, one per active segment. Range: -5000 Hz to 5000 Hz , Unit: Hz

fetch() → List[float]

```
# SCPI: FETCh:EVDO:MEASurement<instance>
↪:MEValuation:LIST:MODulation:FERRor:MAXimum
value: List[float] = driver.multiEval.listPy.modulation.freqError.maximum.
↪fetch()
```

Returns carrier frequency error values for all active list mode segments. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return cfreq_error: float Comma-separated list of values, one per active segment. Range: -5000 Hz to 5000 Hz , Unit: Hz

7.4.7.2.8.4 StandardDev

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:MODulation:FERRor:SDEViation
CALCulate:EVDO:MEASurement<Instance>:MEValuation:LIST:MODulation:FERRor:SDEViation
```

class StandardDev

StandardDev commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

calculate() → List[float]

```
# SCPI: CALCulate:EVDO:MEASurement<instance>
↪:MEValuation:LIST:MODulation:FERRor:SDEViation
value: List[float] = driver.multiEval.listPy.modulation.freqError.standardDev.
↪calculate()
```

Returns carrier frequency error values for all active list mode segments. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return cfreq_error: float Comma-separated list of values, one per active segment. Range: -5000 Hz to 5000 Hz , Unit: Hz

fetch() → List[float]

```
# SCPI: FETCh:EVDO:MEASurement<instance>
↪:MEValuation:LIST:MODulation:FERRor:SDEViation
value: List[float] = driver.multiEval.listPy.modulation.freqError.standardDev.
↪fetch()
```

Returns carrier frequency error values for all active list mode segments. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return cfreq_error: float Comma-separated list of values, one per active segment. Range: -5000 Hz to 5000 Hz , Unit: Hz

7.4.7.2.9 Terror

class Terror

Terror commands group definition. 8 total commands, 4 Sub-groups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.multiEval.listPy.modulation.terror.clone()
```

Subgroups

7.4.7.2.9.1 Current

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:MODulation:TERRor:CURRent
CALCulate:EVDO:MEASurement<Instance>:MEValuation:LIST:MODulation:TERRor:CURRent
```

class Current

Current commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

calculate() → List[float]

```
# SCPI: CALCulate:EVDO:MEASurement<instance>
↪:MEValuation:LIST:MODulation:TERRor:CURRent
value: List[float] = driver.multiEval.listPy.modulation.terror.current.
↪calculate()
```

Returns transmit time error (statistical) values for all active list mode segments. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return trans_time_error: float Comma-separated list of values, one per active segment.
Range: -100 µs to 100 µs, Unit: µs

fetch() → List[float]

```
# SCPI: FETCh:EVDO:MEASurement<instance>
↪:MEValuation:LIST:MODulation:TERRor:CURRent
value: List[float] = driver.multiEval.listPy.modulation.terror.current.fetch()
```

Returns transmit time error (statistical) values for all active list mode segments. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return trans_time_error: float Comma-separated list of values, one per active segment.
Range: -100 µs to 100 µs, Unit: µs

7.4.7.2.9.2 Average

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:MODulation:TERRor:AVERage
CALCulate:EVDO:MEASurement<Instance>:MEValuation:LIST:MODulation:TERRor:AVERage
```

class Average

Average commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

calculate() → List[float]

```
# SCPI: CALCulate:EVDO:MEASurement<instance>
↪:MEValuation:LIST:MODulation:TERRor:AVERage
value: List[float] = driver.multiEval.listPy.modulation.terror.average.
↪calculate()
```

Returns transmit time error (statistical) values for all active list mode segments. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return trans_time_error: float Comma-separated list of values, one per active segment.
Range: -100 µs to 100 µs, Unit: µs

fetch() → List[float]

```
# SCPI: FETCh:EVDO:MEASurement<instance>
↪:MEValuation:LIST:MODulation:TERRor:AVERage
value: List[float] = driver.multiEval.listPy.modulation.terror.average.fetch()
```

Returns transmit time error (statistical) values for all active list mode segments. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return trans_time_error: float Comma-separated list of values, one per active segment.
Range: -100 µs to 100 µs, Unit: µs

7.4.7.2.9.3 Maximum

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:MODulation:TERRor:MAXimum
CALCulate:EVDO:MEASurement<Instance>:MEValuation:LIST:MODulation:TERRor:MAXimum
```

class Maximum

Maximum commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

calculate() → List[float]

```
# SCPI: CALCulate:EVDO:MEASurement<instance>
↪:MEValuation:LIST:MODulation:TERRor:MAXimum
value: List[float] = driver.multiEval.listPy.modulation.terror.maximum.
↪calculate()
```

Returns transmit time error (statistical) values for all active list mode segments. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return trans_time_error: float Comma-separated list of values, one per active segment.
Range: -100 µs to 100 µs, Unit: µs

fetch() → List[float]

```
# SCPI: FETCh:EVDO:MEASurement<instance>
↪:MEValuation:LIST:MODulation:TERRor:MAXimum
value: List[float] = driver.multiEval.listPy.modulation.terror.maximum.fetch()
```

Returns transmit time error (statistical) values for all active list mode segments. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return trans_time_error: float Comma-separated list of values, one per active segment.
Range: -100 µs to 100 µs, Unit: µs

7.4.7.2.9.4 StandardDev

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:MODulation:TERRor:SDEViation
CALCulate:EVDO:MEASurement<Instance>:MEValuation:LIST:MODulation:TERRor:SDEViation
```

class StandardDev

StandardDev commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

calculate() → List[float]

```
# SCPI: CALCulate:EVDO:MEASurement<instance>
↪:MEValuation:LIST:MODulation:TERRor:SDEViation
value: List[float] = driver.multiEval.listPy.modulation.terror.standardDev.
↪calculate()
```

Returns transmit time error (statistical) values for all active list mode segments. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return trans_time_error: float Comma-separated list of values, one per active segment.
Range: -100 µs to 100 µs, Unit: µs

fetch() → List[float]

```
# SCPI: FETCh:EVDO:MEASurement<instance>
↪:MEValuation:LIST:MODulation:TERRor:SDEViation
value: List[float] = driver.multiEval.listPy.modulation.terror.standardDev.
↪fetch()
```

Returns transmit time error (statistical) values for all active list mode segments. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return trans_time_error: float Comma-separated list of values, one per active segment.
Range: -100 µs to 100 µs, Unit: µs

7.4.7.2.10 Wquality

class Wquality

Wquality commands group definition. 12 total commands, 6 Sub-groups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.multiEval.listPy.modulation.wquality.clone()
```

Subgroups

7.4.7.2.10.1 Current

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:MODulation:WQUality:CURRent
CALCulate:EVDO:MEASurement<Instance>:MEValuation:LIST:MODulation:WQUality:CURRent
```

class Current

Current commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

calculate() → List[float]

```
# SCPI: CALCulate:EVDO:MEASurement<instance>
↪:MEValuation:LIST:MODulation:WQUality:CURRent
value: List[float] = driver.multiEval.listPy.modulation.wquality.current.
↪calculate()
```

Returns waveform quality (statistical) values for all active list mode segments - overall, at minimum and at maximum AT power. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return wav_quality: float Range: 0 to 1

fetch() → List[float]

```
# SCPI: FETCh:EVDO:MEASurement<instance>
↪:MEValuation:LIST:MODulation:WQUality:CURRent
value: List[float] = driver.multiEval.listPy.modulation.wquality.current.fetch()
```

Returns waveform quality (statistical) values for all active list mode segments - overall, at minimum and at maximum AT power. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return wav_quality: float Range: 0 to 1

7.4.7.2.10.2 Pmax

class Pmax

Pmax commands group definition. 2 total commands, 1 Sub-groups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.multiEval.listPy.modulation.wquality.pmax.clone()
```

Subgroups

7.4.7.2.10.3 Current

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:MODulation:WQUality:PMAX:CURRent
CALCulate:EVDO:MEASurement<Instance>:MEValuation:LIST:MODulation:WQUality:PMAX:CURRent
```

class Current

Current commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

calculate() → List[float]

```
# SCPI: CALCulate:EVDO:MEASurement<instance>
↪:MEValuation:LIST:MODulation:WQUality:PMAX:CURRent
value: List[float] = driver.multiEval.listPy.modulation.wquality.pmax.current.
↪calculate()
```

Returns waveform quality (statistical) values for all active list mode segments - overall, at minimum and at maximum AT power. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return wav_qual_max_pow: float Range: 0 to 1

fetch() → List[float]

```
# SCPI: FETCh:EVDO:MEASurement<instance>
↪:MEValuation:LIST:MODulation:WQUality:PMAX:CURRent
value: List[float] = driver.multiEval.listPy.modulation.wquality.pmax.current.
↪fetch()
```

Returns waveform quality (statistical) values for all active list mode segments - overall, at minimum and at maximum AT power. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return wav_qual_max_pow: float Range: 0 to 1

7.4.7.2.10.4 Pmin

class Pmin

Pmin commands group definition. 2 total commands, 1 Sub-groups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.multiEval.listPy.modulation.wquality.pmin.clone()
```

Subgroups

7.4.7.2.10.5 Current

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:MODulation:WQUality:PMIN:CURRent
CALCulate:EVDO:MEASurement<Instance>:MEValuation:LIST:MODulation:WQUality:PMIN:CURRent
```

class Current

Current commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

calculate() → List[float]

```
# SCPI: CALCulate:EVDO:MEASurement<instance>
↪:MEValuation:LIST:MODulation:WQUality:PMIN:CURRent
value: List[float] = driver.multiEval.listPy.modulation.wquality.pmin.current.
↪calculate()
```

Returns waveform quality (statistical) values for all active list mode segments - overall, at minimum and at maximum AT power. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return wav_qual_min_power: float Range: 0 to 1

fetch() → List[float]

```
# SCPI: FETCh:EVDO:MEASurement<instance>
↪:MEValuation:LIST:MODulation:WQUality:PMIN:CURRent
value: List[float] = driver.multiEval.listPy.modulation.wquality.pmin.current.
↪fetch()
```

Returns waveform quality (statistical) values for all active list mode segments - overall, at minimum and at maximum AT power. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return wav_qual_min_power: float Range: 0 to 1

7.4.7.2.10.6 Average

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:MODulation:WQUality:AVERage
CALCulate:EVDO:MEASurement<Instance>:MEValuation:LIST:MODulation:WQUality:AVERage
```

class Average

Average commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

calculate() → List[float]

```
# SCPI: CALCulate:EVDO:MEASurement<instance>
↪:MEValuation:LIST:MODulation:WQUality:AVERage
value: List[float] = driver.multiEval.listPy.modulation.wquality.average.
↪calculate()
```

Returns waveform quality (statistical) values for all active list mode segments - overall, at minimum and at maximum AT power. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return wav_quality: float Range: 0 to 1

fetch() → List[float]

```
# SCPI: FETCh:EVDO:MEASurement<instance>
↪:MEValuation:LIST:MODulation:WQUality:AVERage
value: List[float] = driver.multiEval.listPy.modulation.wquality.average.fetch()
```

Returns waveform quality (statistical) values for all active list mode segments - overall, at minimum and at maximum AT power. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return wav_quality: float Range: 0 to 1

7.4.7.2.10.7 Maximum

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:MODulation:WQUality:MAXimum
CALCulate:EVDO:MEASurement<Instance>:MEValuation:LIST:MODulation:WQUality:MAXimum
```

class Maximum

Maximum commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

calculate() → List[float]

```
# SCPI: CALCulate:EVDO:MEASurement<instance>
↪:MEValuation:LIST:MODulation:WQUality:MAXimum
value: List[float] = driver.multiEval.listPy.modulation.wquality.maximum.
↪calculate()
```

Returns waveform quality (statistical) values for all active list mode segments - overall, at minimum and at maximum AT power. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return wav_quality: float Range: 0 to 1

fetch() → List[float]

```
# SCPI: FETCh:EVDO:MEASurement<instance>
↪:MEValuation:LIST:MODulation:WQUality:MAXimum
value: List[float] = driver.multiEval.listPy.modulation.wquality.maximum.fetch()
```

Returns waveform quality (statistical) values for all active list mode segments - overall, at minimum and at maximum AT power. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return wav_quality: float Range: 0 to 1

7.4.7.2.10.8 StandardDev

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:MODulation:WQUality:SDEViation
CALCulate:EVDO:MEASurement<Instance>:MEValuation:LIST:MODulation:WQUality:SDEViation
```

class StandardDev

StandardDev commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

calculate() → List[float]

```
# SCPI: CALCulate:EVDO:MEASurement<instance>
↪:MEValuation:LIST:MODulation:WQUality:SDEViation
value: List[float] = driver.multiEval.listPy.modulation.wquality.standardDev.
↪calculate()
```

Returns waveform quality (statistical) values for all active list mode segments - overall, at minimum and at maximum AT power. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return wav_quality: float Range: 0 to 1

fetch() → List[float]

```
# SCPI: FETCh:EVDO:MEASurement<instance>
↪:MEValuation:LIST:MODulation:WQUality:SDEViation
value: List[float] = driver.multiEval.listPy.modulation.wquality.standardDev.
↪fetch()
```

Returns waveform quality (statistical) values for all active list mode segments - overall, at minimum and at maximum AT power. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return wav_quality: float Range: 0 to 1

7.4.7.2.11 PwBand

class PwBand

PwBand commands group definition. 10 total commands, 5 Sub-groups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.multiEval.listPy.modulation.pwBand.clone()
```

Subgroups

7.4.7.2.11.1 Current

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:MODulation:PWBand:CURRent
CALCulate:EVDO:MEASurement<Instance>:MEValuation:LIST:MODulation:PWBand:CURRent
```

class Current

Current commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

calculate() → List[float]

```
# SCPI: CALCulate:EVDO:MEASurement<instance>
↪:MEValuation:LIST:MODulation:PWBand:CURRent
value: List[float] = driver.multiEval.listPy.modulation.pwBand.current.
↪calculate()
```

Returns AT narrowband and wideband power (statistical) values for all active list mode segments. The narrowband filter is 1.23 MHz, the wideband filter is 8 MHz wide. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return ms_power_wideband: float Comma-separated list of values, one per active segment. Range: -100 dBm to 50 dBm , Unit: dBm

fetch() → List[float]

```
# SCPI: FETCh:EVDO:MEASurement<instance>
↪:MEValuation:LIST:MODulation:PWBand:CURRent
value: List[float] = driver.multiEval.listPy.modulation.pwBand.current.fetch()
```

Returns AT narrowband and wideband power (statistical) values for all active list mode segments. The narrowband filter is 1.23 MHz, the wideband filter is 8 MHz wide. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return ms_power_wideband: float Comma-separated list of values, one per active segment. Range: -100 dBm to 50 dBm , Unit: dBm

7.4.7.2.11.2 Average

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:MODulation:PWBand:AVERage
CALCulate:EVDO:MEASurement<Instance>:MEValuation:LIST:MODulation:PWBand:AVERage
```

class Average

Average commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

calculate() → List[float]

```
# SCPI: CALCulate:EVDO:MEASurement<instance>
↪:MEValuation:LIST:MODulation:PWBand:AVERage
value: List[float] = driver.multiEval.listPy.modulation.pwBand.average.
↪calculate()
```

Returns AT narrowband and wideband power (statistical) values for all active list mode segments. The narrowband filter is 1.23 MHz, the wideband filter is 8 MHz wide. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return ms_power_wideband: float Comma-separated list of values, one per active segment. Range: -100 dBm to 50 dBm , Unit: dBm

fetch() → List[float]

```
# SCPI: FETCh:EVDO:MEASurement<instance>
↪:MEValuation:LIST:MODulation:PWBand:AVERage
value: List[float] = driver.multiEval.listPy.modulation.pwBand.average.fetch()
```

Returns AT narrowband and wideband power (statistical) values for all active list mode segments. The narrowband filter is 1.23 MHz, the wideband filter is 8 MHz wide. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return ms_power_wideband: float Comma-separated list of values, one per active segment. Range: -100 dBm to 50 dBm , Unit: dBm

7.4.7.2.11.3 Maximum

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:MODulation:PWBand:MAXimum
CALCulate:EVDO:MEASurement<Instance>:MEValuation:LIST:MODulation:PWBand:MAXimum
```

class Maximum

Maximum commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

calculate() → List[float]

```
# SCPI: CALCulate:EVDO:MEASurement<instance>
↪:MEValuation:LIST:MODulation:PWBand:MAXimum
value: List[float] = driver.multiEval.listPy.modulation.pwBand.maximum.
↪calculate()
```

Returns AT narrowband and wideband power (statistical) values for all active list mode segments. The narrowband filter is 1.23 MHz, the wideband filter is 8 MHz wide. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return ms_power_wideband: float Comma-separated list of values, one per active segment. Range: -100 dBm to 50 dBm , Unit: dBm

fetch() → List[float]

```
# SCPI: FETCh:EVDO:MEASurement<instance>
↪:MEValuation:LIST:MODulation:PWBand:MAXimum
value: List[float] = driver.multiEval.listPy.modulation.pwBand.maximum.fetch()
```

Returns AT narrowband and wideband power (statistical) values for all active list mode segments. The narrowband filter is 1.23 MHz, the wideband filter is 8 MHz wide. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return ms_power_wideband: float Comma-separated list of values, one per active segment. Range: -100 dBm to 50 dBm , Unit: dBm

7.4.7.2.11.4 Minimum

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:MODulation:PWBand:MINimum
CALCulate:EVDO:MEASurement<Instance>:MEValuation:LIST:MODulation:PWBand:MINimum
```

class Minimum

Minimum commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

calculate() → List[float]

```
# SCPI: CALCulate:EVDO:MEASurement<instance>
↪:MEValuation:LIST:MODulation:PWBand:MINimum
value: List[float] = driver.multiEval.listPy.modulation.pwBand.minimum.
↪calculate()
```

Returns AT narrowband and wideband power (statistical) values for all active list mode segments. The narrowband filter is 1.23 MHz, the wideband filter is 8 MHz wide. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return ms_power_wideband: float Comma-separated list of values, one per active segment. Range: -100 dBm to 50 dBm , Unit: dBm

fetch() → List[float]

```
# SCPI: FETCh:EVDO:MEASurement<instance>
↪:MEValuation:LIST:MODulation:PWBand:MINimum
value: List[float] = driver.multiEval.listPy.modulation.pwBand.minimum.fetch()
```

Returns AT narrowband and wideband power (statistical) values for all active list mode segments. The narrowband filter is 1.23 MHz, the wideband filter is 8 MHz wide. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return ms_power_wideband: float Comma-separated list of values, one per active segment. Range: -100 dBm to 50 dBm , Unit: dBm

7.4.7.2.11.5 StandardDev

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:MODulation:PWBand:SDEViation
CALCulate:EVDO:MEASurement<Instance>:MEValuation:LIST:MODulation:PWBand:SDEViation
```

class StandardDev

StandardDev commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

calculate() → List[float]

```
# SCPI: CALCulate:EVDO:MEASurement<instance>
↪:MEValuation:LIST:MODulation:PWBand:SDEViation
value: List[float] = driver.multiEval.listPy.modulation.pwBand.standardDev.
↪calculate()
```

Returns AT narrowband and wideband power (statistical) values for all active list mode segments. The narrowband filter is 1.23 MHz, the wideband filter is 8 MHz wide. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return ms_power_wideband: float Comma-separated list of values, one per active segment. Range: -100 dBm to 50 dBm , Unit: dBm

fetch() → List[float]

```
# SCPI: FETCh:EVDO:MEASurement<instance>
↪:MEValuation:LIST:MODulation:PWBand:SDEViation
value: List[float] = driver.multiEval.listPy.modulation.pwBand.standardDev.
↪fetch()
```

Returns AT narrowband and wideband power (statistical) values for all active list mode segments. The narrowband filter is 1.23 MHz, the wideband filter is 8 MHz wide. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return ms_power_wideband: float Comma-separated list of values, one per active segment. Range: -100 dBm to 50 dBm , Unit: dBm

7.4.7.2.12 PnBand

class PnBand

PnBand commands group definition. 10 total commands, 5 Sub-groups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.multiEval.listPy.modulation.pnBand.clone()
```

Subgroups

7.4.7.2.12.1 Current

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:MODulation:PNBand:CURRent
CALCulate:EVDO:MEASurement<Instance>:MEValuation:LIST:MODulation:PNBand:CURRent
```

class Current

Current commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

calculate() → List[float]

```
# SCPI: CALCulate:EVDO:MEASurement<instance>
↪:MEValuation:LIST:MODulation:PNBand:CURRent
value: List[float] = driver.multiEval.listPy.modulation.pnBand.current.
↪calculate()
```

Returns AT narrowband and wideband power (statistical) values for all active list mode segments. The narrowband filter is 1.23 MHz, the wideband filter is 8 MHz wide. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return ms_power_1_m_23: float Comma-separated list of values, one per active segment. Range: -100 dBm to 50 dBm , Unit: dBm

fetch() → List[float]

```
# SCPI: FETCh:EVDO:MEASurement<instance>
↪:MEValuation:LIST:MODulation:PNBand:CURRent
value: List[float] = driver.multiEval.listPy.modulation.pnBand.current.fetch()
```

Returns AT narrowband and wideband power (statistical) values for all active list mode segments. The narrowband filter is 1.23 MHz, the wideband filter is 8 MHz wide. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return ms_power_1_m_23: float Comma-separated list of values, one per active segment. Range: -100 dBm to 50 dBm , Unit: dBm

7.4.7.2.12.2 Average

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:MODulation:PNBand:AVERage
CALCulate:EVDO:MEASurement<Instance>:MEValuation:LIST:MODulation:PNBand:AVERage
```

class Average

Average commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

calculate() → List[float]

```
# SCPI: CALCulate:EVDO:MEASurement<instance>
↪:MEValuation:LIST:MODulation:PNBand:AVERage
value: List[float] = driver.multiEval.listPy.modulation.pnBand.average.
↪calculate()
```

Returns AT narrowband and wideband power (statistical) values for all active list mode segments. The narrowband filter is 1.23 MHz, the wideband filter is 8 MHz wide. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return ms_power_1_m_23: float Comma-separated list of values, one per active segment. Range: -100 dBm to 50 dBm , Unit: dBm

fetch() → List[float]

```
# SCPI: FETCh:EVDO:MEASurement<instance>
↪:MEValuation:LIST:MODulation:PNBand:AVERage
value: List[float] = driver.multiEval.listPy.modulation.pnBand.average.fetch()
```

Returns AT narrowband and wideband power (statistical) values for all active list mode segments. The narrowband filter is 1.23 MHz, the wideband filter is 8 MHz wide. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return ms_power_1_m_23: float Comma-separated list of values, one per active segment. Range: -100 dBm to 50 dBm , Unit: dBm

7.4.7.2.12.3 Maximum

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:MODulation:PNBand:MAXimum
CALCulate:EVDO:MEASurement<Instance>:MEValuation:LIST:MODulation:PNBand:MAXimum
```

class Maximum

Maximum commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

calculate() → List[float]

```
# SCPI: CALCulate:EVDO:MEASurement<instance>
↪:MEValuation:LIST:MODulation:PNBand:MAXimum
value: List[float] = driver.multiEval.listPy.modulation.pnBand.maximum.
↪calculate()
```

Returns AT narrowband and wideband power (statistical) values for all active list mode segments. The narrowband filter is 1.23 MHz, the wideband filter is 8 MHz wide. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return ms_power_1_m_23: float Comma-separated list of values, one per active segment. Range: -100 dBm to 50 dBm , Unit: dBm

fetch() → List[float]

```
# SCPI: FETCh:EVDO:MEASurement<instance>
↪:MEValuation:LIST:MODulation:PNBand:MAXimum
value: List[float] = driver.multiEval.listPy.modulation.pnBand.maximum.fetch()
```

Returns AT narrowband and wideband power (statistical) values for all active list mode segments. The narrowband filter is 1.23 MHz, the wideband filter is 8 MHz wide. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return ms_power_1_m_23: float Comma-separated list of values, one per active segment. Range: -100 dBm to 50 dBm , Unit: dBm

7.4.7.2.12.4 Minimum

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:MODulation:PNBand:MINimum
CALCulate:EVDO:MEASurement<Instance>:MEValuation:LIST:MODulation:PNBand:MINimum
```

class Minimum

Minimum commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

calculate() → List[float]

```
# SCPI: CALCulate:EVDO:MEASurement<instance>
↪:MEValuation:LIST:MODulation:PNBand:MINimum
value: List[float] = driver.multiEval.listPy.modulation.pnBand.minimum.
↪calculate()
```

Returns AT narrowband and wideband power (statistical) values for all active list mode segments. The narrowband filter is 1.23 MHz, the wideband filter is 8 MHz wide. The values described below are returned

by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return ms_power_1_m_23: float Comma-separated list of values, one per active segment. Range: -100 dBm to 50 dBm , Unit: dBm

fetch() → List[float]

```
# SCPI: FETCh:EVDO:MEASurement<instance>
↪:MEValuation:LIST:MODulation:PNBand:MINimum
value: List[float] = driver.multiEval.listPy.modulation.pnBand.minimum.fetch()
```

Returns AT narrowband and wideband power (statistical) values for all active list mode segments. The narrowband filter is 1.23 MHz, the wideband filter is 8 MHz wide. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return ms_power_1_m_23: float Comma-separated list of values, one per active segment. Range: -100 dBm to 50 dBm , Unit: dBm

7.4.7.2.12.5 StandardDev

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:MODulation:PNBand:SDEViation
CALCulate:EVDO:MEASurement<Instance>:MEValuation:LIST:MODulation:PNBand:SDEViation
```

class StandardDev

StandardDev commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

calculate() → List[float]

```
# SCPI: CALCulate:EVDO:MEASurement<instance>
↪:MEValuation:LIST:MODulation:PNBand:SDEViation
value: List[float] = driver.multiEval.listPy.modulation.pnBand.standardDev.
↪calculate()
```

Returns AT narrowband and wideband power (statistical) values for all active list mode segments. The narrowband filter is 1.23 MHz, the wideband filter is 8 MHz wide. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return ms_power_1_m_23: float Comma-separated list of values, one per active segment. Range: -100 dBm to 50 dBm , Unit: dBm

fetch() → List[float]

```
# SCPI: FETCh:EVDO:MEASurement<instance>
↪:MEValuation:LIST:MODulation:PNBand:SDEViation
value: List[float] = driver.multiEval.listPy.modulation.pnBand.standardDev.
↪fetch()
```

Returns AT narrowband and wideband power (statistical) values for all active list mode segments. The narrowband filter is 1.23 MHz, the wideband filter is 8 MHz wide. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return ms_power_1_m_23: float Comma-separated list of values, one per active segment. Range: -100 dBm to 50 dBm , Unit: dBm

7.4.7.2.13 Current

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:MODulation:CURRent
CALCulate:EVDO:MEASurement<Instance>:MEValuation:LIST:MODulation:CURRent
```

class Current

Current commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: decimal 'Conventions and General Information'. In list mode, a zero reliability indicator indicates that the results in all measured segments are valid. A non-zero value indicates that an error occurred in at least one of the measured segments.
- Seg_Reliability: List[int]: 0 | 3 | 4 | 8 The segment reliability indicates whether one of the following exceptions occurred in this segment: 0: No error 3: Signal overflow 4: Signal low 8: Synchronization error If a combination of exceptions occurs, the most severe error is indicated.
- Evm_Rms: List[enums.ResultStatus2]: float Error vector magnitude RMS and peak value. Range: 0 % to 100 %, Unit: %
- Evm_Peak: List[enums.ResultStatus2]: float Error vector magnitude RMS and peak value. Range: 0 % to 100 %, Unit: %
- Merr_Rms: List[enums.ResultStatus2]: float Magnitude error RMS value. Range: 0 % to 100 %, Unit: %
- Merr_Peak: List[enums.ResultStatus2]: float Magnitude error peak value. Range: -100 % to +100 % (AVERage: 0% to 100 %, SDEViation: 0 % to 50 %) , Unit: %
- Perr_Rms: List[enums.ResultStatus2]: float Phase error RMS value. Range: 0 deg to 180 deg , Unit: deg
- Perr_Peak: List[enums.ResultStatus2]: float Phase error peak value. Range: -180 deg to 180 deg (AVERage: 0 deg to 180 deg, SDEViation: 0 deg to 90 deg) , Unit: deg
- Iq_Offset: List[enums.ResultStatus2]: float I/Q origin offset Range: -100 dB to 0 dB , Unit: dB
- Iq_Imbalance: List[enums.ResultStatus2]: float I/Q imbalance Range: -100 dB to 0 dB , Unit: dB

- Cfreq_Error: List[enums.ResultStatus2]: float Carrier frequency error Range: -5000 Hz to 5000 Hz , Unit: Hz
- Trans_Time_Err: List[enums.ResultStatus2]: float Transmit time error. Range: -100 µs to 100 µs, Unit: µs
- Ms_Power_1_M_23: List[enums.ResultStatus2]: No parameter help available
- Ms_Power_Wideband: List[enums.ResultStatus2]: No parameter help available
- Wav_Quality: List[enums.ResultStatus2]: float Waveform quality Range: 0 to 1
- Wav_Qual_Max_Pow: List[enums.ResultStatus2]: No parameter help available
- Wav_Qual_Min_Power: List[enums.ResultStatus2]: No parameter help available
- Out_Of_Tol_Count: List[enums.ResultStatus2]: float Out of tolerance result, i.e. percentage of measurement intervals of the statistic count (see [CMDLINK: CONFigure:EVDO:MEASi:MEValuation:SCOunt:MODulation CMDLINK]) exceeding the specified limits, see 'Limits (Modulation) '. Range: 0 % to 100 % , Unit: %
- Cur_Stat_Count: List[enums.ResultStatus2]: decimal Number of evaluated valid slots in this segment. Range: 0 to 1000

class FetchStruct

Response structure. Fields:

- Reliability: int: decimal 'Conventions and General Information'. In list mode, a zero reliability indicator indicates that the results in all measured segments are valid. A non-zero value indicates that an error occurred in at least one of the measured segments.
- Seg_Reliability: List[int]: 0 | 3 | 4 | 8 The segment reliability indicates whether one of the following exceptions occurred in this segment: 0: No error 3: Signal overflow 4: Signal low 8: Synchronization error If a combination of exceptions occurs, the most severe error is indicated.
- Evm_Rms: List[float]: float Error vector magnitude RMS and peak value. Range: 0 % to 100 %, Unit: %
- Evm_Peak: List[float]: float Error vector magnitude RMS and peak value. Range: 0 % to 100 %, Unit: %
- Merr_Rms: List[float]: float Magnitude error RMS value. Range: 0 % to 100 %, Unit: %
- Merr_Peak: List[float]: float Magnitude error peak value. Range: -100 % to +100 % (AVERage: 0% to 100 %, SDEViation: 0 % to 50 %) , Unit: %
- Perr_Rms: List[float]: float Phase error RMS value. Range: 0 deg to 180 deg , Unit: deg
- Perr_Peak: List[float]: float Phase error peak value. Range: -180 deg to 180 deg (AVERage: 0 deg to 180 deg, SDEViation: 0 deg to 90 deg) , Unit: deg
- Iq_Offset: List[float]: float I/Q origin offset Range: -100 dB to 0 dB , Unit: dB
- Iq_Imbalance: List[float]: float I/Q imbalance Range: -100 dB to 0 dB , Unit: dB
- Cfreq_Error: List[float]: float Carrier frequency error Range: -5000 Hz to 5000 Hz , Unit: Hz
- Trans_Time_Err: List[float]: float Transmit time error. Range: -100 µs to 100 µs, Unit: µs
- Ms_Power_1_M_23: List[float]: No parameter help available
- Ms_Power_Wideband: List[float]: No parameter help available
- Wav_Quality: List[float]: float Waveform quality Range: 0 to 1
- Wav_Qual_Max_Pow: List[float]: No parameter help available

- Wav_Qual_Min_Power: List[float]: No parameter help available
- Out_Of_Tol_Count: List[float]: float Out of tolerance result, i.e. percentage of measurement intervals of the statistic count (see [CMDLINK: CONFigure:EVDO:MEASi:MEValuation:SCOunt:MODulation CMDLINK]) exceeding the specified limits, see 'Limits (Modulation) '. Range: 0 % to 100 % , Unit: %
- Cur_Stat_Count: List[int]: decimal Number of evaluated valid slots in this segment. Range: 0 to 1000

calculate() → CalculateStruct

```
# SCPI: CALCulate:EVDO:MEASurement<instance>:MEValuation:LIST:MODulation:CURRent
value: CalculateStruct = driver.multiEval.listPy.modulation.current.calculate()
```

Returns modulation single value results in list mode. To define the statistical length for AVERage, MAXimum, MINimum calculation and enable the calculation of the results use the command method RsCmwEvdoMeas.Configure.MultiEval.ListPy. Segment.Modulation.set. The ranges indicated below apply to all results except standard deviation results. The minimum for standard deviation results equals 0. The maximum equals the width of the indicated range divided by two. Exceptions are explicitly stated. The values listed below in curly brackets {} are returned for each active segment: {...}seg 1, {... }seg 2, ..., {...}seg n. The number of active segments n is determined by method RsCmwEvdoMeas.Configure.MultiEval.ListPy. count. 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:EVDO:MEASurement<instance>:MEValuation:LIST:MODulation:CURRent
value: FetchStruct = driver.multiEval.listPy.modulation.current.fetch()
```

Returns modulation single value results in list mode. To define the statistical length for AVERage, MAXimum, MINimum calculation and enable the calculation of the results use the command method RsCmwEvdoMeas.Configure.MultiEval.ListPy. Segment.Modulation.set. The ranges indicated below apply to all results except standard deviation results. The minimum for standard deviation results equals 0. The maximum equals the width of the indicated range divided by two. Exceptions are explicitly stated. The values listed below in curly brackets {} are returned for each active segment: {...}seg 1, {... }seg 2, ..., {...}seg n. The number of active segments n is determined by method RsCmwEvdoMeas.Configure.MultiEval.ListPy. count. 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.

7.4.7.2.14 Average

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:MODulation:AVERage
CALCulate:EVDO:MEASurement<Instance>:MEValuation:LIST:MODulation:AVERage
```

class Average

Average commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: decimal 'Conventions and General Information'. In list mode, a zero reliability indicator indicates that the results in all measured segments are valid. A non-zero value indicates that an error occurred in at least one of the measured segments.
- Seg_Reliability: List[int]: 0 | 3 | 4 | 8 The segment reliability indicates whether one of the following exceptions occurred in this segment: 0: No error 3: Signal overflow 4: Signal low 8: Synchronization error If a combination of exceptions occurs, the most severe error is indicated.
- Evm_Rms: List[enums.ResultStatus2]: float Error vector magnitude RMS and peak value. Range: 0 % to 100 %, Unit: %
- Evm_Peak: List[enums.ResultStatus2]: float Error vector magnitude RMS and peak value. Range: 0 % to 100 %, Unit: %
- Merr_Rms: List[enums.ResultStatus2]: float Magnitude error RMS value. Range: 0 % to 100 %, Unit: %
- Merr_Peak: List[enums.ResultStatus2]: float Magnitude error peak value. Range: -100 % to +100 % (AVERage: 0% to 100 %, SDEViation: 0 % to 50 %) , Unit: %
- Perr_Rms: List[enums.ResultStatus2]: float Phase error RMS value. Range: 0 deg to 180 deg , Unit: deg
- Perr_Peak: List[enums.ResultStatus2]: float Phase error peak value. Range: -180 deg to 180 deg (AVERage: 0 deg to 180 deg, SDEViation: 0 deg to 90 deg) , Unit: deg
- Iq_Offset: List[enums.ResultStatus2]: float I/Q origin offset Range: -100 dB to 0 dB , Unit: dB
- Iq_Imbalance: List[enums.ResultStatus2]: float I/Q imbalance Range: -100 dB to 0 dB , Unit: dB
- Cfreq_Error: List[enums.ResultStatus2]: float Carrier frequency error Range: -5000 Hz to 5000 Hz , Unit: Hz
- Trans_Time_Err: List[enums.ResultStatus2]: float Transmit time error. Range: -100 µs to 100 µs, Unit: µs
- Ms_Power_1_M_23: List[enums.ResultStatus2]: No parameter help available
- Ms_Power_Wideband: List[enums.ResultStatus2]: No parameter help available
- Wav_Quality: List[enums.ResultStatus2]: float Waveform quality Range: 0 to 1
- Wav_Qual_Max_Pow: List[enums.ResultStatus2]: No parameter help available
- Wav_Qual_Min_Power: List[enums.ResultStatus2]: No parameter help available
- Out_Of_Tol_Count: List[enums.ResultStatus2]: float Out of tolerance result, i.e. percentage of measurement intervals of the statistic count (see [CMDLINK: CONFigure:EVDO:MEASi:MEValuation:SCOunt:MODulation CMDLINK]) exceeding the specified limits, see 'Limits (Modulation) '. Range: 0 % to 100 % , Unit: %
- Cur_Stat_Count: List[enums.ResultStatus2]: decimal Number of evaluated valid slots in this segment. Range: 0 to 1000

class FetchStruct

Response structure. Fields:

- Reliability: int: decimal 'Conventions and General Information'. In list mode, a zero reliability indicator indicates that the results in all measured segments are valid. A non-zero value indicates that an error occurred in at least one of the measured segments.
- Seg_Reliability: List[int]: 0 | 3 | 4 | 8 The segment reliability indicates whether one of the following exceptions occurred in this segment: 0: No error 3: Signal overflow 4: Signal low 8: Synchronization error If a combination of exceptions occurs, the most severe error is indicated.

- Evm_Rms: List[float]: float Error vector magnitude RMS and peak value. Range: 0 % to 100 %, Unit: %
- Evm_Peak: List[float]: float Error vector magnitude RMS and peak value. Range: 0 % to 100 %, Unit: %
- Merr_Rms: List[float]: float Magnitude error RMS value. Range: 0 % to 100 %, Unit: %
- Merr_Peak: List[float]: float Magnitude error peak value. Range: -100 % to +100 % (AVERage: 0% to 100 %, SDEViation: 0 % to 50 %) , Unit: %
- Perr_Rms: List[float]: float Phase error RMS value. Range: 0 deg to 180 deg , Unit: deg
- Perr_Peak: List[float]: float Phase error peak value. Range: -180 deg to 180 deg (AVERage: 0 deg to 180 deg, SDEViation: 0 deg to 90 deg) , Unit: deg
- Iq_Offset: List[float]: float I/Q origin offset Range: -100 dB to 0 dB , Unit: dB
- Iq_Imbalance: List[float]: float I/Q imbalance Range: -100 dB to 0 dB , Unit: dB
- Cfreq_Error: List[float]: float Carrier frequency error Range: -5000 Hz to 5000 Hz , Unit: Hz
- Trans_Time_Err: List[float]: float Transmit time error. Range: -100 µs to 100 µs, Unit: µs
- Ms_Power_1_M_23: List[float]: No parameter help available
- Ms_Power_Wideband: List[float]: No parameter help available
- Wav_Quality: List[float]: float Waveform quality Range: 0 to 1
- Wav_Qual_Max_Pow: List[float]: No parameter help available
- Wav_Qual_Min_Power: List[float]: No parameter help available
- Out_Of_Tol_Count: List[float]: float Out of tolerance result, i.e. percentage of measurement intervals of the statistic count (see [CMDLINK: CONFigure:EVDO:MEASi:MEValuation:SCOunt:MODulation CMDLINK]) exceeding the specified limits, see 'Limits (Modulation) '. Range: 0 % to 100 % , Unit: %
- Cur_Stat_Count: List[int]: decimal Number of evaluated valid slots in this segment. Range: 0 to 1000

calculate() → CalculateStruct

```
# SCPI: CALCulate:EVDO:MEASurement<instance>:MEValuation:LIST:MODulation:AVERage
value: CalculateStruct = driver.multiEval.listPy.modulation.average.calculate()
```

Returns modulation single value results in list mode. To define the statistical length for AVERage, MAXimum, MINimum calculation and enable the calculation of the results use the command method RsCmwEvdoMeas.Configure.MultiEval.ListPy. Segment.Modulation.set. The ranges indicated below apply to all results except standard deviation results. The minimum for standard deviation results equals 0. The maximum equals the width of the indicated range divided by two. Exceptions are explicitly stated. The values listed below in curly brackets { } are returned for each active segment: {...}seg 1, {... }seg 2, ..., {...}seg n. The number of active segments n is determined by method RsCmwEvdoMeas.Configure.MultiEval.ListPy. count. 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:EVDO:MEASurement<instance>:MEValuation:LIST:MODulation:AVERage
value: FetchStruct = driver.multiEval.listPy.modulation.average.fetch()
```

Returns modulation single value results in list mode. To define the statistical length for AVERage, MAXimum, MINimum calculation and enable the calculation of the results use the command method RsCmwEvdoMeas.Configure.MultiEval.ListPy. Segment.Modulation.set. The ranges indicated below apply to all results except standard deviation results. The minimum for standard deviation results equals 0. The maximum equals the width of the indicated range divided by two. Exceptions are explicitly stated. The values listed below in curly brackets {} are returned for each active segment: {...}seg 1, {... }seg 2, ..., {...}seg n. The number of active segments n is determined by method RsCmwEvdoMeas.Configure.MultiEval.ListPy. count. 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.

7.4.7.2.15 Maximum

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:MODulation:MAXimum
CALCulate:EVDO:MEASurement<Instance>:MEValuation:LIST:MODulation:MAXimum
```

class Maximum

Maximum commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: decimal 'Conventions and General Information'. In list mode, a zero reliability indicator indicates that the results in all measured segments are valid. A non-zero value indicates that an error occurred in at least one of the measured segments.
- Seg_Reliability: List[int]: 0 | 3 | 4 | 8 The segment reliability indicates whether one of the following exceptions occurred in this segment: 0: No error 3: Signal overflow 4: Signal low 8: Synchronization error If a combination of exceptions occurs, the most severe error is indicated.
- Evm_Rms: List[enums.ResultStatus2]: float Error vector magnitude RMS and peak value. Range: 0 % to 100 %, Unit: %
- Evm_Peak: List[enums.ResultStatus2]: float Error vector magnitude RMS and peak value. Range: 0 % to 100 %, Unit: %
- Merr_Rms: List[enums.ResultStatus2]: float Magnitude error RMS value. Range: 0 % to 100 %, Unit: %
- Merr_Peak: List[enums.ResultStatus2]: float Magnitude error peak value. Range: -100 % to +100 % (AVERage: 0% to 100 %, SDEViation: 0 % to 50 %) , Unit: %
- Perr_Rms: List[enums.ResultStatus2]: float Phase error RMS value. Range: 0 deg to 180 deg , Unit: deg
- Perr_Peak: List[enums.ResultStatus2]: float Phase error peak value. Range: -180 deg to 180 deg (AVERage: 0 deg to 180 deg, SDEViation: 0 deg to 90 deg) , Unit: deg
- Iq_Offset: List[enums.ResultStatus2]: float I/Q origin offset Range: -100 dB to 0 dB , Unit: dB
- Iq_Imbalance: List[enums.ResultStatus2]: float I/Q imbalance Range: -100 dB to 0 dB , Unit: dB

- Cfreq_Error: List[enums.ResultStatus2]: float Carrier frequency error Range: -5000 Hz to 5000 Hz , Unit: Hz
- Trans_Time_Err: List[enums.ResultStatus2]: float Transmit time error. Range: -100 µs to 100 µs, Unit: µs
- Ms_Power_1_M_23: List[enums.ResultStatus2]: No parameter help available
- Ms_Power_Wideband: List[enums.ResultStatus2]: No parameter help available
- Wav_Quality: List[enums.ResultStatus2]: float Waveform quality Range: 0 to 1
- Wav_Qual_Max_Pow: List[enums.ResultStatus2]: No parameter help available
- Wav_Qual_Min_Power: List[enums.ResultStatus2]: No parameter help available
- Out_Of_Tol_Count: List[enums.ResultStatus2]: float Out of tolerance result, i.e. percentage of measurement intervals of the statistic count (see [CMDLINK: CONFigure:EVDO:MEASi:MEValuation:SCOunt:MODulation CMDLINK]) exceeding the specified limits, see 'Limits (Modulation) '. Range: 0 % to 100 % , Unit: %
- Cur_Stat_Count: List[enums.ResultStatus2]: decimal Number of evaluated valid slots in this segment. Range: 0 to 1000

class FetchStruct

Response structure. Fields:

- Reliability: int: decimal 'Conventions and General Information'. In list mode, a zero reliability indicator indicates that the results in all measured segments are valid. A non-zero value indicates that an error occurred in at least one of the measured segments.
- Seg_Reliability: List[int]: 0 | 3 | 4 | 8 The segment reliability indicates whether one of the following exceptions occurred in this segment: 0: No error 3: Signal overflow 4: Signal low 8: Synchronization error If a combination of exceptions occurs, the most severe error is indicated.
- Evm_Rms: List[float]: float Error vector magnitude RMS and peak value. Range: 0 % to 100 %, Unit: %
- Evm_Peak: List[float]: float Error vector magnitude RMS and peak value. Range: 0 % to 100 %, Unit: %
- Merr_Rms: List[float]: float Magnitude error RMS value. Range: 0 % to 100 %, Unit: %
- Merr_Peak: List[float]: float Magnitude error peak value. Range: -100 % to +100 % (AVERage: 0% to 100 %, SDEViation: 0 % to 50 %) , Unit: %
- Perr_Rms: List[float]: float Phase error RMS value. Range: 0 deg to 180 deg , Unit: deg
- Perr_Peak: List[float]: float Phase error peak value. Range: -180 deg to 180 deg (AVERage: 0 deg to 180 deg, SDEViation: 0 deg to 90 deg) , Unit: deg
- Iq_Offset: List[float]: float I/Q origin offset Range: -100 dB to 0 dB , Unit: dB
- Iq_Imbalance: List[float]: float I/Q imbalance Range: -100 dB to 0 dB , Unit: dB
- Cfreq_Error: List[float]: float Carrier frequency error Range: -5000 Hz to 5000 Hz , Unit: Hz
- Trans_Time_Err: List[float]: float Transmit time error. Range: -100 µs to 100 µs, Unit: µs
- Ms_Power_1_M_23: List[float]: No parameter help available
- Ms_Power_Wideband: List[float]: No parameter help available
- Wav_Quality: List[float]: float Waveform quality Range: 0 to 1
- Wav_Qual_Max_Pow: List[float]: No parameter help available

- Wav_Qual_Min_Power: List[float]: No parameter help available
- Out_Of_Tol_Count: List[float]: float Out of tolerance result, i.e. percentage of measurement intervals of the statistic count (see [CMDLINK: CONFigure:EVDO:MEASi:MEValuation:SCOunt:MODulation CMDLINK]) exceeding the specified limits, see 'Limits (Modulation) '. Range: 0 % to 100 % , Unit: %
- Cur_Stat_Count: List[int]: decimal Number of evaluated valid slots in this segment. Range: 0 to 1000

calculate() → CalculateStruct

```
# SCPI: CALCulate:EVDO:MEASurement<instance>:MEValuation:LIST:MODulation:MAXimum
value: CalculateStruct = driver.multiEval.listPy.modulation.maximum.calculate()
```

Returns modulation single value results in list mode. To define the statistical length for AVERage, MAXimum, MINimum calculation and enable the calculation of the results use the command method RsCmwEvdoMeas.Configure.MultiEval.ListPy. Segment.Modulation.set. The ranges indicated below apply to all results except standard deviation results. The minimum for standard deviation results equals 0. The maximum equals the width of the indicated range divided by two. Exceptions are explicitly stated. The values listed below in curly brackets {} are returned for each active segment: {...}seg 1, {... }seg 2, ..., {...}seg n. The number of active segments n is determined by method RsCmwEvdoMeas.Configure.MultiEval.ListPy. count. 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:EVDO:MEASurement<instance>:MEValuation:LIST:MODulation:MAXimum
value: FetchStruct = driver.multiEval.listPy.modulation.maximum.fetch()
```

Returns modulation single value results in list mode. To define the statistical length for AVERage, MAXimum, MINimum calculation and enable the calculation of the results use the command method RsCmwEvdoMeas.Configure.MultiEval.ListPy. Segment.Modulation.set. The ranges indicated below apply to all results except standard deviation results. The minimum for standard deviation results equals 0. The maximum equals the width of the indicated range divided by two. Exceptions are explicitly stated. The values listed below in curly brackets {} are returned for each active segment: {...}seg 1, {... }seg 2, ..., {...}seg n. The number of active segments n is determined by method RsCmwEvdoMeas.Configure.MultiEval.ListPy. count. 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.

7.4.7.2.16 Minimum

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:MODulation:MINimum
CALCulate:EVDO:MEASurement<Instance>:MEValuation:LIST:MODulation:MINimum
```

class Minimum

Minimum commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: decimal 'Conventions and General Information'. In list mode, a zero reliability indicator indicates that the results in all measured segments are valid. A non-zero value indicates that an error occurred in at least one of the measured segments.
- Seg_Reliability: List[int]: 0 | 3 | 4 | 8 The segment reliability indicates whether one of the following exceptions occurred in this segment: 0: No error 3: Signal overflow 4: Signal low 8: Synchronization error If a combination of exceptions occurs, the most severe error is indicated.
- Evm_Rms: List[enums.ResultStatus2]: float Error vector magnitude RMS and peak value. Range: 0 % to 100 %, Unit: %
- Evm_Peak: List[enums.ResultStatus2]: float Error vector magnitude RMS and peak value. Range: 0 % to 100 %, Unit: %
- Merr_Rms: List[enums.ResultStatus2]: float Magnitude error RMS value. Range: 0 % to 100 %, Unit: %
- Merr_Peak: List[enums.ResultStatus2]: float Magnitude error peak value. Range: -100 % to +100 % (AVERage: 0% to 100 %, SDEViation: 0 % to 50 %) , Unit: %
- Perr_Rms: List[enums.ResultStatus2]: float Phase error RMS value. Range: 0 deg to 180 deg , Unit: deg
- Perr_Peak: List[enums.ResultStatus2]: float Phase error peak value. Range: -180 deg to 180 deg (AVERage: 0 deg to 180 deg, SDEViation: 0 deg to 90 deg) , Unit: deg
- Iq_Offset: List[enums.ResultStatus2]: float I/Q origin offset Range: -100 dB to 0 dB , Unit: dB
- Iq_Imbalance: List[enums.ResultStatus2]: float I/Q imbalance Range: -100 dB to 0 dB , Unit: dB
- Cfreq_Error: List[enums.ResultStatus2]: float Carrier frequency error Range: -5000 Hz to 5000 Hz , Unit: Hz
- Trans_Time_Err: List[enums.ResultStatus2]: float Transmit time error. Range: -100 µs to 100 µs, Unit: µs
- Ms_Power_1_M_23: List[enums.ResultStatus2]: No parameter help available
- Ms_Power_Wideband: List[enums.ResultStatus2]: No parameter help available
- Wav_Quality: List[enums.ResultStatus2]: float Waveform quality Range: 0 to 1
- Wav_Qual_Max_Pow: List[enums.ResultStatus2]: No parameter help available
- Wav_Qual_Min_Power: List[enums.ResultStatus2]: No parameter help available
- Out_Of_Tol_Count: List[enums.ResultStatus2]: float Out of tolerance result, i.e. percentage of measurement intervals of the statistic count (see [CMDLINK: CONFigure:EVDO:MEASi:MEValuation:SCOunt:MODulation CMDLINK]) exceeding the specified limits, see 'Limits (Modulation) '. Range: 0 % to 100 % , Unit: %
- Cur_Stat_Count: List[enums.ResultStatus2]: decimal Number of evaluated valid slots in this segment. Range: 0 to 1000

class FetchStruct

Response structure. Fields:

- Reliability: int: decimal 'Conventions and General Information'. In list mode, a zero reliability indicator indicates that the results in all measured segments are valid. A non-zero value indicates that an error occurred in at least one of the measured segments.
- Seg_Reliability: List[int]: 0 | 3 | 4 | 8 The segment reliability indicates whether one of the following exceptions occurred in this segment: 0: No error 3: Signal overflow 4: Signal low 8: Synchronization error If a combination of exceptions occurs, the most severe error is indicated.

- Evm_Rms: List[float]: float Error vector magnitude RMS and peak value. Range: 0 % to 100 %, Unit: %
- Evm_Peak: List[float]: float Error vector magnitude RMS and peak value. Range: 0 % to 100 %, Unit: %
- Merr_Rms: List[float]: float Magnitude error RMS value. Range: 0 % to 100 %, Unit: %
- Merr_Peak: List[float]: float Magnitude error peak value. Range: -100 % to +100 % (AVERage: 0% to 100 %, SDEViation: 0 % to 50 %) , Unit: %
- Perr_Rms: List[float]: float Phase error RMS value. Range: 0 deg to 180 deg , Unit: deg
- Perr_Peak: List[float]: float Phase error peak value. Range: -180 deg to 180 deg (AVERage: 0 deg to 180 deg, SDEViation: 0 deg to 90 deg) , Unit: deg
- Iq_Offset: List[float]: float I/Q origin offset Range: -100 dB to 0 dB , Unit: dB
- Iq_Imbalance: List[float]: float I/Q imbalance Range: -100 dB to 0 dB , Unit: dB
- Cfreq_Error: List[float]: float Carrier frequency error Range: -5000 Hz to 5000 Hz , Unit: Hz
- Trans_Time_Err: List[float]: float Transmit time error. Range: -100 µs to 100 µs, Unit: µs
- Ms_Power_1_M_23: List[float]: No parameter help available
- Ms_Power_Wideband: List[float]: No parameter help available
- Wav_Quality: List[float]: float Waveform quality Range: 0 to 1
- Wav_Qual_Max_Pow: List[float]: No parameter help available
- Wav_Qual_Min_Power: List[float]: No parameter help available
- Out_Of_Tol_Count: List[float]: float Out of tolerance result, i.e. percentage of measurement intervals of the statistic count (see [CMDLINK: CONFigure:EVDO:MEASi:MEValuation:SCOunt:MODulation CMDLINK]) exceeding the specified limits, see 'Limits (Modulation) '. Range: 0 % to 100 % , Unit: %
- Cur_Stat_Count: List[int]: decimal Number of evaluated valid slots in this segment. Range: 0 to 1000

calculate() → CalculateStruct

```
# SCPI: CALCulate:EVDO:MEASurement<instance>:MEValuation:LIST:MODulation:MINimum
value: CalculateStruct = driver.multiEval.listPy.modulation.minimum.calculate()
```

Returns modulation single value results in list mode. To define the statistical length for AVERage, MAXimum, MINimum calculation and enable the calculation of the results use the command method RsCmwEvdoMeas.Configure.MultiEval.ListPy. Segment.Modulation.set. The ranges indicated below apply to all results except standard deviation results. The minimum for standard deviation results equals 0. The maximum equals the width of the indicated range divided by two. Exceptions are explicitly stated. The values listed below in curly brackets {} are returned for each active segment: {...}seg 1, {... }seg 2, ..., {...}seg n. The number of active segments n is determined by method RsCmwEvdoMeas.Configure.MultiEval.ListPy. count. 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:EVDO:MEASurement<instance>:MEValuation:LIST:MODulation:MINimum
value: FetchStruct = driver.multiEval.listPy.modulation.minimum.fetch()
```

Returns modulation single value results in list mode. To define the statistical length for AVERage, MAXimum, MINimum calculation and enable the calculation of the results use the command method RsCmwEvdoMeas.Configure.MultiEval.ListPy. Segment.Modulation.set. The ranges indicated below apply to all results except standard deviation results. The minimum for standard deviation results equals 0. The maximum equals the width of the indicated range divided by two. Exceptions are explicitly stated. The values listed below in curly brackets {} are returned for each active segment: {...}seg 1, {... }seg 2, ..., {...}seg n. The number of active segments n is determined by method RsCmwEvdoMeas.Configure.MultiEval.ListPy. count. 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.

7.4.7.2.17 StandardDev

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:MODulation:SDEViation
CALCulate:EVDO:MEASurement<Instance>:MEValuation:LIST:MODulation:SDEViation
```

class StandardDev

StandardDev commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: decimal 'Conventions and General Information'. In list mode, a zero reliability indicator indicates that the results in all measured segments are valid. A non-zero value indicates that an error occurred in at least one of the measured segments.
- Seg_Reliability: List[int]: 0 | 3 | 4 | 8 The segment reliability indicates whether one of the following exceptions occurred in this segment: 0: No error 3: Signal overflow 4: Signal low 8: Synchronization error If a combination of exceptions occurs, the most severe error is indicated.
- Evm_Rms: List[enums.ResultStatus2]: float Error vector magnitude RMS and peak value. Range: 0 % to 100 %, Unit: %
- Evm_Peak: List[enums.ResultStatus2]: float Error vector magnitude RMS and peak value. Range: 0 % to 100 %, Unit: %
- Merr_Rms: List[enums.ResultStatus2]: float Magnitude error RMS value. Range: 0 % to 100 %, Unit: %
- Merr_Peak: List[enums.ResultStatus2]: float Magnitude error peak value. Range: -100 % to +100 % (AVERage: 0% to 100 %, SDEViation: 0 % to 50 %) , Unit: %
- Perr_Rms: List[enums.ResultStatus2]: float Phase error RMS value. Range: 0 deg to 180 deg , Unit: deg
- Perr_Peak: List[enums.ResultStatus2]: float Phase error peak value. Range: -180 deg to 180 deg (AVERage: 0 deg to 180 deg, SDEViation: 0 deg to 90 deg) , Unit: deg
- Iq_Offset: List[enums.ResultStatus2]: float I/Q origin offset Range: -100 dB to 0 dB , Unit: dB
- Iq_Imbalance: List[enums.ResultStatus2]: float I/Q imbalance Range: -100 dB to 0 dB , Unit: dB

- Cfreq_Error: List[enums.ResultStatus2]: float Carrier frequency error Range: -5000 Hz to 5000 Hz , Unit: Hz
- Trans_Time_Err: List[enums.ResultStatus2]: float Transmit time error. Range: -100 µs to 100 µs, Unit: µs
- Ms_Power_1_M_23: List[enums.ResultStatus2]: No parameter help available
- Ms_Power_Wideband: List[enums.ResultStatus2]: No parameter help available
- Wav_Quality: List[enums.ResultStatus2]: float Waveform quality Range: 0 to 1
- Wav_Qual_Max_Pow: List[enums.ResultStatus2]: No parameter help available
- Wav_Qual_Min_Power: List[enums.ResultStatus2]: No parameter help available
- Out_Of_Tol_Count: List[enums.ResultStatus2]: float Out of tolerance result, i.e. percentage of measurement intervals of the statistic count (see [CMDLINK: CONFigure:EVDO:MEASi:MEValuation:SCOunt:MODulation CMDLINK]) exceeding the specified limits, see 'Limits (Modulation) '. Range: 0 % to 100 % , Unit: %
- Cur_Stat_Count: List[enums.ResultStatus2]: decimal Number of evaluated valid slots in this segment. Range: 0 to 1000

class FetchStruct

Response structure. Fields:

- Reliability: int: decimal 'Conventions and General Information'. In list mode, a zero reliability indicator indicates that the results in all measured segments are valid. A non-zero value indicates that an error occurred in at least one of the measured segments.
- Seg_Reliability: List[int]: 0 | 3 | 4 | 8 The segment reliability indicates whether one of the following exceptions occurred in this segment: 0: No error 3: Signal overflow 4: Signal low 8: Synchronization error If a combination of exceptions occurs, the most severe error is indicated.
- Evm_Rms: List[float]: float Error vector magnitude RMS and peak value. Range: 0 % to 100 %, Unit: %
- Evm_Peak: List[float]: float Error vector magnitude RMS and peak value. Range: 0 % to 100 %, Unit: %
- Merr_Rms: List[float]: float Magnitude error RMS value. Range: 0 % to 100 %, Unit: %
- Merr_Peak: List[float]: float Magnitude error peak value. Range: -100 % to +100 % (AVERage: 0% to 100 %, SDEViation: 0 % to 50 %) , Unit: %
- Perr_Rms: List[float]: float Phase error RMS value. Range: 0 deg to 180 deg , Unit: deg
- Perr_Peak: List[float]: float Phase error peak value. Range: -180 deg to 180 deg (AVERage: 0 deg to 180 deg, SDEViation: 0 deg to 90 deg) , Unit: deg
- Iq_Offset: List[float]: float I/Q origin offset Range: -100 dB to 0 dB , Unit: dB
- Iq_Imbalance: List[float]: float I/Q imbalance Range: -100 dB to 0 dB , Unit: dB
- Cfreq_Error: List[float]: float Carrier frequency error Range: -5000 Hz to 5000 Hz , Unit: Hz
- Trans_Time_Err: List[float]: float Transmit time error. Range: -100 µs to 100 µs, Unit: µs
- Ms_Power_1_M_23: List[float]: No parameter help available
- Ms_Power_Wideband: List[float]: No parameter help available
- Wav_Quality: List[float]: float Waveform quality Range: 0 to 1
- Wav_Qual_Max_Pow: List[float]: No parameter help available

- Wav_Qual_Min_Power: List[float]: No parameter help available
- Out_Of_Tol_Count: List[float]: float Out of tolerance result, i.e. percentage of measurement intervals of the statistic count (see [CMDLINK: CONFigure:EVDO:MEASi:MEValuation:SCOunt:MODulation CMDLINK]) exceeding the specified limits, see 'Limits (Modulation) '. Range: 0 % to 100 % , Unit: %
- Cur_Stat_Count: List[int]: decimal Number of evaluated valid slots in this segment. Range: 0 to 1000

calculate() → CalculateStruct

```
# SCPI: CALCulate:EVDO:MEASurement<instance>
↪:MEValuation:LIST:MODulation:SDEViation
value: CalculateStruct = driver.multiEval.listPy.modulation.standardDev.
↪calculate()
```

Returns modulation single value results in list mode. To define the statistical length for AVERage, MAXimum, MINimum calculation and enable the calculation of the results use the command method RsCmwEvdoMeas.Configure.MultiEval.ListPy. Segment.Modulation.set. The ranges indicated below apply to all results except standard deviation results. The minimum for standard deviation results equals 0. The maximum equals the width of the indicated range divided by two. Exceptions are explicitly stated. The values listed below in curly brackets {} are returned for each active segment: {...}seg 1, {... }seg 2, ..., {...}seg n. The number of active segments n is determined by method RsCmwEvdoMeas.Configure.MultiEval.ListPy. count. 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:EVDO:MEASurement<instance>:MEValuation:LIST:MODulation:SDEViation
value: FetchStruct = driver.multiEval.listPy.modulation.standardDev.fetch()
```

Returns modulation single value results in list mode. To define the statistical length for AVERage, MAXimum, MINimum calculation and enable the calculation of the results use the command method RsCmwEvdoMeas.Configure.MultiEval.ListPy. Segment.Modulation.set. The ranges indicated below apply to all results except standard deviation results. The minimum for standard deviation results equals 0. The maximum equals the width of the indicated range divided by two. Exceptions are explicitly stated. The values listed below in curly brackets {} are returned for each active segment: {...}seg 1, {... }seg 2, ..., {...}seg n. The number of active segments n is determined by method RsCmwEvdoMeas.Configure.MultiEval.ListPy. count. 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.

7.4.7.3 Segment

RepCap Settings

```
# Range: Nr1 .. Nr200
rc = driver.multiEval.listPy.segment.repcap_segment_get()
driver.multiEval.listPy.segment.repcap_segment_set(repcap.Segment.Nr1)
```

class Segment
Segment commands group definition. 55 total commands, 5 Sub-groups, 0 group commands Repeated Capability: Segment, default value after init: Segment.Nr1

Cloning the Group

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

Subgroups

7.4.7.3.1 Modulation

class Modulation
Modulation commands group definition. 10 total commands, 5 Sub-groups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.multiEval.listPy.segment.modulation.clone()
```

Subgroups

7.4.7.3.1.1 Current

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:SEGMent<Segment>:MODulation:CURRent
CALCulate:EVDO:MEASurement<Instance>:MEValuation:LIST:SEGMent<Segment>:MODulation:CURRent
```

class Current
Current commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

class CalculateStruct
Response structure. Fields:

- Reliability: int: No parameter help available
- Seg_Reliability: int: 0 | 3 | 4 | 8 The segment reliability indicates whether one of the following exceptions occurred in this segment: 0: No error 3: Signal overflow 4: Signal low 8: Synchronization error If a combination of exceptions occurs, the most severe error is indicated.

- Evm_Rms: enums.ResultStatus2: float Error vector magnitude RMS and peak value. Range: 0 % to 100 %, Unit: %
- Evm_Peak: enums.ResultStatus2: float Error vector magnitude RMS and peak value. Range: 0 % to 100 %, Unit: %
- Merr_Rms: enums.ResultStatus2: float Magnitude error RMS value. Range: 0 % to 100 %, Unit: %
- Merr_Peak: enums.ResultStatus2: float Magnitude error peak value. Range: -100 % to +100 % (AVERage: 0% to 100 %, SDEViation: 0 % to 50 %) , Unit: %
- Perr_Rms: enums.ResultStatus2: float Phase error RMS value. Range: 0 deg to 180 deg , Unit: deg
- Perr_Peak: enums.ResultStatus2: float Phase error peak value. Range: -180 deg to 180 deg (AVERage: 0 deg to 180 deg, SDEViation: 0 deg to 90 deg) , Unit: deg
- Iq_Offset: enums.ResultStatus2: float I/Q origin offset Range: -100 dB to 0 dB , Unit: dB
- Iq_Imbalance: enums.ResultStatus2: float I/Q imbalance Range: -100 dB to 0 dB , Unit: dB
- Cfreq_Error: enums.ResultStatus2: float Carrier frequency error Range: -5000 Hz to 5000 Hz , Unit: Hz
- Trans_Time_Err: enums.ResultStatus2: float Transmit time error. Only for future use. Therefore the return value is always NCAP Range: NCAP , Unit: µs
- Ms_Power_1_M_23: enums.ResultStatus2: No parameter help available
- Ms_Power_Wideband: enums.ResultStatus2: No parameter help available
- Wav_Quality: enums.ResultStatus2: float Waveform quality Range: 0 to 1
- Wav_Qual_Max_Pow: enums.ResultStatus2: No parameter help available
- Wav_Qual_Min_Power: enums.ResultStatus2: No parameter help available
- Out_Of_Tol_Count: enums.ResultStatus2: float Out of tolerance result, i.e. percentage of measurement intervals of the statistic count (see [CMDLINK: CONFigure:EVDO:MEASi:MEValuation:SCOunt:MODulation CMDLINK]) exceeding the specified limits, see 'Limits (Modulation) '. Range: 0 % to 100 % , Unit: %
- Cur_Stat_Count: enums.ResultStatus2: decimal Number of evaluated valid slots in this segment. Range: 0 to 1000

class FetchStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Seg_Reliability: int: 0 | 3 | 4 | 8 The segment reliability indicates whether one of the following exceptions occurred in this segment: 0: No error 3: Signal overflow 4: Signal low 8: Synchronization error If a combination of exceptions occurs, the most severe error is indicated.
- Evm_Rms: float: float Error vector magnitude RMS and peak value. Range: 0 % to 100 %, Unit: %
- Evm_Peak: float: float Error vector magnitude RMS and peak value. Range: 0 % to 100 %, Unit: %
- Merr_Rms: float: float Magnitude error RMS value. Range: 0 % to 100 %, Unit: %
- Merr_Peak: float: float Magnitude error peak value. Range: -100 % to +100 % (AVERage: 0% to 100 %, SDEViation: 0 % to 50 %) , Unit: %
- Perr_Rms: float: float Phase error RMS value. Range: 0 deg to 180 deg , Unit: deg
- Perr_Peak: float: float Phase error peak value. Range: -180 deg to 180 deg (AVERage: 0 deg to 180 deg, SDEViation: 0 deg to 90 deg) , Unit: deg

- Iq_Offset: float: float I/Q origin offset Range: -100 dB to 0 dB , Unit: dB
- Iq_Imbalance: float: float I/Q imbalance Range: -100 dB to 0 dB , Unit: dB
- Cfreq_Error: float: float Carrier frequency error Range: -5000 Hz to 5000 Hz , Unit: Hz
- Trans_Time_Err: float: float Transmit time error. Only for future use. Therefore the return value is always NCAP Range: NCAP , Unit: µs
- Ms_Power_1_M_23: float: No parameter help available
- Ms_Power_Wideband: float: No parameter help available
- Wav_Quality: float: float Waveform quality Range: 0 to 1
- Wav_Qual_Max_Pow: float: No parameter help available
- Wav_Qual_Min_Power: float: No parameter help available
- Out_Of_Tol_Count: float: float Out of tolerance result, i.e. percentage of measurement intervals of the statistic count (see [CMDLINK: CONFigure:EVDO:MEASi:MEValuation:SCOunt:MODulation CMDLINK]) exceeding the specified limits, see 'Limits (Modulation) '. Range: 0 % to 100 % , Unit: %
- Cur_Stat_Count: int: decimal Number of evaluated valid slots in this segment. Range: 0 to 1000

calculate(*segment=<Segment.Default: -1>*) → CalculateStruct

```
# SCPI: CALCulate:EVDO:MEASurement<instance>:MEValuation:LIST:SEGMent<nr>
↪:MODulation:CURRent
value: CalculateStruct = driver.multiEval.listPy.segment.modulation.current.
↪calculate(segment = repcap.Segment.Default)
```

Returns modulation single value results for segment <no> in list mode. To define the statistical length for AVERage, MAXimum, MINimum calculation and enable the calculation of the results use the command method RsCmwEvdoMeas.Configure. MultiEval.ListPy.Segment.Modulation.set. The ranges indicated below apply to all results except standard deviation results. The minimum for standard deviation results equals 0. The maximum equals the width of the indicated range divided by two. Exceptions are explicitly stated. 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.

param segment optional repeated capability selector. Default value: Nr1 (settable in the interface 'Segment')

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

fetch(*segment=<Segment.Default: -1>*) → FetchStruct

```
# SCPI: FETCh:EVDO:MEASurement<instance>:MEValuation:LIST:SEGMent<nr>
↪:MODulation:CURRent
value: FetchStruct = driver.multiEval.listPy.segment.modulation.current.
↪fetch(segment = repcap.Segment.Default)
```

Returns modulation single value results for segment <no> in list mode. To define the statistical length for AVERage, MAXimum, MINimum calculation and enable the calculation of the results use the command method RsCmwEvdoMeas.Configure. MultiEval.ListPy.Segment.Modulation.set. The ranges indicated below apply to all results except standard deviation results. The minimum for standard deviation results equals 0. The maximum equals the width of the indicated range divided by two. Exceptions are explicitly

stated. 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.

param segment optional repeated capability selector. Default value: Nr1 (settable in the interface 'Segment')

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

7.4.7.3.1.2 Average

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:SEGMent<Segment>:MODulation:AVERage
CALCulate:EVDO:MEASurement<Instance>:MEValuation:LIST:SEGMent<Segment>:MODulation:AVERage
```

class Average

Average commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Seg_Reliability: int: 0 | 3 | 4 | 8 The segment reliability indicates whether one of the following exceptions occurred in this segment: 0: No error 3: Signal overflow 4: Signal low 8: Synchronization error If a combination of exceptions occurs, the most severe error is indicated.
- Evm_Rms: enums.ResultStatus2: float Error vector magnitude RMS and peak value. Range: 0 % to 100 %, Unit: %
- Evm_Peak: enums.ResultStatus2: float Error vector magnitude RMS and peak value. Range: 0 % to 100 %, Unit: %
- Merr_Rms: enums.ResultStatus2: float Magnitude error RMS value. Range: 0 % to 100 %, Unit: %
- Merr_Peak: enums.ResultStatus2: float Magnitude error peak value. Range: -100 % to +100 % (AVERage: 0% to 100 %, SDEViation: 0 % to 50 %) , Unit: %
- Perr_Rms: enums.ResultStatus2: float Phase error RMS value. Range: 0 deg to 180 deg , Unit: deg
- Perr_Peak: enums.ResultStatus2: float Phase error peak value. Range: -180 deg to 180 deg (AVERage: 0 deg to 180 deg, SDEViation: 0 deg to 90 deg) , Unit: deg
- Iq_Offset: enums.ResultStatus2: float I/Q origin offset Range: -100 dB to 0 dB , Unit: dB
- Iq_Imbalance: enums.ResultStatus2: float I/Q imbalance Range: -100 dB to 0 dB , Unit: dB
- Cfreq_Error: enums.ResultStatus2: float Carrier frequency error Range: -5000 Hz to 5000 Hz , Unit: Hz
- Trans_Time_Err: enums.ResultStatus2: float Transmit time error. Only for future use. Therefore the return value is always NCAP Range: NCAP , Unit: µs
- Ms_Power_1_M_23: enums.ResultStatus2: No parameter help available
- Ms_Power_Wideband: enums.ResultStatus2: No parameter help available
- Wav_Quality: enums.ResultStatus2: float Waveform quality Range: 0 to 1
- Wav_Qual_Max_Pow: enums.ResultStatus2: No parameter help available
- Wav_Qual_Min_Power: enums.ResultStatus2: No parameter help available

- Out_Of_Tol_Count: enums.ResultStatus2: float Out of tolerance result, i.e. percentage of measurement intervals of the statistic count (see [CMDLINK: CONFigure:EVDO:MEASi:MEValuation:SCOunt:MODulation CMDLINK]) exceeding the specified limits, see 'Limits (Modulation) '. Range: 0 % to 100 % , Unit: %
- Cur_Stat_Count: enums.ResultStatus2: decimal Number of evaluated valid slots in this segment. Range: 0 to 1000

class FetchStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Seg_Reliability: int: 0 | 3 | 4 | 8 The segment reliability indicates whether one of the following exceptions occurred in this segment: 0: No error 3: Signal overflow 4: Signal low 8: Synchronization error If a combination of exceptions occurs, the most severe error is indicated.
- Evm_Rms: float: float Error vector magnitude RMS and peak value. Range: 0 % to 100 %, Unit: %
- Evm_Peak: float: float Error vector magnitude RMS and peak value. Range: 0 % to 100 %, Unit: %
- Merr_Rms: float: float Magnitude error RMS value. Range: 0 % to 100 %, Unit: %
- Merr_Peak: float: float Magnitude error peak value. Range: -100 % to +100 % (AVERage: 0% to 100 %, SDEViation: 0 % to 50 %) , Unit: %
- Perr_Rms: float: float Phase error RMS value. Range: 0 deg to 180 deg , Unit: deg
- Perr_Peak: float: float Phase error peak value. Range: -180 deg to 180 deg (AVERage: 0 deg to 180 deg, SDEViation: 0 deg to 90 deg) , Unit: deg
- Iq_Offset: float: float I/Q origin offset Range: -100 dB to 0 dB , Unit: dB
- Iq_Imbalance: float: float I/Q imbalance Range: -100 dB to 0 dB , Unit: dB
- Cfreq_Error: float: float Carrier frequency error Range: -5000 Hz to 5000 Hz , Unit: Hz
- Trans_Time_Err: float: float Transmit time error. Only for future use. Therefore the return value is always NCAP Range: NCAP , Unit: µs
- Ms_Power_1_M_23: float: No parameter help available
- Ms_Power_Wideband: float: No parameter help available
- Wav_Quality: float: float Waveform quality Range: 0 to 1
- Wav_Qual_Max_Pow: float: No parameter help available
- Wav_Qual_Min_Power: float: No parameter help available
- Out_Of_Tol_Count: float: float Out of tolerance result, i.e. percentage of measurement intervals of the statistic count (see [CMDLINK: CONFigure:EVDO:MEASi:MEValuation:SCOunt:MODulation CMDLINK]) exceeding the specified limits, see 'Limits (Modulation) '. Range: 0 % to 100 % , Unit: %
- Cur_Stat_Count: int: decimal Number of evaluated valid slots in this segment. Range: 0 to 1000

calculate(*segment*=<*Segment.Default: -1*>) → CalculateStruct

```
# SCPI: CALCulate:EVDO:MEASurement<instance>:MEValuation:LIST:SEGMent<nr>
↪:MODulation:AVERage
value: CalculateStruct = driver.multiEval.listPy.segment.modulation.average.
↪calculate(segment = repcap.Segment.Default)
```

Returns modulation single value results for segment <no> in list mode. To define the statistical length for AVERage, MAXimum, MINimum calculation and enable the calculation of the results use the command method RsCmwEvdoMeas.Configure. MultiEval.ListPy.Segment.Modulation.set. The ranges indicated below apply to all results except standard deviation results. The minimum for standard deviation results equals 0. The maximum equals the width of the indicated range divided by two. Exceptions are explicitly stated. 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.

param segment optional repeated capability selector. Default value: Nr1 (settable in the interface 'Segment')

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

fetch(*segment=<Segment.Default: -1>*) → FetchStruct

```
# SCPI: FETCh:EVDO:MEASurement<instance>:MEValuation:LIST:SEGMent<nr>
↪:MODulation:AVERage
value: FetchStruct = driver.multiEval.listPy.segment.modulation.average.
↪fetch(segment = repcap.Segment.Default)
```

Returns modulation single value results for segment <no> in list mode. To define the statistical length for AVERage, MAXimum, MINimum calculation and enable the calculation of the results use the command method RsCmwEvdoMeas.Configure. MultiEval.ListPy.Segment.Modulation.set. The ranges indicated below apply to all results except standard deviation results. The minimum for standard deviation results equals 0. The maximum equals the width of the indicated range divided by two. Exceptions are explicitly stated. 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.

param segment optional repeated capability selector. Default value: Nr1 (settable in the interface 'Segment')

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

7.4.7.3.1.3 Maximum

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:SEGMent<Segment>:MODulation:MAXimum
CALCulate:EVDO:MEASurement<Instance>:MEValuation:LIST:SEGMent<Segment>:MODulation:MAXimum
```

class Maximum

Maximum commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Seg_Reliability: int: 0 | 3 | 4 | 8 The segment reliability indicates whether one of the following exceptions occurred in this segment: 0: No error 3: Signal overflow 4: Signal low 8: Synchronization error If a combination of exceptions occurs, the most severe error is indicated.
- Evm_Rms: enums.ResultStatus2: float Error vector magnitude RMS and peak value. Range: 0 % to 100 %, Unit: %
- Evm_Peak: enums.ResultStatus2: float Error vector magnitude RMS and peak value. Range: 0 % to 100 %, Unit: %

- Merr_Rms: enums.ResultStatus2: float Magnitude error RMS value. Range: 0 % to 100 %, Unit: %
- Merr_Peak: enums.ResultStatus2: float Magnitude error peak value. Range: -100 % to +100 % (AVERage: 0% to 100 %, SDEViation: 0 % to 50 %) , Unit: %
- Perr_Rms: enums.ResultStatus2: float Phase error RMS value. Range: 0 deg to 180 deg , Unit: deg
- Perr_Peak: enums.ResultStatus2: float Phase error peak value. Range: -180 deg to 180 deg (AVERage: 0 deg to 180 deg, SDEViation: 0 deg to 90 deg) , Unit: deg
- Iq_Offset: enums.ResultStatus2: float I/Q origin offset Range: -100 dB to 0 dB , Unit: dB
- Iq_Imbalance: enums.ResultStatus2: float I/Q imbalance Range: -100 dB to 0 dB , Unit: dB
- Cfreq_Error: enums.ResultStatus2: float Carrier frequency error Range: -5000 Hz to 5000 Hz , Unit: Hz
- Trans_Time_Err: enums.ResultStatus2: float Transmit time error. Only for future use. Therefore the return value is always NCAP Range: NCAP , Unit: µs
- Ms_Power_1_M_23: enums.ResultStatus2: No parameter help available
- Ms_Power_Wideband: enums.ResultStatus2: No parameter help available
- Wav_Quality: enums.ResultStatus2: float Waveform quality Range: 0 to 1
- Wav_Qual_Max_Pow: enums.ResultStatus2: No parameter help available
- Wav_Qual_Min_Power: enums.ResultStatus2: No parameter help available
- Out_Of_Tol_Count: enums.ResultStatus2: float Out of tolerance result, i.e. percentage of measurement intervals of the statistic count (see [CMDLINK: CONFigure:EVDO:MEASi:MEValuation:SCOunt:MODulation CMDLINK]) exceeding the specified limits, see 'Limits (Modulation) '. Range: 0 % to 100 % , Unit: %
- Cur_Stat_Count: enums.ResultStatus2: decimal Number of evaluated valid slots in this segment. Range: 0 to 1000

class FetchStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Seg_Reliability: int: 0 | 3 | 4 | 8 The segment reliability indicates whether one of the following exceptions occurred in this segment: 0: No error 3: Signal overflow 4: Signal low 8: Synchronization error If a combination of exceptions occurs, the most severe error is indicated.
- Evm_Rms: float: float Error vector magnitude RMS and peak value. Range: 0 % to 100 %, Unit: %
- Evm_Peak: float: float Error vector magnitude RMS and peak value. Range: 0 % to 100 %, Unit: %
- Merr_Rms: float: float Magnitude error RMS value. Range: 0 % to 100 %, Unit: %
- Merr_Peak: float: float Magnitude error peak value. Range: -100 % to +100 % (AVERage: 0% to 100 %, SDEViation: 0 % to 50 %) , Unit: %
- Perr_Rms: float: float Phase error RMS value. Range: 0 deg to 180 deg , Unit: deg
- Perr_Peak: float: float Phase error peak value. Range: -180 deg to 180 deg (AVERage: 0 deg to 180 deg, SDEViation: 0 deg to 90 deg) , Unit: deg
- Iq_Offset: float: float I/Q origin offset Range: -100 dB to 0 dB , Unit: dB
- Iq_Imbalance: float: float I/Q imbalance Range: -100 dB to 0 dB , Unit: dB
- Cfreq_Error: float: float Carrier frequency error Range: -5000 Hz to 5000 Hz , Unit: Hz

- Trans_Time_Err: float: float Transmit time error. Only for future use. Therefore the return value is always NCAP Range: NCAP , Unit: µs
- Ms_Power_1_M_23: float: No parameter help available
- Ms_Power_Wideband: float: No parameter help available
- Wav_Quality: float: float Waveform quality Range: 0 to 1
- Wav_Qual_Max_Pow: float: No parameter help available
- Wav_Qual_Min_Power: float: No parameter help available
- Out_Of_Tol_Count: float: float Out of tolerance result, i.e. percentage of measurement intervals of the statistic count (see [CMDLINK: CONFigure:EVDO:MEASi:MEValuation:SCOunt:MODulation CMDLINK]) exceeding the specified limits, see 'Limits (Modulation) '. Range: 0 % to 100 % , Unit: %
- Cur_Stat_Count: int: decimal Number of evaluated valid slots in this segment. Range: 0 to 1000

calculate(*segment=<Segment.Default: -1>*) → CalculateStruct

```
# SCPI: CALCulate:EVDO:MEASurement<instance>:MEValuation:LIST:SEGMent<nr>
↪:MODulation:MAXimum
value: CalculateStruct = driver.multiEval.listPy.segment.modulation.maximum.
↪calculate(segment = repcap.Segment.Default)
```

Returns modulation single value results for segment <no> in list mode. To define the statistical length for AVERage, MAXimum, MINimum calculation and enable the calculation of the results use the command method RsCmwEvdoMeas.Configure. MultiEval.ListPy.Segment.Modulation.set. The ranges indicated below apply to all results except standard deviation results. The minimum for standard deviation results equals 0. The maximum equals the width of the indicated range divided by two. Exceptions are explicitly stated. 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.

param segment optional repeated capability selector. Default value: Nr1 (settable in the interface 'Segment')

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

fetch(*segment=<Segment.Default: -1>*) → FetchStruct

```
# SCPI: FETCh:EVDO:MEASurement<instance>:MEValuation:LIST:SEGMent<nr>
↪:MODulation:MAXimum
value: FetchStruct = driver.multiEval.listPy.segment.modulation.maximum.
↪fetch(segment = repcap.Segment.Default)
```

Returns modulation single value results for segment <no> in list mode. To define the statistical length for AVERage, MAXimum, MINimum calculation and enable the calculation of the results use the command method RsCmwEvdoMeas.Configure. MultiEval.ListPy.Segment.Modulation.set. The ranges indicated below apply to all results except standard deviation results. The minimum for standard deviation results equals 0. The maximum equals the width of the indicated range divided by two. Exceptions are explicitly stated. 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.

param segment optional repeated capability selector. Default value: Nr1 (settable in the interface 'Segment')

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

7.4.7.3.1.4 Minimum

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:SEGMent<Segment>:MODulation:MINimum
CALCulate:EVDO:MEASurement<Instance>:MEValuation:LIST:SEGMent<Segment>:MODulation:MINimum
```

class Minimum

Minimum commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Seg_Reliability: int: 0 | 3 | 4 | 8 The segment reliability indicates whether one of the following exceptions occurred in this segment: 0: No error 3: Signal overflow 4: Signal low 8: Synchronization error If a combination of exceptions occurs, the most severe error is indicated.
- Evm_Rms: enums.ResultStatus2: float Error vector magnitude RMS and peak value. Range: 0 % to 100 %, Unit: %
- Evm_Peak: enums.ResultStatus2: float Error vector magnitude RMS and peak value. Range: 0 % to 100 %, Unit: %
- Merr_Rms: enums.ResultStatus2: float Magnitude error RMS value. Range: 0 % to 100 %, Unit: %
- Merr_Peak: enums.ResultStatus2: float Magnitude error peak value. Range: -100 % to +100 % (AVERage: 0% to 100 %, SDEViation: 0 % to 50 %) , Unit: %
- Perr_Rms: enums.ResultStatus2: float Phase error RMS value. Range: 0 deg to 180 deg , Unit: deg
- Perr_Peak: enums.ResultStatus2: float Phase error peak value. Range: -180 deg to 180 deg (AVERage: 0 deg to 180 deg, SDEViation: 0 deg to 90 deg) , Unit: deg
- Iq_Offset: enums.ResultStatus2: float I/Q origin offset Range: -100 dB to 0 dB , Unit: dB
- Iq_Imbalance: enums.ResultStatus2: float I/Q imbalance Range: -100 dB to 0 dB , Unit: dB
- Cfreq_Error: enums.ResultStatus2: float Carrier frequency error Range: -5000 Hz to 5000 Hz , Unit: Hz
- Trans_Time_Err: enums.ResultStatus2: float Transmit time error. Only for future use. Therefore the return value is always NCAP Range: NCAP , Unit: µs
- Ms_Power_1_M_23: enums.ResultStatus2: No parameter help available
- Ms_Power_Wideband: enums.ResultStatus2: No parameter help available
- Wav_Quality: enums.ResultStatus2: float Waveform quality Range: 0 to 1
- Wav_Qual_Max_Pow: enums.ResultStatus2: No parameter help available
- Wav_Qual_Min_Power: enums.ResultStatus2: No parameter help available
- Out_Of_Tol_Count: enums.ResultStatus2: float Out of tolerance result, i.e. percentage of measurement intervals of the statistic count (see [CMDLINK: CONFigure:EVDO:MEASi:MEValuation:SCOunt:MODulation CMDLINK]) exceeding the specified limits, see 'Limits (Modulation) '. Range: 0 % to 100 % , Unit: %
- Cur_Stat_Count: enums.ResultStatus2: decimal Number of evaluated valid slots in this segment. Range: 0 to 1000

class FetchStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Seg_Reliability: int: 0 | 3 | 4 | 8 The segment reliability indicates whether one of the following exceptions occurred in this segment: 0: No error 3: Signal overflow 4: Signal low 8: Synchronization error If a combination of exceptions occurs, the most severe error is indicated.
- Evm_Rms: float: float Error vector magnitude RMS and peak value. Range: 0 % to 100 %, Unit: %
- Evm_Peak: float: float Error vector magnitude RMS and peak value. Range: 0 % to 100 %, Unit: %
- Merr_Rms: float: float Magnitude error RMS value. Range: 0 % to 100 %, Unit: %
- Merr_Peak: float: float Magnitude error peak value. Range: -100 % to +100 % (AVERage: 0% to 100 %, SDEViation: 0 % to 50 %) , Unit: %
- Perr_Rms: float: float Phase error RMS value. Range: 0 deg to 180 deg , Unit: deg
- Perr_Peak: float: float Phase error peak value. Range: -180 deg to 180 deg (AVERage: 0 deg to 180 deg, SDEViation: 0 deg to 90 deg) , Unit: deg
- Iq_Offset: float: float I/Q origin offset Range: -100 dB to 0 dB , Unit: dB
- Iq_Imbalance: float: float I/Q imbalance Range: -100 dB to 0 dB , Unit: dB
- Cfreq_Error: float: float Carrier frequency error Range: -5000 Hz to 5000 Hz , Unit: Hz
- Trans_Time_Err: float: float Transmit time error. Only for future use. Therefore the return value is always NCAP Range: NCAP , Unit: µs
- Ms_Power_1_M_23: float: No parameter help available
- Ms_Power_Wideband: float: No parameter help available
- Wav_Quality: float: float Waveform quality Range: 0 to 1
- Wav_Qual_Max_Pow: float: No parameter help available
- Wav_Qual_Min_Power: float: No parameter help available
- Out_Of_Tol_Count: float: float Out of tolerance result, i.e. percentage of measurement intervals of the statistic count (see [CMDLINK: CONFigure:EVDO:MEASi:MEValuation:SCOunt:MODulation CMDLINK]) exceeding the specified limits, see 'Limits (Modulation) '. Range: 0 % to 100 % , Unit: %
- Cur_Stat_Count: int: decimal Number of evaluated valid slots in this segment. Range: 0 to 1000

calculate(*segment*=<*Segment.Default: -1*>) → CalculateStruct

```
# SCPI: CALCulate:EVDO:MEASurement<instance>:MEValuation:LIST:SEGMent<nr>
↪:MODulation:MINimum
value: CalculateStruct = driver.multiEval.listPy.segment.modulation.minimum.
↪calculate(segment = repcap.Segment.Default)
```

Returns modulation single value results for segment <no> in list mode. To define the statistical length for AVERage, MAXimum, MINimum calculation and enable the calculation of the results use the command method RsCmwEvdoMeas.Configure. MultiEval.ListPy.Segment.Modulation.set. The ranges indicated below apply to all results except standard deviation results. The minimum for standard deviation results equals 0. The maximum equals the width of the indicated range divided by two. Exceptions are explicitly stated. 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.

param segment optional repeated capability selector. Default value: Nr1 (settable in the interface 'Segment')

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

fetch(*segment=<Segment.Default: -1>*) → FetchStruct

```
# SCPI: FETCh:EVDO:MEASurement<instance>:MEValuation:LIST:SEGMent<nr>
↪:MODulation:MINimum
value: FetchStruct = driver.multiEval.listPy.segment.modulation.minimum.
↪fetch(segment = repcap.Segment.Default)
```

Returns modulation single value results for segment <no> in list mode. To define the statistical length for AVERage, MAXimum, MINimum calculation and enable the calculation of the results use the command method RsCmwEvdoMeas.Configure. MultiEval.ListPy.Segment.Modulation.set. The ranges indicated below apply to all results except standard deviation results. The minimum for standard deviation results equals 0. The maximum equals the width of the indicated range divided by two. Exceptions are explicitly stated. 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.

param segment optional repeated capability selector. Default value: Nr1 (settable in the interface 'Segment')

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

7.4.7.3.1.5 StandardDev

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:SEGMent<Segment>:MODulation:SDEViation
CALCulate:EVDO:MEASurement<Instance>:MEValuation:LIST:SEGMent<Segment>
↪:MODulation:SDEViation
```

class StandardDev

StandardDev commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Seg_Reliability: int: 0 | 3 | 4 | 8 The segment reliability indicates whether one of the following exceptions occurred in this segment: 0: No error 3: Signal overflow 4: Signal low 8: Synchronization error If a combination of exceptions occurs, the most severe error is indicated.
- Evm_Rms: enums.ResultStatus2: float Error vector magnitude RMS and peak value. Range: 0 % to 100 %, Unit: %
- Evm_Peak: enums.ResultStatus2: float Error vector magnitude RMS and peak value. Range: 0 % to 100 %, Unit: %
- Merr_Rms: enums.ResultStatus2: float Magnitude error RMS value. Range: 0 % to 100 %, Unit: %
- Merr_Peak: enums.ResultStatus2: float Magnitude error peak value. Range: -100 % to +100 % (AVERage: 0% to 100 %, SDEViation: 0 % to 50 %) , Unit: %
- Perr_Rms: enums.ResultStatus2: float Phase error RMS value. Range: 0 deg to 180 deg , Unit: deg

- Perr_Peak: enums.ResultStatus2: float Phase error peak value. Range: -180 deg to 180 deg (AVERage: 0 deg to 180 deg, SDEViation: 0 deg to 90 deg) , Unit: deg
- Iq_Offset: enums.ResultStatus2: float I/Q origin offset Range: -100 dB to 0 dB , Unit: dB
- Iq_Imbalance: enums.ResultStatus2: float I/Q imbalance Range: -100 dB to 0 dB , Unit: dB
- Cfreq_Error: enums.ResultStatus2: float Carrier frequency error Range: -5000 Hz to 5000 Hz , Unit: Hz
- Trans_Time_Err: enums.ResultStatus2: float Transmit time error. Only for future use. Therefore the return value is always NCAP Range: NCAP , Unit: µs
- Ms_Power_1_M_23: enums.ResultStatus2: No parameter help available
- Ms_Power_Wideband: enums.ResultStatus2: No parameter help available
- Wav_Quality: enums.ResultStatus2: float Waveform quality Range: 0 to 1
- Wav_Qual_Max_Pow: enums.ResultStatus2: No parameter help available
- Wav_Qual_Min_Power: enums.ResultStatus2: No parameter help available
- Out_Of_Tol_Count: enums.ResultStatus2: float Out of tolerance result, i.e. percentage of measurement intervals of the statistic count (see [CMDLINK: CONFigure:EVDO:MEASi:MEValuation:SCOunt:MODulation CMDLINK]) exceeding the specified limits, see 'Limits (Modulation) '. Range: 0 % to 100 % , Unit: %
- Cur_Stat_Count: enums.ResultStatus2: decimal Number of evaluated valid slots in this segment. Range: 0 to 1000

class FetchStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Seg_Reliability: int: 0 | 3 | 4 | 8 The segment reliability indicates whether one of the following exceptions occurred in this segment: 0: No error 3: Signal overflow 4: Signal low 8: Synchronization error If a combination of exceptions occurs, the most severe error is indicated.
- Evm_Rms: float: float Error vector magnitude RMS and peak value. Range: 0 % to 100 %, Unit: %
- Evm_Peak: float: float Error vector magnitude RMS and peak value. Range: 0 % to 100 %, Unit: %
- Merr_Rms: float: float Magnitude error RMS value. Range: 0 % to 100 %, Unit: %
- Merr_Peak: float: float Magnitude error peak value. Range: -100 % to +100 % (AVERage: 0% to 100 %, SDEViation: 0 % to 50 %) , Unit: %
- Perr_Rms: float: float Phase error RMS value. Range: 0 deg to 180 deg , Unit: deg
- Perr_Peak: float: float Phase error peak value. Range: -180 deg to 180 deg (AVERage: 0 deg to 180 deg, SDEViation: 0 deg to 90 deg) , Unit: deg
- Iq_Offset: float: float I/Q origin offset Range: -100 dB to 0 dB , Unit: dB
- Iq_Imbalance: float: float I/Q imbalance Range: -100 dB to 0 dB , Unit: dB
- Cfreq_Error: float: float Carrier frequency error Range: -5000 Hz to 5000 Hz , Unit: Hz
- Trans_Time_Err: float: float Transmit time error. Only for future use. Therefore the return value is always NCAP Range: NCAP , Unit: µs
- Ms_Power_1_M_23: float: No parameter help available
- Ms_Power_Wideband: float: No parameter help available

- Wav_Quality: float: float Waveform quality Range: 0 to 1
- Wav_Qual_Max_Pow: float: No parameter help available
- Wav_Qual_Min_Power: float: No parameter help available
- Out_Of_Tol_Count: float: float Out of tolerance result, i.e. percentage of measurement intervals of the statistic count (see [CMDLINK: CONFigure:EVDO:MEASi:MEValuation:SCOunt:MODulation CMDLINK]) exceeding the specified limits, see 'Limits (Modulation) '. Range: 0 % to 100 % , Unit: %
- Cur_Stat_Count: int: decimal Number of evaluated valid slots in this segment. Range: 0 to 1000

calculate(*segment*=<*Segment.Default: -1*>) → CalculateStruct

```
# SCPI: CALCulate:EVDO:MEASurement<instance>:MEValuation:LIST:SEGMent<nr>
↪:MODulation:SDEViation
value: CalculateStruct = driver.multiEval.listPy.segment.modulation.standardDev.
↪calculate(segment = repcap.Segment.Default)
```

Returns modulation single value results for segment <no> in list mode. To define the statistical length for AVERage, MAXimum, MINimum calculation and enable the calculation of the results use the command method RsCmwEvdoMeas.Configure. MultiEval.ListPy.Segment.Modulation.set. The ranges indicated below apply to all results except standard deviation results. The minimum for standard deviation results equals 0. The maximum equals the width of the indicated range divided by two. Exceptions are explicitly stated. 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.

param segment optional repeated capability selector. Default value: Nr1 (settable in the interface 'Segment')

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

fetch(*segment*=<*Segment.Default: -1*>) → FetchStruct

```
# SCPI: FETCh:EVDO:MEASurement<instance>:MEValuation:LIST:SEGMent<nr>
↪:MODulation:SDEViation
value: FetchStruct = driver.multiEval.listPy.segment.modulation.standardDev.
↪fetch(segment = repcap.Segment.Default)
```

Returns modulation single value results for segment <no> in list mode. To define the statistical length for AVERage, MAXimum, MINimum calculation and enable the calculation of the results use the command method RsCmwEvdoMeas.Configure. MultiEval.ListPy.Segment.Modulation.set. The ranges indicated below apply to all results except standard deviation results. The minimum for standard deviation results equals 0. The maximum equals the width of the indicated range divided by two. Exceptions are explicitly stated. 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.

param segment optional repeated capability selector. Default value: Nr1 (settable in the interface 'Segment')

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

7.4.7.3.2 Cp

class Cp

Cp commands group definition. 9 total commands, 5 Sub-groups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.multiEval.listPy.segment.cp.clone()
```

Subgroups

7.4.7.3.2.1 Current

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:SEGMent<Segment>:CP:CURRent
CALCulate:EVDO:MEASurement<Instance>:MEValuation:LIST:SEGMent<Segment>:CP:CURRent
```

class Current

Current commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

class CalculateStruct

Response structure. Fields:

- Reliabiltiy: int: decimal 'Conventions and General Information'. In list mode, a zero reliability indicator indicates that the results in all measured segments are valid. A non-zero value indicates that an error occurred in at least one of the measured segments.
- Seg_Reliability: int: 0 | 3 | 4 | 8 The segment reliability indicates whether one of the following exceptions occurred in this segment: 0: No error 3: Signal overflow 4: Signal low 8: Synchronization error If a combination of exceptions occurs, the most severe error is indicated.
- Rri: enums.ResultStatus2: float RMS channel power values for the physical channels. Depending on the subtype 0/1 or 2, some channels exist or not. If not available, no power measurement is possible therefore NAV is returned. Range: -60 dB to +10 dB , Unit: dB
- Pilot: enums.ResultStatus2: float RMS channel power values for the physical channels. Depending on the subtype 0/1 or 2, some channels exist or not. If not available, no power measurement is possible therefore NAV is returned. Range: -60 dB to +10 dB , Unit: dB
- Ack_Dsc: enums.ResultStatus2: float RMS channel power values for the physical channels. Depending on the subtype 0/1 or 2, some channels exist or not. If not available, no power measurement is possible therefore NAV is returned. Range: -60 dB to +10 dB , Unit: dB
- Aux_Pilot: enums.ResultStatus2: float RMS channel power values for the physical channels. Depending on the subtype 0/1 or 2, some channels exist or not. If not available, no power measurement is possible therefore NAV is returned. Range: -60 dB to +10 dB , Unit: dB
- Drc: enums.ResultStatus2: float RMS channel power values for the physical channels. Depending on the subtype 0/1 or 2, some channels exist or not. If not available, no power measurement is possible therefore NAV is returned. Range: -60 dB to +10 dB , Unit: dB

- Data: enums.ResultStatus2: float RMS channel power values for the physical channels. Depending on the subtype 0/1 or 2, some channels exist or not. If not available, no power measurement is possible therefore NAV is returned. Range: -60 dB to +10 dB , Unit: dB

class FetchStruct

Response structure. Fields:

- Reliabiltiy: int: decimal 'Conventions and General Information'. In list mode, a zero reliability indicator indicates that the results in all measured segments are valid. A non-zero value indicates that an error occurred in at least one of the measured segments.
- Seg_Reliability: int: 0 | 3 | 4 | 8 The segment reliability indicates whether one of the following exceptions occurred in this segment: 0: No error 3: Signal overflow 4: Signal low 8: Synchronization error If a combination of exceptions occurs, the most severe error is indicated.
- Rri: float: float RMS channel power values for the physical channels. Depending on the subtype 0/1 or 2, some channels exist or not. If not available, no power measurement is possible therefore NAV is returned. Range: -60 dB to +10 dB , Unit: dB
- Pilot: float: float RMS channel power values for the physical channels. Depending on the subtype 0/1 or 2, some channels exist or not. If not available, no power measurement is possible therefore NAV is returned. Range: -60 dB to +10 dB , Unit: dB
- Ack_Dsc: float: float RMS channel power values for the physical channels. Depending on the subtype 0/1 or 2, some channels exist or not. If not available, no power measurement is possible therefore NAV is returned. Range: -60 dB to +10 dB , Unit: dB
- Aux_Pilot: float: float RMS channel power values for the physical channels. Depending on the subtype 0/1 or 2, some channels exist or not. If not available, no power measurement is possible therefore NAV is returned. Range: -60 dB to +10 dB , Unit: dB
- Drc: float: float RMS channel power values for the physical channels. Depending on the subtype 0/1 or 2, some channels exist or not. If not available, no power measurement is possible therefore NAV is returned. Range: -60 dB to +10 dB , Unit: dB
- Data: float: float RMS channel power values for the physical channels. Depending on the subtype 0/1 or 2, some channels exist or not. If not available, no power measurement is possible therefore NAV is returned. Range: -60 dB to +10 dB , Unit: dB

calculate(*segment=<Segment.Default: -1>*) → CalculateStruct

```
# SCPI: CALCulate:EVDO:MEASurement<instance>:MEValuation:LIST:SEGMent<nr>
↪:CP:CURRent
value: CalculateStruct = driver.multiEval.listPy.segment.cp.current.
↪calculate(segment = repcap.Segment.Default)
```

Returns channel power (CP) results for the segment <no> in list mode. To define the statistical length for AVERage, MAXimum, MINimum calculation and enable the calculation of the results use the command method RsCmwEvdoMeas.Configure. MultiEval.ListPy.Segment.Modulation.set. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

param segment optional repeated capability selector. Default value: Nr1 (settable in the interface 'Segment')

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

fetch(*segment=<Segment.Default: -1>*) → FetchStruct

```
# SCPI: FETCh:EVDO:MEASurement<instance>:MEValuation:LIST:SEGMent<nr>:CP:CURRent
value: FetchStruct = driver.multiEval.listPy.segment.cp.current.fetch(segment =␣
↪repcap.Segment.Default)
```

Returns channel power (CP) results for the segment <no> in list mode. To define the statistical length for AVERage, MAXimum, MINimum calculation and enable the calculation of the results use the command method RsCmwEvdoMeas.Configure. MultiEval.ListPy.Segment.Modulation.set. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

param segment optional repeated capability selector. Default value: Nr1 (settable in the interface 'Segment')

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

7.4.7.3.2.2 Average

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:SEGMent<Segment>:CP:AVERage
CALCulate:EVDO:MEASurement<Instance>:MEValuation:LIST:SEGMent<Segment>:CP:AVERage
```

class Average

Average commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Seg_Reliability: int: 0 | 3 | 4 | 8 The segment reliability indicates whether one of the following exceptions occurred in this segment: 0: No error 3: Signal overflow 4: Signal low 8: Synchronization error If a combination of exceptions occurs, the most severe error is indicated.
- Rri: enums.ResultStatus2: float RMS channel power values for the physical channels. Depending on the subtype 0/1 or 2, some channels exist or not. If not available, no power measurement is possible therefore NAV is returned. Range: -60 dB to +10 dB , Unit: dB
- Pilot: enums.ResultStatus2: float RMS channel power values for the physical channels. Depending on the subtype 0/1 or 2, some channels exist or not. If not available, no power measurement is possible therefore NAV is returned. Range: -60 dB to +10 dB , Unit: dB
- Ack_Dsc: enums.ResultStatus2: float RMS channel power values for the physical channels. Depending on the subtype 0/1 or 2, some channels exist or not. If not available, no power measurement is possible therefore NAV is returned. Range: -60 dB to +10 dB , Unit: dB
- Aux_Pilot: enums.ResultStatus2: float RMS channel power values for the physical channels. Depending on the subtype 0/1 or 2, some channels exist or not. If not available, no power measurement is possible therefore NAV is returned. Range: -60 dB to +10 dB , Unit: dB
- Drc: enums.ResultStatus2: float RMS channel power values for the physical channels. Depending on the subtype 0/1 or 2, some channels exist or not. If not available, no power measurement is possible therefore NAV is returned. Range: -60 dB to +10 dB , Unit: dB
- Data: enums.ResultStatus2: float RMS channel power values for the physical channels. Depending on the subtype 0/1 or 2, some channels exist or not. If not available, no power measurement is possible therefore NAV is returned. Range: -60 dB to +10 dB , Unit: dB

class FetchStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Seg_Reliability: int: 0 | 3 | 4 | 8 The segment reliability indicates whether one of the following exceptions occurred in this segment: 0: No error 3: Signal overflow 4: Signal low 8: Synchronization error If a combination of exceptions occurs, the most severe error is indicated.
- Rri: float: float RMS channel power values for the physical channels. Depending on the subtype 0/1 or 2, some channels exist or not. If not available, no power measurement is possible therefore NAV is returned. Range: -60 dB to +10 dB , Unit: dB
- Pilot: float: float RMS channel power values for the physical channels. Depending on the subtype 0/1 or 2, some channels exist or not. If not available, no power measurement is possible therefore NAV is returned. Range: -60 dB to +10 dB , Unit: dB
- Ack_Dsc: float: float RMS channel power values for the physical channels. Depending on the subtype 0/1 or 2, some channels exist or not. If not available, no power measurement is possible therefore NAV is returned. Range: -60 dB to +10 dB , Unit: dB
- Aux_Pilot: float: float RMS channel power values for the physical channels. Depending on the subtype 0/1 or 2, some channels exist or not. If not available, no power measurement is possible therefore NAV is returned. Range: -60 dB to +10 dB , Unit: dB
- Drc: float: float RMS channel power values for the physical channels. Depending on the subtype 0/1 or 2, some channels exist or not. If not available, no power measurement is possible therefore NAV is returned. Range: -60 dB to +10 dB , Unit: dB
- Data: float: float RMS channel power values for the physical channels. Depending on the subtype 0/1 or 2, some channels exist or not. If not available, no power measurement is possible therefore NAV is returned. Range: -60 dB to +10 dB , Unit: dB

calculate(*segment=<Segment.Default: -1>*) → CalculateStruct

```
# SCPI: CALCulate:EVDO:MEASurement<instance>:MEValuation:LIST:SEGMent<nr>
↪:CP:AVERage
value: CalculateStruct = driver.multiEval.listPy.segment.cp.average.
↪calculate(segment = repcap.Segment.Default)
```

Returns channel power (CP) results for the segment <no> in list mode. To define the statistical length for AVERage, MAXimum, MINimum calculation and enable the calculation of the results use the command method RsCmwEvdoMeas.Configure. MultiEval.ListPy.Segment.Modulation.set. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

param segment optional repeated capability selector. Default value: Nr1 (settable in the interface 'Segment')

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

fetch(*segment=<Segment.Default: -1>*) → FetchStruct

```
# SCPI: FETCh:EVDO:MEASurement<instance>:MEValuation:LIST:SEGMent<nr>:CP:AVERage
value: FetchStruct = driver.multiEval.listPy.segment.cp.average.fetch(segment =␣
↪repcap.Segment.Default)
```

Returns channel power (CP) results for the segment <no> in list mode. To define the statistical length for AVERage, MAXimum, MINimum calculation and enable the calculation of the results use the command method RsCmwEvdoMeas.Configure. MultiEval.ListPy.Segment.Modulation.set. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

param segment optional repeated capability selector. Default value: Nr1 (settable in the interface 'Segment')

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

7.4.7.3.2.3 Maximum

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:SEGMent<Segment>:CP:MAXimum
CALCulate:EVDO:MEASurement<Instance>:MEValuation:LIST:SEGMent<Segment>:CP:MAXimum
```

class Maximum

Maximum commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Seg_Reliability: int: 0 | 3 | 4 | 8 The segment reliability indicates whether one of the following exceptions occurred in this segment: 0: No error 3: Signal overflow 4: Signal low 8: Synchronization error If a combination of exceptions occurs, the most severe error is indicated.
- Rri: enums.ResultStatus2: float RMS channel power values for the physical channels. Depending on the subtype 0/1 or 2, some channels exist or not. If not available, no power measurement is possible therefore NAV is returned. Range: -60 dB to +10 dB , Unit: dB
- Pilot: enums.ResultStatus2: float RMS channel power values for the physical channels. Depending on the subtype 0/1 or 2, some channels exist or not. If not available, no power measurement is possible therefore NAV is returned. Range: -60 dB to +10 dB , Unit: dB
- Ack_Dsc: enums.ResultStatus2: float RMS channel power values for the physical channels. Depending on the subtype 0/1 or 2, some channels exist or not. If not available, no power measurement is possible therefore NAV is returned. Range: -60 dB to +10 dB , Unit: dB
- Aux_Pilot: enums.ResultStatus2: float RMS channel power values for the physical channels. Depending on the subtype 0/1 or 2, some channels exist or not. If not available, no power measurement is possible therefore NAV is returned. Range: -60 dB to +10 dB , Unit: dB
- Drc: enums.ResultStatus2: float RMS channel power values for the physical channels. Depending on the subtype 0/1 or 2, some channels exist or not. If not available, no power measurement is possible therefore NAV is returned. Range: -60 dB to +10 dB , Unit: dB
- Data: enums.ResultStatus2: float RMS channel power values for the physical channels. Depending on the subtype 0/1 or 2, some channels exist or not. If not available, no power measurement is possible therefore NAV is returned. Range: -60 dB to +10 dB , Unit: dB

class FetchStruct

Response structure. Fields:

- Reliability: int: No parameter help available

- Seg_Reliability: int: 0 | 3 | 4 | 8 The segment reliability indicates whether one of the following exceptions occurred in this segment: 0: No error 3: Signal overflow 4: Signal low 8: Synchronization error If a combination of exceptions occurs, the most severe error is indicated.
- Rri: float: float RMS channel power values for the physical channels. Depending on the subtype 0/1 or 2, some channels exist or not. If not available, no power measurement is possible therefore NAV is returned. Range: -60 dB to +10 dB , Unit: dB
- Pilot: float: float RMS channel power values for the physical channels. Depending on the subtype 0/1 or 2, some channels exist or not. If not available, no power measurement is possible therefore NAV is returned. Range: -60 dB to +10 dB , Unit: dB
- Ack_Dsc: float: float RMS channel power values for the physical channels. Depending on the subtype 0/1 or 2, some channels exist or not. If not available, no power measurement is possible therefore NAV is returned. Range: -60 dB to +10 dB , Unit: dB
- Aux_Pilot: float: float RMS channel power values for the physical channels. Depending on the subtype 0/1 or 2, some channels exist or not. If not available, no power measurement is possible therefore NAV is returned. Range: -60 dB to +10 dB , Unit: dB
- Drc: float: float RMS channel power values for the physical channels. Depending on the subtype 0/1 or 2, some channels exist or not. If not available, no power measurement is possible therefore NAV is returned. Range: -60 dB to +10 dB , Unit: dB
- Data: float: float RMS channel power values for the physical channels. Depending on the subtype 0/1 or 2, some channels exist or not. If not available, no power measurement is possible therefore NAV is returned. Range: -60 dB to +10 dB , Unit: dB

calculate(*segment=<Segment.Default: -1>*) → CalculateStruct

```
# SCPI: CALCulate:EVDO:MEASurement<instance>:MEValuation:LIST:SEGMent<nr>
↪:CP:MAXimum
value: CalculateStruct = driver.multiEval.listPy.segment.cp.maximum.
↪calculate(segment = repcap.Segment.Default)
```

Returns channel power (CP) results for the segment <no> in list mode. To define the statistical length for AVERage, MAXimum, MINimum calculation and enable the calculation of the results use the command method RsCmwEvdoMeas.Configure. MultiEval.ListPy.Segment.Modulation.set. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

param segment optional repeated capability selector. Default value: Nr1 (settable in the interface 'Segment')

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

fetch(*segment=<Segment.Default: -1>*) → FetchStruct

```
# SCPI: FETCh:EVDO:MEASurement<instance>:MEValuation:LIST:SEGMent<nr>:CP:MAXimum
value: FetchStruct = driver.multiEval.listPy.segment.cp.maximum.fetch(segment =␣
↪repcap.Segment.Default)
```

Returns channel power (CP) results for the segment <no> in list mode. To define the statistical length for AVERage, MAXimum, MINimum calculation and enable the calculation of the results use the command method RsCmwEvdoMeas.Configure. MultiEval.ListPy.Segment.Modulation.set. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

param segment optional repeated capability selector. Default value: Nr1 (settable in the interface 'Segment')

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

7.4.7.3.2.4 Minimum

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:SEGMent<Segment>:CP:MINimum
CALCulate:EVDO:MEASurement<Instance>:MEValuation:LIST:SEGMent<Segment>:CP:MINimum
```

class Minimum

Minimum commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Seg_Reliability: int: 0 | 3 | 4 | 8 The segment reliability indicates whether one of the following exceptions occurred in this segment: 0: No error 3: Signal overflow 4: Signal low 8: Synchronization error If a combination of exceptions occurs, the most severe error is indicated.
- Rri: enums.ResultStatus2: float RMS channel power values for the physical channels. Depending on the subtype 0/1 or 2, some channels exist or not. If not available, no power measurement is possible therefore NAV is returned. Range: -60 dB to +10 dB , Unit: dB
- Pilot: enums.ResultStatus2: float RMS channel power values for the physical channels. Depending on the subtype 0/1 or 2, some channels exist or not. If not available, no power measurement is possible therefore NAV is returned. Range: -60 dB to +10 dB , Unit: dB
- Ack_Dsc: enums.ResultStatus2: float RMS channel power values for the physical channels. Depending on the subtype 0/1 or 2, some channels exist or not. If not available, no power measurement is possible therefore NAV is returned. Range: -60 dB to +10 dB , Unit: dB
- Aux_Pilot: enums.ResultStatus2: float RMS channel power values for the physical channels. Depending on the subtype 0/1 or 2, some channels exist or not. If not available, no power measurement is possible therefore NAV is returned. Range: -60 dB to +10 dB , Unit: dB
- Drc: enums.ResultStatus2: float RMS channel power values for the physical channels. Depending on the subtype 0/1 or 2, some channels exist or not. If not available, no power measurement is possible therefore NAV is returned. Range: -60 dB to +10 dB , Unit: dB
- Data: enums.ResultStatus2: float RMS channel power values for the physical channels. Depending on the subtype 0/1 or 2, some channels exist or not. If not available, no power measurement is possible therefore NAV is returned. Range: -60 dB to +10 dB , Unit: dB

class FetchStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Seg_Reliability: int: 0 | 3 | 4 | 8 The segment reliability indicates whether one of the following exceptions occurred in this segment: 0: No error 3: Signal overflow 4: Signal low 8: Synchronization error If a combination of exceptions occurs, the most severe error is indicated.
- Rri: float: float RMS channel power values for the physical channels. Depending on the subtype 0/1 or 2, some channels exist or not. If not available, no power measurement is possible therefore NAV is returned. Range: -60 dB to +10 dB , Unit: dB

- Pilot: float: float RMS channel power values for the physical channels. Depending on the subtype 0/1 or 2, some channels exist or not. If not available, no power measurement is possible therefore NAV is returned. Range: -60 dB to +10 dB , Unit: dB
- Ack_Dsc: float: float RMS channel power values for the physical channels. Depending on the subtype 0/1 or 2, some channels exist or not. If not available, no power measurement is possible therefore NAV is returned. Range: -60 dB to +10 dB , Unit: dB
- Aux_Pilot: float: float RMS channel power values for the physical channels. Depending on the subtype 0/1 or 2, some channels exist or not. If not available, no power measurement is possible therefore NAV is returned. Range: -60 dB to +10 dB , Unit: dB
- Drc: float: float RMS channel power values for the physical channels. Depending on the subtype 0/1 or 2, some channels exist or not. If not available, no power measurement is possible therefore NAV is returned. Range: -60 dB to +10 dB , Unit: dB
- Data: float: float RMS channel power values for the physical channels. Depending on the subtype 0/1 or 2, some channels exist or not. If not available, no power measurement is possible therefore NAV is returned. Range: -60 dB to +10 dB , Unit: dB

calculate(*segment=<Segment.Default: -1>*) → CalculateStruct

```
# SCPI: CALCulate:EVDO:MEASurement<instance>:MEValuation:LIST:SEGMent<nr>
↪:CP:MINimum
value: CalculateStruct = driver.multiEval.listPy.segment.cp.minimum.
↪calculate(segment = repcap.Segment.Default)
```

Returns channel power (CP) results for the segment <no> in list mode. To define the statistical length for AVERage, MAXimum, MINimum calculation and enable the calculation of the results use the command method RsCmwEvdoMeas.Configure. MultiEval.ListPy.Segment.Modulation.set. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

param segment optional repeated capability selector. Default value: Nr1 (settable in the interface 'Segment')

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

fetch(*segment=<Segment.Default: -1>*) → FetchStruct

```
# SCPI: FETCh:EVDO:MEASurement<instance>:MEValuation:LIST:SEGMent<nr>:CP:MINimum
value: FetchStruct = driver.multiEval.listPy.segment.cp.minimum.fetch(segment =␣
↪repcap.Segment.Default)
```

Returns channel power (CP) results for the segment <no> in list mode. To define the statistical length for AVERage, MAXimum, MINimum calculation and enable the calculation of the results use the command method RsCmwEvdoMeas.Configure. MultiEval.ListPy.Segment.Modulation.set. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

param segment optional repeated capability selector. Default value: Nr1 (settable in the interface 'Segment')

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

7.4.7.3.2.5 State

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:SEGMent<Segment>:CP:STATe
```

class State

State commands group definition. 1 total commands, 0 Sub-groups, 1 group commands

class FetchStruct

Response structure. Fields:

- Reliabiltiy: int: decimal 'Conventions and General Information'. In list mode, a zero reliability indicator indicates that the results in all measured segments are valid. A non-zero value indicates that an error occurred in at least one of the measured segments.
- Seg_Reliability: int: 0 | 3 | 4 | 8 The segment reliability indicates whether one of the following exceptions occurred in this segment: 0: No error 3: Signal overflow 4: Signal low 8: Synchronization error If a combination of exceptions occurs, the most severe error is indicated.
- Rri: enums.ResultStateA: INVisible | ACTive | IACTive 'INV': No channel available 'ACTive': Active channel 'IACtive': Inactive channel
- Pilot: enums.ResultStateA: INVisible | ACTive | IACTive 'INV': No channel available 'ACTive': Active channel 'IACtive': Inactive channel
- Ack_Dsc: enums.ResultStateA: INVisible | ACTive | IACTive 'INV': No channel available 'ACTive': Active channel 'IACtive': Inactive channel
- Aux_Pilot: enums.ResultStateA: INVisible | ACTive | IACTive 'INV': No channel available 'ACTive': Active channel 'IACtive': Inactive channel
- Drc: enums.ResultStateA: INVisible | ACTive | IACTive 'INV': No channel available 'ACTive': Active channel 'IACtive': Inactive channel
- Data: enums.ResultStateA: INVisible | ACTive | IACTive 'INV': No channel available 'ACTive': Active channel 'IACtive': Inactive channel

fetch(*segment=<Segment.Default: -1>*) → FetchStruct

```
# SCPI: FETCh:EVDO:MEASurement<instance>:MEValuation:LIST:SEGMent<nr>:CP:STATe
value: FetchStruct = driver.multiEval.listPy.segment.cp.state.fetch(segment =␣
↪repcap.Segment.Default)
```

Return the states of the channels for power measurement (CP) .

param segment optional repeated capability selector. Default value: Nr1 (settable in the interface 'Segment')

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

7.4.7.3.3 Dwcp

class Dwcp
Dwcp commands group definition. 8 total commands, 4 Sub-groups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.multiEval.listPy.segment.dwcp.clone()
```

Subgroups

7.4.7.3.3.1 Current

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:SEGMent<Segment>:DWCP:CURRent
CALCulate:EVDO:MEASurement<Instance>:MEValuation:LIST:SEGMent<Segment>:DWCP:CURRent
```

class Current
Current commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

class CalculateStruct
Response structure. Fields:

- Reliabiltiy: int: decimal 'Conventions and General Information'. In list mode, a zero reliability indicator indicates that the results in all measured segments are valid. A non-zero value indicates that an error occurred in at least one of the measured segments.
- Seg_Reliability: int: 0 | 3 | 4 | 8 The segment reliability indicates whether one of the following exceptions occurred in this segment: 0: No error 3: Signal overflow 4: Signal low 8: Synchronization error If a combination of exceptions occurs, the most severe error is indicated.
- W_24_I: enums.ResultStatus2: No parameter help available
- W_24_Q: enums.ResultStatus2: No parameter help available
- W_12_I: enums.ResultStatus2: No parameter help available
- W_12_Q: enums.ResultStatus2: No parameter help available

class FetchStruct
Response structure. Fields:

- Reliabiltiy: int: decimal 'Conventions and General Information'. In list mode, a zero reliability indicator indicates that the results in all measured segments are valid. A non-zero value indicates that an error occurred in at least one of the measured segments.
- Seg_Reliability: int: 0 | 3 | 4 | 8 The segment reliability indicates whether one of the following exceptions occurred in this segment: 0: No error 3: Signal overflow 4: Signal low 8: Synchronization error If a combination of exceptions occurs, the most severe error is indicated.
- W_24_I: float: No parameter help available
- W_24_Q: float: No parameter help available
- W_12_I: float: No parameter help available

- W_12_Q: float: No parameter help available

calculate(*segment*=<*Segment.Default: -1*>) → CalculateStruct

```
# SCPI: CALCulate:EVDO:MEASurement<instance>:MEValuation:LIST:SEGMent<nr>
↪:DWCP:CURRent
value: CalculateStruct = driver.multiEval.listPy.segment.dwcp.current.
↪calculate(segment = repcap.Segment.Default)
```

Returns the scalar channel power for the data channel results for the segment <no> in list mode. The result is extended to the W24 and W12 I/Q values of the data channel. Only available if subtype 2 or 3 is selected. Otherwise NAV is returned in list mode. To define the statistical length for AVERage, MAXimum, MINimum calculation and to enable the calculation of the results, use the command method RsCmwEvdoMeas.Configure.MultiEval.ListPy.Segment.Modulation.set. 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.

param segment optional repeated capability selector. Default value: Nr1 (settable in the interface 'Segment')

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

fetch(*segment*=<*Segment.Default: -1*>) → FetchStruct

```
# SCPI: FETCh:EVDO:MEASurement<instance>:MEValuation:LIST:SEGMent<nr>
↪:DWCP:CURRent
value: FetchStruct = driver.multiEval.listPy.segment.dwcp.current.fetch(segment␣
↪= repcap.Segment.Default)
```

Returns the scalar channel power for the data channel results for the segment <no> in list mode. The result is extended to the W24 and W12 I/Q values of the data channel. Only available if subtype 2 or 3 is selected. Otherwise NAV is returned in list mode. To define the statistical length for AVERage, MAXimum, MINimum calculation and to enable the calculation of the results, use the command method RsCmwEvdoMeas.Configure.MultiEval.ListPy.Segment.Modulation.set. 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.

param segment optional repeated capability selector. Default value: Nr1 (settable in the interface 'Segment')

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

7.4.7.3.3.2 Average

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:SEGMent<Segment>:DWCP:AVERage
CALCulate:EVDO:MEASurement<Instance>:MEValuation:LIST:SEGMent<Segment>:DWCP:AVERage
```

class Average

Average commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

class CalculateStruct

Response structure. Fields:

- Reliabiltiy: int: decimal 'Conventions and General Information'. In list mode, a zero reliability indicator indicates that the results in all measured segments are valid. A non-zero value indicates that an error occurred in at least one of the measured segments.
- Seg_Reliability: int: 0 | 3 | 4 | 8 The segment reliability indicates whether one of the following exceptions occurred in this segment: 0: No error 3: Signal overflow 4: Signal low 8: Synchronization error If a combination of exceptions occurs, the most severe error is indicated.
- W_24_I: enums.ResultStatus2: No parameter help available
- W_24_Q: enums.ResultStatus2: No parameter help available
- W_12_I: enums.ResultStatus2: No parameter help available
- W_12_Q: enums.ResultStatus2: No parameter help available

class FetchStruct

Response structure. Fields:

- Reliabiltiy: int: decimal 'Conventions and General Information'. In list mode, a zero reliability indicator indicates that the results in all measured segments are valid. A non-zero value indicates that an error occurred in at least one of the measured segments.
- Seg_Reliability: int: 0 | 3 | 4 | 8 The segment reliability indicates whether one of the following exceptions occurred in this segment: 0: No error 3: Signal overflow 4: Signal low 8: Synchronization error If a combination of exceptions occurs, the most severe error is indicated.
- W_24_I: float: No parameter help available
- W_24_Q: float: No parameter help available
- W_12_I: float: No parameter help available
- W_12_Q: float: No parameter help available

calculate(*segment=<Segment.Default: -1>*) → CalculateStruct

```
# SCPI: CALCulate:EVDO:MEASurement<instance>:MEValuation:LIST:SEGMent<nr>
↪:DWCP:AVERage
value: CalculateStruct = driver.multiEval.listPy.segment.dwcp.average.
↪calculate(segment = repcap.Segment.Default)
```

Returns the scalar channel power for the data channel results for the segment <no> in list mode. The result is extended to the W24 and W12 I/Q values of the data channel. Only available if subtype 2 or 3 is selected. Otherwise NAV is returned in list mode. To define the statistical length for AVERage, MAXimum, MINimum calculation and to enable the calculation of the results, use the command method RsCmwEvdoMeas.Configure.MultiEval.ListPy.Segment.Modulation.set. 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.

param segment optional repeated capability selector. Default value: Nr1 (settable in the interface 'Segment')

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

fetch(*segment=<Segment.Default: -1>*) → FetchStruct

```
# SCPI: FETCh:EVDO:MEASurement<instance>:MEValuation:LIST:SEGMent<nr>
↪:DWCP:AVERage
value: FetchStruct = driver.multiEval.listPy.segment.dwcp.average.fetch(segment␣
↪= repcap.Segment.Default)
```

(continues on next page)

(continued from previous page)

Returns the scalar channel power for the data channel results for the segment <no> in list mode. The result is extended to the W24 and W12 I/Q values of the data channel. Only available if subtype 2 or 3 is selected. Otherwise NAV is returned in list mode. To define the statistical length for AVERage, MAXimum, MINimum calculation and to enable the calculation of the results, use the command method RsCmwEvdoMeas.Configure.MultiEval.ListPy.Segment.Modulation.set. 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.

param segment optional repeated capability selector. Default value: Nr1 (settable in the interface 'Segment')

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

7.4.7.3.3.3 Maximum

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:SEGMent<Segment>:DWCP:MAXimum
CALCulate:EVDO:MEASurement<Instance>:MEValuation:LIST:SEGMent<Segment>:DWCP:MAXimum
```

class Maximum

Maximum commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

class CalculateStruct

Response structure. Fields:

- Reliabiltiy: int: decimal 'Conventions and General Information'. In list mode, a zero reliability indicator indicates that the results in all measured segments are valid. A non-zero value indicates that an error occurred in at least one of the measured segments.
- Seg_Reliability: int: 0 | 3 | 4 | 8 The segment reliability indicates whether one of the following exceptions occurred in this segment: 0: No error 3: Signal overflow 4: Signal low 8: Synchronization error If a combination of exceptions occurs, the most severe error is indicated.
- W_24_I: enums.ResultStatus2: No parameter help available
- W_24_Q: enums.ResultStatus2: No parameter help available
- W_12_I: enums.ResultStatus2: No parameter help available
- W_12_Q: enums.ResultStatus2: No parameter help available

class FetchStruct

Response structure. Fields:

- Reliabiltiy: int: decimal 'Conventions and General Information'. In list mode, a zero reliability indicator indicates that the results in all measured segments are valid. A non-zero value indicates that an error occurred in at least one of the measured segments.
- Seg_Reliability: int: 0 | 3 | 4 | 8 The segment reliability indicates whether one of the following exceptions occurred in this segment: 0: No error 3: Signal overflow 4: Signal low 8: Synchronization error If a combination of exceptions occurs, the most severe error is indicated.
- W_24_I: float: No parameter help available
- W_24_Q: float: No parameter help available

- W_12_I: float: No parameter help available
- W_12_Q: float: No parameter help available

calculate(*segment*=<*Segment.Default: -1*>) → CalculateStruct

```
# SCPI: CALCulate:EVDO:MEASurement<instance>:MEValuation:LIST:SEGMent<nr>
↪:DWCP:MAXimum
value: CalculateStruct = driver.multiEval.listPy.segment.dwcp.maximum.
↪calculate(segment = repcap.Segment.Default)
```

Returns the scalar channel power for the data channel results for the segment <no> in list mode. The result is extended to the W24 and W12 I/Q values of the data channel. Only available if subtype 2 or 3 is selected. Otherwise NAV is returned in list mode. To define the statistical length for AVERage, MAXimum, MINimum calculation and to enable the calculation of the results, use the command method RsCmwEvdoMeas.Configure.MultiEval.ListPy.Segment.Modulation.set. 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.

param segment optional repeated capability selector. Default value: Nr1 (settable in the interface 'Segment')

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

fetch(*segment*=<*Segment.Default: -1*>) → FetchStruct

```
# SCPI: FETCh:EVDO:MEASurement<instance>:MEValuation:LIST:SEGMent<nr>
↪:DWCP:MAXimum
value: FetchStruct = driver.multiEval.listPy.segment.dwcp.maximum.fetch(segment␣
↪= repcap.Segment.Default)
```

Returns the scalar channel power for the data channel results for the segment <no> in list mode. The result is extended to the W24 and W12 I/Q values of the data channel. Only available if subtype 2 or 3 is selected. Otherwise NAV is returned in list mode. To define the statistical length for AVERage, MAXimum, MINimum calculation and to enable the calculation of the results, use the command method RsCmwEvdoMeas.Configure.MultiEval.ListPy.Segment.Modulation.set. 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.

param segment optional repeated capability selector. Default value: Nr1 (settable in the interface 'Segment')

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

7.4.7.3.3.4 Minimum

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:SEGMent<Segment>:DWCP:MINimum
CALCulate:EVDO:MEASurement<Instance>:MEValuation:LIST:SEGMent<Segment>:DWCP:MINimum
```

class Minimum

Minimum commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

class CalculateStruct
Response structure. Fields:

- Reliabiltiy: int: decimal 'Conventions and General Information'. In list mode, a zero reliability indicator indicates that the results in all measured segments are valid. A non-zero value indicates that an error occurred in at least one of the measured segments.
- Seg_Reliability: int: 0 | 3 | 4 | 8 The segment reliability indicates whether one of the following exceptions occurred in this segment: 0: No error 3: Signal overflow 4: Signal low 8: Synchronization error If a combination of exceptions occurs, the most severe error is indicated.
- W_24_I: enums.ResultStatus2: No parameter help available
- W_24_Q: enums.ResultStatus2: No parameter help available
- W_12_I: enums.ResultStatus2: No parameter help available
- W_12_Q: enums.ResultStatus2: No parameter help available

class FetchStruct
Response structure. Fields:

- Reliabiltiy: int: decimal 'Conventions and General Information'. In list mode, a zero reliability indicator indicates that the results in all measured segments are valid. A non-zero value indicates that an error occurred in at least one of the measured segments.
- Seg_Reliability: int: 0 | 3 | 4 | 8 The segment reliability indicates whether one of the following exceptions occurred in this segment: 0: No error 3: Signal overflow 4: Signal low 8: Synchronization error If a combination of exceptions occurs, the most severe error is indicated.
- W_24_I: float: No parameter help available
- W_24_Q: float: No parameter help available
- W_12_I: float: No parameter help available
- W_12_Q: float: No parameter help available

calculate(*segment*=<*Segment.Default: -1*>) → CalculateStruct

```
# SCPI: CALCulate:EVDO:MEASurement<instance>:MEValuation:LIST:SEGMent<nr>
↪:DWCP:MINimum
value: CalculateStruct = driver.multiEval.listPy.segment.dwcp.minimum.
↪calculate(segment = repcap.Segment.Default)
```

Returns the scalar channel power for the data channel results for the segment <no> in list mode. The result is extended to the W24 and W12 I/Q values of the data channel. Only available if subtype 2 or 3 is selected. Otherwise NAV is returned in list mode. To define the statistical length for AVERage, MAXimum, MINimum calculation and to enable the calculation of the results, use the command method RsCmwEvdoMeas.Configure.MultiEval.ListPy.Segment.Modulation.set. 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.

param segment optional repeated capability selector. Default value: Nr1 (settable in the interface 'Segment')

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

fetch(*segment*=<*Segment.Default: -1*>) → FetchStruct

```
# SCPI: FETCh:EVDO:MEASurement<instance>:MEValuation:LIST:SEGMent<nr>
↪:DWCP:MINimum
value: FetchStruct = driver.multiEval.listPy.segment.dwcp.minimum.fetch(segment␣
↪= repcap.Segment.Default)
```

Returns the scalar channel power for the data channel results for the segment <no> in list mode. The result is extended to the W24 and W12 I/Q values of the data channel. Only available if subtype 2 or 3 is selected. Otherwise NAV is returned in list mode. To define the statistical length for AVERage, MAXimum, MINimum calculation and to enable the calculation of the results, use the command method RsCmwEvdoMeas.Configure.MultiEval.ListPy.Segment.Modulation.set. 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.

param segment optional repeated capability selector. Default value: Nr1 (settable in the interface 'Segment')

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

7.4.7.3.4 Acp

class Acp

Acp commands group definition. 20 total commands, 6 Sub-groups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.multiEval.listPy.segment.acp.clone()
```

Subgroups

7.4.7.3.4.1 Current

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:SEGMent<Segment>:ACP:CURRent
CALCulate:EVDO:MEASurement<Instance>:MEValuation:LIST:SEGMent<Segment>:ACP:CURRent
```

class Current

Current commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Seg_Reliability: int: 0 | 3 | 4 | 8 The segment reliability indicates whether one of the following exceptions occurred in this segment: 0: No error 3: Signal overflow 4: Signal low 8: Synchronization error If a combination of exceptions occurs, the most severe error is indicated.
- Current_Acp: List[enums.ResultStatus2]: No parameter help available
- Ms_Power_Wide: enums.ResultStatus2: No parameter help available
- Ms_Power_Narrow: enums.ResultStatus2: No parameter help available

- Out_Of_Tol_Count: enums.ResultStatus2: float Out of tolerance result, i.e. percentage of measurement intervals of the statistic count (see [CMDLINK: CONFigure:EVDO:MEASi:MEValuation:SCOunt:SPECtrum CMDLINK]) exceeding the specified limits, see 'Limits (Spectrum) '. Range: 0 % to 100 %, Unit: %
- Cur_Stat_Count: enums.ResultStatus2: decimal Number of evaluated valid slots in this segment. Range: 0 to 1000

class FetchStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Seg_Reliability: int: 0 | 3 | 4 | 8 The segment reliability indicates whether one of the following exceptions occurred in this segment: 0: No error 3: Signal overflow 4: Signal low 8: Synchronization error If a combination of exceptions occurs, the most severe error is indicated.
- Current_Acp: List[float]: No parameter help available
- Ms_Power_Wide: float: No parameter help available
- Ms_Power_Narrow: float: No parameter help available
- Out_Of_Tol_Count: float: float Out of tolerance result, i.e. percentage of measurement intervals of the statistic count (see [CMDLINK: CONFigure:EVDO:MEASi:MEValuation:SCOunt:SPECtrum CMDLINK]) exceeding the specified limits, see 'Limits (Spectrum) '. Range: 0 % to 100 %, Unit: %
- Cur_Stat_Count: int: decimal Number of evaluated valid slots in this segment. Range: 0 to 1000

calculate(*segment*=<*Segment.Default: -1*>) → CalculateStruct

```
# SCPI: CALCulate:EVDO:MEASurement<instance>:MEValuation:LIST:SEGMent<nr>
↪:ACP:CURRent
value: CalculateStruct = driver.multiEval.listPy.segment.acp.current.
↪calculate(segment = repcap.Segment.Default)
```

Returns all ACP value results for the segment <no> in list mode. To define the statistical length for AVERage, MAXimum, MINimum calculation and enable the calculation of the results use the command method RsCmwEvdoMeas.Configure.MultiEval. ListPy.Segment.Spectrum.set. The ranges indicated below apply to all results except standard deviation results. The minimum for standard deviation results equals 0. The maximum equals the width of the indicated range divided by two. Exceptions are explicitly stated. 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. For the out of tolerance and code channel filter match ratio, results retrieved via the CURRent, AVERage and MAXimum command are identical.

param segment optional repeated capability selector. Default value: Nr1 (settable in the interface 'Segment')

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

fetch(*segment*=<*Segment.Default: -1*>) → FetchStruct

```
# SCPI: FETCh:EVDO:MEASurement<instance>:MEValuation:LIST:SEGMent<nr>
↪:ACP:CURRent
value: FetchStruct = driver.multiEval.listPy.segment.acp.current.fetch(segment␣
↪= repcap.Segment.Default)
```

Returns all ACP value results for the segment <no> in list mode. To define the statistical length for AVERage, MAXimum, MINimum calculation and enable the calculation of the results use the command method

RsCmwEvdoMeas.Configure.MultiEval. ListPy.Segment.Spectrum.set. The ranges indicated below apply to all results except standard deviation results. The minimum for standard deviation results equals 0. The maximum equals the width of the indicated range divided by two. Exceptions are explicitly stated. 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. For the out of tolerance and code channel filter match ratio, results retrieved via the CURRent, AVERage and MAXimum command are identical.

param segment optional repeated capability selector. Default value: Nr1 (settable in the interface 'Segment')

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

7.4.7.3.4.2 Extended

class Extended

Extended commands group definition. 10 total commands, 5 Sub-groups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.multiEval.listPy.segment.acp.extended.clone()
```

Subgroups

7.4.7.3.4.3 Current

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:SEGMent<Segment>:ACP:EXTended:CURRent
CALCulate:EVDO:MEASurement<Instance>:MEValuation:LIST:SEGMent<Segment>
↪:ACP:EXTended:CURRent
```

class Current

Current commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Seg_Reliability: int: 0 | 3 | 4 | 8 The segment reliability indicates whether one of the following exceptions occurred in this segment: 0: No error 3: Signal overflow 4: Signal low 8: Synchronization error If a combination of exceptions occurs, the most severe error is indicated.
- Current_Acp: List[enums.ResultStatus2]: No parameter help available
- Ms_Power_Wide: enums.ResultStatus2: No parameter help available
- Ms_Power_Narrow: enums.ResultStatus2: No parameter help available
- Out_Of_Tol_Count: enums.ResultStatus2: float Out of tolerance result, i.e. percentage of measurement intervals of the statistic count (see [CMDLINK: CONFigure:EVDO:MEASi:MEValuation:SCOunt:SPECtrum CMDLINK]) exceeding the specified limits, see 'Limits (Spectrum) '. Range: 0 % to 100 %, Unit: %

- Cur_Stat_Count: enums.ResultStatus2: decimal Number of evaluated valid slots in this segment. Range: 0 to 1000

class FetchStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Seg_Reliability: int: 0 | 3 | 4 | 8 The segment reliability indicates whether one of the following exceptions occurred in this segment: 0: No error 3: Signal overflow 4: Signal low 8: Synchronization error If a combination of exceptions occurs, the most severe error is indicated.
- Current_Acp: List[float]: No parameter help available
- Ms_Power_Wide: float: No parameter help available
- Ms_Power_Narrow: float: No parameter help available
- Out_Of_Tol_Count: float: float Out of tolerance result, i.e. percentage of measurement intervals of the statistic count (see [CMDLINK: CONFigure:EVDO:MEASi:MEValuation:SCOunt:SPECtrum CMDLINK]) exceeding the specified limits, see 'Limits (Spectrum) '. Range: 0 % to 100 %, Unit: %
- Cur_Stat_Count: int: decimal Number of evaluated valid slots in this segment. Range: 0 to 1000

calculate(*segment=<Segment.Default: -1>*) → CalculateStruct

```
# SCPI: CALCulate:EVDO:MEASurement<instance>:MEValuation:LIST:SEGMent<nr>
↪:ACP:EXTended:CURRent
value: CalculateStruct = driver.multiEval.listPy.segment.acp.extended.current.
↪calculate(segment = repcap.Segment.Default)
```

Returns all ACP value results for the segment <no> in list mode. To define the statistical length for AVERage, MAXimum, MINimum calculation and enable the calculation of the results use the command method RsCmwEvdoMeas.Configure.MultiEval. ListPy.Segment.Spectrum.set. The ranges indicated below apply to all results except standard deviation results. The minimum for standard deviation results equals 0. The maximum equals the width of the indicated range divided by two. Exceptions are explicitly stated. 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. For the out of tolerance and code channel filter match ratio, results retrieved via the CURRent, AVERage and MAXimum command are identical.

param segment optional repeated capability selector. Default value: Nr1 (settable in the interface 'Segment')

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

fetch(*segment=<Segment.Default: -1>*) → FetchStruct

```
# SCPI: FETCh:EVDO:MEASurement<instance>:MEValuation:LIST:SEGMent<nr>
↪:ACP:EXTended:CURRent
value: FetchStruct = driver.multiEval.listPy.segment.acp.extended.current.
↪fetch(segment = repcap.Segment.Default)
```

Returns all ACP value results for the segment <no> in list mode. To define the statistical length for AVERage, MAXimum, MINimum calculation and enable the calculation of the results use the command method RsCmwEvdoMeas.Configure.MultiEval. ListPy.Segment.Spectrum.set. The ranges indicated below apply to all results except standard deviation results. The minimum for standard deviation results equals 0. The maximum equals the width of the indicated range divided by two. Exceptions are explicitly stated. 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. For the out of tolerance and code channel filter match ratio, results retrieved via the CURRent, AVERage and MAXimum command are identical.

param segment optional repeated capability selector. Default value: Nr1 (settable in the interface 'Segment')

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

7.4.7.3.4.4 Average

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:SEGMent<Segment>:ACP:EXTended:AVERage
CALCulate:EVDO:MEASurement<Instance>:MEValuation:LIST:SEGMent<Segment>
↪:ACP:EXTended:AVERage
```

class Average

Average commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Seg_Reliability: int: 0 | 3 | 4 | 8 The segment reliability indicates whether one of the following exceptions occurred in this segment: 0: No error 3: Signal overflow 4: Signal low 8: Synchronization error If a combination of exceptions occurs, the most severe error is indicated.
- Current_Acp: List[enums.ResultStatus2]: No parameter help available
- Ms_Power_Wide: enums.ResultStatus2: No parameter help available
- Ms_Power_Narrow: enums.ResultStatus2: No parameter help available
- Out_Of_Tol_Count: enums.ResultStatus2: float Out of tolerance result, i.e. percentage of measurement intervals of the statistic count (see [CMDLINK: CONFigure:EVDO:MEASi:MEValuation:SCOunt:SPECtrum CMDLINK]) exceeding the specified limits, see 'Limits (Spectrum) '. Range: 0 % to 100 %, Unit: %
- Cur_Stat_Count: enums.ResultStatus2: decimal Number of evaluated valid slots in this segment. Range: 0 to 1000

class FetchStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Seg_Reliability: int: 0 | 3 | 4 | 8 The segment reliability indicates whether one of the following exceptions occurred in this segment: 0: No error 3: Signal overflow 4: Signal low 8: Synchronization error If a combination of exceptions occurs, the most severe error is indicated.
- Current_Acp: List[float]: No parameter help available
- Ms_Power_Wide: float: No parameter help available
- Ms_Power_Narrow: float: No parameter help available
- Out_Of_Tol_Count: float: float Out of tolerance result, i.e. percentage of measurement intervals of the statistic count (see [CMDLINK: CONFigure:EVDO:MEASi:MEValuation:SCOunt:SPECtrum CMDLINK]) exceeding the specified limits, see 'Limits (Spectrum) '. Range: 0 % to 100 %, Unit: %
- Cur_Stat_Count: int: decimal Number of evaluated valid slots in this segment. Range: 0 to 1000

calculate(*segment*=<*Segment.Default: -1*>) → CalculateStruct

```
# SCPI: CALCulate:EVDO:MEASurement<instance>:MEValuation:LIST:SEGMent<nr>
↪:ACP:EXTended:AVERage
value: CalculateStruct = driver.multiEval.listPy.segment.acp.extended.average.
↪calculate(segment = repcap.Segment.Default)
```

Returns all ACP value results for the segment <no> in list mode. To define the statistical length for AVERage, MAXimum, MINimum calculation and enable the calculation of the results use the command method RsCmwEvdoMeas.Configure.MultiEval. ListPy.Segment.Spectrum.set. The ranges indicated below apply to all results except standard deviation results. The minimum for standard deviation results equals 0. The maximum equals the width of the indicated range divided by two. Exceptions are explicitly stated. 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. For the out of tolerance and code channel filter match ratio, results retrieved via the CURRent, AVERage and MAXimum command are identical.

param segment optional repeated capability selector. Default value: Nr1 (settable in the interface 'Segment')

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

fetch(*segment*=<*Segment.Default: -1*>) → FetchStruct

```
# SCPI: FETCh:EVDO:MEASurement<instance>:MEValuation:LIST:SEGMent<nr>
↪:ACP:EXTended:AVERage
value: FetchStruct = driver.multiEval.listPy.segment.acp.extended.average.
↪fetch(segment = repcap.Segment.Default)
```

Returns all ACP value results for the segment <no> in list mode. To define the statistical length for AVERage, MAXimum, MINimum calculation and enable the calculation of the results use the command method RsCmwEvdoMeas.Configure.MultiEval. ListPy.Segment.Spectrum.set. The ranges indicated below apply to all results except standard deviation results. The minimum for standard deviation results equals 0. The maximum equals the width of the indicated range divided by two. Exceptions are explicitly stated. 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. For the out of tolerance and code channel filter match ratio, results retrieved via the CURRent, AVERage and MAXimum command are identical.

param segment optional repeated capability selector. Default value: Nr1 (settable in the interface 'Segment')

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

7.4.7.3.4.5 Maximum

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:SEGMent<Segment>:ACP:EXTended:MAXimum
CALCulate:EVDO:MEASurement<Instance>:MEValuation:LIST:SEGMent<Segment>
↪:ACP:EXTended:MAXimum
```

class Maximum

Maximum commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

class CalculateStruct
Response structure. Fields:

- Reliability: int: No parameter help available
- Seg_Reliability: int: 0 | 3 | 4 | 8 The segment reliability indicates whether one of the following exceptions occurred in this segment: 0: No error 3: Signal overflow 4: Signal low 8: Synchronization error If a combination of exceptions occurs, the most severe error is indicated.
- Current_Acp: List[enums.ResultStatus2]: No parameter help available
- Ms_Power_Wide: enums.ResultStatus2: No parameter help available
- Ms_Power_Narrow: enums.ResultStatus2: No parameter help available
- Out_Of_Tol_Count: enums.ResultStatus2: float Out of tolerance result, i.e. percentage of measurement intervals of the statistic count (see [CMDLINK: CONFigure:EVDO:MEASi:MEValuation:SCOunt:SPECtrum CMDLINK]) exceeding the specified limits, see 'Limits (Spectrum) '. Range: 0 % to 100 %, Unit: %
- Cur_Stat_Count: enums.ResultStatus2: decimal Number of evaluated valid slots in this segment. Range: 0 to 1000

class FetchStruct
Response structure. Fields:

- Reliability: int: No parameter help available
- Seg_Reliability: int: 0 | 3 | 4 | 8 The segment reliability indicates whether one of the following exceptions occurred in this segment: 0: No error 3: Signal overflow 4: Signal low 8: Synchronization error If a combination of exceptions occurs, the most severe error is indicated.
- Current_Acp: List[float]: No parameter help available
- Ms_Power_Wide: float: No parameter help available
- Ms_Power_Narrow: float: No parameter help available
- Out_Of_Tol_Count: float: float Out of tolerance result, i.e. percentage of measurement intervals of the statistic count (see [CMDLINK: CONFigure:EVDO:MEASi:MEValuation:SCOunt:SPECtrum CMDLINK]) exceeding the specified limits, see 'Limits (Spectrum) '. Range: 0 % to 100 %, Unit: %
- Cur_Stat_Count: int: decimal Number of evaluated valid slots in this segment. Range: 0 to 1000

calculate(*segment*=<*Segment.Default: -1*>) → CalculateStruct

```
# SCPI: CALCulate:EVDO:MEASurement<instance>:MEValuation:LIST:SEGMent<nr>
↪:ACP:EXTended:MAXimum
value: CalculateStruct = driver.multiEval.listPy.segment.acp.extended.maximum.
↪calculate(segment = repcap.Segment.Default)
```

Returns all ACP value results for the segment <no> in list mode. To define the statistical length for AVERage, MAXimum, MINimum calculation and enable the calculation of the results use the command method RsCmwEvdoMeas.Configure.MultiEval. ListPy.Segment.Spectrum.set. The ranges indicated below apply to all results except standard deviation results. The minimum for standard deviation results equals 0. The maximum equals the width of the indicated range divided by two. Exceptions are explicitly stated. 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. For the out of tolerance and code channel filter match ratio, results retrieved via the CURRent, AVERage and MAXimum command are identical.

param segment optional repeated capability selector. Default value: Nr1 (settable in the interface 'Segment')

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

fetch(*segment*=<*Segment.Default: -1*>) → FetchStruct

```
# SCPI: FETCh:EVDO:MEASurement<instance>:MEValuation:LIST:SEGMent<nr>
↪:ACP:EXTended:MAXimum
value: FetchStruct = driver.multiEval.listPy.segment.acp.extended.maximum.
↪fetch(segment = repcap.Segment.Default)
```

Returns all ACP value results for the segment <no> in list mode. To define the statistical length for AVERage, MAXimum, MINimum calculation and enable the calculation of the results use the command method RsCmwEvdoMeas.Configure.MultiEval. ListPy.Segment.Spectrum.set. The ranges indicated below apply to all results except standard deviation results. The minimum for standard deviation results equals 0. The maximum equals the width of the indicated range divided by two. Exceptions are explicitly stated. 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. For the out of tolerance and code channel filter match ratio, results retrieved via the CURRent, AVERage and MAXimum command are identical.

param segment optional repeated capability selector. Default value: Nr1 (settable in the interface 'Segment')

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

7.4.7.3.4.6 Minimum

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:SEGMent<Segment>:ACP:EXTended:MINimum
CALCulate:EVDO:MEASurement<Instance>:MEValuation:LIST:SEGMent<Segment>
↪:ACP:EXTended:MINimum
```

class Minimum

Minimum commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Seg_Reliability: int: 0 | 3 | 4 | 8 The segment reliability indicates whether one of the following exceptions occurred in this segment: 0: No error 3: Signal overflow 4: Signal low 8: Synchronization error If a combination of exceptions occurs, the most severe error is indicated.
- Current_Acp: List[enums.ResultStatus2]: No parameter help available
- Ms_Power_Wide: enums.ResultStatus2: No parameter help available
- Ms_Power_Narrow: enums.ResultStatus2: No parameter help available
- Out_Of_Tol_Count: enums.ResultStatus2: float Out of tolerance result, i.e. percentage of measurement intervals of the statistic count (see [CMDLINK: CONFigure:EVDO:MEASi:MEValuation:SCOunt:SPECtrum CMDLINK]) exceeding the specified limits, see 'Limits (Spectrum) '. Range: 0 % to 100 %, Unit: %
- Cur_Stat_Count: enums.ResultStatus2: decimal Number of evaluated valid slots in this segment. Range: 0 to 1000

class FetchStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Seg_Reliability: int: 0 | 3 | 4 | 8 The segment reliability indicates whether one of the following exceptions occurred in this segment: 0: No error 3: Signal overflow 4: Signal low 8: Synchronization error If a combination of exceptions occurs, the most severe error is indicated.
- Current_Acp: List[float]: No parameter help available
- Ms_Power_Wide: float: No parameter help available
- Ms_Power_Narrow: float: No parameter help available
- Out_Of_Tol_Count: float: float Out of tolerance result, i.e. percentage of measurement intervals of the statistic count (see [CMDLINK: CONFigure:EVDO:MEASi:MEValuation:SCOunt:SPECtrum CMDLINK]) exceeding the specified limits, see 'Limits (Spectrum) '. Range: 0 % to 100 %, Unit: %
- Cur_Stat_Count: int: decimal Number of evaluated valid slots in this segment. Range: 0 to 1000

calculate(*segment*=<*Segment.Default: -1*>) → CalculateStruct

```
# SCPI: CALCulate:EVDO:MEASurement<instance>:MEValuation:LIST:SEGMent<nr>
↪:ACP:EXTended:MINimum
value: CalculateStruct = driver.multiEval.listPy.segment.acp.extended.minimum.
↪calculate(segment = repcap.Segment.Default)
```

Returns all ACP value results for the segment <no> in list mode. To define the statistical length for AVERage, MAXimum, MINimum calculation and enable the calculation of the results use the command method RsCmwEvdoMeas.Configure.MultiEval. ListPy.Segment.Spectrum.set. The ranges indicated below apply to all results except standard deviation results. The minimum for standard deviation results equals 0. The maximum equals the width of the indicated range divided by two. Exceptions are explicitly stated. 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. For the out of tolerance and code channel filter match ratio, results retrieved via the CURRent, AVERage and MAXimum command are identical.

param segment optional repeated capability selector. Default value: Nr1 (settable in the interface 'Segment')

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

fetch(*segment*=<*Segment.Default: -1*>) → FetchStruct

```
# SCPI: FETCh:EVDO:MEASurement<instance>:MEValuation:LIST:SEGMent<nr>
↪:ACP:EXTended:MINimum
value: FetchStruct = driver.multiEval.listPy.segment.acp.extended.minimum.
↪fetch(segment = repcap.Segment.Default)
```

Returns all ACP value results for the segment <no> in list mode. To define the statistical length for AVERage, MAXimum, MINimum calculation and enable the calculation of the results use the command method RsCmwEvdoMeas.Configure.MultiEval. ListPy.Segment.Spectrum.set. The ranges indicated below apply to all results except standard deviation results. The minimum for standard deviation results equals 0. The maximum equals the width of the indicated range divided by two. Exceptions are explicitly stated. 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. For the out of tolerance and code channel filter match ratio, results retrieved via the CURRent, AVERage and MAXimum command are identical.

param segment optional repeated capability selector. Default value: Nr1 (settable in the interface 'Segment')

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

7.4.7.3.4.7 StandardDev

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:SEGMent<Segment>
↪:ACP:EXTended:SDEViation
CALCulate:EVDO:MEASurement<Instance>:MEValuation:LIST:SEGMent<Segment>
↪:ACP:EXTended:SDEViation
```

class StandardDev

StandardDev commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Seg_Reliability: int: 0 | 3 | 4 | 8 The segment reliability indicates whether one of the following exceptions occurred in this segment: 0: No error 3: Signal overflow 4: Signal low 8: Synchronization error If a combination of exceptions occurs, the most severe error is indicated.
- Current_Acp: List[enums.ResultStatus2]: No parameter help available
- Ms_Power_Wide: enums.ResultStatus2: No parameter help available
- Ms_Power_Narrow: enums.ResultStatus2: No parameter help available
- Out_Of_Tol_Count: enums.ResultStatus2: float Out of tolerance result, i.e. percentage of measurement intervals of the statistic count (see [CMDLINK: CONFigure:EVDO:MEASi:MEValuation:SCOunt:SPECtrum CMDLINK]) exceeding the specified limits, see 'Limits (Spectrum) '. Range: 0 % to 100 %, Unit: %
- Cur_Stat_Count: enums.ResultStatus2: decimal Number of evaluated valid slots in this segment. Range: 0 to 1000

class FetchStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Seg_Reliability: int: 0 | 3 | 4 | 8 The segment reliability indicates whether one of the following exceptions occurred in this segment: 0: No error 3: Signal overflow 4: Signal low 8: Synchronization error If a combination of exceptions occurs, the most severe error is indicated.
- Current_Acp: List[float]: No parameter help available
- Ms_Power_Wide: float: No parameter help available
- Ms_Power_Narrow: float: No parameter help available
- Out_Of_Tol_Count: float: float Out of tolerance result, i.e. percentage of measurement intervals of the statistic count (see [CMDLINK: CONFigure:EVDO:MEASi:MEValuation:SCOunt:SPECtrum CMDLINK]) exceeding the specified limits, see 'Limits (Spectrum) '. Range: 0 % to 100 %, Unit: %
- Cur_Stat_Count: int: decimal Number of evaluated valid slots in this segment. Range: 0 to 1000

calculate(*segment*=<*Segment.Default: -1*>) → CalculateStruct

```
# SCPI: CALCulate:EVDO:MEASurement<instance>:MEValuation:LIST:SEGMent<nr>
↪:ACP:EXTended:SDEViation
value: CalculateStruct = driver.multiEval.listPy.segment.acp.extended.
↪standardDev.calculate(segment = repcap.Segment.Default)
```

Returns all ACP value results for the segment <no> in list mode. To define the statistical length for AVERage, MAXimum, MINimum calculation and enable the calculation of the results use the command method RsCmwEvdoMeas.Configure.MultiEval. ListPy.Segment.Spectrum.set. The ranges indicated below apply to all results except standard deviation results. The minimum for standard deviation results equals 0. The maximum equals the width of the indicated range divided by two. Exceptions are explicitly stated. 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. For the out of tolerance and code channel filter match ratio, results retrieved via the CURRent, AVERage and MAXimum command are identical.

param segment optional repeated capability selector. Default value: Nr1 (settable in the interface 'Segment')

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

fetch(*segment*=<*Segment.Default: -1*>) → FetchStruct

```
# SCPI: FETCh:EVDO:MEASurement<instance>:MEValuation:LIST:SEGMent<nr>
↪:ACP:EXTended:SDEViation
value: FetchStruct = driver.multiEval.listPy.segment.acp.extended.standardDev.
↪fetch(segment = repcap.Segment.Default)
```

Returns all ACP value results for the segment <no> in list mode. To define the statistical length for AVERage, MAXimum, MINimum calculation and enable the calculation of the results use the command method RsCmwEvdoMeas.Configure.MultiEval. ListPy.Segment.Spectrum.set. The ranges indicated below apply to all results except standard deviation results. The minimum for standard deviation results equals 0. The maximum equals the width of the indicated range divided by two. Exceptions are explicitly stated. 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. For the out of tolerance and code channel filter match ratio, results retrieved via the CURRent, AVERage and MAXimum command are identical.

param segment optional repeated capability selector. Default value: Nr1 (settable in the interface 'Segment')

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

7.4.7.3.4.8 Average

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:SEGMent<Segment>:ACP:AVERage
CALCulate:EVDO:MEASurement<Instance>:MEValuation:LIST:SEGMent<Segment>:ACP:AVERage
```

class Average

Average commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Seg_Reliability: int: 0 | 3 | 4 | 8 The segment reliability indicates whether one of the following exceptions occurred in this segment: 0: No error 3: Signal overflow 4: Signal low 8: Synchronization error If a combination of exceptions occurs, the most severe error is indicated.
- Current_Acp: List[enums.ResultStatus2]: No parameter help available
- Ms_Power_Wide: enums.ResultStatus2: No parameter help available
- Ms_Power_Narrow: enums.ResultStatus2: No parameter help available
- Out_Of_Tol_Count: enums.ResultStatus2: float Out of tolerance result, i.e. percentage of measurement intervals of the statistic count (see [CMDLINK: CONFigure:EVDO:MEASi:MEValuation:SCOunt:SPECtrum CMDLINK]) exceeding the specified limits, see 'Limits (Spectrum) '. Range: 0 % to 100 %, Unit: %
- Cur_Stat_Count: enums.ResultStatus2: decimal Number of evaluated valid slots in this segment. Range: 0 to 1000

class FetchStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Seg_Reliability: int: 0 | 3 | 4 | 8 The segment reliability indicates whether one of the following exceptions occurred in this segment: 0: No error 3: Signal overflow 4: Signal low 8: Synchronization error If a combination of exceptions occurs, the most severe error is indicated.
- Current_Acp: List[float]: No parameter help available
- Ms_Power_Wide: float: No parameter help available
- Ms_Power_Narrow: float: No parameter help available
- Out_Of_Tol_Count: float: float Out of tolerance result, i.e. percentage of measurement intervals of the statistic count (see [CMDLINK: CONFigure:EVDO:MEASi:MEValuation:SCOunt:SPECtrum CMDLINK]) exceeding the specified limits, see 'Limits (Spectrum) '. Range: 0 % to 100 %, Unit: %
- Cur_Stat_Count: int: decimal Number of evaluated valid slots in this segment. Range: 0 to 1000

calculate(*segment=<Segment.Default: -1>*) → CalculateStruct

```
# SCPI: CALCulate:EVDO:MEASurement<instance>:MEValuation:LIST:SEGMent<nr>
↪:ACP:AVERage
value: CalculateStruct = driver.multiEval.listPy.segment.acp.average.
↪calculate(segment = repcap.Segment.Default)
```

Returns all ACP value results for the segment <no> in list mode. To define the statistical length for AVERage, MAXimum, MINimum calculation and enable the calculation of the results use the command method RsCmwEvdoMeas.Configure.MultiEval. ListPy.Segment.Spectrum.set. The ranges indicated below apply to all results except standard deviation results. The minimum for standard deviation results equals 0. The maximum equals the width of the indicated range divided by two. Exceptions are explicitly stated. 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. For the out of tolerance and code channel filter match ratio, results retrieved via the CURRent, AVERage and MAXimum command are identical.

param segment optional repeated capability selector. Default value: Nr1 (settable in the interface 'Segment')

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

fetch(*segment=<Segment.Default: -1>*) → FetchStruct

```
# SCPI: FETCh:EVDO:MEASurement<instance>:MEValuation:LIST:SEGMent<nr>
↪:ACP:AVERage
value: FetchStruct = driver.multiEval.listPy.segment.acp.average.fetch(segment␣
↪= repcap.Segment.Default)
```

Returns all ACP value results for the segment <no> in list mode. To define the statistical length for AVERage, MAXimum, MINimum calculation and enable the calculation of the results use the command method RsCmwEvdoMeas.Configure.MultiEval. ListPy.Segment.Spectrum.set. The ranges indicated below apply to all results except standard deviation results. The minimum for standard deviation results equals 0. The maximum equals the width of the indicated range divided by two. Exceptions are explicitly stated. 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. For the out of tolerance and code channel filter match ratio, results retrieved via the CURRent, AVERage and MAXimum command are identical.

param segment optional repeated capability selector. Default value: Nr1 (settable in the interface 'Segment')

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

7.4.7.3.4.9 Maximum

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:SEGMent<Segment>:ACP:MAXimum
CALCulate:EVDO:MEASurement<Instance>:MEValuation:LIST:SEGMent<Segment>:ACP:MAXimum
```

class Maximum

Maximum commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Seg_Reliability: int: 0 | 3 | 4 | 8 The segment reliability indicates whether one of the following exceptions occurred in this segment: 0: No error 3: Signal overflow 4: Signal low 8: Synchronization error If a combination of exceptions occurs, the most severe error is indicated.
- Current_Acp: List[enums.ResultStatus2]: No parameter help available
- Ms_Power_Wide: enums.ResultStatus2: No parameter help available
- Ms_Power_Narrow: enums.ResultStatus2: No parameter help available
- Out_Of_Tol_Count: enums.ResultStatus2: float Out of tolerance result, i.e. percentage of measurement intervals of the statistic count (see [CMDLINK: CONFigure:EVDO:MEASi:MEValuation:SCOunt:SPECtrum CMDLINK]) exceeding the specified limits, see 'Limits (Spectrum) '. Range: 0 % to 100 %, Unit: %
- Cur_Stat_Count: enums.ResultStatus2: decimal Number of evaluated valid slots in this segment. Range: 0 to 1000

class FetchStruct

Response structure. Fields:

- Reliability: int: No parameter help available

- Seg_Reliability: int: 0 | 3 | 4 | 8 The segment reliability indicates whether one of the following exceptions occurred in this segment: 0: No error 3: Signal overflow 4: Signal low 8: Synchronization error If a combination of exceptions occurs, the most severe error is indicated.
- Current_Acp: List[float]: No parameter help available
- Ms_Power_Wide: float: No parameter help available
- Ms_Power_Narrow: float: No parameter help available
- Out_Of_Tol_Count: float: float Out of tolerance result, i.e. percentage of measurement intervals of the statistic count (see [CMDLINK: CONFigure:EVDO:MEASi:MEValuation:SCOunt:SPECtrum CMDLINK]) exceeding the specified limits, see 'Limits (Spectrum) '. Range: 0 % to 100 %, Unit: %
- Cur_Stat_Count: int: decimal Number of evaluated valid slots in this segment. Range: 0 to 1000

calculate(*segment=<Segment.Default: -1>*) → CalculateStruct

```
# SCPI: CALCulate:EVDO:MEASurement<instance>:MEValuation:LIST:SEGMent<nr>
↪:ACP:MAXimum
value: CalculateStruct = driver.multiEval.listPy.segment.acp.maximum.
↪calculate(segment = repcap.Segment.Default)
```

Returns all ACP value results for the segment <no> in list mode. To define the statistical length for AVERage, MAXimum, MINimum calculation and enable the calculation of the results use the command method RsCmwEvdoMeas.Configure.MultiEval. ListPy.Segment.Spectrum.set. The ranges indicated below apply to all results except standard deviation results. The minimum for standard deviation results equals 0. The maximum equals the width of the indicated range divided by two. Exceptions are explicitly stated. 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. For the out of tolerance and code channel filter match ratio, results retrieved via the CURRent, AVERage and MAXimum command are identical.

param segment optional repeated capability selector. Default value: Nr1 (settable in the interface 'Segment')

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

fetch(*segment=<Segment.Default: -1>*) → FetchStruct

```
# SCPI: FETCh:EVDO:MEASurement<instance>:MEValuation:LIST:SEGMent<nr>
↪:ACP:MAXimum
value: FetchStruct = driver.multiEval.listPy.segment.acp.maximum.fetch(segment␣
↪= repcap.Segment.Default)
```

Returns all ACP value results for the segment <no> in list mode. To define the statistical length for AVERage, MAXimum, MINimum calculation and enable the calculation of the results use the command method RsCmwEvdoMeas.Configure.MultiEval. ListPy.Segment.Spectrum.set. The ranges indicated below apply to all results except standard deviation results. The minimum for standard deviation results equals 0. The maximum equals the width of the indicated range divided by two. Exceptions are explicitly stated. 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. For the out of tolerance and code channel filter match ratio, results retrieved via the CURRent, AVERage and MAXimum command are identical.

param segment optional repeated capability selector. Default value: Nr1 (settable in the interface 'Segment')

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

7.4.7.3.4.10 Minimum

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:SEGMent<Segment>:ACP:MINimum
CALCulate:EVDO:MEASurement<Instance>:MEValuation:LIST:SEGMent<Segment>:ACP:MINimum
```

class Minimum

Minimum commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Seg_Reliability: int: 0 | 3 | 4 | 8 The segment reliability indicates whether one of the following exceptions occurred in this segment: 0: No error 3: Signal overflow 4: Signal low 8: Synchronization error If a combination of exceptions occurs, the most severe error is indicated.
- Current_Acp: List[enums.ResultStatus2]: No parameter help available
- Ms_Power_Wide: enums.ResultStatus2: No parameter help available
- Ms_Power_Narrow: enums.ResultStatus2: No parameter help available
- Out_Of_Tol_Count: enums.ResultStatus2: float Out of tolerance result, i.e. percentage of measurement intervals of the statistic count (see [CMDLINK: CONFigure:EVDO:MEASi:MEValuation:SCOunt:SPECtrum CMDLINK]) exceeding the specified limits, see 'Limits (Spectrum) '. Range: 0 % to 100 %, Unit: %
- Cur_Stat_Count: enums.ResultStatus2: decimal Number of evaluated valid slots in this segment. Range: 0 to 1000

class FetchStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Seg_Reliability: int: 0 | 3 | 4 | 8 The segment reliability indicates whether one of the following exceptions occurred in this segment: 0: No error 3: Signal overflow 4: Signal low 8: Synchronization error If a combination of exceptions occurs, the most severe error is indicated.
- Current_Acp: List[float]: No parameter help available
- Ms_Power_Wide: float: No parameter help available
- Ms_Power_Narrow: float: No parameter help available
- Out_Of_Tol_Count: float: float Out of tolerance result, i.e. percentage of measurement intervals of the statistic count (see [CMDLINK: CONFigure:EVDO:MEASi:MEValuation:SCOunt:SPECtrum CMDLINK]) exceeding the specified limits, see 'Limits (Spectrum) '. Range: 0 % to 100 %, Unit: %
- Cur_Stat_Count: int: decimal Number of evaluated valid slots in this segment. Range: 0 to 1000

calculate(*segment=<Segment.Default: -1>*) → CalculateStruct

```
# SCPI: CALCulate:EVDO:MEASurement<instance>:MEValuation:LIST:SEGMent<nr>
↪:ACP:MINimum
value: CalculateStruct = driver.multiEval.listPy.segment.acp.minimum.
↪calculate(segment = repcap.Segment.Default)
```

Returns all ACP value results for the segment <no> in list mode. To define the statistical length for AVERage, MAXimum, MINimum calculation and enable the calculation of the results use the command method RsCmwEvdoMeas.Configure.MultiEval. ListPy.Segment.Spectrum.set. The ranges indicated below apply to all results except standard deviation results. The minimum for standard deviation results equals 0. The maximum equals the width of the indicated range divided by two. Exceptions are explicitly stated. 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. For the out of tolerance and code channel filter match ratio, results retrieved via the CURRent, AVERage and MAXimum command are identical.

param segment optional repeated capability selector. Default value: Nr1 (settable in the interface 'Segment')

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

fetch(*segment=<Segment.Default: -1>*) → FetchStruct

```
# SCPI: FETCh:EVDO:MEASurement<instance>:MEValuation:LIST:SEGMent<nr>
↪:ACP:MINimum
value: FetchStruct = driver.multiEval.listPy.segment.acp.minimum.fetch(segment␣
↪= repcap.Segment.Default)
```

Returns all ACP value results for the segment <no> in list mode. To define the statistical length for AVERage, MAXimum, MINimum calculation and enable the calculation of the results use the command method RsCmwEvdoMeas.Configure.MultiEval. ListPy.Segment.Spectrum.set. The ranges indicated below apply to all results except standard deviation results. The minimum for standard deviation results equals 0. The maximum equals the width of the indicated range divided by two. Exceptions are explicitly stated. 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. For the out of tolerance and code channel filter match ratio, results retrieved via the CURRent, AVERage and MAXimum command are identical.

param segment optional repeated capability selector. Default value: Nr1 (settable in the interface 'Segment')

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

7.4.7.3.4.11 StandardDev

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:SEGMent<Segment>:ACP:SDEViation
CALCulate:EVDO:MEASurement<Instance>:MEValuation:LIST:SEGMent<Segment>:ACP:SDEViation
```

class StandardDev

StandardDev commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Seg_Reliability: int: 0 | 3 | 4 | 8 The segment reliability indicates whether one of the following exceptions occurred in this segment: 0: No error 3: Signal overflow 4: Signal low 8: Synchronization error If a combination of exceptions occurs, the most severe error is indicated.
- Current_Acp: List[enums.ResultStatus2]: No parameter help available
- Ms_Power_Wide: enums.ResultStatus2: No parameter help available

- Ms_Power_Narrow: enums.ResultStatus2: No parameter help available
- Out_Of_Tol_Count: enums.ResultStatus2: float Out of tolerance result, i.e. percentage of measurement intervals of the statistic count (see [CMDLINK: CONFigure:EVDO:MEASi:MEValuation:SCOunt:SPECtrum CMDLINK]) exceeding the specified limits, see 'Limits (Spectrum) '. Range: 0 % to 100 %, Unit: %
- Cur_Stat_Count: enums.ResultStatus2: decimal Number of evaluated valid slots in this segment. Range: 0 to 1000

class FetchStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Seg_Reliability: int: 0 | 3 | 4 | 8 The segment reliability indicates whether one of the following exceptions occurred in this segment: 0: No error 3: Signal overflow 4: Signal low 8: Synchronization error If a combination of exceptions occurs, the most severe error is indicated.
- Current_Acp: List[float]: No parameter help available
- Ms_Power_Wide: float: No parameter help available
- Ms_Power_Narrow: float: No parameter help available
- Out_Of_Tol_Count: float: float Out of tolerance result, i.e. percentage of measurement intervals of the statistic count (see [CMDLINK: CONFigure:EVDO:MEASi:MEValuation:SCOunt:SPECtrum CMDLINK]) exceeding the specified limits, see 'Limits (Spectrum) '. Range: 0 % to 100 %, Unit: %
- Cur_Stat_Count: int: decimal Number of evaluated valid slots in this segment. Range: 0 to 1000

calculate(*segment=<Segment.Default: -1>*) → CalculateStruct

```
# SCPI: CALCulate:EVDO:MEASurement<instance>:MEValuation:LIST:SEGMent<nr>
↪:ACP:SDEViation
value: CalculateStruct = driver.multiEval.listPy.segment.acp.standardDev.
↪calculate(segment = repcap.Segment.Default)
```

Returns all ACP value results for the segment <no> in list mode. To define the statistical length for AVERage, MAXimum, MINimum calculation and enable the calculation of the results use the command method RsCmwEvdoMeas.Configure.MultiEval. ListPy.Segment.Spectrum.set. The ranges indicated below apply to all results except standard deviation results. The minimum for standard deviation results equals 0. The maximum equals the width of the indicated range divided by two. Exceptions are explicitly stated. 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. For the out of tolerance and code channel filter match ratio, results retrieved via the CURRent, AVERage and MAXimum command are identical.

param segment optional repeated capability selector. Default value: Nr1 (settable in the interface 'Segment')

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

fetch(*segment=<Segment.Default: -1>*) → FetchStruct

```
# SCPI: FETCh:EVDO:MEASurement<instance>:MEValuation:LIST:SEGMent<nr>
↪:ACP:SDEViation
value: FetchStruct = driver.multiEval.listPy.segment.acp.standardDev.
↪fetch(segment = repcap.Segment.Default)
```

Returns all ACP value results for the segment <no> in list mode. To define the statistical length for AVERage, MAXimum, MINimum calculation and enable the calculation of the results use the command method RsCmwEvdoMeas.Configure.MultiEval. ListPy.Segment.Spectrum.set. The ranges indicated below apply to all results except standard deviation results. The minimum for standard deviation results equals 0. The maximum equals the width of the indicated range divided by two. Exceptions are explicitly stated. 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. For the out of tolerance and code channel filter match ratio, results retrieved via the CURRent, AVERage and MAXimum command are identical.

param segment optional repeated capability selector. Default value: Nr1 (settable in the interface 'Segment')

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

7.4.7.3.5 Obw

class Obw

Obw commands group definition. 8 total commands, 4 Sub-groups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.multiEval.listPy.segment.obw.clone()
```

Subgroups

7.4.7.3.5.1 Current

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:SEGMent<Segment>:OBW:CURRent
CALCulate:EVDO:MEASurement<Instance>:MEValuation:LIST:SEGMent<Segment>:OBW:CURRent
```

class Current

Current commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: decimal 'Conventions and General Information'. In list mode, a zero reliability indicator indicates that the results in all measured segments are valid. A non-zero value indicates that an error occurred in at least one of the measured segments.
- Seg_Reliability: int: 0 | 3 | 4 | 8 The segment reliability indicates whether one of the following exceptions occurred in this segment: 0: No error 3: Signal overflow 4: Signal low 8: Synchronization error If a combination of exceptions occurs, the most severe error is indicated.
- Obw: enums.ResultStatus2: float Occupied bandwidth Range: 0 MHz to 16 MHz , Unit: MHz
- Lower_Freq: float: float Lower edge of the selected carrier's or carrier group's occupied bandwidth. For single-carrier configurations, the lower and upper edges are returned as frequency offsets related to the carrier's center frequency. For multi-carrier configurations, absolute frequency values are returned. Range: - 16 MHz to 6000 MHz , Unit: MHz

- Upper_Freq: float: float Upper edge of the selected carrier's or carrier group's occupied bandwidth. For single-carrier configurations, the lower and upper edges are returned as frequency offsets related to the carrier's center frequency. For multi-carrier configurations, absolute frequency values are returned. Range: - 16 MHz to 6000 MHz , Unit: MHz

class FetchStruct

Response structure. Fields:

- Reliability: int: decimal 'Conventions and General Information'. In list mode, a zero reliability indicator indicates that the results in all measured segments are valid. A non-zero value indicates that an error occurred in at least one of the measured segments.
- Seg_Reliability: int: 0 | 3 | 4 | 8 The segment reliability indicates whether one of the following exceptions occurred in this segment: 0: No error 3: Signal overflow 4: Signal low 8: Synchronization error If a combination of exceptions occurs, the most severe error is indicated.
- Obw: float: float Occupied bandwidth Range: 0 MHz to 16 MHz , Unit: MHz
- Lower_Freq: float: float Lower edge of the selected carrier's or carrier group's occupied bandwidth. For single-carrier configurations, the lower and upper edges are returned as frequency offsets related to the carrier's center frequency. For multi-carrier configurations, absolute frequency values are returned. Range: - 16 MHz to 6000 MHz , Unit: MHz
- Upper_Freq: float: float Upper edge of the selected carrier's or carrier group's occupied bandwidth. For single-carrier configurations, the lower and upper edges are returned as frequency offsets related to the carrier's center frequency. For multi-carrier configurations, absolute frequency values are returned. Range: - 16 MHz to 6000 MHz , Unit: MHz

calculate(*segment=<Segment.Default: -1>*) → CalculateStruct

```
# SCPI: CALCulate:EVDO:MEASurement<instance>:MEValuation:LIST:SEGMent<nr>
↪:OBW:CURRent
value: CalculateStruct = driver.multiEval.listPy.segment.obw.current.
↪calculate(segment = repcap.Segment.Default)
```

Returns occupied bandwidth (OBW) results for the segment <no> in list mode. To enable the calculation of the results, use the command method RsCmwEvdoMeas.Configure.MultiEval.ListPy.Segment.Modulation.set. For single-carrier configurations (i. e. only carrier 0 is active) the <Number> suffix can be omitted to obtain the OBW results. For multi-carrier configurations the <Number> suffixes are used as follows:

INTRO_CMD_HELP: Starts, stops, or aborts the measurement:

- For an active and isolated carrier i, use <Number> = i+1 to get its OBW results (i = 0, 1, 2)
- Use <Number> = 4 to display the OBW results of the 'Overall Carrier'
- If all active carriers are adjacent, use <Number>=4 to get the group (overall) OBW results
- If three carriers are active and exactly two carriers i,j with 0i<j2 are adjacent, use <Number> = i+1 for the joint OBW results of adjacent carriers.

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. For details, refer to 'Multi-Evaluation Measurement Results'.

param segment optional repeated capability selector. Default value: Nr1 (settable in the interface 'Segment')

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

fetch(*segment=<Segment.Default: -1>*) → FetchStruct

```
# SCPI: FETCh:EVDO:MEASurement<instance>:MEValuation:LIST:SEGMent<nr>
↪:OBW:CURRent
value: FetchStruct = driver.multiEval.listPy.segment.obw.current.fetch(segment␣
↪= repcap.Segment.Default)
```

Returns occupied bandwidth (OBW) results for the segment <no> in list mode. To enable the calculation of the results, use the command method RsCmwEvdoMeas.Configure.MultiEval.ListPy.Segment.Modulation.set. For single-carrier configurations (i. e. only carrier 0 is active) the <Number> suffix can be omitted to obtain the OBW results. For multi-carrier configurations the <Number> suffixes are used as follows:

INTRO_CMD_HELP: Starts, stops, or aborts the measurement:

- For an active and isolated carrier i, use <Number> = i+1 to get its OBW results (i = 0, 1, 2)
- Use <Number> = 4 to display the OBW results of the 'Overall Carrier'
- If all active carriers are adjacent, use <Number>=4 to get the group (overall) OBW results
- If three carriers are active and exactly two carriers i,j with 0i<j2 are adjacent, use <Number> = i+1 for the joint OBW results of adjacent carriers.

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. For details, refer to 'Multi-Evaluation Measurement Results'.

param segment optional repeated capability selector. Default value: Nr1 (settable in the interface 'Segment')

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

7.4.7.3.5.2 Average

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:SEGMent<Segment>:OBW:AVERage
CALCulate:EVDO:MEASurement<Instance>:MEValuation:LIST:SEGMent<Segment>:OBW:AVERage
```

class Average

Average commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Seg_Reliability: int: 0 | 3 | 4 | 8 The segment reliability indicates whether one of the following exceptions occurred in this segment: 0: No error 3: Signal overflow 4: Signal low 8: Synchronization error If a combination of exceptions occurs, the most severe error is indicated.
- Obw: enums.ResultStatus2: float Occupied bandwidth Range: 0 MHz to 16 MHz (SDEViation 0 MHz to 8 MHz) , Unit: MHz

class FetchStruct

Response structure. Fields:

- Reliability: int: No parameter help available

- Seg_Reliability: int: 0 | 3 | 4 | 8 The segment reliability indicates whether one of the following exceptions occurred in this segment: 0: No error 3: Signal overflow 4: Signal low 8: Synchronization error If a combination of exceptions occurs, the most severe error is indicated.
- Obw: float: float Occupied bandwidth Range: 0 MHz to 16 MHz (SDEViation 0 MHz to 8 MHz) , Unit: MHz

calculate(*segment=<Segment.Default: -1>*) → CalculateStruct

```
# SCPI: CALCulate:EVDO:MEASurement<instance>:MEValuation:LIST:SEGMent<nr>
↪:OBW:AVERage
value: CalculateStruct = driver.multiEval.listPy.segment.obw.average.
↪calculate(segment = repcap.Segment.Default)
```

In list mode, returns the occupied bandwidth (OBW) result for segment <no>. To define the statistical length for AVERage, MAXimum calculation and to enable the calculation of the results, use the command method RsCmwEvdoMeas.Configure. MultiEval.ListPy.Segment.Modulation.set. The ranges indicated below apply to all results except standard deviation results. The minimum for standard deviation results equals 0. The maximum equals the width of the indicated range divided by two. Exceptions are explicitly stated. 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.

param segment optional repeated capability selector. Default value: Nr1 (settable in the interface 'Segment')

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

fetch(*segment=<Segment.Default: -1>*) → FetchStruct

```
# SCPI: FETCh:EVDO:MEASurement<instance>:MEValuation:LIST:SEGMent<nr>
↪:OBW:AVERage
value: FetchStruct = driver.multiEval.listPy.segment.obw.average.fetch(segment␣
↪= repcap.Segment.Default)
```

In list mode, returns the occupied bandwidth (OBW) result for segment <no>. To define the statistical length for AVERage, MAXimum calculation and to enable the calculation of the results, use the command method RsCmwEvdoMeas.Configure. MultiEval.ListPy.Segment.Modulation.set. The ranges indicated below apply to all results except standard deviation results. The minimum for standard deviation results equals 0. The maximum equals the width of the indicated range divided by two. Exceptions are explicitly stated. 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.

param segment optional repeated capability selector. Default value: Nr1 (settable in the interface 'Segment')

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

7.4.7.3.5.3 Maximum

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:SEGMent<Segment>:OBW:MAXimum
CALCulate:EVDO:MEASurement<Instance>:MEValuation:LIST:SEGMent<Segment>:OBW:MAXimum
```

class Maximum

Maximum commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Seg_Reliability: int: 0 | 3 | 4 | 8 The segment reliability indicates whether one of the following exceptions occurred in this segment: 0: No error 3: Signal overflow 4: Signal low 8: Synchronization error If a combination of exceptions occurs, the most severe error is indicated.
- Obw: enums.ResultStatus2: float Occupied bandwidth Range: 0 MHz to 16 MHz (SDEViation 0 MHz to 8 MHz) , Unit: MHz

class FetchStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Seg_Reliability: int: 0 | 3 | 4 | 8 The segment reliability indicates whether one of the following exceptions occurred in this segment: 0: No error 3: Signal overflow 4: Signal low 8: Synchronization error If a combination of exceptions occurs, the most severe error is indicated.
- Obw: float: float Occupied bandwidth Range: 0 MHz to 16 MHz (SDEViation 0 MHz to 8 MHz) , Unit: MHz

calculate(*segment*=<*Segment.Default: -1*>) → CalculateStruct

```
# SCPI: CALCulate:EVDO:MEASurement<instance>:MEValuation:LIST:SEGMent<nr>
↪:OBW:MAXimum
value: CalculateStruct = driver.multiEval.listPy.segment.obw.maximum.
↪calculate(segment = repcap.Segment.Default)
```

In list mode, returns the occupied bandwidth (OBW) result for segment <no>. To define the statistical length for AVERage, MAXimum calculation and to enable the calculation of the results, use the command method RsCmwEvdoMeas.Configure. MultiEval.ListPy.Segment.Modulation.set. The ranges indicated below apply to all results except standard deviation results. The minimum for standard deviation results equals 0. The maximum equals the width of the indicated range divided by two. Exceptions are explicitly stated. 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.

param segment optional repeated capability selector. Default value: Nr1 (settable in the interface 'Segment')

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

fetch(*segment*=<*Segment.Default: -1*>) → FetchStruct

```
# SCPI: FETCh:EVDO:MEASurement<instance>:MEValuation:LIST:SEGMent<nr>
↪:OBW:MAXimum
value: FetchStruct = driver.multiEval.listPy.segment.obw.maximum.fetch(segment␣
↪= repcap.Segment.Default)
```

In list mode, returns the occupied bandwidth (OBW) result for segment <no>. To define the statistical length for AVERage, MAXimum calculation and to enable the calculation of the results, use the command method RsCmwEvdoMeas.Configure. MultiEval.ListPy.Segment.Modulation.set. The ranges indicated below apply to all results except standard deviation results. The minimum for standard deviation results equals 0. The maximum equals the width of the indicated range divided by two. Exceptions are explicitly stated. 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.

param segment optional repeated capability selector. Default value: Nr1 (settable in the interface 'Segment')

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

7.4.7.3.5.4 StandardDev

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:SEGMent<Segment>:OBW:SDEViation
CALCulate:EVDO:MEASurement<Instance>:MEValuation:LIST:SEGMent<Segment>:OBW:SDEViation
```

class StandardDev

StandardDev commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Seg_Reliability: int: 0 | 3 | 4 | 8 The segment reliability indicates whether one of the following exceptions occurred in this segment: 0: No error 3: Signal overflow 4: Signal low 8: Synchronization error If a combination of exceptions occurs, the most severe error is indicated.
- Obw: enums.ResultStatus2: float Occupied bandwidth Range: 0 MHz to 16 MHz (SDEViation 0 MHz to 8 MHz) , Unit: MHz

class FetchStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Seg_Reliability: int: 0 | 3 | 4 | 8 The segment reliability indicates whether one of the following exceptions occurred in this segment: 0: No error 3: Signal overflow 4: Signal low 8: Synchronization error If a combination of exceptions occurs, the most severe error is indicated.
- Obw: float: float Occupied bandwidth Range: 0 MHz to 16 MHz (SDEViation 0 MHz to 8 MHz) , Unit: MHz

calculate(*segment*=<*Segment.Default: -1*>) → CalculateStruct

```
# SCPI: CALCulate:EVDO:MEASurement<instance>:MEValuation:LIST:SEGMent<nr>
↪:OBW:SDEViation
value: CalculateStruct = driver.multiEval.listPy.segment.obw.standardDev.
↪calculate(segment = repcap.Segment.Default)
```

In list mode, returns the occupied bandwidth (OBW) result for segment <no>. To define the statistical length for AVERage, MAXimum calculation and to enable the calculation of the results, use the command method RsCmwEvdoMeas.Configure. MultiEval.ListPy.Segment.Modulation.set. The ranges indicated below apply to all results except standard deviation results. The minimum for standard deviation results equals 0. The maximum equals the width of the indicated range divided by two. Exceptions are explicitly stated. 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.

param segment optional repeated capability selector. Default value: Nr1 (settable in the interface 'Segment')

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

fetch(*segment*=<*Segment.Default: -1*>) → FetchStruct

```
# SCPI: FETCh:EVDO:MEASurement<instance>:MEValuation:LIST:SEGMent<nr>
↪:OBW:SDEViation
value: FetchStruct = driver.multiEval.listPy.segment.obw.standardDev.
↪fetch(segment = repcap.Segment.Default)
```

In list mode, returns the occupied bandwidth (OBW) result for segment <no>. To define the statistical length for AVERage, MAXimum calculation and to enable the calculation of the results, use the command method RsCmwEvdoMeas.Configure. MultiEval.ListPy.Segment.Modulation.set. The ranges indicated below apply to all results except standard deviation results. The minimum for standard deviation results equals 0. The maximum equals the width of the indicated range divided by two. Exceptions are explicitly stated. 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.

param segment optional repeated capability selector. Default value: Nr1 (settable in the interface 'Segment')

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

7.4.7.4 Cp

class Cp

Cp commands group definition. 63 total commands, 11 Sub-groups, 0 group commands

Cloning the Group

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

Subgroups

7.4.7.4.1 Rri

class Rri
Rri commands group definition. 9 total commands, 5 Sub-groups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.multiEval.listPy.cp.rri.clone()
```

Subgroups

7.4.7.4.1.1 State

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:CP:RRI:STATe
```

class State
State commands group definition. 1 total commands, 0 Sub-groups, 1 group commands

fetch() → List[RsCmwEvdoMeas.enums.ResultStateA]

```
# SCPI: FETCh:EVDO:MEASurement<instance>:MEValuation:LIST:CP:RRI:STATe
value: List[enums.ResultStateA] = driver.multiEval.listPy.cp.rri.state.fetch()
```

Returns the state of a particular reverse link channel (ACK/DSC, auxiliary pilot, data, DRC, PILOT, RRI) in the channel power measurement for all active segments.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return rri: INVisible | ACTive | IACTive Comma-separated list of values, one per active segment.

7.4.7.4.1.2 Current

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:CP:RRI:CURRent
CALCulate:EVDO:MEASurement<Instance>:MEValuation:LIST:CP:RRI:CURRent
```

class Current
Current commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

calculate() → List[RsCmwEvdoMeas.enums.ResultStatus2]

```
# SCPI: CALCulate:EVDO:MEASurement<instance>:MEValuation:LIST:CP:RRI:CURRent
value: List[enums.ResultStatus2] = driver.multiEval.listPy.cp.rri.current.
↪calculate()
```

Returns the RMS power (statistical values) of the reverse rate indicator channel for all active list mode segments. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return rri: float Comma-separated list of values, one per active segment. Range: -60 dB to +10 dB , Unit: dB

fetch() → List[float]

```
# SCPI: FETCh:EVDO:MEASurement<instance>:MEValuation:LIST:CP:RRI:CURRent
value: List[float] = driver.multiEval.listPy.cp.rri.current.fetch()
```

Returns the RMS power (statistical values) of the reverse rate indicator channel for all active list mode segments. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return rri: float Comma-separated list of values, one per active segment. Range: -60 dB to +10 dB , Unit: dB

7.4.7.4.1.3 Average

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:CP:RRI:AVERage
CALCulate:EVDO:MEASurement<Instance>:MEValuation:LIST:CP:RRI:AVERage
```

class Average

Average commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

calculate() → List[RsCmwEvdoMeas.enums.ResultStatus2]

```
# SCPI: CALCulate:EVDO:MEASurement<instance>:MEValuation:LIST:CP:RRI:AVERage
value: List[enums.ResultStatus2] = driver.multiEval.listPy.cp.rri.average.
↪calculate()
```

Returns the RMS power (statistical values) of the reverse rate indicator channel for all active list mode segments. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return rpi_ch: float Comma-separated list of values, one per active segment. Range: -60 dB to +10 dB , Unit: dB

fetch() → List[float]

```
# SCPI: FETCh:EVDO:MEASurement<instance>:MEValuation:LIST:CP:RRI:AVERage
value: List[float] = driver.multiEval.listPy.cp.rri.average.fetch()
```

Returns the RMS power (statistical values) of the reverse rate indicator channel for all active list mode segments. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return rpi_ch: float Comma-separated list of values, one per active segment. Range: -60 dB to +10 dB , Unit: dB

7.4.7.4.1.4 Maximum

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:CP:RRI:MAXimum
CALCulate:EVDO:MEASurement<Instance>:MEValuation:LIST:CP:RRI:MAXimum
```

class Maximum

Maximum commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

calculate() → List[RsCmwEvdoMeas.enums.ResultStatus2]

```
# SCPI: CALCulate:EVDO:MEASurement<instance>:MEValuation:LIST:CP:RRI:MAXimum
value: List[enums.ResultStatus2] = driver.multiEval.listPy.cp.rri.maximum.
↪calculate()
```

Returns the RMS power (statistical values) of the reverse rate indicator channel for all active list mode segments. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return rri: float Comma-separated list of values, one per active segment. Range: -60 dB to +10 dB , Unit: dB

fetch() → List[float]

```
# SCPI: FETCh:EVDO:MEASurement<instance>:MEValuation:LIST:CP:RRI:MAXimum
value: List[float] = driver.multiEval.listPy.cp.rri.maximum.fetch()
```

Returns the RMS power (statistical values) of the reverse rate indicator channel for all active list mode segments. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return rri: float Comma-separated list of values, one per active segment. Range: -60 dB to +10 dB , Unit: dB

7.4.7.4.1.5 Minimum

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:CP:RRI:MINimum
CALCulate:EVDO:MEASurement<Instance>:MEValuation:LIST:CP:RRI:MINimum
```

class Minimum

Minimum commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

calculate() → List[RsCmwEvdoMeas.enums.ResultStatus2]

```
# SCPI: CALCulate:EVDO:MEASurement<instance>:MEValuation:LIST:CP:RRI:MINimum
value: List[enums.ResultStatus2] = driver.multiEval.listPy.cp.rri.minimum.
↪calculate()
```

Returns the RMS power (statistical values) of the reverse rate indicator channel for all active list mode segments. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return rri: float Comma-separated list of values, one per active segment. Range: -60 dB to +10 dB , Unit: dB

fetch() → List[float]

```
# SCPI: FETCh:EVDO:MEASurement<instance>:MEValuation:LIST:CP:RRI:MINimum
value: List[float] = driver.multiEval.listPy.cp.rri.minimum.fetch()
```

Returns the RMS power (statistical values) of the reverse rate indicator channel for all active list mode segments. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return rri: float Comma-separated list of values, one per active segment. Range: -60 dB to +10 dB , Unit: dB

7.4.7.4.2 Pilot

class Pilot

Pilot commands group definition. 9 total commands, 5 Sub-groups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.multiEval.listPy.cp.pilot.clone()
```

Subgroups

7.4.7.4.2.1 State

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:CP:PILot:STATe
```

class State

State commands group definition. 1 total commands, 0 Sub-groups, 1 group commands

fetch() → List[RsCmwEvdoMeas.enums.ResultStateA]

```
# SCPI: FETCh:EVDO:MEASurement<instance>:MEValuation:LIST:CP:PILot:STATe
value: List[enums.ResultStateA] = driver.multiEval.listPy.cp.pilot.state.fetch()
```

Returns the state of a particular reverse link channel (ACK/DSC, auxiliary pilot, data, DRC, PILOT, RRI) in the channel power measurement for all active segments.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return pilot: INVisible | ACTive | IACTive Comma-separated list of values, one per active segment.

7.4.7.4.2.2 Current

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:CP:PILot:CURRent
CALCulate:EVDO:MEASurement<Instance>:MEValuation:LIST:CP:PILot:CURRent
```

class Current

Current commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

calculate() → List[RsCmwEvdoMeas.enums.ResultStatus2]

```
# SCPI: CALCulate:EVDO:MEASurement<instance>:MEValuation:LIST:CP:PILot:CURRent
value: List[enums.ResultStatus2] = driver.multiEval.listPy.cp.pilot.current.
↪calculate()
```

Returns the RMS power (statistical values) of the pilot channel for all active list mode segments. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return pilot: float Comma-separated list of values, one per active segment. Range: -60 dB to +10 dB , Unit: dB

fetch() → List[float]

```
# SCPI: FETCh:EVDO:MEASurement<instance>:MEValuation:LIST:CP:PILot:CURRent
value: List[float] = driver.multiEval.listPy.cp.pilot.current.fetch()
```

Returns the RMS power (statistical values) of the pilot channel for all active list mode segments. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return pilot: float Comma-separated list of values, one per active segment. Range: -60 dB to +10 dB , Unit: dB

7.4.7.4.2.3 Average

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:CP:PILot:AVERage
CALCulate:EVDO:MEASurement<Instance>:MEValuation:LIST:CP:PILot:AVERage
```

class Average

Average commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

calculate() → List[RsCmwEvdoMeas.enums.ResultStatus2]

```
# SCPI: CALCulate:EVDO:MEASurement<instance>:MEValuation:LIST:CP:PILot:AVERage
value: List[enums.ResultStatus2] = driver.multiEval.listPy.cp.pilot.average.
↪calculate()
```

Returns the RMS power (statistical values) of the pilot channel for all active list mode segments. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return pilot: float Comma-separated list of values, one per active segment. Range: -60 dB to +10 dB , Unit: dB

fetch() → List[float]

```
# SCPI: FETCh:EVDO:MEASurement<instance>:MEValuation:LIST:CP:PILot:AVERage
value: List[float] = driver.multiEval.listPy.cp.pilot.average.fetch()
```

Returns the RMS power (statistical values) of the pilot channel for all active list mode segments. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return pilot: float Comma-separated list of values, one per active segment. Range: -60 dB to +10 dB , Unit: dB

7.4.7.4.2.4 Maximum

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:CP:PILot:MAXimum
CALCulate:EVDO:MEASurement<Instance>:MEValuation:LIST:CP:PILot:MAXimum
```

class Maximum

Maximum commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

calculate() → List[RsCmwEvdoMeas.enums.ResultStatus2]

```
# SCPI: CALCulate:EVDO:MEASurement<instance>:MEValuation:LIST:CP:PILot:MAXimum
value: List[enums.ResultStatus2] = driver.multiEval.listPy.cp.pilot.maximum.
↪calculate()
```

Returns the RMS power (statistical values) of the pilot channel for all active list mode segments. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return pilot: float Comma-separated list of values, one per active segment. Range: -60 dB to +10 dB , Unit: dB

fetch() → List[float]

```
# SCPI: FETCh:EVDO:MEASurement<instance>:MEValuation:LIST:CP:PILot:MAXimum
value: List[float] = driver.multiEval.listPy.cp.pilot.maximum.fetch()
```

Returns the RMS power (statistical values) of the pilot channel for all active list mode segments. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return pilot: float Comma-separated list of values, one per active segment. Range: -60 dB to +10 dB , Unit: dB

7.4.7.4.2.5 Minimum

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:CP:PILot:MINimum
CALCulate:EVDO:MEASurement<Instance>:MEValuation:LIST:CP:PILot:MINimum
```

class Minimum

Minimum commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

calculate() → List[RsCmwEvdoMeas.enums.ResultStatus2]

```
# SCPI: CALCulate:EVDO:MEASurement<instance>:MEValuation:LIST:CP:PILot:MINimum
value: List[enums.ResultStatus2] = driver.multiEval.listPy.cp.pilot.minimum.
↪calculate()
```

Returns the RMS power (statistical values) of the pilot channel for all active list mode segments. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return pilot: float Comma-separated list of values, one per active segment. Range: -60 dB to +10 dB , Unit: dB

fetch() → List[float]

```
# SCPI: FETCh:EVDO:MEASurement<instance>:MEValuation:LIST:CP:PILot:MINimum
value: List[float] = driver.multiEval.listPy.cp.pilot.minimum.fetch()
```

Returns the RMS power (statistical values) of the pilot channel for all active list mode segments. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return pilot: float Comma-separated list of values, one per active segment. Range: -60 dB to +10 dB , Unit: dB

7.4.7.4.3 Adsc

class Adsc

Adsc commands group definition. 9 total commands, 5 Sub-groups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.multiEval.listPy.cp.adsc.clone()
```

Subgroups

7.4.7.4.3.1 State

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:CP:ADSC:STATe
```

class State

State commands group definition. 1 total commands, 0 Sub-groups, 1 group commands

fetch() → List[RsCmwEvdoMeas.enums.ResultStateA]

```
# SCPI: FETCh:EVDO:MEASurement<instance>:MEValuation:LIST:CP:ADSC:STATe
value: List[enums.ResultStateA] = driver.multiEval.listPy.cp.adsc.state.fetch()
```

Returns the state of a particular reverse link channel (ACK/DSC, auxiliary pilot, data, DRC, PILOT, RRI) in the channel power measurement for all active segments.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return ack_dsc: INVisible | ACTive | IACTive Comma-separated list of values, one per active segment.

7.4.7.4.3.2 Current

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:CP:ADSC:CURRent
CALCulate:EVDO:MEASurement<Instance>:MEValuation:LIST:CP:ADSC:CURRent
```

class Current

Current commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

calculate() → List[RsCmwEvdoMeas.enums.ResultStatus2]

```
# SCPI: CALCulate:EVDO:MEASurement<instance>:MEValuation:LIST:CP:ADSC:CURRent
value: List[enums.ResultStatus2] = driver.multiEval.listPy.cp.adsc.current.
↪calculate()
```

Returns the RMS power (statistical values) of the acknowledgment / data source control channel for all active list mode segments. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return ack_dsc: float Comma-separated list of values, one per active segment. Range: -60 dB to +10 dB , Unit: dB

fetch() → List[float]

```
# SCPI: FETCh:EVDO:MEASurement<instance>:MEValuation:LIST:CP:ADSC:CURRent
value: List[float] = driver.multiEval.listPy.cp.adsc.current.fetch()
```

Returns the RMS power (statistical values) of the acknowledgment / data source control channel for all active list mode segments. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return ack_dsc: float Comma-separated list of values, one per active segment. Range: -60 dB to +10 dB , Unit: dB

7.4.7.4.3.3 Average

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:CP:ADSC:AVERage
CALCulate:EVDO:MEASurement<Instance>:MEValuation:LIST:CP:ADSC:AVERage
```

class Average

Average commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

calculate() → List[RsCmwEvdoMeas.enums.ResultStatus2]

```
# SCPI: CALCulate:EVDO:MEASurement<instance>:MEValuation:LIST:CP:ADSC:AVERage
value: List[enums.ResultStatus2] = driver.multiEval.listPy.cp.adsc.average.
↪calculate()
```

Returns the RMS power (statistical values) of the acknowledgment / data source control channel for all active list mode segments. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return ack_dsc: float Comma-separated list of values, one per active segment. Range: -60 dB to +10 dB , Unit: dB

fetch() → List[float]

```
# SCPI: FETCh:EVDO:MEASurement<instance>:MEValuation:LIST:CP:ADSC:AVERage
value: List[float] = driver.multiEval.listPy.cp.adsc.average.fetch()
```

Returns the RMS power (statistical values) of the acknowledgment / data source control channel for all active list mode segments. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return ack_dsc: float Comma-separated list of values, one per active segment. Range: -60 dB to +10 dB , Unit: dB

7.4.7.4.3.4 Maximum

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:CP:ADSC:MAXimum
CALCulate:EVDO:MEASurement<Instance>:MEValuation:LIST:CP:ADSC:MAXimum
```

class Maximum

Maximum commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

calculate() → List[RsCmwEvdoMeas.enums.ResultStatus2]

```
# SCPI: CALCulate:EVDO:MEASurement<instance>:MEValuation:LIST:CP:ADSC:MAXimum
value: List[enums.ResultStatus2] = driver.multiEval.listPy.cp.adsc.maximum.
↪calculate()
```

Returns the RMS power (statistical values) of the acknowledgment / data source control channel for all active list mode segments. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return ack_dsc: float Comma-separated list of values, one per active segment. Range: -60 dB to +10 dB , Unit: dB

fetch() → List[float]

```
# SCPI: FETCh:EVDO:MEASurement<instance>:MEValuation:LIST:CP:ADSC:MAXimum
value: List[float] = driver.multiEval.listPy.cp.adsc.maximum.fetch()
```

Returns the RMS power (statistical values) of the acknowledgment / data source control channel for all active list mode segments. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return ack_dsc: float Comma-separated list of values, one per active segment. Range: -60 dB to +10 dB , Unit: dB

7.4.7.4.3.5 Minimum

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:CP:ADSC:MINimum
CALCulate:EVDO:MEASurement<Instance>:MEValuation:LIST:CP:ADSC:MINimum
```

class Minimum

Minimum commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

calculate() → List[RsCmwEvdoMeas.enums.ResultStatus2]

```
# SCPI: CALCulate:EVDO:MEASurement<instance>:MEValuation:LIST:CP:ADSC:MINimum
value: List[enums.ResultStatus2] = driver.multiEval.listPy.cp.adsc.minimum.
↪calculate()
```

Returns the RMS power (statistical values) of the acknowledgment / data source control channel for all active list mode segments. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return ack_dsc: float Comma-separated list of values, one per active segment. Range: -60 dB to +10 dB , Unit: dB

fetch() → List[float]

```
# SCPI: FETCh:EVDO:MEASurement<instance>:MEValuation:LIST:CP:ADSC:MINimum
value: List[float] = driver.multiEval.listPy.cp.adsc.minimum.fetch()
```

Returns the RMS power (statistical values) of the acknowledgment / data source control channel for all active list mode segments. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return ack_dsc: float Comma-separated list of values, one per active segment. Range: -60 dB to +10 dB , Unit: dB

7.4.7.4.4 Apilot

class Apilot

Apilot commands group definition. 9 total commands, 5 Sub-groups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.multiEval.listPy.cp.apilot.clone()
```

Subgroups

7.4.7.4.4.1 State

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:CP:APILot:STATe
```

class State

State commands group definition. 1 total commands, 0 Sub-groups, 1 group commands

fetch() → List[RsCmwEvdoMeas.enums.ResultStateA]

```
# SCPI: FETCh:EVDO:MEASurement<instance>:MEValuation:LIST:CP:APILot:STATe
value: List[enums.ResultStateA] = driver.multiEval.listPy.cp.apilot.state.
↪fetch()
```

Returns the state of a particular reverse link channel (ACK/DSC, auxiliary pilot, data, DRC, PILOT, RRI) in the channel power measurement for all active segments.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return aux_pilot: INVisible | ACTive | IACTive Comma-separated list of values, one per active segment.

7.4.7.4.4.2 Current

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:CP:APILot:CURRent
CALCulate:EVDO:MEASurement<Instance>:MEValuation:LIST:CP:APILot:CURRent
```

class Current

Current commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

calculate() → List[RsCmwEvdoMeas.enums.ResultStatus2]

```
# SCPI: CALCulate:EVDO:MEASurement<instance>:MEValuation:LIST:CP:APILot:CURRent
value: List[enums.ResultStatus2] = driver.multiEval.listPy.cp.apilot.current.
↪calculate()
```

Returns the RMS power (statistical values) of the auxiliary pilot channel for all active list mode segments. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return aux_pilot: float Comma-separated list of values, one per active segment. Range: -60 dB to +10 dB , Unit: dB

fetch() → List[float]

```
# SCPI: FETCh:EVDO:MEASurement<instance>:MEValuation:LIST:CP:APILot:CURRent
value: List[float] = driver.multiEval.listPy.cp.apilot.current.fetch()
```

Returns the RMS power (statistical values) of the auxiliary pilot channel for all active list mode segments. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return aux_pilot: float Comma-separated list of values, one per active segment. Range: -60 dB to +10 dB , Unit: dB

7.4.7.4.4.3 Average

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:CP:APILot:AVERage
CALCulate:EVDO:MEASurement<Instance>:MEValuation:LIST:CP:APILot:AVERage
```

class Average

Average commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

calculate() → List[RsCmwEvdoMeas.enums.ResultStatus2]

```
# SCPI: CALCulate:EVDO:MEASurement<instance>:MEValuation:LIST:CP:APILot:AVERage
value: List[enums.ResultStatus2] = driver.multiEval.listPy.cp.apilot.average.
↪calculate()
```

Returns the RMS power (statistical values) of the auxiliary pilot channel for all active list mode segments. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return aux_pilot: float Comma-separated list of values, one per active segment. Range: -60 dB to +10 dB , Unit: dB

fetch() → List[float]

```
# SCPI: FETCh:EVDO:MEASurement<instance>:MEValuation:LIST:CP:APILot:AVERage
value: List[float] = driver.multiEval.listPy.cp.apilot.average.fetch()
```

Returns the RMS power (statistical values) of the auxiliary pilot channel for all active list mode segments. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return aux_pilot: float Comma-separated list of values, one per active segment. Range: -60 dB to +10 dB , Unit: dB

7.4.7.4.4.4 Maximum

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:CP:APILot:MAXimum
CALCulate:EVDO:MEASurement<Instance>:MEValuation:LIST:CP:APILot:MAXimum
```

class Maximum

Maximum commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

calculate() → List[RsCmwEvdoMeas.enums.ResultStatus2]

```
# SCPI: CALCulate:EVDO:MEASurement<instance>:MEValuation:LIST:CP:APILot:MAXimum
value: List[enums.ResultStatus2] = driver.multiEval.listPy.cp.apilot.maximum.
↪calculate()
```

Returns the RMS power (statistical values) of the auxiliary pilot channel for all active list mode segments. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return aux_pilot: float Comma-separated list of values, one per active segment. Range: -60 dB to +10 dB , Unit: dB

fetch() → List[float]

```
# SCPI: FETCh:EVDO:MEASurement<instance>:MEValuation:LIST:CP:APILot:MAXimum
value: List[float] = driver.multiEval.listPy.cp.apilot.maximum.fetch()
```

Returns the RMS power (statistical values) of the auxiliary pilot channel for all active list mode segments. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return aux_pilot: float Comma-separated list of values, one per active segment. Range: -60 dB to +10 dB , Unit: dB

7.4.7.4.4.5 Minimum

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:CP:APILot:MINimum
CALCulate:EVDO:MEASurement<Instance>:MEValuation:LIST:CP:APILot:MINimum
```

class Minimum

Minimum commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

calculate() → List[RsCmwEvdoMeas.enums.ResultStatus2]

```
# SCPI: CALCulate:EVDO:MEASurement<instance>:MEValuation:LIST:CP:APILot:MINimum
value: List[enums.ResultStatus2] = driver.multiEval.listPy.cp.apilot.minimum.
↪calculate()
```

Returns the RMS power (statistical values) of the auxiliary pilot channel for all active list mode segments. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return aux_pilot: float Comma-separated list of values, one per active segment. Range: -60 dB to +10 dB , Unit: dB

fetch() → List[float]

```
# SCPI: FETCh:EVDO:MEASurement<instance>:MEValuation:LIST:CP:APILot:MINimum
value: List[float] = driver.multiEval.listPy.cp.apilot.minimum.fetch()
```

Returns the RMS power (statistical values) of the auxiliary pilot channel for all active list mode segments. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return aux_pilot: float Comma-separated list of values, one per active segment. Range: -60 dB to +10 dB , Unit: dB

7.4.7.4.5 Drc

class Drc
Drc commands group definition. 9 total commands, 5 Sub-groups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.multiEval.listPy.cp.drc.clone()
```

Subgroups

7.4.7.4.5.1 State

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:CP:DRC:STATe
```

class State
State commands group definition. 1 total commands, 0 Sub-groups, 1 group commands

fetch() → List[RsCmwEvdoMeas.enums.ResultStateA]

```
# SCPI: FETCh:EVDO:MEASurement<instance>:MEValuation:LIST:CP:DRC:STATe
value: List[enums.ResultStateA] = driver.multiEval.listPy.cp.drc.state.fetch()
```

Returns the state of a particular reverse link channel (ACK/DSC, auxiliary pilot, data, DRC, PILOT, RRI) in the channel power measurement for all active segments.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return drc: INVisible | ACTive | IACTive Comma-separated list of values, one per active segment.

7.4.7.4.5.2 Current

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:CP:DRC:CURRent
CALCulate:EVDO:MEASurement<Instance>:MEValuation:LIST:CP:DRC:CURRent
```

class Current
Current commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

calculate() → List[RsCmwEvdoMeas.enums.ResultStatus2]

```
# SCPI: CALCulate:EVDO:MEASurement<instance>:MEValuation:LIST:CP:DRC:CURRent
value: List[enums.ResultStatus2] = driver.multiEval.listPy.cp.drc.current.
↪calculate()
```

Returns the RMS power (statistical values) of the data rate control channel for all active list mode segments. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return drc: float Comma-separated list of values, one per active segment. Range: -60 dB to +10 dB , Unit: dB

fetch() → List[float]

```
# SCPI: FETCh:EVDO:MEASurement<instance>:MEValuation:LIST:CP:DRC:CURRent
value: List[float] = driver.multiEval.listPy.cp.drc.current.fetch()
```

Returns the RMS power (statistical values) of the data rate control channel for all active list mode segments. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return drc: float Comma-separated list of values, one per active segment. Range: -60 dB to +10 dB , Unit: dB

7.4.7.4.5.3 Average

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:CP:DRC:AVERage
CALCulate:EVDO:MEASurement<Instance>:MEValuation:LIST:CP:DRC:AVERage
```

class Average

Average commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

calculate() → List[RsCmwEvdoMeas.enums.ResultStatus2]

```
# SCPI: CALCulate:EVDO:MEASurement<instance>:MEValuation:LIST:CP:DRC:AVERage
value: List[enums.ResultStatus2] = driver.multiEval.listPy.cp.drc.average.
↪calculate()
```

Returns the RMS power (statistical values) of the data rate control channel for all active list mode segments. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return drc: float Comma-separated list of values, one per active segment. Range: -60 dB to +10 dB , Unit: dB

fetch() → List[float]

```
# SCPI: FETCh:EVDO:MEASurement<instance>:MEValuation:LIST:CP:DRC:AVERage
value: List[float] = driver.multiEval.listPy.cp.drc.average.fetch()
```

Returns the RMS power (statistical values) of the data rate control channel for all active list mode segments. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return drc: float Comma-separated list of values, one per active segment. Range: -60 dB to +10 dB , Unit: dB

7.4.7.4.5.4 Maximum

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:CP:DRC:MAXimum
CALCulate:EVDO:MEASurement<Instance>:MEValuation:LIST:CP:DRC:MAXimum
```

class Maximum

Maximum commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

calculate() → List[RsCmwEvdoMeas.enums.ResultStatus2]

```
# SCPI: CALCulate:EVDO:MEASurement<instance>:MEValuation:LIST:CP:DRC:MAXimum
value: List[enums.ResultStatus2] = driver.multiEval.listPy.cp.drc.maximum.
↪calculate()
```

Returns the RMS power (statistical values) of the data rate control channel for all active list mode segments. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return drc: float Comma-separated list of values, one per active segment. Range: -60 dB to +10 dB , Unit: dB

fetch() → List[float]

```
# SCPI: FETCh:EVDO:MEASurement<instance>:MEValuation:LIST:CP:DRC:MAXimum
value: List[float] = driver.multiEval.listPy.cp.drc.maximum.fetch()
```

Returns the RMS power (statistical values) of the data rate control channel for all active list mode segments. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return drc: float Comma-separated list of values, one per active segment. Range: -60 dB to +10 dB , Unit: dB

7.4.7.4.5.5 Minimum

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:CP:DRC:MINimum
CALCulate:EVDO:MEASurement<Instance>:MEValuation:LIST:CP:DRC:MINimum
```

class Minimum

Minimum commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

calculate() → List[RsCmwEvdoMeas.enums.ResultStatus2]

```
# SCPI: CALCulate:EVDO:MEASurement<instance>:MEValuation:LIST:CP:DRC:MINimum
value: List[enums.ResultStatus2] = driver.multiEval.listPy.cp.drc.minimum.
↪calculate()
```

Returns the RMS power (statistical values) of the data rate control channel for all active list mode segments. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return drc: float Comma-separated list of values, one per active segment. Range: -60 dB to +10 dB , Unit: dB

fetch() → List[float]

```
# SCPI: FETCh:EVDO:MEASurement<instance>:MEValuation:LIST:CP:DRC:MINimum
value: List[float] = driver.multiEval.listPy.cp.drc.minimum.fetch()
```

Returns the RMS power (statistical values) of the data rate control channel for all active list mode segments. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return drc: float Comma-separated list of values, one per active segment. Range: -60 dB to +10 dB , Unit: dB

7.4.7.4.6 Data

class Data

Data commands group definition. 9 total commands, 5 Sub-groups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.multiEval.listPy.cp.data.clone()
```

Subgroups

7.4.7.4.6.1 State

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:CP:DATA:STATe
```

class State

State commands group definition. 1 total commands, 0 Sub-groups, 1 group commands

fetch() → List[RsCmwEvdoMeas.enums.ResultStateA]

```
# SCPI: FETCh:EVDO:MEASurement<instance>:MEValuation:LIST:CP:DATA:STATe
value: List[enums.ResultStateA] = driver.multiEval.listPy.cp.data.state.fetch()
```

Returns the state of a particular reverse link channel (ACK/DSC, auxiliary pilot, data, DRC, PILOT, RRI) in the channel power measurement for all active segments.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return data: INVisible | ACTive | IACTive Comma-separated list of values, one per active segment.

7.4.7.4.6.2 Current

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:CP:DATA:CURRent
CALCulate:EVDO:MEASurement<Instance>:MEValuation:LIST:CP:DATA:CURRent
```

class Current

Current commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

calculate() → List[RsCmwEvdoMeas.enums.ResultStatus2]

```
# SCPI: CALCulate:EVDO:MEASurement<instance>:MEValuation:LIST:CP:DATA:CURRent
value: List[enums.ResultStatus2] = driver.multiEval.listPy.cp.data.current.
↪calculate()
```

Returns the RMS power (statistical values) of the data channel for all active list mode segments. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return data: float Comma-separated list of values, one per active segment. Range: -60 dB to +10 dB , Unit: dB

fetch() → List[float]

```
# SCPI: FETCh:EVDO:MEASurement<instance>:MEValuation:LIST:CP:DATA:CURRent
value: List[float] = driver.multiEval.listPy.cp.data.current.fetch()
```

Returns the RMS power (statistical values) of the data channel for all active list mode segments. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return data: float Comma-separated list of values, one per active segment. Range: -60 dB to +10 dB , Unit: dB

7.4.7.4.6.3 Average

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:CP:DATA:AVERage
CALCulate:EVDO:MEASurement<Instance>:MEValuation:LIST:CP:DATA:AVERage
```

class Average

Average commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

calculate() → List[RsCmwEvdoMeas.enums.ResultStatus2]

```
# SCPI: CALCulate:EVDO:MEASurement<instance>:MEValuation:LIST:CP:DATA:AVERage
value: List[enums.ResultStatus2] = driver.multiEval.listPy.cp.data.average.
↪calculate()
```

Returns the RMS power (statistical values) of the data channel for all active list mode segments. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return data: float Comma-separated list of values, one per active segment. Range: -60 dB to +10 dB , Unit: dB

fetch() → List[float]

```
# SCPI: FETCh:EVDO:MEASurement<instance>:MEValuation:LIST:CP:DATA:AVERage
value: List[float] = driver.multiEval.listPy.cp.data.average.fetch()
```

Returns the RMS power (statistical values) of the data channel for all active list mode segments. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return data: float Comma-separated list of values, one per active segment. Range: -60 dB to +10 dB , Unit: dB

7.4.7.4.6.4 Maximum

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:CP:DATA:MAXimum
CALCulate:EVDO:MEASurement<Instance>:MEValuation:LIST:CP:DATA:MAXimum
```

class Maximum

Maximum commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

calculate() → List[RsCmwEvdoMeas.enums.ResultStatus2]

```
# SCPI: CALCulate:EVDO:MEASurement<instance>:MEValuation:LIST:CP:DATA:MAXimum
value: List[enums.ResultStatus2] = driver.multiEval.listPy.cp.data.maximum.
↪calculate()
```

Returns the RMS power (statistical values) of the data channel for all active list mode segments. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return data: float Comma-separated list of values, one per active segment. Range: -60 dB to +10 dB , Unit: dB

fetch() → List[float]

```
# SCPI: FETCh:EVDO:MEASurement<instance>:MEValuation:LIST:CP:DATA:MAXimum
value: List[float] = driver.multiEval.listPy.cp.data.maximum.fetch()
```

Returns the RMS power (statistical values) of the data channel for all active list mode segments. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return data: float Comma-separated list of values, one per active segment. Range: -60 dB to +10 dB , Unit: dB

7.4.7.4.6.5 Minimum

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:CP:DATA:MINimum
CALCulate:EVDO:MEASurement<Instance>:MEValuation:LIST:CP:DATA:MINimum
```

class Minimum

Minimum commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

calculate() → List[RsCmwEvdoMeas.enums.ResultStatus2]

```
# SCPI: CALCulate:EVDO:MEASurement<instance>:MEValuation:LIST:CP:DATA:MINimum
value: List[enums.ResultStatus2] = driver.multiEval.listPy.cp.data.minimum.
↪calculate()
```

Returns the RMS power (statistical values) of the data channel for all active list mode segments. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return data: float Comma-separated list of values, one per active segment. Range: -60 dB to +10 dB , Unit: dB

fetch() → List[float]

```
# SCPI: FETCh:EVDO:MEASurement<instance>:MEValuation:LIST:CP:DATA:MINimum
value: List[float] = driver.multiEval.listPy.cp.data.minimum.fetch()
```

Returns the RMS power (statistical values) of the data channel for all active list mode segments. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return data: float Comma-separated list of values, one per active segment. Range: -60 dB to +10 dB , Unit: dB

7.4.7.4.7 Current

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:CP:CURRent
CALCulate:EVDO:MEASurement<Instance>:MEValuation:LIST:CP:CURRent
```

class Current

Current commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

class CalculateStruct

Response structure. Fields:

- Reliabiltiy: int: decimal 'Conventions and General Information'. In list mode, a zero reliability indicator indicates that the results in all measured segments are valid. A non-zero value indicates that an error occurred in at least one of the measured segments.
- Seg_Reliability: List[int]: 0 | 3 | 4 | 8 The segment reliability indicates whether one of the following exceptions occurred in this segment: 0: No error 3: Signal overflow 4: Signal low 8: Synchronization error If a combination of exceptions occurs, the most severe error is indicated.
- Rri: List[enums.ResultStatus2]: float RMS channel power values for the physical channels. Depending on the subtype 0/1 or 2, some channels exist or not. If not available, no power measurement is possible therefore NAV is returned. Range: -60 dB to +10 dB , Unit: dB
- Pilot: List[enums.ResultStatus2]: float RMS channel power values for the physical channels. Depending on the subtype 0/1 or 2, some channels exist or not. If not available, no power measurement is possible therefore NAV is returned. Range: -60 dB to +10 dB , Unit: dB

- Ack_Dsc: List[enums.ResultStatus2]: float RMS channel power values for the physical channels. Depending on the subtype 0/1 or 2, some channels exist or not. If not available, no power measurement is possible therefore NAV is returned. Range: -60 dB to +10 dB , Unit: dB
- Aux_Pilot: List[enums.ResultStatus2]: float RMS channel power values for the physical channels. Depending on the subtype 0/1 or 2, some channels exist or not. If not available, no power measurement is possible therefore NAV is returned. Range: -60 dB to +10 dB , Unit: dB
- Drc: List[enums.ResultStatus2]: float RMS channel power values for the physical channels. Depending on the subtype 0/1 or 2, some channels exist or not. If not available, no power measurement is possible therefore NAV is returned. Range: -60 dB to +10 dB , Unit: dB
- Data: List[enums.ResultStatus2]: float RMS channel power values for the physical channels. Depending on the subtype 0/1 or 2, some channels exist or not. If not available, no power measurement is possible therefore NAV is returned. Range: -60 dB to +10 dB , Unit: dB

class FetchStruct

Response structure. Fields:

- Reliabiltiy: int: decimal 'Conventions and General Information'. In list mode, a zero reliability indicator indicates that the results in all measured segments are valid. A non-zero value indicates that an error occurred in at least one of the measured segments.
- Seg_Reliability: List[int]: 0 | 3 | 4 | 8 The segment reliability indicates whether one of the following exceptions occurred in this segment: 0: No error 3: Signal overflow 4: Signal low 8: Synchronization error If a combination of exceptions occurs, the most severe error is indicated.
- Rri: List[float]: float RMS channel power values for the physical channels. Depending on the subtype 0/1 or 2, some channels exist or not. If not available, no power measurement is possible therefore NAV is returned. Range: -60 dB to +10 dB , Unit: dB
- Pilot: List[float]: float RMS channel power values for the physical channels. Depending on the subtype 0/1 or 2, some channels exist or not. If not available, no power measurement is possible therefore NAV is returned. Range: -60 dB to +10 dB , Unit: dB
- Ack_Dsc: List[float]: float RMS channel power values for the physical channels. Depending on the subtype 0/1 or 2, some channels exist or not. If not available, no power measurement is possible therefore NAV is returned. Range: -60 dB to +10 dB , Unit: dB
- Aux_Pilot: List[float]: float RMS channel power values for the physical channels. Depending on the subtype 0/1 or 2, some channels exist or not. If not available, no power measurement is possible therefore NAV is returned. Range: -60 dB to +10 dB , Unit: dB
- Drc: List[float]: float RMS channel power values for the physical channels. Depending on the subtype 0/1 or 2, some channels exist or not. If not available, no power measurement is possible therefore NAV is returned. Range: -60 dB to +10 dB , Unit: dB
- Data: List[float]: float RMS channel power values for the physical channels. Depending on the subtype 0/1 or 2, some channels exist or not. If not available, no power measurement is possible therefore NAV is returned. Range: -60 dB to +10 dB , Unit: dB

calculate() → CalculateStruct

```
# SCPI: CALCulate:EVDO:MEASurement<instance>:MEValuation:LIST:CP:CURRent
value: CalculateStruct = driver.multiEval.listPy.cp.current.calculate()
```

Returns the channel power (CP) value results in list mode. To define the statistical length for AVERage, MAXimum, MINimum calculation and enable the calculation of the results use the command method RsCmwEvdoMeas.Configure.MultiEval.ListPy. Segment.Modulation.set. The values listed below in curly brackets {} are returned for each active segment: {...}seg 1, {.. .}seg 2, ..., {...}seg n. The number of

active segments n is determined by method RsCmwEvdoMeas.Configure.MultiEval. ListPy.count. 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:EVDO:MEASurement<instance>:MEValuation:LIST:CP:CURRent
value: FetchStruct = driver.multiEval.listPy.cp.current.fetch()
```

Returns the channel power (CP) value results in list mode. To define the statistical length for AVERage, MAXimum, MINimum calculation and enable the calculation of the results use the command method RsCmwEvdoMeas.Configure.MultiEval.ListPy. Segment.Modulation.set. The values listed below in curly brackets {} are returned for each active segment: {...}seg 1, {.. .}seg 2, ..., {...}seg n. The number of active segments n is determined by method RsCmwEvdoMeas.Configure.MultiEval. ListPy.count. 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.

7.4.7.4.8 Average

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:CP:AVERage
CALCulate:EVDO:MEASurement<Instance>:MEValuation:LIST:CP:AVERage
```

class Average

Average commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

class CalculateStruct

Response structure. Fields:

- Reliabiltiy: int: decimal 'Conventions and General Information'. In list mode, a zero reliability indicator indicates that the results in all measured segments are valid. A non-zero value indicates that an error occurred in at least one of the measured segments.
- Seg_Reliability: List[int]: 0 | 3 | 4 | 8 The segment reliability indicates whether one of the following exceptions occurred in this segment: 0: No error 3: Signal overflow 4: Signal low 8: Synchronization error If a combination of exceptions occurs, the most severe error is indicated.
- Rri: List[enums.ResultStatus2]: float RMS channel power values for the physical channels. Depending on the subtype 0/1 or 2, some channels exist or not. If not available, no power measurement is possible therefore NAV is returned. Range: -60 dB to +10 dB , Unit: dB
- Pilot: List[enums.ResultStatus2]: float RMS channel power values for the physical channels. Depending on the subtype 0/1 or 2, some channels exist or not. If not available, no power measurement is possible therefore NAV is returned. Range: -60 dB to +10 dB , Unit: dB
- Ack_Dsc: List[enums.ResultStatus2]: float RMS channel power values for the physical channels. Depending on the subtype 0/1 or 2, some channels exist or not. If not available, no power measurement is possible therefore NAV is returned. Range: -60 dB to +10 dB , Unit: dB
- Aux_Pilot: List[enums.ResultStatus2]: float RMS channel power values for the physical channels. Depending on the subtype 0/1 or 2, some channels exist or not. If not available, no power measurement is possible therefore NAV is returned. Range: -60 dB to +10 dB , Unit: dB

- Drc: List[enums.ResultStatus2]: float RMS channel power values for the physical channels. Depending on the subtype 0/1 or 2, some channels exist or not. If not available, no power measurement is possible therefore NAV is returned. Range: -60 dB to +10 dB , Unit: dB
- Data: List[enums.ResultStatus2]: float RMS channel power values for the physical channels. Depending on the subtype 0/1 or 2, some channels exist or not. If not available, no power measurement is possible therefore NAV is returned. Range: -60 dB to +10 dB , Unit: dB

class FetchStruct

Response structure. Fields:

- Reliabiltiy: int: decimal 'Conventions and General Information'. In list mode, a zero reliability indicator indicates that the results in all measured segments are valid. A non-zero value indicates that an error occurred in at least one of the measured segments.
- Seg_Reliability: List[int]: 0 | 3 | 4 | 8 The segment reliability indicates whether one of the following exceptions occurred in this segment: 0: No error 3: Signal overflow 4: Signal low 8: Synchronization error If a combination of exceptions occurs, the most severe error is indicated.
- Rri: List[float]: float RMS channel power values for the physical channels. Depending on the subtype 0/1 or 2, some channels exist or not. If not available, no power measurement is possible therefore NAV is returned. Range: -60 dB to +10 dB , Unit: dB
- Pilot: List[float]: float RMS channel power values for the physical channels. Depending on the subtype 0/1 or 2, some channels exist or not. If not available, no power measurement is possible therefore NAV is returned. Range: -60 dB to +10 dB , Unit: dB
- Ack_Dsc: List[float]: float RMS channel power values for the physical channels. Depending on the subtype 0/1 or 2, some channels exist or not. If not available, no power measurement is possible therefore NAV is returned. Range: -60 dB to +10 dB , Unit: dB
- Aux_Pilot: List[float]: float RMS channel power values for the physical channels. Depending on the subtype 0/1 or 2, some channels exist or not. If not available, no power measurement is possible therefore NAV is returned. Range: -60 dB to +10 dB , Unit: dB
- Drc: List[float]: float RMS channel power values for the physical channels. Depending on the subtype 0/1 or 2, some channels exist or not. If not available, no power measurement is possible therefore NAV is returned. Range: -60 dB to +10 dB , Unit: dB
- Data: List[float]: float RMS channel power values for the physical channels. Depending on the subtype 0/1 or 2, some channels exist or not. If not available, no power measurement is possible therefore NAV is returned. Range: -60 dB to +10 dB , Unit: dB

calculate() → CalculateStruct

```
# SCPI: CALCulate:EVDO:MEASurement<instance>:MEValuation:LIST:CP:AVERage
value: CalculateStruct = driver.multiEval.listPy.cp.average.calculate()
```

Returns the channel power (CP) value results in list mode. To define the statistical length for AVERage, MAXimum, MINimum calculation and enable the calculation of the results use the command method RsCmwEvdoMeas.Configure.MultiEval.ListPy. Segment.Modulation.set. The values listed below in curly brackets {} are returned for each active segment: {...}seg 1, {.. .}seg 2, ..., {...}seg n. The number of active segments n is determined by method RsCmwEvdoMeas.Configure.MultiEval. ListPy.count. 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:EVDO:MEASurement<instance>:MEValuation:LIST:CP:AVERage
value: FetchStruct = driver.multiEval.listPy.cp.average.fetch()
```

Returns the channel power (CP) value results in list mode. To define the statistical length for AVERage, MAXimum, MINimum calculation and enable the calculation of the results use the command method RsCmwEvdoMeas.Configure.MultiEval.ListPy. Segment.Modulation.set. The values listed below in curly brackets {} are returned for each active segment: {...}seg 1, {.. .}seg 2, ..., {...}seg n. The number of active segments n is determined by method RsCmwEvdoMeas.Configure.MultiEval. ListPy.count. 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.

7.4.7.4.9 Maximum

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:CP:MAXimum
CALCulate:EVDO:MEASurement<Instance>:MEValuation:LIST:CP:MAXimum
```

class Maximum

Maximum commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

class CalculateStruct

Response structure. Fields:

- Reliabiltiy: int: decimal 'Conventions and General Information'. In list mode, a zero reliability indicator indicates that the results in all measured segments are valid. A non-zero value indicates that an error occurred in at least one of the measured segments.
- Seg_Reliability: List[int]: 0 | 3 | 4 | 8 The segment reliability indicates whether one of the following exceptions occurred in this segment: 0: No error 3: Signal overflow 4: Signal low 8: Synchronization error If a combination of exceptions occurs, the most severe error is indicated.
- Rri: List[enums.ResultStatus2]: float RMS channel power values for the physical channels. Depending on the subtype 0/1 or 2, some channels exist or not. If not available, no power measurement is possible therefore NAV is returned. Range: -60 dB to +10 dB , Unit: dB
- Pilot: List[enums.ResultStatus2]: float RMS channel power values for the physical channels. Depending on the subtype 0/1 or 2, some channels exist or not. If not available, no power measurement is possible therefore NAV is returned. Range: -60 dB to +10 dB , Unit: dB
- Ack_Dsc: List[enums.ResultStatus2]: float RMS channel power values for the physical channels. Depending on the subtype 0/1 or 2, some channels exist or not. If not available, no power measurement is possible therefore NAV is returned. Range: -60 dB to +10 dB , Unit: dB
- Aux_Pilot: List[enums.ResultStatus2]: float RMS channel power values for the physical channels. Depending on the subtype 0/1 or 2, some channels exist or not. If not available, no power measurement is possible therefore NAV is returned. Range: -60 dB to +10 dB , Unit: dB
- Drc: List[enums.ResultStatus2]: float RMS channel power values for the physical channels. Depending on the subtype 0/1 or 2, some channels exist or not. If not available, no power measurement is possible therefore NAV is returned. Range: -60 dB to +10 dB , Unit: dB
- Data: List[enums.ResultStatus2]: float RMS channel power values for the physical channels. Depending on the subtype 0/1 or 2, some channels exist or not. If not available, no power measurement is possible therefore NAV is returned. Range: -60 dB to +10 dB , Unit: dB

class FetchStruct
Response structure. Fields:

- Reliabiltiy: int: decimal 'Conventions and General Information'. In list mode, a zero reliability indicator indicates that the results in all measured segments are valid. A non-zero value indicates that an error occurred in at least one of the measured segments.
- Seg_Reliability: List[int]: 0 | 3 | 4 | 8 The segment reliability indicates whether one of the following exceptions occurred in this segment: 0: No error 3: Signal overflow 4: Signal low 8: Synchronization error If a combination of exceptions occurs, the most severe error is indicated.
- Rri: List[float]: float RMS channel power values for the physical channels. Depending on the subtype 0/1 or 2, some channels exist or not. If not available, no power measurement is possible therefore NAV is returned. Range: -60 dB to +10 dB , Unit: dB
- Pilot: List[float]: float RMS channel power values for the physical channels. Depending on the subtype 0/1 or 2, some channels exist or not. If not available, no power measurement is possible therefore NAV is returned. Range: -60 dB to +10 dB , Unit: dB
- Ack_Dsc: List[float]: float RMS channel power values for the physical channels. Depending on the subtype 0/1 or 2, some channels exist or not. If not available, no power measurement is possible therefore NAV is returned. Range: -60 dB to +10 dB , Unit: dB
- Aux_Pilot: List[float]: float RMS channel power values for the physical channels. Depending on the subtype 0/1 or 2, some channels exist or not. If not available, no power measurement is possible therefore NAV is returned. Range: -60 dB to +10 dB , Unit: dB
- Drc: List[float]: float RMS channel power values for the physical channels. Depending on the subtype 0/1 or 2, some channels exist or not. If not available, no power measurement is possible therefore NAV is returned. Range: -60 dB to +10 dB , Unit: dB
- Data: List[float]: float RMS channel power values for the physical channels. Depending on the subtype 0/1 or 2, some channels exist or not. If not available, no power measurement is possible therefore NAV is returned. Range: -60 dB to +10 dB , Unit: dB

calculate() → CalculateStruct

```
# SCPI: CALCulate:EVDO:MEASurement<instance>:MEValuation:LIST:CP:MAXimum
value: CalculateStruct = driver.multiEval.listPy.cp.maximum.calculate()
```

Returns the channel power (CP) value results in list mode. To define the statistical length for AVERage, MAXimum, MINimum calculation and enable the calculation of the results use the command method RsCmwEvdoMeas.Configure.MultiEval.ListPy. Segment.Modulation.set. The values listed below in curly brackets {} are returned for each active segment: {...}seg 1, {.. .}seg 2, ..., {...}seg n. The number of active segments n is determined by method RsCmwEvdoMeas.Configure.MultiEval. ListPy.count. 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:EVDO:MEASurement<instance>:MEValuation:LIST:CP:MAXimum
value: FetchStruct = driver.multiEval.listPy.cp.maximum.fetch()
```

Returns the channel power (CP) value results in list mode. To define the statistical length for AVERage, MAXimum, MINimum calculation and enable the calculation of the results use the command method RsCmwEvdoMeas.Configure.MultiEval.ListPy. Segment.Modulation.set. The values listed below in curly

brackets {} are returned for each active segment: {...}seg 1, {.. .}seg 2, ..., {...}seg n. The number of active segments n is determined by method RsCmwEvdoMeas.Configure.MultiEval. ListPy.count. 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.

7.4.7.4.10 Minimum

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:CP:MINimum
CALCulate:EVDO:MEASurement<Instance>:MEValuation:LIST:CP:MINimum
```

class Minimum

Minimum commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

class CalculateStruct

Response structure. Fields:

- Reliabiltiy: int: decimal 'Conventions and General Information'. In list mode, a zero reliability indicator indicates that the results in all measured segments are valid. A non-zero value indicates that an error occurred in at least one of the measured segments.
- Seg_Reliability: List[int]: 0 | 3 | 4 | 8 The segment reliability indicates whether one of the following exceptions occurred in this segment: 0: No error 3: Signal overflow 4: Signal low 8: Synchronization error If a combination of exceptions occurs, the most severe error is indicated.
- Rri: List[enums.ResultStatus2]: float RMS channel power values for the physical channels. Depending on the subtype 0/1 or 2, some channels exist or not. If not available, no power measurement is possible therefore NAV is returned. Range: -60 dB to +10 dB , Unit: dB
- Pilot: List[enums.ResultStatus2]: float RMS channel power values for the physical channels. Depending on the subtype 0/1 or 2, some channels exist or not. If not available, no power measurement is possible therefore NAV is returned. Range: -60 dB to +10 dB , Unit: dB
- Ack_Dsc: List[enums.ResultStatus2]: float RMS channel power values for the physical channels. Depending on the subtype 0/1 or 2, some channels exist or not. If not available, no power measurement is possible therefore NAV is returned. Range: -60 dB to +10 dB , Unit: dB
- Aux_Pilot: List[enums.ResultStatus2]: float RMS channel power values for the physical channels. Depending on the subtype 0/1 or 2, some channels exist or not. If not available, no power measurement is possible therefore NAV is returned. Range: -60 dB to +10 dB , Unit: dB
- Drc: List[enums.ResultStatus2]: float RMS channel power values for the physical channels. Depending on the subtype 0/1 or 2, some channels exist or not. If not available, no power measurement is possible therefore NAV is returned. Range: -60 dB to +10 dB , Unit: dB
- Data: List[enums.ResultStatus2]: float RMS channel power values for the physical channels. Depending on the subtype 0/1 or 2, some channels exist or not. If not available, no power measurement is possible therefore NAV is returned. Range: -60 dB to +10 dB , Unit: dB

class FetchStruct

Response structure. Fields:

- Reliabiltiy: int: decimal 'Conventions and General Information'. In list mode, a zero reliability indicator indicates that the results in all measured segments are valid. A non-zero value indicates that an error occurred in at least one of the measured segments.

- Seg_Reliability: List[int]: 0 | 3 | 4 | 8 The segment reliability indicates whether one of the following exceptions occurred in this segment: 0: No error 3: Signal overflow 4: Signal low 8: Synchronization error If a combination of exceptions occurs, the most severe error is indicated.
- Rri: List[float]: float RMS channel power values for the physical channels. Depending on the subtype 0/1 or 2, some channels exist or not. If not available, no power measurement is possible therefore NAV is returned. Range: -60 dB to +10 dB , Unit: dB
- Pilot: List[float]: float RMS channel power values for the physical channels. Depending on the subtype 0/1 or 2, some channels exist or not. If not available, no power measurement is possible therefore NAV is returned. Range: -60 dB to +10 dB , Unit: dB
- Ack_Dsc: List[float]: float RMS channel power values for the physical channels. Depending on the subtype 0/1 or 2, some channels exist or not. If not available, no power measurement is possible therefore NAV is returned. Range: -60 dB to +10 dB , Unit: dB
- Aux_Pilot: List[float]: float RMS channel power values for the physical channels. Depending on the subtype 0/1 or 2, some channels exist or not. If not available, no power measurement is possible therefore NAV is returned. Range: -60 dB to +10 dB , Unit: dB
- Drc: List[float]: float RMS channel power values for the physical channels. Depending on the subtype 0/1 or 2, some channels exist or not. If not available, no power measurement is possible therefore NAV is returned. Range: -60 dB to +10 dB , Unit: dB
- Data: List[float]: float RMS channel power values for the physical channels. Depending on the subtype 0/1 or 2, some channels exist or not. If not available, no power measurement is possible therefore NAV is returned. Range: -60 dB to +10 dB , Unit: dB

calculate() → CalculateStruct

```
# SCPI: CALCulate:EVDO:MEASurement<instance>:MEValuation:LIST:CP:MINimum
value: CalculateStruct = driver.multiEval.listPy.cp.minimum.calculate()
```

Returns the channel power (CP) value results in list mode. To define the statistical length for AVERage, MAXimum, MINimum calculation and enable the calculation of the results use the command method RsCmwEvdoMeas.Configure.MultiEval.ListPy. Segment.Modulation.set. The values listed below in curly brackets {} are returned for each active segment: {...}seg 1, {.. .}seg 2, ..., {...}seg n. The number of active segments n is determined by method RsCmwEvdoMeas.Configure.MultiEval. ListPy.count. 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:EVDO:MEASurement<instance>:MEValuation:LIST:CP:MINimum
value: FetchStruct = driver.multiEval.listPy.cp.minimum.fetch()
```

Returns the channel power (CP) value results in list mode. To define the statistical length for AVERage, MAXimum, MINimum calculation and enable the calculation of the results use the command method RsCmwEvdoMeas.Configure.MultiEval.ListPy. Segment.Modulation.set. The values listed below in curly brackets {} are returned for each active segment: {...}seg 1, {.. .}seg 2, ..., {...}seg n. The number of active segments n is determined by method RsCmwEvdoMeas.Configure.MultiEval. ListPy.count. 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.

7.4.7.4.11 State

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:CP:STATe
```

class State

State commands group definition. 1 total commands, 0 Sub-groups, 1 group commands

class FetchStruct

Response structure. Fields:

- Reliabiltiy: int: decimal 'Conventions and General Information'. In list mode, a zero reliability indicator indicates that the results in all measured segments are valid. A non-zero value indicates that an error occurred in at least one of the measured segments.
- Seg_Reliability: List[int]: 0 | 3 | 4 | 8 The segment reliability indicates whether one of the following exceptions occurred in this segment: 0: No error 3: Signal overflow 4: Signal low 8: Synchronization error If a combination of exceptions occurs, the most severe error is indicated.
- Rri: List[enums.ResultStateA]: INVisible | ACTive | IACTive 'INV': No channel available 'ACTive': Active channel 'IACtive': Inactive channel
- Pilot: List[enums.ResultStateA]: INVisible | ACTive | IACTive 'INV': No channel available 'ACTive': Active channel 'IACtive': Inactive channel
- Ack_Dsc: List[enums.ResultStateA]: INVisible | ACTive | IACTive 'INV': No channel available 'ACTive': Active channel 'IACtive': Inactive channel
- Aux_Pilot: List[enums.ResultStateA]: INVisible | ACTive | IACTive 'INV': No channel available 'ACTive': Active channel 'IACtive': Inactive channel
- Drc: List[enums.ResultStateA]: INVisible | ACTive | IACTive 'INV': No channel available 'ACTive': Active channel 'IACtive': Inactive channel
- Data: List[enums.ResultStateA]: INVisible | ACTive | IACTive 'INV': No channel available 'ACTive': Active channel 'IACtive': Inactive channel

fetch() → FetchStruct

```
# SCPI: FETCh:EVDO:MEASurement<instance>:MEValuation:LIST:CP:STATe
value: FetchStruct = driver.multiEval.listPy.cp.state.fetch()
```

Return the states of the channels for power measurement (CP) . The values listed below in curly brackets {} are returned for each active segment: {...}seg 1, {...}seg 2, ..., {...}seg n. The number of active segments n is determined by method RsCmwEvdoMeas.Configure.MultiEval.ListPy.count. The number to the left of each result parameter is provided for easy identification of the parameter position within the result array.

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

7.4.7.5 Dwcp

class Dwcp
Dwcp commands group definition. 40 total commands, 8 Sub-groups, 0 group commands

Cloning the Group

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

Subgroups

7.4.7.5.1 Wtfi

class Wtfi
Wtfi commands group definition. 8 total commands, 4 Sub-groups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.multiEval.listPy.dwcp.wtfi.clone()
```

Subgroups

7.4.7.5.1.1 Current

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:DWCP:WTFI:CURRent
CALCulate:EVDO:MEASurement<Instance>:MEValuation:LIST:DWCP:WTFI:CURRent
```

class Current
Current commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

class CalculateStruct
Response structure. Fields:

- Reliabiltiy: int: No parameter help available
- W_24_I: List[enums.ResultStatus2]: No parameter help available

class FetchStruct
Response structure. Fields:

- Reliabiltiy: int: No parameter help available
- W_24_I: List[float]: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:EVDO:MEASurement<instance>:MEValuation:LIST:DWCP:WTFI:CURRent
value: CalculateStruct = driver.multiEval.listPy.dwcp.wtfi.current.calculate()
```

Returns the scalar channel power for the data channel results for all active list mode segments. WOT represents W12 (Walsh code one-two) and WTF represents W24 (Walsh code two-four) . The values described below are returned by FETCh 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:EVDO:MEASurement<instance>:MEValuation:LIST:DWCP:WTFI:CURRent
value: FetchStruct = driver.multiEval.listPy.dwcp.wtfi.current.fetch()
```

Returns the scalar channel power for the data channel results for all active list mode segments. WOT represents W12 (Walsh code one-two) and WTF represents W24 (Walsh code two-four) . The values described below are returned by FETCh 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.

7.4.7.5.1.2 Average

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:DWCP:WTFI:AVERage
CALCulate:EVDO:MEASurement<Instance>:MEValuation:LIST:DWCP:WTFI:AVERage
```

class Average

Average commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

class CalculateStruct

Response structure. Fields:

- Reliabiltiy: int: No parameter help available
- W_24_I: List[enums.ResultStatus2]: No parameter help available

class FetchStruct

Response structure. Fields:

- Reliabiltiy: int: No parameter help available
- W_24_I: List[float]: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:EVDO:MEASurement<instance>:MEValuation:LIST:DWCP:WTFI:AVERage
value: CalculateStruct = driver.multiEval.listPy.dwcp.wtfi.average.calculate()
```

Returns the scalar channel power for the data channel results for all active list mode segments. WOT represents W12 (Walsh code one-two) and WTF represents W24 (Walsh code two-four) . The values described below are returned by FETCh 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:EVDO:MEASurement<instance>:MEValuation:LIST:DWCP:WTFI:AVERage
value: FetchStruct = driver.multiEval.listPy.dwcp.wtfi.average.fetch()
```

Returns the scalar channel power for the data channel results for all active list mode segments. WOT represents W12 (Walsh code one-two) and WTF represents W24 (Walsh code two-four) . The values described below are returned by FETCh 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.

7.4.7.5.1.3 Maximum

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:DWCP:WTFI:MAXimum
CALCulate:EVDO:MEASurement<Instance>:MEValuation:LIST:DWCP:WTFI:MAXimum
```

class Maximum

Maximum commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

class CalculateStruct

Response structure. Fields:

- Reliabiltiy: int: No parameter help available
- W_24_I: List[enums.ResultStatus2]: No parameter help available

class FetchStruct

Response structure. Fields:

- Reliabiltiy: int: No parameter help available
- W_24_I: List[float]: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:EVDO:MEASurement<instance>:MEValuation:LIST:DWCP:WTFI:MAXimum
value: CalculateStruct = driver.multiEval.listPy.dwcp.wtfi.maximum.calculate()
```

Returns the scalar channel power for the data channel results for all active list mode segments. WOT represents W12 (Walsh code one-two) and WTF represents W24 (Walsh code two-four) . The values described below are returned by FETCh 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:EVDO:MEASurement<instance>:MEValuation:LIST:DWCP:WTFI:MAXimum
value: FetchStruct = driver.multiEval.listPy.dwcp.wtfi.maximum.fetch()
```

Returns the scalar channel power for the data channel results for all active list mode segments. WOT represents W12 (Walsh code one-two) and WTF represents W24 (Walsh code two-four) . The values described below are returned by FETCh 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.

7.4.7.5.1.4 Minimum

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:DWCP:WTFI:MINimum
CALCulate:EVDO:MEASurement<Instance>:MEValuation:LIST:DWCP:WTFI:MINimum
```

class Minimum

Minimum commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

class CalculateStruct

Response structure. Fields:

- Reliabiltiy: int: No parameter help available
- W_24_I: List[enums.ResultStatus2]: No parameter help available

class FetchStruct

Response structure. Fields:

- Reliabiltiy: int: No parameter help available
- W_24_I: List[float]: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:EVDO:MEASurement<instance>:MEValuation:LIST:DWCP:WTFI:MINimum
value: CalculateStruct = driver.multiEval.listPy.dwcp.wtfi.minimum.calculate()
```

Returns the scalar channel power for the data channel results for all active list mode segments. WOT represents W12 (Walsh code one-two) and WTF represents W24 (Walsh code two-four) . The values described below are returned by FETCh 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:EVDO:MEASurement<instance>:MEValuation:LIST:DWCP:WTFI:MINimum
value: FetchStruct = driver.multiEval.listPy.dwcp.wtfi.minimum.fetch()
```

Returns the scalar channel power for the data channel results for all active list mode segments. WOT represents W12 (Walsh code one-two) and WTF represents W24 (Walsh code two-four) . The values described below are returned by FETCh 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.

7.4.7.5.2 Wtfq

class Wtfq
Wtfq commands group definition. 8 total commands, 4 Sub-groups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.multiEval.listPy.dwcp.wtfq.clone()
```

Subgroups

7.4.7.5.2.1 Current

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:DWCP:WTFQ:CURRent
CALCulate:EVDO:MEASurement<Instance>:MEValuation:LIST:DWCP:WTFQ:CURRent
```

class Current
Current commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

class CalculateStruct
Response structure. Fields:

- Reliabiltiy: int: No parameter help available
- W_24_Q: List[enums.ResultStatus2]: No parameter help available

class FetchStruct
Response structure. Fields:

- Reliabiltiy: int: No parameter help available
- W_24_Q: List[float]: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:EVDO:MEASurement<instance>:MEValuation:LIST:DWCP:WTFQ:CURRent
value: CalculateStruct = driver.multiEval.listPy.dwcp.wtfq.current.calculate()
```

Returns the scalar channel power for the data channel results for all active list mode segments. WOT represents W12 (Walsh code one-two) and WTF represents W24 (Walsh code two-four) . The values described below are returned by FETCh 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:EVDO:MEASurement<instance>:MEValuation:LIST:DWCP:WTFQ:CURRent
value: FetchStruct = driver.multiEval.listPy.dwcp.wtfq.current.fetch()
```

Returns the scalar channel power for the data channel results for all active list mode segments. WOT represents W12 (Walsh code one-two) and WTF represents W24 (Walsh code two-four) . The values described below are returned by FETCh 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.

7.4.7.5.2.2 Average

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:DWCP:WTFQ:AVERage
CALCulate:EVDO:MEASurement<Instance>:MEValuation:LIST:DWCP:WTFQ:AVERage
```

class Average

Average commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

class CalculateStruct

Response structure. Fields:

- Reliabiltiy: int: No parameter help available
- W_24_Q: List[enums.ResultStatus2]: No parameter help available

class FetchStruct

Response structure. Fields:

- Reliabiltiy: int: No parameter help available
- W_24_Q: List[float]: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:EVDO:MEASurement<instance>:MEValuation:LIST:DWCP:WTFQ:AVERage
value: CalculateStruct = driver.multiEval.listPy.dwcp.wtfq.average.calculate()
```

Returns the scalar channel power for the data channel results for all active list mode segments. WOT represents W12 (Walsh code one-two) and WTF represents W24 (Walsh code two-four) . The values described below are returned by FETCh 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:EVDO:MEASurement<instance>:MEValuation:LIST:DWCP:WTFQ:AVERage
value: FetchStruct = driver.multiEval.listPy.dwcp.wtfq.average.fetch()
```

Returns the scalar channel power for the data channel results for all active list mode segments. WOT represents W12 (Walsh code one-two) and WTF represents W24 (Walsh code two-four) . The values described below are returned by FETCh 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.

7.4.7.5.2.3 Maximum

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:DWCP:WTFQ:MAXimum
CALCulate:EVDO:MEASurement<Instance>:MEValuation:LIST:DWCP:WTFQ:MAXimum
```

class Maximum

Maximum commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

class CalculateStruct

Response structure. Fields:

- Reliabiltiy: int: No parameter help available
- W_24_Q: List[enums.ResultStatus2]: No parameter help available

class FetchStruct

Response structure. Fields:

- Reliabiltiy: int: No parameter help available
- W_24_Q: List[float]: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:EVDO:MEASurement<instance>:MEValuation:LIST:DWCP:WTFQ:MAXimum
value: CalculateStruct = driver.multiEval.listPy.dwcp.wtfq.maximum.calculate()
```

Returns the scalar channel power for the data channel results for all active list mode segments. WOT represents W12 (Walsh code one-two) and WTF represents W24 (Walsh code two-four) . The values described below are returned by FETCh 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:EVDO:MEASurement<instance>:MEValuation:LIST:DWCP:WTFQ:MAXimum
value: FetchStruct = driver.multiEval.listPy.dwcp.wtfq.maximum.fetch()
```

Returns the scalar channel power for the data channel results for all active list mode segments. WOT represents W12 (Walsh code one-two) and WTF represents W24 (Walsh code two-four) . The values described below are returned by FETCh 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.

7.4.7.5.2.4 Minimum

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:DWCP:WTFQ:MINimum
CALCulate:EVDO:MEASurement<Instance>:MEValuation:LIST:DWCP:WTFQ:MINimum
```

class Minimum

Minimum commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

class CalculateStruct

Response structure. Fields:

- Reliabiltiy: int: No parameter help available
- W_24_Q: List[enums.ResultStatus2]: No parameter help available

class FetchStruct

Response structure. Fields:

- Reliabiltiy: int: No parameter help available
- W_24_Q: List[float]: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:EVDO:MEASurement<instance>:MEValuation:LIST:DWCP:WTFQ:MINimum
value: CalculateStruct = driver.multiEval.listPy.dwcp.wtfq.minimum.calculate()
```

Returns the scalar channel power for the data channel results for all active list mode segments. WOT represents W12 (Walsh code one-two) and WTF represents W24 (Walsh code two-four) . The values described below are returned by FETCh 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:EVDO:MEASurement<instance>:MEValuation:LIST:DWCP:WTFQ:MINimum
value: FetchStruct = driver.multiEval.listPy.dwcp.wtfq.minimum.fetch()
```

Returns the scalar channel power for the data channel results for all active list mode segments. WOT represents W12 (Walsh code one-two) and WTF represents W24 (Walsh code two-four) . The values described below are returned by FETCh 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.

7.4.7.5.3 Woti

class Woti
Woti commands group definition. 8 total commands, 4 Sub-groups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.multiEval.listPy.dwcp.woti.clone()
```

Subgroups

7.4.7.5.3.1 Current

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:DWCP:WOTI:CURRent
CALCulate:EVDO:MEASurement<Instance>:MEValuation:LIST:DWCP:WOTI:CURRent
```

class Current
Current commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

class CalculateStruct
Response structure. Fields:

- Reliabiltiy: int: No parameter help available
- W_12_I: List[enums.ResultStatus2]: No parameter help available

class FetchStruct
Response structure. Fields:

- Reliabiltiy: int: No parameter help available
- W_12_I: List[float]: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:EVDO:MEASurement<instance>:MEValuation:LIST:DWCP:WOTI:CURRent
value: CalculateStruct = driver.multiEval.listPy.dwcp.woti.current.calculate()
```

Returns the scalar channel power for the data channel results for all active list mode segments. WOT represents W12 (Walsh code one-two) and WTF represents W24 (Walsh code two-four) . The values described below are returned by FETCh 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:EVDO:MEASurement<instance>:MEValuation:LIST:DWCP:WOTI:CURRent
value: FetchStruct = driver.multiEval.listPy.dwcp.woti.current.fetch()
```

Returns the scalar channel power for the data channel results for all active list mode segments. WOT represents W12 (Walsh code one-two) and WTF represents W24 (Walsh code two-four) . The values described below are returned by FETCh 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.

7.4.7.5.3.2 Average

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:DWCP:WOTI:AVERage
CALCulate:EVDO:MEASurement<Instance>:MEValuation:LIST:DWCP:WOTI:AVERage
```

class Average

Average commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

class CalculateStruct

Response structure. Fields:

- Reliabiltiy: int: No parameter help available
- W_12_I: List[enums.ResultStatus2]: No parameter help available

class FetchStruct

Response structure. Fields:

- Reliabiltiy: int: No parameter help available
- W_12_I: List[float]: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:EVDO:MEASurement<instance>:MEValuation:LIST:DWCP:WOTI:AVERage
value: CalculateStruct = driver.multiEval.listPy.dwcp.woti.average.calculate()
```

Returns the scalar channel power for the data channel results for all active list mode segments. WOT represents W12 (Walsh code one-two) and WTF represents W24 (Walsh code two-four) . The values described below are returned by FETCh 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:EVDO:MEASurement<instance>:MEValuation:LIST:DWCP:WOTI:AVERage
value: FetchStruct = driver.multiEval.listPy.dwcp.woti.average.fetch()
```

Returns the scalar channel power for the data channel results for all active list mode segments. WOT represents W12 (Walsh code one-two) and WTF represents W24 (Walsh code two-four) . The values described below are returned by FETCh 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.

7.4.7.5.3.3 Maximum

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:DWCP:WOTI:MAXimum
CALCulate:EVDO:MEASurement<Instance>:MEValuation:LIST:DWCP:WOTI:MAXimum
```

class Maximum

Maximum commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

class CalculateStruct

Response structure. Fields:

- Reliabiltiy: int: No parameter help available
- W_12_I: List[enums.ResultStatus2]: No parameter help available

class FetchStruct

Response structure. Fields:

- Reliabiltiy: int: No parameter help available
- W_12_I: List[float]: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:EVDO:MEASurement<instance>:MEValuation:LIST:DWCP:WOTI:MAXimum
value: CalculateStruct = driver.multiEval.listPy.dwcp.woti.maximum.calculate()
```

Returns the scalar channel power for the data channel results for all active list mode segments. WOT represents W12 (Walsh code one-two) and WTF represents W24 (Walsh code two-four) . The values described below are returned by FETCh 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:EVDO:MEASurement<instance>:MEValuation:LIST:DWCP:WOTI:MAXimum
value: FetchStruct = driver.multiEval.listPy.dwcp.woti.maximum.fetch()
```

Returns the scalar channel power for the data channel results for all active list mode segments. WOT represents W12 (Walsh code one-two) and WTF represents W24 (Walsh code two-four) . The values described below are returned by FETCh 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.

7.4.7.5.3.4 Minimum

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:DWCP:WOTI:MINimum
CALCulate:EVDO:MEASurement<Instance>:MEValuation:LIST:DWCP:WOTI:MINimum
```

class Minimum

Minimum commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

class CalculateStruct

Response structure. Fields:

- Reliabiltiy: int: No parameter help available
- W_12_I: List[enums.ResultStatus2]: No parameter help available

class FetchStruct

Response structure. Fields:

- Reliabiltiy: int: No parameter help available
- W_12_I: List[float]: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:EVDO:MEASurement<instance>:MEValuation:LIST:DWCP:WOTI:MINimum
value: CalculateStruct = driver.multiEval.listPy.dwcp.woti.minimum.calculate()
```

Returns the scalar channel power for the data channel results for all active list mode segments. WOT represents W12 (Walsh code one-two) and WTF represents W24 (Walsh code two-four) . The values described below are returned by FETCh 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:EVDO:MEASurement<instance>:MEValuation:LIST:DWCP:WOTI:MINimum
value: FetchStruct = driver.multiEval.listPy.dwcp.woti.minimum.fetch()
```

Returns the scalar channel power for the data channel results for all active list mode segments. WOT represents W12 (Walsh code one-two) and WTF represents W24 (Walsh code two-four) . The values described below are returned by FETCh 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.

7.4.7.5.4 Wotq

class Wotq
Wotq commands group definition. 8 total commands, 4 Sub-groups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.multiEval.listPy.dwcp.wotq.clone()
```

Subgroups

7.4.7.5.4.1 Current

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:DWCP:WOTQ:CURRent
CALCulate:EVDO:MEASurement<Instance>:MEValuation:LIST:DWCP:WOTQ:CURRent
```

class Current
Current commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

class CalculateStruct
Response structure. Fields:

- Reliabiltiy: int: No parameter help available
- W_12_Q: List[enums.ResultStatus2]: No parameter help available

class FetchStruct
Response structure. Fields:

- Reliabiltiy: int: No parameter help available
- W_12_Q: List[float]: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:EVDO:MEASurement<instance>:MEValuation:LIST:DWCP:WOTQ:CURRent
value: CalculateStruct = driver.multiEval.listPy.dwcp.wotq.current.calculate()
```

Returns the scalar channel power for the data channel results for all active list mode segments. WOT represents W12 (Walsh code one-two) and WTF represents W24 (Walsh code two-four) . The values described below are returned by FETCh 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:EVDO:MEASurement<instance>:MEValuation:LIST:DWCP:WOTQ:CURRent
value: FetchStruct = driver.multiEval.listPy.dwcp.wotq.current.fetch()
```

Returns the scalar channel power for the data channel results for all active list mode segments. WOT represents W12 (Walsh code one-two) and WTF represents W24 (Walsh code two-four) . The values described below are returned by FETCh 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.

7.4.7.5.4.2 Average

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:DWCP:WOTQ:AVERage
CALCulate:EVDO:MEASurement<Instance>:MEValuation:LIST:DWCP:WOTQ:AVERage
```

class Average

Average commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

class CalculateStruct

Response structure. Fields:

- Reliabiltiy: int: No parameter help available
- W_12_Q: List[enums.ResultStatus2]: No parameter help available

class FetchStruct

Response structure. Fields:

- Reliabiltiy: int: No parameter help available
- W_12_Q: List[float]: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:EVDO:MEASurement<instance>:MEValuation:LIST:DWCP:WOTQ:AVERage
value: CalculateStruct = driver.multiEval.listPy.dwcp.wotq.average.calculate()
```

Returns the scalar channel power for the data channel results for all active list mode segments. WOT represents W12 (Walsh code one-two) and WTF represents W24 (Walsh code two-four) . The values described below are returned by FETCh 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:EVDO:MEASurement<instance>:MEValuation:LIST:DWCP:WOTQ:AVERage
value: FetchStruct = driver.multiEval.listPy.dwcp.wotq.average.fetch()
```

Returns the scalar channel power for the data channel results for all active list mode segments. WOT represents W12 (Walsh code one-two) and WTF represents W24 (Walsh code two-four) . The values described below are returned by FETCh 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.

7.4.7.5.4.3 Maximum

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:DWCP:WOTQ:MAXimum
CALCulate:EVDO:MEASurement<Instance>:MEValuation:LIST:DWCP:WOTQ:MAXimum
```

class Maximum

Maximum commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

class CalculateStruct

Response structure. Fields:

- Reliabiltiy: int: No parameter help available
- W_12_Q: List[enums.ResultStatus2]: No parameter help available

class FetchStruct

Response structure. Fields:

- Reliabiltiy: int: No parameter help available
- W_12_Q: List[float]: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:EVDO:MEASurement<instance>:MEValuation:LIST:DWCP:WOTQ:MAXimum
value: CalculateStruct = driver.multiEval.listPy.dwcp.wotq.maximum.calculate()
```

Returns the scalar channel power for the data channel results for all active list mode segments. WOT represents W12 (Walsh code one-two) and WTF represents W24 (Walsh code two-four) . The values described below are returned by FETCh 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:EVDO:MEASurement<instance>:MEValuation:LIST:DWCP:WOTQ:MAXimum
value: FetchStruct = driver.multiEval.listPy.dwcp.wotq.maximum.fetch()
```

Returns the scalar channel power for the data channel results for all active list mode segments. WOT represents W12 (Walsh code one-two) and WTF represents W24 (Walsh code two-four) . The values described below are returned by FETCh 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.

7.4.7.5.4.4 Minimum

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:DWCP:WOTQ:MINimum
CALCulate:EVDO:MEASurement<Instance>:MEValuation:LIST:DWCP:WOTQ:MINimum
```

class Minimum

Minimum commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

class CalculateStruct

Response structure. Fields:

- Reliabiltiy: int: No parameter help available
- W_12_Q: List[enums.ResultStatus2]: No parameter help available

class FetchStruct

Response structure. Fields:

- Reliabiltiy: int: No parameter help available
- W_12_Q: List[float]: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:EVDO:MEASurement<instance>:MEValuation:LIST:DWCP:WOTQ:MINimum
value: CalculateStruct = driver.multiEval.listPy.dwcp.wotq.minimum.calculate()
```

Returns the scalar channel power for the data channel results for all active list mode segments. WOT represents W12 (Walsh code one-two) and WTF represents W24 (Walsh code two-four) . The values described below are returned by FETCh 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:EVDO:MEASurement<instance>:MEValuation:LIST:DWCP:WOTQ:MINimum
value: FetchStruct = driver.multiEval.listPy.dwcp.wotq.minimum.fetch()
```

Returns the scalar channel power for the data channel results for all active list mode segments. WOT represents W12 (Walsh code one-two) and WTF represents W24 (Walsh code two-four) . The values described below are returned by FETCh 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.

7.4.7.5.5 Current

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:DWCP:CURRent
CALCulate:EVDO:MEASurement<Instance>:MEValuation:LIST:DWCP:CURRent
```

class Current

Current commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

class CalculateStruct

Response structure. Fields:

- Reliabiltiy: int: decimal 'Conventions and General Information'. In list mode, a zero reliability indicator indicates that the results in all measured segments are valid. A non-zero value indicates that an error occurred in at least one of the measured segments.
- Seg_Reliability: List[int]: 0 | 3 | 4 | 8 The segment reliability indicates whether one of the following exceptions occurred in this segment: 0: No error 3: Signal overflow 4: Signal low 8: Synchronization error If a combination of exceptions occurs, the most severe error is indicated.
- W_24_I: List[enums.ResultStatus2]: No parameter help available
- W_24_Q: List[enums.ResultStatus2]: No parameter help available
- W_12_I: List[enums.ResultStatus2]: No parameter help available
- W_12_Q: List[enums.ResultStatus2]: No parameter help available

class FetchStruct

Response structure. Fields:

- Reliabiltiy: int: decimal 'Conventions and General Information'. In list mode, a zero reliability indicator indicates that the results in all measured segments are valid. A non-zero value indicates that an error occurred in at least one of the measured segments.
- Seg_Reliability: List[int]: 0 | 3 | 4 | 8 The segment reliability indicates whether one of the following exceptions occurred in this segment: 0: No error 3: Signal overflow 4: Signal low 8: Synchronization error If a combination of exceptions occurs, the most severe error is indicated.
- W_24_I: List[float]: No parameter help available
- W_24_Q: List[float]: No parameter help available
- W_12_I: List[float]: No parameter help available
- W_12_Q: List[float]: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:EVDO:MEASurement<instance>:MEValuation:LIST:DWCP:CURRent
value: CalculateStruct = driver.multiEval.listPy.dwcp.current.calculate()
```

Returns the scalar channel power for the data channel results in list mode. The result is extended to the W24 and W12 I/Q values of the data channel. Only available if subtype 2 or 3 is selected. Otherwise NAV is returned in list mode. To define the statistical length for AVERage, MAXimum, MINimum calculation and enable the calculation of the results use the command method RsCmwEvdoMeas.Configure.MultiEval.ListPy.Segment.Modulation.set. The values listed below in curly brackets {} are returned for each active segment: {...}seg 1, {...}seg 2, ..., {...}seg n. The number of active

segments n is determined by method RsCmwEvdoMeas.Configure.MultiEval.ListPy.count. 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:EVDO:MEASurement<instance>:MEValuation:LIST:DWCP:CURRent
value: FetchStruct = driver.multiEval.listPy.dwcp.current.fetch()
```

Returns the scalar channel power for the data channel results in list mode. The result is extended to the W24 and W12 I/Q values of the data channel. Only available if subtype 2 or 3 is selected. Otherwise NAV is returned in list mode. To define the statistical length for AVERage, MAXimum, MINimum calculation and enable the calculation of the results use the command method RsCmwEvdoMeas.Configure.MultiEval.ListPy.Segment.Modulation.set. The values listed below in curly brackets {} are returned for each active segment: {...}seg 1, {...}seg 2, ..., {...}seg n. The number of active segments n is determined by method RsCmwEvdoMeas.Configure.MultiEval.ListPy.count. 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.

7.4.7.5.6 Average

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:DWCP:AVERage
CALCulate:EVDO:MEASurement<Instance>:MEValuation:LIST:DWCP:AVERage
```

class Average

Average commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

class CalculateStruct

Response structure. Fields:

- Reliabiltiy: int: decimal 'Conventions and General Information'. In list mode, a zero reliability indicator indicates that the results in all measured segments are valid. A non-zero value indicates that an error occurred in at least one of the measured segments.
- Seg_Reliability: List[int]: 0 | 3 | 4 | 8 The segment reliability indicates whether one of the following exceptions occurred in this segment: 0: No error 3: Signal overflow 4: Signal low 8: Synchronization error If a combination of exceptions occurs, the most severe error is indicated.
- W_24_I: List[enums.ResultStatus2]: No parameter help available
- W_24_Q: List[enums.ResultStatus2]: No parameter help available
- W_12_I: List[enums.ResultStatus2]: No parameter help available
- W_12_Q: List[enums.ResultStatus2]: No parameter help available

class FetchStruct

Response structure. Fields:

- Reliabiltiy: int: decimal 'Conventions and General Information'. In list mode, a zero reliability indicator indicates that the results in all measured segments are valid. A non-zero value indicates that an error occurred in at least one of the measured segments.

- Seg_Reliability: List[int]: 0 | 3 | 4 | 8 The segment reliability indicates whether one of the following exceptions occurred in this segment: 0: No error 3: Signal overflow 4: Signal low 8: Synchronization error If a combination of exceptions occurs, the most severe error is indicated.
- W_24_I: List[float]: No parameter help available
- W_24_Q: List[float]: No parameter help available
- W_12_I: List[float]: No parameter help available
- W_12_Q: List[float]: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:EVDO:MEASurement<instance>:MEValuation:LIST:DWCP:AVERage
value: CalculateStruct = driver.multiEval.listPy.dwcp.average.calculate()
```

Returns the scalar channel power for the data channel results in list mode. The result is extended to the W24 and W12 I/Q values of the data channel. Only available if subtype 2 or 3 is selected. Otherwise NAV is returned in list mode. To define the statistical length for AVERage, MAXimum, MINimum calculation and enable the calculation of the results use the command method RsCmwEvdoMeas.Configure.MultiEval.ListPy.Segment.Modulation.set. The values listed below in curly brackets {} are returned for each active segment: {...}seg 1, {...}seg 2, ..., {...}seg n. The number of active segments n is determined by method RsCmwEvdoMeas.Configure.MultiEval.ListPy.count. 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:EVDO:MEASurement<instance>:MEValuation:LIST:DWCP:AVERage
value: FetchStruct = driver.multiEval.listPy.dwcp.average.fetch()
```

Returns the scalar channel power for the data channel results in list mode. The result is extended to the W24 and W12 I/Q values of the data channel. Only available if subtype 2 or 3 is selected. Otherwise NAV is returned in list mode. To define the statistical length for AVERage, MAXimum, MINimum calculation and enable the calculation of the results use the command method RsCmwEvdoMeas.Configure.MultiEval.ListPy.Segment.Modulation.set. The values listed below in curly brackets {} are returned for each active segment: {...}seg 1, {...}seg 2, ..., {...}seg n. The number of active segments n is determined by method RsCmwEvdoMeas.Configure.MultiEval.ListPy.count. 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.

7.4.7.5.7 Maximum

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:DWCP:MAXimum
CALCulate:EVDO:MEASurement<Instance>:MEValuation:LIST:DWCP:MAXimum
```

class Maximum

Maximum commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

class CalculateStruct

Response structure. Fields:

- Reliabiltiy: int: decimal 'Conventions and General Information'. In list mode, a zero reliability indicator indicates that the results in all measured segments are valid. A non-zero value indicates that an error occurred in at least one of the measured segments.
- Seg_Reliability: List[int]: 0 | 3 | 4 | 8 The segment reliability indicates whether one of the following exceptions occurred in this segment: 0: No error 3: Signal overflow 4: Signal low 8: Synchronization error If a combination of exceptions occurs, the most severe error is indicated.
- W_24_I: List[enums.ResultStatus2]: No parameter help available
- W_24_Q: List[enums.ResultStatus2]: No parameter help available
- W_12_I: List[enums.ResultStatus2]: No parameter help available
- W_12_Q: List[enums.ResultStatus2]: No parameter help available

class FetchStruct

Response structure. Fields:

- Reliabiltiy: int: decimal 'Conventions and General Information'. In list mode, a zero reliability indicator indicates that the results in all measured segments are valid. A non-zero value indicates that an error occurred in at least one of the measured segments.
- Seg_Reliability: List[int]: 0 | 3 | 4 | 8 The segment reliability indicates whether one of the following exceptions occurred in this segment: 0: No error 3: Signal overflow 4: Signal low 8: Synchronization error If a combination of exceptions occurs, the most severe error is indicated.
- W_24_I: List[float]: No parameter help available
- W_24_Q: List[float]: No parameter help available
- W_12_I: List[float]: No parameter help available
- W_12_Q: List[float]: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:EVDO:MEASurement<instance>:MEValuation:LIST:DWCP:MAXimum
value: CalculateStruct = driver.multiEval.listPy.dwcp.maximum.calculate()
```

Returns the scalar channel power for the data channel results in list mode. The result is extended to the W24 and W12 I/Q values of the data channel. Only available if subtype 2 or 3 is selected. Otherwise NAV is returned in list mode. To define the statistical length for AVERage, MAXimum, MINimum calculation and enable the calculation of the results use the command method RsCmwEvdoMeas.Configure.MultiEval.ListPy.Segment.Modulation.set. The values listed below in curly brackets {} are returned for each active segment: {...}seg 1, {...}seg 2, ..., {...}seg n. The number of active

segments n is determined by method RsCmwEvdoMeas.Configure.MultiEval.ListPy.count. 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:EVDO:MEASurement<instance>:MEValuation:LIST:DWCP:MAXimum
value: FetchStruct = driver.multiEval.listPy.dwcp.maximum.fetch()
```

Returns the scalar channel power for the data channel results in list mode. The result is extended to the W24 and W12 I/Q values of the data channel. Only available if subtype 2 or 3 is selected. Otherwise NAV is returned in list mode. To define the statistical length for AVERage, MAXimum, MINimum calculation and enable the calculation of the results use the command method RsCmwEvdoMeas.Configure.MultiEval.ListPy.Segment.Modulation.set. The values listed below in curly brackets {} are returned for each active segment: {...}seg 1, {...}seg 2, ..., {...}seg n. The number of active segments n is determined by method RsCmwEvdoMeas.Configure.MultiEval.ListPy.count. 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.

7.4.7.5.8 Minimum

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:DWCP:MINimum
CALCulate:EVDO:MEASurement<Instance>:MEValuation:LIST:DWCP:MINimum
```

class Minimum

Minimum commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

class CalculateStruct

Response structure. Fields:

- Reliabiltiy: int: decimal 'Conventions and General Information'. In list mode, a zero reliability indicator indicates that the results in all measured segments are valid. A non-zero value indicates that an error occurred in at least one of the measured segments.
- Seg_Reliability: List[int]: 0 | 3 | 4 | 8 The segment reliability indicates whether one of the following exceptions occurred in this segment: 0: No error 3: Signal overflow 4: Signal low 8: Synchronization error If a combination of exceptions occurs, the most severe error is indicated.
- W_24_I: List[enums.ResultStatus2]: No parameter help available
- W_24_Q: List[enums.ResultStatus2]: No parameter help available
- W_12_I: List[enums.ResultStatus2]: No parameter help available
- W_12_Q: List[enums.ResultStatus2]: No parameter help available

class FetchStruct

Response structure. Fields:

- Reliabiltiy: int: decimal 'Conventions and General Information'. In list mode, a zero reliability indicator indicates that the results in all measured segments are valid. A non-zero value indicates that an error occurred in at least one of the measured segments.

- Seg_Reliability: List[int]: 0 | 3 | 4 | 8 The segment reliability indicates whether one of the following exceptions occurred in this segment: 0: No error 3: Signal overflow 4: Signal low 8: Synchronization error If a combination of exceptions occurs, the most severe error is indicated.
- W_24_I: List[float]: No parameter help available
- W_24_Q: List[float]: No parameter help available
- W_12_I: List[float]: No parameter help available
- W_12_Q: List[float]: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:EVDO:MEASurement<instance>:MEValuation:LIST:DWCP:MINimum
value: CalculateStruct = driver.multiEval.listPy.dwcp.minimum.calculate()
```

Returns the scalar channel power for the data channel results in list mode. The result is extended to the W24 and W12 I/Q values of the data channel. Only available if subtype 2 or 3 is selected. Otherwise NAV is returned in list mode. To define the statistical length for AVERage, MAXimum, MINimum calculation and enable the calculation of the results use the command method RsCmwEvdoMeas.Configure.MultiEval.ListPy.Segment.Modulation.set. The values listed below in curly brackets {} are returned for each active segment: {...}seg 1, {...}seg 2, ..., {...}seg n. The number of active segments n is determined by method RsCmwEvdoMeas.Configure.MultiEval.ListPy.count. 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:EVDO:MEASurement<instance>:MEValuation:LIST:DWCP:MINimum
value: FetchStruct = driver.multiEval.listPy.dwcp.minimum.fetch()
```

Returns the scalar channel power for the data channel results in list mode. The result is extended to the W24 and W12 I/Q values of the data channel. Only available if subtype 2 or 3 is selected. Otherwise NAV is returned in list mode. To define the statistical length for AVERage, MAXimum, MINimum calculation and enable the calculation of the results use the command method RsCmwEvdoMeas.Configure.MultiEval.ListPy.Segment.Modulation.set. The values listed below in curly brackets {} are returned for each active segment: {...}seg 1, {...}seg 2, ..., {...}seg n. The number of active segments n is determined by method RsCmwEvdoMeas.Configure.MultiEval.ListPy.count. 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.

7.4.7.6 Acp

class Acp
Acp commands group definition. 66 total commands, 12 Sub-groups, 0 group commands

Cloning the Group

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

Subgroups

7.4.7.6.1 Otolerance

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:ACP:OTOLerance
```

class Otolerance
Otolerance commands group definition. 1 total commands, 0 Sub-groups, 1 group commands

fetch() → List[int]

```
# SCPI: FETCh:EVDO:MEASurement<instance>:MEValuation:LIST:ACP:OTOLerance
value: List[int] = driver.multiEval.listPy.acp.otolerance.fetch()
```

Returns the out of tolerance percentages in the adjacent channel power measurement for all active list mode segments.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return out_of_tol_count: decimal Comma-separated list of values, one per active segment. Range: 0 % to 100 %, Unit: %

7.4.7.6.2 StCount

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:ACP:STCount
```

class StCount
StCount commands group definition. 1 total commands, 0 Sub-groups, 1 group commands

fetch() → List[int]

```
# SCPI: FETCh:EVDO:MEASurement<instance>:MEValuation:LIST:ACP:STCount
value: List[int] = driver.multiEval.listPy.acp.stCount.fetch()
```

Returns the statistic count of the spectrum measurement for all active list mode segments.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return statistic_count: decimal Number of evaluated valid slots. Comma-separated list of values, one per active segment.

7.4.7.6.3 Acpm<AcpMinus>

RepCap Settings

```
# Range: Ch1 .. Ch20
rc = driver.multiEval.listPy.acp.acpm.repcap_acpMinus_get()
driver.multiEval.listPy.acp.acpm.repcap_acpMinus_set(repcap.AcpMinus.Ch1)
```

class Acpm

Acpm commands group definition. 6 total commands, 3 Sub-groups, 0 group commands Repeated Capability: AcpMinus, default value after init: AcpMinus.Ch1

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.multiEval.listPy.acp.acpm.clone()
```

Subgroups

7.4.7.6.3.1 Current

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:ACP:ACPM<AcpMinus>:CURRent
CALCulate:EVDO:MEASurement<Instance>:MEValuation:LIST:ACP:ACPM<AcpMinus>:CURRent
```

class Current

Current commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

calculate(*acpMinus=<AcpMinus.Default: -1>*) → List[float]

```
# SCPI: CALCulate:EVDO:MEASurement<instance>:MEValuation:LIST:ACP:ACPM
↪<freqpoint>:CURRent
value: List[float] = driver.multiEval.listPy.acp.acpm.current.
↪calculate(acpMinus = repcap.AcpMinus.Default)
```

Returns adjacent channel power (statistical) values for the selected off-center channels and all active list mode segments. The mnemonic ACPM refers to the channels in the 'minus' direction, the mnemonic ACPP to the channels in the 'plus' direction relative to the current channel. The offset is selected with the <freqpoint> suffix (±10 channels) . The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

param acpMinus optional repeated capability selector. Default value: Ch1 (settable in the interface 'Acpm')

return acpm: float Comma-separated list of values, one per active segment. Range: -999 dB to 999 dB, Unit: dB

fetch(*acpMinus*=<*AcpMinus.Default: -1*>) → List[float]

```
# SCPI: FETCh:EVDO:MEASurement<instance>:MEValuation:LIST:ACP:ACPM<freqpoint>
↪:CURRent
value: List[float] = driver.multiEval.listPy.acp.acpm.current.fetch(acpMinus =␣
↪repcap.AcpMinus.Default)
```

Returns adjacent channel power (statistical) values for the selected off-center channels and all active list mode segments. The mnemonic ACPM refers to the channels in the 'minus' direction, the mnemonic ACPP to the channels in the 'plus' direction relative to the current channel. The offset is selected with the <freqpoint> suffix (±10 channels) . The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

param acpMinus optional repeated capability selector. Default value: Ch1 (settable in the interface 'Acpm')

return acpm: float Comma-separated list of values, one per active segment. Range: -999 dB to 999 dB, Unit: dB

7.4.7.6.3.2 Average

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:ACP:ACPM<AcpMinus>:AVERage
CALCulate:EVDO:MEASurement<Instance>:MEValuation:LIST:ACP:ACPM<AcpMinus>:AVERage
```

class Average

Average commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

calculate(*acpMinus*=<*AcpMinus.Default: -1*>) → List[float]

```
# SCPI: CALCulate:EVDO:MEASurement<instance>:MEValuation:LIST:ACP:ACPM
↪<freqpoint>:AVERage
value: List[float] = driver.multiEval.listPy.acp.acpm.average.
↪calculate(acpMinus = repcap.AcpMinus.Default)
```

Returns adjacent channel power (statistical) values for the selected off-center channels and all active list mode segments. The mnemonic ACPM refers to the channels in the 'minus' direction, the mnemonic ACPP to the channels in the 'plus' direction relative to the current channel. The offset is selected with the <freqpoint> suffix (±10 channels) . The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

param acpMinus optional repeated capability selector. Default value: Ch1 (settable in the interface 'Acpm')

return acpm: float Comma-separated list of values, one per active segment. Range: -999 dB to 999 dB, Unit: dB

fetch(*acpMinus=<AcpMinus.Default: -1>*) → List[float]

```
# SCPI: FETCh:EVDO:MEASurement<instance>:MEValuation:LIST:ACP:ACPM<freqpoint>
↪:AVERage
value: List[float] = driver.multiEval.listPy.acp.acpm.average.fetch(acpMinus =␣
↪repcap.AcpMinus.Default)
```

Returns adjacent channel power (statistical) values for the selected off-center channels and all active list mode segments. The mnemonic ACPM refers to the channels in the 'minus' direction, the mnemonic ACPP to the channels in the 'plus' direction relative to the current channel. The offset is selected with the <freqpoint> suffix (±10 channels) . The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

param acpMinus optional repeated capability selector. Default value: Ch1 (settable in the interface 'Acpm')

return acpm: float Comma-separated list of values, one per active segment. Range: -999 dB to 999 dB, Unit: dB

7.4.7.6.3.3 Maximum

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:ACP:ACPM<AcpMinus>:MAXimum
CALCulate:EVDO:MEASurement<Instance>:MEValuation:LIST:ACP:ACPM<AcpMinus>:MAXimum
```

class Maximum

Maximum commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

calculate(*acpMinus=<AcpMinus.Default: -1>*) → List[float]

```
# SCPI: CALCulate:EVDO:MEASurement<instance>:MEValuation:LIST:ACP:ACPM
↪<freqpoint>:MAXimum
value: List[float] = driver.multiEval.listPy.acp.acpm.maximum.
↪calculate(acpMinus = repcap.AcpMinus.Default)
```

Returns adjacent channel power (statistical) values for the selected off-center channels and all active list mode segments. The mnemonic ACPM refers to the channels in the 'minus' direction, the mnemonic ACPP to the channels in the 'plus' direction relative to the current channel. The offset is selected with the <freqpoint> suffix (±10 channels) . The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

param acpMinus optional repeated capability selector. Default value: Ch1 (settable in the interface 'Acpm')

return acpm: float Comma-separated list of values, one per active segment. Range: -999 dB to 999 dB, Unit: dB

fetch(*acpMinus=<AcpMinus.Default: -1>*) → List[float]

```
# SCPI: FETCh:EVDO:MEASurement<instance>:MEValuation:LIST:ACP:ACPM<freqpoint>
↪:MAXimum
value: List[float] = driver.multiEval.listPy.acp.acpm.maximum.fetch(acpMinus =␣
↪repcap.AcpMinus.Default)
```

Returns adjacent channel power (statistical) values for the selected off-center channels and all active list mode segments. The mnemonic ACPM refers to the channels in the 'minus' direction, the mnemonic ACPP to the channels in the 'plus' direction relative to the current channel. The offset is selected with the <freqpoint> suffix (±10 channels) . The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

param acpMinus optional repeated capability selector. Default value: Ch1 (settable in the interface 'Acpm')

return acpm: float Comma-separated list of values, one per active segment. Range: -999 dB to 999 dB, Unit: dB

7.4.7.6.4 Extended

class Extended

Extended commands group definition. 22 total commands, 7 Sub-groups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.multiEval.listPy.acp.extended.clone()
```

Subgroups

7.4.7.6.4.1 Acpm<AcpMinus>

RepCap Settings

```
# Range: Ch1 .. Ch20
rc = driver.multiEval.listPy.acp.extended.acpm.repcap_acpMinus_get()
driver.multiEval.listPy.acp.extended.acpm.repcap_acpMinus_set(repcap.AcpMinus.Ch1)
```

class Acpm

Acpm commands group definition. 6 total commands, 3 Sub-groups, 0 group commands Repeated Capability: AcpMinus, default value after init: AcpMinus.Ch1

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.multiEval.listPy.acp.extended.acpm.clone()
```

Subgroups

7.4.7.6.4.2 Current

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:ACP:EXTended:ACPM<AcpMinus>:CURRent
CALCulate:EVDO:MEASurement<Instance>:MEValuation:LIST:ACP:EXTended:ACPM<AcpMinus>:CURRent
```

class Current

Current commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

calculate(*acpMinus*=<*AcpMinus.Default: -1*>) → List[float]

```
# SCPI: CALCulate:EVDO:MEASurement<instance>:MEValuation:LIST:ACP:EXTended:ACPM
↪<freqpoint>:CURRent
value: List[float] = driver.multiEval.listPy.acp.extended.acpm.current.
↪calculate(acpMinus = repcap.AcpMinus.Default)
```

Returns adjacent channel power (statistical) values for the selected off-center channels and all active list mode segments. The mnemonic ACPM refers to the channels in the 'minus' direction, the mnemonic ACPP to the channels in the 'plus' direction relative to the current channel. The offset is selected with the <freqpoint> suffix (±20 channels) . The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

param acpMinus optional repeated capability selector. Default value: Ch1 (settable in the interface 'Acpm')

return acpm: float Comma-separated list of values, one per active segment. Range: -999 dB to 999 dB, Unit: dB

fetch(*acpMinus*=<*AcpMinus.Default: -1*>) → List[float]

```
# SCPI: FETCh:EVDO:MEASurement<instance>:MEValuation:LIST:ACP:EXTended:ACPM
↪<freqpoint>:CURRent
value: List[float] = driver.multiEval.listPy.acp.extended.acpm.current.
↪fetch(acpMinus = repcap.AcpMinus.Default)
```

Returns adjacent channel power (statistical) values for the selected off-center channels and all active list mode segments. The mnemonic ACPM refers to the channels in the 'minus' direction, the mnemonic ACPP to the channels in the 'plus' direction relative to the current channel. The offset is selected with the <freqpoint> suffix (±20 channels) . The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

param acpMinus optional repeated capability selector. Default value: Ch1 (settable in the interface 'Acpm')

return acpm: float Comma-separated list of values, one per active segment. Range: -999 dB to 999 dB, Unit: dB

7.4.7.6.4.3 Average

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:ACP:EXTended:ACPM<AcpMinus>:AVERage
CALCulate:EVDO:MEASurement<Instance>:MEValuation:LIST:ACP:EXTended:ACPM<AcpMinus>:AVERage
```

class Average

Average commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

calculate(*acpMinus*=<*AcpMinus.Default: -1*>) → List[float]

```
# SCPI: CALCulate:EVDO:MEASurement<instance>:MEValuation:LIST:ACP:EXTended:ACPM
↪<freqpoint>:AVERage
value: List[float] = driver.multiEval.listPy.acp.extended.acpm.average.
↪calculate(acpMinus = repcap.AcpMinus.Default)
```

Returns adjacent channel power (statistical) values for the selected off-center channels and all active list mode segments. The mnemonic ACPM refers to the channels in the 'minus' direction, the mnemonic ACPP to the channels in the 'plus' direction relative to the current channel. The offset is selected with the <freqpoint> suffix (±20 channels) . The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

param acpMinus optional repeated capability selector. Default value: Ch1 (settable in the interface 'Acpm')

return acpm: float Comma-separated list of values, one per active segment. Range: -999 dB to 999 dB, Unit: dB

fetch(*acpMinus*=<*AcpMinus.Default: -1*>) → List[float]

```
# SCPI: FETCh:EVDO:MEASurement<instance>:MEValuation:LIST:ACP:EXTended:ACPM
↪<freqpoint>:AVERage
value: List[float] = driver.multiEval.listPy.acp.extended.acpm.average.
↪fetch(acpMinus = repcap.AcpMinus.Default)
```

Returns adjacent channel power (statistical) values for the selected off-center channels and all active list mode segments. The mnemonic ACPM refers to the channels in the 'minus' direction, the mnemonic ACPP to the channels in the 'plus' direction relative to the current channel. The offset is selected with the <freqpoint> suffix (±20 channels) . The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

param acpMinus optional repeated capability selector. Default value: Ch1 (settable in the interface 'Acpm')

return acpm: float Comma-separated list of values, one per active segment. Range: -999 dB to 999 dB, Unit: dB

7.4.7.6.4.4 Maximum

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:ACP:EXTended:ACPM<AcpMinus>:MAXimum
CALCulate:EVDO:MEASurement<Instance>:MEValuation:LIST:ACP:EXTended:ACPM<AcpMinus>:MAXimum
```

class Maximum

Maximum commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

calculate(*acpMinus*=<*AcpMinus.Default: -1*>) → List[float]

```
# SCPI: CALCulate:EVDO:MEASurement<instance>:MEValuation:LIST:ACP:EXTended:ACPM
↪<freqpoint>:MAXimum
value: List[float] = driver.multiEval.listPy.acp.extended.acpm.maximum.
↪calculate(acpMinus = repcap.AcpMinus.Default)
```

Returns adjacent channel power (statistical) values for the selected off-center channels and all active list mode segments. The mnemonic ACPM refers to the channels in the 'minus' direction, the mnemonic ACPP to the channels in the 'plus' direction relative to the current channel. The offset is selected with the <freqpoint> suffix (±20 channels) . The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

param acpMinus optional repeated capability selector. Default value: Ch1 (settable in the interface 'Acpm')

return acpm: float Comma-separated list of values, one per active segment. Range: -999 dB to 999 dB, Unit: dB

fetch(*acpMinus*=<*AcpMinus.Default: -1*>) → List[float]

```
# SCPI: FETCh:EVDO:MEASurement<instance>:MEValuation:LIST:ACP:EXTended:ACPM
↪<freqpoint>:MAXimum
value: List[float] = driver.multiEval.listPy.acp.extended.acpm.maximum.
↪fetch(acpMinus = repcap.AcpMinus.Default)
```

Returns adjacent channel power (statistical) values for the selected off-center channels and all active list mode segments. The mnemonic ACPM refers to the channels in the 'minus' direction, the mnemonic ACPP to the channels in the 'plus' direction relative to the current channel. The offset is selected with the <freqpoint> suffix (±20 channels) . The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

param acpMinus optional repeated capability selector. Default value: Ch1 (settable in the interface 'Acpm')

return acpm: float Comma-separated list of values, one per active segment. Range: -999 dB to 999 dB, Unit: dB

7.4.7.6.4.5 Acpp<AcpPlus>

RepCap Settings

```
# Range: Ch1 .. Ch20
rc = driver.multiEval.listPy.acp.extended.acpp.repcap_acpPlus_get()
driver.multiEval.listPy.acp.extended.acpp.repcap_acpPlus_set(repcap.AcpPlus.Ch1)
```

class Acpp

Acpp commands group definition. 6 total commands, 3 Sub-groups, 0 group commands Repeated Capability: AcpPlus, default value after init: AcpPlus.Ch1

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.multiEval.listPy.acp.extended.acpp.clone()
```

Subgroups

7.4.7.6.4.6 Current

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:ACP:EXTended:ACPP<AcpPlus>:CURRent
CALCulate:EVDO:MEASurement<Instance>:MEValuation:LIST:ACP:EXTended:ACPP<AcpPlus>:CURRent
```

class Current

Current commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

calculate(*acpPlus=<AcpPlus.Default: -1>*) → List[float]

```
# SCPI: CALCulate:EVDO:MEASurement<instance>:MEValuation:LIST:ACP:EXTended:ACPP
↪<freqpoint>:CURRent
value: List[float] = driver.multiEval.listPy.acp.extended.acpp.current.
↪calculate(acpPlus = repcap.AcpPlus.Default)
```

Returns adjacent channel power (statistical) values for the selected off-center channels and all active list mode segments. The mnemonic ACPM refers to the channels in the 'minus' direction, the mnemonic ACPP to the channels in the 'plus' direction relative to the current channel. The offset is selected with the <freqpoint> suffix (±20 channels) . The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

param acpPlus optional repeated capability selector. Default value: Ch1 (settable in the interface 'Acpp')

return acpm: float Comma-separated list of values, one per active segment. Range: -999 dB to 999 dB, Unit: dB

fetch(*acpPlus=<AcpPlus.Default: -1>*) → List[float]

```
# SCPI: FETCh:EVDO:MEASurement<instance>:MEValuation:LIST:ACP:EXTended:ACPP
↪<freqpoint>:CURRent
value: List[float] = driver.multiEval.listPy.acp.extended.acpp.current.
↪fetch(acpPlus = repcap.AcpPlus.Default)
```

Returns adjacent channel power (statistical) values for the selected off-center channels and all active list mode segments. The mnemonic ACPM refers to the channels in the 'minus' direction, the mnemonic ACPP to the channels in the 'plus' direction relative to the current channel. The offset is selected with the <freqpoint> suffix (±20 channels) . The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

param acpPlus optional repeated capability selector. Default value: Ch1 (settable in the interface 'Acpp')

return acpm: float Comma-separated list of values, one per active segment. Range: -999 dB to 999 dB, Unit: dB

7.4.7.6.4.7 Average

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:ACP:EXTended:ACPP<AcpPlus>:AVERage
CALCulate:EVDO:MEASurement<Instance>:MEValuation:LIST:ACP:EXTended:ACPP<AcpPlus>:AVERage
```

class Average

Average commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

calculate(*acpPlus=<AcpPlus.Default: -1>*) → List[float]

```
# SCPI: CALCulate:EVDO:MEASurement<instance>:MEValuation:LIST:ACP:EXTended:ACPP
↪<freqpoint>:AVERage
value: List[float] = driver.multiEval.listPy.acp.extended.acpp.average.
↪calculate(acpPlus = repcap.AcpPlus.Default)
```

Returns adjacent channel power (statistical) values for the selected off-center channels and all active list mode segments. The mnemonic ACPM refers to the channels in the 'minus' direction, the mnemonic ACPP to the channels in the 'plus' direction relative to the current channel. The offset is selected with the <freqpoint> suffix (±20 channels) . The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

param acpPlus optional repeated capability selector. Default value: Ch1 (settable in the interface 'Acpp')

return acpm: float Comma-separated list of values, one per active segment. Range: -999 dB to 999 dB, Unit: dB

fetch(*acpPlus=<AcpPlus.Default: -1>*) → List[float]

```
# SCPI: FETCh:EVDO:MEASurement<instance>:MEValuation:LIST:ACP:EXTended:ACPP
↪<freqpoint>:AVERage
value: List[float] = driver.multiEval.listPy.acp.extended.acpp.average.
↪fetch(acpPlus = repcap.AcpPlus.Default)
```

Returns adjacent channel power (statistical) values for the selected off-center channels and all active list mode segments. The mnemonic ACPM refers to the channels in the 'minus' direction, the mnemonic ACPP to the channels in the 'plus' direction relative to the current channel. The offset is selected with the <freqpoint> suffix (±20 channels) . The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

param acpPlus optional repeated capability selector. Default value: Ch1 (settable in the interface 'Acpp')

return acpm: float Comma-separated list of values, one per active segment. Range: -999 dB to 999 dB, Unit: dB

7.4.7.6.4.8 Maximum

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:ACP:EXTended:ACPP<AcpPlus>:MAXimum
CALCulate:EVDO:MEASurement<Instance>:MEValuation:LIST:ACP:EXTended:ACPP<AcpPlus>:MAXimum
```

class Maximum

Maximum commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

calculate(*acpPlus=<AcpPlus.Default: -1>*) → List[float]

```
# SCPI: CALCulate:EVDO:MEASurement<instance>:MEValuation:LIST:ACP:EXTended:ACPP
↪<freqpoint>:MAXimum
value: List[float] = driver.multiEval.listPy.acp.extended.acpp.maximum.
↪calculate(acpPlus = repcap.AcpPlus.Default)
```

Returns adjacent channel power (statistical) values for the selected off-center channels and all active list mode segments. The mnemonic ACPM refers to the channels in the 'minus' direction, the mnemonic ACPP to the channels in the 'plus' direction relative to the current channel. The offset is selected with the <freqpoint> suffix (±20 channels) . The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

param acpPlus optional repeated capability selector. Default value: Ch1 (settable in the interface 'Acpp')

return acpm: float Comma-separated list of values, one per active segment. Range: -999 dB to 999 dB, Unit: dB

fetch(*acpPlus=<AcpPlus.Default: -1>*) → List[float]

```
# SCPI: FETCh:EVDO:MEASurement<instance>:MEValuation:LIST:ACP:EXTended:ACPP
↪<freqpoint>:MAXimum
value: List[float] = driver.multiEval.listPy.acp.extended.acpp.maximum.
↪fetch(acpPlus = repcap.AcpPlus.Default)
```

Returns adjacent channel power (statistical) values for the selected off-center channels and all active list mode segments. The mnemonic ACPM refers to the channels in the 'minus' direction, the mnemonic ACPP to the channels in the 'plus' direction relative to the current channel. The offset is selected with the <freqpoint> suffix (±20 channels) . The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

param acpPlus optional repeated capability selector. Default value: Ch1 (settable in the interface 'Acpp')

return acpm: float Comma-separated list of values, one per active segment. Range: -999 dB to 999 dB, Unit: dB

7.4.7.6.4.9 Current

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:ACP:EXTended:CURRent
CALCulate:EVDO:MEASurement<Instance>:MEValuation:LIST:ACP:EXTended:CURRent
```

class Current

Current commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: decimal 'Conventions and General Information'. In list mode, a zero reliability indicator indicates that the results in all measured segments are valid. A non-zero value indicates that an error occurred in at least one of the measured segments.
- Seg_Reliability: List[int]: decimal The segment reliability indicates whether one of the following exceptions occurred in this segment: 0: No error 3: Signal overflow 4: Signal low 8: Synchronization error If a combination of exceptions occurs, the most severe error is indicated.
- Acp: List[float]: No parameter help available
- Ms_Power_Wide: List[enums.ResultStatus2]: No parameter help available
- Ms_Power_Narrow: List[enums.ResultStatus2]: No parameter help available
- Out_Of_Tol_Count: List[enums.ResultStatus2]: Out of tolerance result, i.e. percentage of measurement intervals of the statistic count (see [CMDLINK: CONFigure:EVDO:MEASi:MEValuation:SCOunt:SPECtrum CMDLINK]) exceeding the specified limits, see 'Limits (Spectrum) '. Range: 0 % to 100 %, Unit: %
- Cur_Stat_Count: List[enums.ResultStatus2]: decimal Number of evaluated valid slots in this segment.

class FetchStruct

Response structure. Fields:

- Reliability: int: decimal 'Conventions and General Information'. In list mode, a zero reliability indicator indicates that the results in all measured segments are valid. A non-zero value indicates that an error occurred in at least one of the measured segments.

- Seg_Reliability: List[int]: decimal The segment reliability indicates whether one of the following exceptions occurred in this segment: 0: No error 3: Signal overflow 4: Signal low 8: Synchronization error If a combination of exceptions occurs, the most severe error is indicated.
- Acp: List[float]: No parameter help available
- Ms_Power_Wide: List[float]: No parameter help available
- Ms_Power_Narrow: List[float]: No parameter help available
- Out_Of_Tol_Count: List[float]: Out of tolerance result, i.e. percentage of measurement intervals of the statistic count (see [CMDLINK: CONFigure:EVDO:MEASi:MEValuation:SCOunt:SPECtrum CMDLINK]) exceeding the specified limits, see 'Limits (Spectrum) '. Range: 0 % to 100 %, Unit: %
- Cur_Stat_Count: List[int]: decimal Number of evaluated valid slots in this segment.

calculate() → CalculateStruct

```
# SCPI: CALCulate:EVDO:MEASurement<instance>
↪:MEValuation:LIST:ACP:EXTended:CURRent
value: CalculateStruct = driver.multiEval.listPy.acp.extended.current.
↪calculate()
```

Returns all ACP value results in list mode. To define the statistical length for AVERage, MAXimum, MINimum calculation and enable the calculation of the results use the command method RsCmwEvdoMeas.Configure.MultiEval.ListPy.Segment. Spectrum.set. The ranges indicated below apply to all results except standard deviation results. The minimum for standard deviation results equals 0. The maximum equals the width of the indicated range divided by two. Exceptions are explicitly stated. The values listed below in curly brackets {} are returned for each active segment: {...}seg 1, {...}seg 2, ..., {. ..}seg n. The number of active segments n is determined by method RsCmwEvdoMeas.Configure.MultiEval.ListPy.count. 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. The number to the left of each result parameter is provided for easy identification of the parameter position within the result array. For the out of tolerance and code channel filter match ratio, results retrieved via the CURRent, AVERage and MAXimum commands are identical.

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

fetch() → FetchStruct

```
# SCPI: FETCh:EVDO:MEASurement<instance>:MEValuation:LIST:ACP:EXTended:CURRent
value: FetchStruct = driver.multiEval.listPy.acp.extended.current.fetch()
```

Returns all ACP value results in list mode. To define the statistical length for AVERage, MAXimum, MINimum calculation and enable the calculation of the results use the command method RsCmwEvdoMeas.Configure.MultiEval.ListPy.Segment. Spectrum.set. The ranges indicated below apply to all results except standard deviation results. The minimum for standard deviation results equals 0. The maximum equals the width of the indicated range divided by two. Exceptions are explicitly stated. The values listed below in curly brackets {} are returned for each active segment: {...}seg 1, {...}seg 2, ..., {. ..}seg n. The number of active segments n is determined by method RsCmwEvdoMeas.Configure.MultiEval.ListPy.count. 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. The number to the left of each result parameter is provided for easy identification of the parameter position within the result array. For the out of tolerance and code channel filter match ratio, results retrieved via the CURRent, AVERage and MAXimum commands are identical.

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

7.4.7.6.4.10 Average

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:ACP:EXTended:AVERage
CALCulate:EVDO:MEASurement<Instance>:MEValuation:LIST:ACP:EXTended:AVERage
```

class Average

Average commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: decimal 'Conventions and General Information'. In list mode, a zero reliability indicator indicates that the results in all measured segments are valid. A non-zero value indicates that an error occurred in at least one of the measured segments.
- Seg_Reliability: List[int]: decimal The segment reliability indicates whether one of the following exceptions occurred in this segment: 0: No error 3: Signal overflow 4: Signal low 8: Synchronization error If a combination of exceptions occurs, the most severe error is indicated.
- Acp: List[float]: No parameter help available
- Ms_Power_Wide: List[enums.ResultStatus2]: No parameter help available
- Ms_Power_Narrow: List[enums.ResultStatus2]: No parameter help available
- Out_Of_Tol_Count: List[enums.ResultStatus2]: Out of tolerance result, i.e. percentage of measurement intervals of the statistic count (see [CMDLINK: CONFigure:EVDO:MEASi:MEValuation:SCOunt:SPECtrum CMDLINK]) exceeding the specified limits, see 'Limits (Spectrum) '. Range: 0 % to 100 %, Unit: %
- Cur_Stat_Count: List[enums.ResultStatus2]: decimal Number of evaluated valid slots in this segment.

class FetchStruct

Response structure. Fields:

- Reliability: int: decimal 'Conventions and General Information'. In list mode, a zero reliability indicator indicates that the results in all measured segments are valid. A non-zero value indicates that an error occurred in at least one of the measured segments.
- Seg_Reliability: List[int]: decimal The segment reliability indicates whether one of the following exceptions occurred in this segment: 0: No error 3: Signal overflow 4: Signal low 8: Synchronization error If a combination of exceptions occurs, the most severe error is indicated.
- Acp: List[float]: No parameter help available
- Ms_Power_Wide: List[float]: No parameter help available
- Ms_Power_Narrow: List[float]: No parameter help available
- Out_Of_Tol_Count: List[float]: Out of tolerance result, i.e. percentage of measurement intervals of the statistic count (see [CMDLINK: CONFigure:EVDO:MEASi:MEValuation:SCOunt:SPECtrum CMDLINK]) exceeding the specified limits, see 'Limits (Spectrum) '. Range: 0 % to 100 %, Unit: %
- Cur_Stat_Count: List[int]: decimal Number of evaluated valid slots in this segment.

calculate() → CalculateStruct

```
# SCPI: CALCulate:EVDO:MEASurement<instance>
↪:MEValuation:LIST:ACP:EXTended:AVERage
value: CalculateStruct = driver.multiEval.listPy.acp.extended.average.
↪calculate()
```

Returns all ACP value results in list mode. To define the statistical length for AVERage, MAXimum, MINimum calculation and enable the calculation of the results use the command method RsCmwEvdoMeas.Configure.MultiEval.ListPy.Segment. Spectrum.set. The ranges indicated below apply to all results except standard deviation results. The minimum for standard deviation results equals 0. The maximum equals the width of the indicated range divided by two. Exceptions are explicitly stated. The values listed below in curly brackets {} are returned for each active segment: {...}seg 1, {...}seg 2, ..., {. ..}seg n. The number of active segments n is determined by method RsCmwEvdoMeas.Configure.MultiEval.ListPy.count. 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. The number to the left of each result parameter is provided for easy identification of the parameter position within the result array. For the out of tolerance and code channel filter match ratio, results retrieved via the CURRent, AVERage and MAXimum commands are identical.

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

fetch() → FetchStruct

```
# SCPI: FETCh:EVDO:MEASurement<instance>:MEValuation:LIST:ACP:EXTended:AVERage
value: FetchStruct = driver.multiEval.listPy.acp.extended.average.fetch()
```

Returns all ACP value results in list mode. To define the statistical length for AVERage, MAXimum, MINimum calculation and enable the calculation of the results use the command method RsCmwEvdoMeas.Configure.MultiEval.ListPy.Segment. Spectrum.set. The ranges indicated below apply to all results except standard deviation results. The minimum for standard deviation results equals 0. The maximum equals the width of the indicated range divided by two. Exceptions are explicitly stated. The values listed below in curly brackets {} are returned for each active segment: {...}seg 1, {...}seg 2, ..., {. ..}seg n. The number of active segments n is determined by method RsCmwEvdoMeas.Configure.MultiEval.ListPy.count. 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. The number to the left of each result parameter is provided for easy identification of the parameter position within the result array. For the out of tolerance and code channel filter match ratio, results retrieved via the CURRent, AVERage and MAXimum commands are identical.

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

7.4.7.6.4.11 Maximum

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:ACP:EXTended:MAXimum
CALCulate:EVDO:MEASurement<Instance>:MEValuation:LIST:ACP:EXTended:MAXimum
```

class Maximum

Maximum commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: decimal 'Conventions and General Information'. In list mode, a zero reliability indicator indicates that the results in all measured segments are valid. A non-zero value indicates that an error occurred in at least one of the measured segments.
- Seg_Reliability: List[int]: decimal The segment reliability indicates whether one of the following exceptions occurred in this segment: 0: No error 3: Signal overflow 4: Signal low 8: Synchronization error If a combination of exceptions occurs, the most severe error is indicated.
- Acp: List[float]: No parameter help available
- Ms_Power_Wide: List[enums.ResultStatus2]: No parameter help available
- Ms_Power_Narrow: List[enums.ResultStatus2]: No parameter help available
- Out_Of_Tol_Count: List[enums.ResultStatus2]: Out of tolerance result, i.e. percentage of measurement intervals of the statistic count (see [CMDLINK: CONFigure:EVDO:MEASi:MEValuation:SCOunt:SPECtrum CMDLINK]) exceeding the specified limits, see 'Limits (Spectrum) '. Range: 0 % to 100 %, Unit: %
- Cur_Stat_Count: List[enums.ResultStatus2]: decimal Number of evaluated valid slots in this segment.

class FetchStruct

Response structure. Fields:

- Reliability: int: decimal 'Conventions and General Information'. In list mode, a zero reliability indicator indicates that the results in all measured segments are valid. A non-zero value indicates that an error occurred in at least one of the measured segments.
- Seg_Reliability: List[int]: decimal The segment reliability indicates whether one of the following exceptions occurred in this segment: 0: No error 3: Signal overflow 4: Signal low 8: Synchronization error If a combination of exceptions occurs, the most severe error is indicated.
- Acp: List[float]: No parameter help available
- Ms_Power_Wide: List[float]: No parameter help available
- Ms_Power_Narrow: List[float]: No parameter help available
- Out_Of_Tol_Count: List[float]: Out of tolerance result, i.e. percentage of measurement intervals of the statistic count (see [CMDLINK: CONFigure:EVDO:MEASi:MEValuation:SCOunt:SPECtrum CMDLINK]) exceeding the specified limits, see 'Limits (Spectrum) '. Range: 0 % to 100 %, Unit: %
- Cur_Stat_Count: List[int]: decimal Number of evaluated valid slots in this segment.

calculate() → CalculateStruct

```
# SCPI: CALCulate:EVDO:MEASurement<instance>
↪:MEValuation:LIST:ACP:EXTended:MAXimum
value: CalculateStruct = driver.multiEval.listPy.acp.extended.maximum.
↪calculate()
```

Returns all ACP value results in list mode. To define the statistical length for AVERage, MAXimum, MINimum calculation and enable the calculation of the results use the command method RsCmwEvdoMeas.Configure.MultiEval.ListPy.Segment. Spectrum.set. The ranges indicated below apply to all results except standard deviation results. The minimum for standard deviation results equals 0. The maximum equals the width of the indicated range divided by two. Exceptions are explicitly stated. The values listed below in curly brackets {} are returned for each active segment: {...}seg 1, {...}seg 2, ..., {. ..}seg n. The number of active segments n is determined by method RsCmwEvdoMeas.Configure.MultiEval.ListPy.count. 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. The number to the left of each result parameter is provided for easy identification of the parameter position

within the result array. For the out of tolerance and code channel filter match ratio, results retrieved via the CURRent, AVERage and MAXimum commands are identical.

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

fetch() → FetchStruct

```
# SCPI: FETCh:EVDO:MEASurement<instance>:MEValuation:LIST:ACP:EXTended:MAXimum
value: FetchStruct = driver.multiEval.listPy.acp.extended.maximum.fetch()
```

Returns all ACP value results in list mode. To define the statistical length for AVERage, MAXimum, MINimum calculation and enable the calculation of the results use the command method RsCmwEvdoMeas.Configure.MultiEval.ListPy.Segment. Spectrum.set. The ranges indicated below apply to all results except standard deviation results. The minimum for standard deviation results equals 0. The maximum equals the width of the indicated range divided by two. Exceptions are explicitly stated. The values listed below in curly brackets {} are returned for each active segment: {...}seg 1, {...}seg 2, ..., {. ..}seg n. The number of active segments n is determined by method RsCmwEvdoMeas.Configure.MultiEval.ListPy.count. 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. The number to the left of each result parameter is provided for easy identification of the parameter position within the result array. For the out of tolerance and code channel filter match ratio, results retrieved via the CURRent, AVERage and MAXimum commands are identical.

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

7.4.7.6.4.12 Minimum

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:ACP:EXTended:MINimum
CALCulate:EVDO:MEASurement<Instance>:MEValuation:LIST:ACP:EXTended:MINimum
```

class Minimum

Minimum commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: decimal 'Conventions and General Information'. In list mode, a zero reliability indicator indicates that the results in all measured segments are valid. A non-zero value indicates that an error occurred in at least one of the measured segments.
- Seg_Reliability: List[int]: decimal The segment reliability indicates whether one of the following exceptions occurred in this segment: 0: No error 3: Signal overflow 4: Signal low 8: Synchronization error If a combination of exceptions occurs, the most severe error is indicated.
- Acp: List[float]: No parameter help available
- Ms_Power_Wide: List[enums.ResultStatus2]: No parameter help available
- Ms_Power_Narrow: List[enums.ResultStatus2]: No parameter help available
- Out_Of_Tol_Count: List[enums.ResultStatus2]: Out of tolerance result, i.e. percentage of measurement intervals of the statistic count (see [CMDLINK: CONFigure:EVDO:MEASi:MEValuation:SCOunt:SPECtrum CMDLINK]) exceeding the specified limits, see 'Limits (Spectrum) '. Range: 0 % to 100 %, Unit: %

- Cur_Stat_Count: List[enums.ResultStatus2]: decimal Number of evaluated valid slots in this segment.

class FetchStruct

Response structure. Fields:

- Reliability: int: decimal 'Conventions and General Information'. In list mode, a zero reliability indicator indicates that the results in all measured segments are valid. A non-zero value indicates that an error occurred in at least one of the measured segments.
- Seg_Reliability: List[int]: decimal The segment reliability indicates whether one of the following exceptions occurred in this segment: 0: No error 3: Signal overflow 4: Signal low 8: Synchronization error If a combination of exceptions occurs, the most severe error is indicated.
- Acp: List[float]: No parameter help available
- Ms_Power_Wide: List[float]: No parameter help available
- Ms_Power_Narrow: List[float]: No parameter help available
- Out_Of_Tol_Count: List[float]: Out of tolerance result, i.e. percentage of measurement intervals of the statistic count (see [CMDLINK: CONFigure:EVDO:MEASi:MEValuation:SCOunt:SPECtrum CMDLINK]) exceeding the specified limits, see 'Limits (Spectrum) '. Range: 0 % to 100 %, Unit: %
- Cur_Stat_Count: List[int]: decimal Number of evaluated valid slots in this segment.

calculate() → CalculateStruct

```
# SCPI: CALCulate:EVDO:MEASurement<instance>
↪:MEValuation:LIST:ACP:EXTended:MINimum
value: CalculateStruct = driver.multiEval.listPy.acp.extended.minimum.
↪calculate()
```

Returns all ACP value results in list mode. To define the statistical length for AVERage, MAXimum, MINimum calculation and enable the calculation of the results use the command method RsCmwEvdoMeas.Configure.MultiEval.ListPy.Segment. Spectrum.set. The ranges indicated below apply to all results except standard deviation results. The minimum for standard deviation results equals 0. The maximum equals the width of the indicated range divided by two. Exceptions are explicitly stated. The values listed below in curly brackets {} are returned for each active segment: {...}seg 1, {...}seg 2, ..., {. ..}seg n. The number of active segments n is determined by method RsCmwEvdoMeas.Configure.MultiEval.ListPy.count. 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. The number to the left of each result parameter is provided for easy identification of the parameter position within the result array. For the out of tolerance and code channel filter match ratio, results retrieved via the CURRent, AVERage and MAXimum commands are identical.

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

fetch() → FetchStruct

```
# SCPI: FETCh:EVDO:MEASurement<instance>:MEValuation:LIST:ACP:EXTended:MINimum
value: FetchStruct = driver.multiEval.listPy.acp.extended.minimum.fetch()
```

Returns all ACP value results in list mode. To define the statistical length for AVERage, MAXimum, MINimum calculation and enable the calculation of the results use the command method RsCmwEvdoMeas.Configure.MultiEval.ListPy.Segment. Spectrum.set. The ranges indicated below apply to all results except standard deviation results. The minimum for standard deviation results equals 0. The maximum equals the width of the indicated range divided by two. Exceptions are explicitly stated. The values listed below in curly brackets {} are returned for each active segment: {...}seg 1,

{...}seg 2, ..., {. ..}seg n. The number of active segments n is determined by method RsCmwEvdoMeas.Configure.MultiEval.ListPy.count. 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. The number to the left of each result parameter is provided for easy identification of the parameter position within the result array. For the out of tolerance and code channel filter match ratio, results retrieved via the CURRent, AVERage and MAXimum commands are identical.

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

7.4.7.6.4.13 StandardDev

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:ACP:EXTended:SDEViation
CALCulate:EVDO:MEASurement<Instance>:MEValuation:LIST:ACP:EXTended:SDEViation
```

class StandardDev

StandardDev commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: decimal 'Conventions and General Information'. In list mode, a zero reliability indicator indicates that the results in all measured segments are valid. A non-zero value indicates that an error occurred in at least one of the measured segments.
- Seg_Reliability: List[int]: decimal The segment reliability indicates whether one of the following exceptions occurred in this segment: 0: No error 3: Signal overflow 4: Signal low 8: Synchronization error If a combination of exceptions occurs, the most severe error is indicated.
- Acp: List[float]: No parameter help available
- Ms_Power_Wide: List[enums.ResultStatus2]: No parameter help available
- Ms_Power_Narrow: List[enums.ResultStatus2]: No parameter help available
- Out_Of_Tol_Count: List[enums.ResultStatus2]: Out of tolerance result, i.e. percentage of measurement intervals of the statistic count (see [CMDLINK: CONFigure:EVDO:MEASi:MEValuation:SCOunt:SPECtrum CMDLINK]) exceeding the specified limits, see 'Limits (Spectrum) '. Range: 0 % to 100 %, Unit: %
- Cur_Stat_Count: List[enums.ResultStatus2]: decimal Number of evaluated valid slots in this segment.

class FetchStruct

Response structure. Fields:

- Reliability: int: decimal 'Conventions and General Information'. In list mode, a zero reliability indicator indicates that the results in all measured segments are valid. A non-zero value indicates that an error occurred in at least one of the measured segments.
- Seg_Reliability: List[int]: decimal The segment reliability indicates whether one of the following exceptions occurred in this segment: 0: No error 3: Signal overflow 4: Signal low 8: Synchronization error If a combination of exceptions occurs, the most severe error is indicated.
- Acp: List[float]: No parameter help available
- Ms_Power_Wide: List[float]: No parameter help available
- Ms_Power_Narrow: List[float]: No parameter help available

- Out_Of_Tol_Count: List[float]: Out of tolerance result, i.e. percentage of measurement intervals of the statistic count (see [CMDLINK: CONFigure:EVDO:MEASi:MEValuation:SCOunt:SPECtrum CMDLINK]) exceeding the specified limits, see 'Limits (Spectrum) '. Range: 0 % to 100 %, Unit: %
- Cur_Stat_Count: List[int]: decimal Number of evaluated valid slots in this segment.

calculate() → CalculateStruct

```
# SCPI: CALCulate:EVDO:MEASurement<instance>
↪:MEValuation:LIST:ACP:EXTended:SDEViation
value: CalculateStruct = driver.multiEval.listPy.acp.extended.standardDev.
↪calculate()
```

Returns all ACP value results in list mode. To define the statistical length for AVERage, MAXimum, MINimum calculation and enable the calculation of the results use the command method RsCmwEvdoMeas.Configure.MultiEval.ListPy.Segment. Spectrum.set. The ranges indicated below apply to all results except standard deviation results. The minimum for standard deviation results equals 0. The maximum equals the width of the indicated range divided by two. Exceptions are explicitly stated. The values listed below in curly brackets {} are returned for each active segment: {...}seg 1, {...}seg 2, ..., {. ..}seg n. The number of active segments n is determined by method RsCmwEvdoMeas.Configure.MultiEval.ListPy.count. 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. The number to the left of each result parameter is provided for easy identification of the parameter position within the result array. For the out of tolerance and code channel filter match ratio, results retrieved via the CURRent, AVERage and MAXimum commands are identical.

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

fetch() → FetchStruct

```
# SCPI: FETCh:EVDO:MEASurement<instance>
↪:MEValuation:LIST:ACP:EXTended:SDEViation
value: FetchStruct = driver.multiEval.listPy.acp.extended.standardDev.fetch()
```

Returns all ACP value results in list mode. To define the statistical length for AVERage, MAXimum, MINimum calculation and enable the calculation of the results use the command method RsCmwEvdoMeas.Configure.MultiEval.ListPy.Segment. Spectrum.set. The ranges indicated below apply to all results except standard deviation results. The minimum for standard deviation results equals 0. The maximum equals the width of the indicated range divided by two. Exceptions are explicitly stated. The values listed below in curly brackets {} are returned for each active segment: {...}seg 1, {...}seg 2, ..., {. ..}seg n. The number of active segments n is determined by method RsCmwEvdoMeas.Configure.MultiEval.ListPy.count. 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. The number to the left of each result parameter is provided for easy identification of the parameter position within the result array. For the out of tolerance and code channel filter match ratio, results retrieved via the CURRent, AVERage and MAXimum commands are identical.

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

7.4.7.6.5 Acpp<AcpPlus>

RepCap Settings

```
# Range: Ch1 .. Ch20
rc = driver.multiEval.listPy.acp.acpp.repcap_acpPlus_get()
driver.multiEval.listPy.acp.acpp.repcap_acpPlus_set(repcap.AcpPlus.Ch1)
```

class Acpp

Acpp commands group definition. 6 total commands, 3 Sub-groups, 0 group commands Repeated Capability: AcpPlus, default value after init: AcpPlus.Ch1

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.multiEval.listPy.acp.acpp.clone()
```

Subgroups

7.4.7.6.5.1 Current

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:ACP:ACPP<AcpPlus>:CURRent
CALCulate:EVDO:MEASurement<Instance>:MEValuation:LIST:ACP:ACPP<AcpPlus>:CURRent
```

class Current

Current commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

calculate(*acpPlus=<AcpPlus.Default: -1>*) → List[float]

```
# SCPI: CALCulate:EVDO:MEASurement<instance>:MEValuation:LIST:ACP:ACPP
↪<freqpoint>:CURRent
value: List[float] = driver.multiEval.listPy.acp.acpp.current.calculate(acpPlus␣
↪= repcap.AcpPlus.Default)
```

Returns adjacent channel power (statistical) values for the selected off-center channels and all active list mode segments. The mnemonic ACPM refers to the channels in the 'minus' direction, the mnemonic ACPP to the channels in the 'plus' direction relative to the current channel. The offset is selected with the <freqpoint> suffix (±10 channels) . The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

param acpPlus optional repeated capability selector. Default value: Ch1 (settable in the interface 'Acpp')

return acpm: float Comma-separated list of values, one per active segment. Range: -999 dB to 999 dB, Unit: dB

fetch(*acpPlus=<AcpPlus.Default: -1>*) → List[float]

```
# SCPI: FETCh:EVDO:MEASurement<instance>:MEValuation:LIST:ACP:ACPP<freqpoint>
↪:CURRent
value: List[float] = driver.multiEval.listPy.acp.acpp.current.fetch(acpPlus =␣
↪repcap.AcpPlus.Default)
```

Returns adjacent channel power (statistical) values for the selected off-center channels and all active list mode segments. The mnemonic ACPM refers to the channels in the 'minus' direction, the mnemonic ACPP to the channels in the 'plus' direction relative to the current channel. The offset is selected with the <freqpoint> suffix (±10 channels) . The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

param acpPlus optional repeated capability selector. Default value: Ch1 (settable in the interface 'Acpp')

return acpm: float Comma-separated list of values, one per active segment. Range: -999 dB to 999 dB, Unit: dB

7.4.7.6.5.2 Average

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:ACP:ACPP<AcpPlus>:AVERage
CALCulate:EVDO:MEASurement<Instance>:MEValuation:LIST:ACP:ACPP<AcpPlus>:AVERage
```

class Average

Average commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

calculate(*acpPlus=<AcpPlus.Default: -1>*) → List[float]

```
# SCPI: CALCulate:EVDO:MEASurement<instance>:MEValuation:LIST:ACP:ACPP
↪<freqpoint>:AVERage
value: List[float] = driver.multiEval.listPy.acp.acpp.average.calculate(acpPlus␣
↪= repcap.AcpPlus.Default)
```

Returns adjacent channel power (statistical) values for the selected off-center channels and all active list mode segments. The mnemonic ACPM refers to the channels in the 'minus' direction, the mnemonic ACPP to the channels in the 'plus' direction relative to the current channel. The offset is selected with the <freqpoint> suffix (±10 channels) . The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

param acpPlus optional repeated capability selector. Default value: Ch1 (settable in the interface 'Acpp')

return acpm: float Comma-separated list of values, one per active segment. Range: -999 dB to 999 dB, Unit: dB

fetch(*acpPlus=<AcpPlus.Default: -1>*) → List[float]

```
# SCPI: FETCh:EVDO:MEASurement<instance>:MEValuation:LIST:ACP:ACPP<freqpoint>
↪:AVERage
value: List[float] = driver.multiEval.listPy.acp.acpp.average.fetch(acpPlus =␣
↪repcap.AcpPlus.Default)
```

Returns adjacent channel power (statistical) values for the selected off-center channels and all active list mode segments. The mnemonic ACPM refers to the channels in the 'minus' direction, the mnemonic ACPP to the channels in the 'plus' direction relative to the current channel. The offset is selected with the <freqpoint> suffix (±10 channels) . The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

param acpPlus optional repeated capability selector. Default value: Ch1 (settable in the interface 'Acpp')

return acpm: float Comma-separated list of values, one per active segment. Range: -999 dB to 999 dB, Unit: dB

7.4.7.6.5.3 Maximum

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:ACP:ACPP<AcpPlus>:MAXimum
CALCulate:EVDO:MEASurement<Instance>:MEValuation:LIST:ACP:ACPP<AcpPlus>:MAXimum
```

class Maximum

Maximum commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

calculate(*acpPlus=<AcpPlus.Default: -1>*) → List[float]

```
# SCPI: CALCulate:EVDO:MEASurement<instance>:MEValuation:LIST:ACP:ACPP
↪<freqpoint>:MAXimum
value: List[float] = driver.multiEval.listPy.acp.acpp.maximum.calculate(acpPlus␣
↪= repcap.AcpPlus.Default)
```

Returns adjacent channel power (statistical) values for the selected off-center channels and all active list mode segments. The mnemonic ACPM refers to the channels in the 'minus' direction, the mnemonic ACPP to the channels in the 'plus' direction relative to the current channel. The offset is selected with the <freqpoint> suffix (±10 channels) . The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

param acpPlus optional repeated capability selector. Default value: Ch1 (settable in the interface 'Acpp')

return acpm: float Comma-separated list of values, one per active segment. Range: -999 dB to 999 dB, Unit: dB

fetch(*acpPlus=<AcpPlus.Default: -1>*) → List[float]

```
# SCPI: FETCh:EVDO:MEASurement<instance>:MEValuation:LIST:ACP:ACPP<freqpoint>
↪:MAXimum
value: List[float] = driver.multiEval.listPy.acp.acpp.maximum.fetch(acpPlus =␣
↪repcap.AcpPlus.Default)
```

Returns adjacent channel power (statistical) values for the selected off-center channels and all active list mode segments. The mnemonic ACPM refers to the channels in the 'minus' direction, the mnemonic ACPP to the channels in the 'plus' direction relative to the current channel. The offset is selected with the <freqpoint> suffix (±10 channels) . The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

param acpPlus optional repeated capability selector. Default value: Ch1 (settable in the interface 'Acpp')

return acpm: float Comma-separated list of values, one per active segment. Range: -999 dB to 999 dB, Unit: dB

7.4.7.6.6 Npow

class Npow

Npow commands group definition. 10 total commands, 5 Sub-groups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.multiEval.listPy.acp.npow.clone()
```

Subgroups

7.4.7.6.6.1 Current

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:ACP:NPOW:CURRent
CALCulate:EVDO:MEASurement<Instance>:MEValuation:LIST:ACP:NPOW:CURRent
```

class Current

Current commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

calculate() → List[RsCmwEvdoMeas.enums.ResultStatus2]

```
# SCPI: CALCulate:EVDO:MEASurement<instance>:MEValuation:LIST:ACP:NPOW:CURRent
value: List[enums.ResultStatus2] = driver.multiEval.listPy.acp.npow.current.
↪calculate()
```

Returns narrowband and wideband (statistical) power values in the ACP measurement for all active list mode segments. The narrowband filter is 1.23 MHz, the wideband filter is 8 MHz wide. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return narrow_power: float Comma-separated list of values, one per active segment. Range: -100 dBm to 50 dBm, Unit: dBm

fetch() → List[float]

```
# SCPI: FETCh:EVDO:MEASurement<instance>:MEValuation:LIST:ACP:NPOW:CURRent
value: List[float] = driver.multiEval.listPy.acp.npow.current.fetch()
```

Returns narrowband and wideband (statistical) power values in the ACP measurement for all active list mode segments. The narrowband filter is 1.23 MHz, the wideband filter is 8 MHz wide. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return narrow_power: float Comma-separated list of values, one per active segment. Range: -100 dBm to 50 dBm, Unit: dBm

7.4.7.6.6.2 Average

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:ACP:NPOW:AVERage
CALCulate:EVDO:MEASurement<Instance>:MEValuation:LIST:ACP:NPOW:AVERage
```

class Average

Average commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

calculate() → List[RsCmwEvdoMeas.enums.ResultStatus2]

```
# SCPI: CALCulate:EVDO:MEASurement<instance>:MEValuation:LIST:ACP:NPOW:AVERage
value: List[enums.ResultStatus2] = driver.multiEval.listPy.acp.npow.average.
↪calculate()
```

Returns narrowband and wideband (statistical) power values in the ACP measurement for all active list mode segments. The narrowband filter is 1.23 MHz, the wideband filter is 8 MHz wide. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return narrow_power: float Comma-separated list of values, one per active segment. Range: -100 dBm to 50 dBm, Unit: dBm

fetch() → List[float]

```
# SCPI: FETCh:EVDO:MEASurement<instance>:MEValuation:LIST:ACP:NPOW:AVERage
value: List[float] = driver.multiEval.listPy.acp.npow.average.fetch()
```

Returns narrowband and wideband (statistical) power values in the ACP measurement for all active list mode segments. The narrowband filter is 1.23 MHz, the wideband filter is 8 MHz wide. The values described

below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return narrow_power: float Comma-separated list of values, one per active segment. Range: -100 dBm to 50 dBm, Unit: dBm

7.4.7.6.6.3 Maximum

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:ACP:NPOW:MAXimum
CALCulate:EVDO:MEASurement<Instance>:MEValuation:LIST:ACP:NPOW:MAXimum
```

class Maximum

Maximum commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

calculate() → List[RsCmwEvdoMeas.enums.ResultStatus2]

```
# SCPI: CALCulate:EVDO:MEASurement<instance>:MEValuation:LIST:ACP:NPOW:MAXimum
value: List[enums.ResultStatus2] = driver.multiEval.listPy.acp.npow.maximum.
↪calculate()
```

Returns narrowband and wideband (statistical) power values in the ACP measurement for all active list mode segments. The narrowband filter is 1.23 MHz, the wideband filter is 8 MHz wide. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return narrow_power: float Comma-separated list of values, one per active segment. Range: -100 dBm to 50 dBm, Unit: dBm

fetch() → List[float]

```
# SCPI: FETCh:EVDO:MEASurement<instance>:MEValuation:LIST:ACP:NPOW:MAXimum
value: List[float] = driver.multiEval.listPy.acp.npow.maximum.fetch()
```

Returns narrowband and wideband (statistical) power values in the ACP measurement for all active list mode segments. The narrowband filter is 1.23 MHz, the wideband filter is 8 MHz wide. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return narrow_power: float Comma-separated list of values, one per active segment. Range: -100 dBm to 50 dBm, Unit: dBm

7.4.7.6.6.4 Minimum

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:ACP:NPOW:MINimum
CALCulate:EVDO:MEASurement<Instance>:MEValuation:LIST:ACP:NPOW:MINimum
```

class Minimum

Minimum commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

calculate() → List[RsCmwEvdoMeas.enums.ResultStatus2]

```
# SCPI: CALCulate:EVDO:MEASurement<instance>:MEValuation:LIST:ACP:NPOW:MINimum
value: List[enums.ResultStatus2] = driver.multiEval.listPy.acp.npow.minimum.
↪calculate()
```

Returns narrowband and wideband (statistical) power values in the ACP measurement for all active list mode segments. The narrowband filter is 1.23 MHz, the wideband filter is 8 MHz wide. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return narrow_power: float Comma-separated list of values, one per active segment. Range: -100 dBm to 50 dBm, Unit: dBm

fetch() → List[float]

```
# SCPI: FETCh:EVDO:MEASurement<instance>:MEValuation:LIST:ACP:NPOW:MINimum
value: List[float] = driver.multiEval.listPy.acp.npow.minimum.fetch()
```

Returns narrowband and wideband (statistical) power values in the ACP measurement for all active list mode segments. The narrowband filter is 1.23 MHz, the wideband filter is 8 MHz wide. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return narrow_power: float Comma-separated list of values, one per active segment. Range: -100 dBm to 50 dBm, Unit: dBm

7.4.7.6.6.5 StandardDev

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:ACP:NPOW:SDEViation
CALCulate:EVDO:MEASurement<Instance>:MEValuation:LIST:ACP:NPOW:SDEViation
```

class StandardDev

StandardDev commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

calculate() → List[RsCmwEvdoMeas.enums.ResultStatus2]

```
# SCPI: CALCulate:EVDO:MEASurement<instance>
↪:MEValuation:LIST:ACP:NPOW:SDEViation
value: List[enums.ResultStatus2] = driver.multiEval.listPy.acp.npow.standardDev.
↪calculate()
```

Returns narrowband and wideband (statistical) power values in the ACP measurement for all active list mode segments. The narrowband filter is 1.23 MHz, the wideband filter is 8 MHz wide. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return narrow_power: float Comma-separated list of values, one per active segment. Range: -100 dBm to 50 dBm, Unit: dBm

fetch() → List[float]

```
# SCPI: FETCh:EVDO:MEASurement<instance>:MEValuation:LIST:ACP:NPOW:SDEViation
value: List[float] = driver.multiEval.listPy.acp.npow.standardDev.fetch()
```

Returns narrowband and wideband (statistical) power values in the ACP measurement for all active list mode segments. The narrowband filter is 1.23 MHz, the wideband filter is 8 MHz wide. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return narrow_power: float Comma-separated list of values, one per active segment. Range: -100 dBm to 50 dBm, Unit: dBm

7.4.7.6.7 Wpow

class Wpow

Wpow commands group definition. 10 total commands, 5 Sub-groups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.multiEval.listPy.acp.wpow.clone()
```

Subgroups

7.4.7.6.7.1 Current

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:ACP:WPOW:CURRent
CALCulate:EVDO:MEASurement<Instance>:MEValuation:LIST:ACP:WPOW:CURRent
```

class Current

Current commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

calculate() → List[RsCmwEvdoMeas.enums.ResultStatus2]

```
# SCPI: CALCulate:EVDO:MEASurement<instance>:MEValuation:LIST:ACP:WPOW:CURRent
value: List[enums.ResultStatus2] = driver.multiEval.listPy.acp.wpow.current.
↪calculate()
```

Returns narrowband and wideband (statistical) power values in the ACP measurement for all active list mode segments. The narrowband filter is 1.23 MHz, the wideband filter is 8 MHz wide. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return wideband_power: float Comma-separated list of values, one per active segment. Range: -100 dBm to 50 dBm, Unit: dBm

fetch() → List[float]

```
# SCPI: FETCh:EVDO:MEASurement<instance>:MEValuation:LIST:ACP:WPOW:CURRent
value: List[float] = driver.multiEval.listPy.acp.wpow.current.fetch()
```

Returns narrowband and wideband (statistical) power values in the ACP measurement for all active list mode segments. The narrowband filter is 1.23 MHz, the wideband filter is 8 MHz wide. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return wideband_power: float Comma-separated list of values, one per active segment. Range: -100 dBm to 50 dBm, Unit: dBm

7.4.7.6.7.2 Average

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:ACP:WPOW:AVERage
CALCulate:EVDO:MEASurement<Instance>:MEValuation:LIST:ACP:WPOW:AVERage
```

class Average

Average commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

calculate() → List[RsCmwEvdoMeas.enums.ResultStatus2]

```
# SCPI: CALCulate:EVDO:MEASurement<instance>:MEValuation:LIST:ACP:WPOW:AVERage
value: List[enums.ResultStatus2] = driver.multiEval.listPy.acp.wpow.average.
↪calculate()
```

Returns narrowband and wideband (statistical) power values in the ACP measurement for all active list mode segments. The narrowband filter is 1.23 MHz, the wideband filter is 8 MHz wide. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return wideband_power: float Comma-separated list of values, one per active segment.
Range: -100 dBm to 50 dBm, Unit: dBm

fetch() → List[float]

```
# SCPI: FETCh:EVDO:MEASurement<instance>:MEValuation:LIST:ACP:WPOW:AVERage
value: List[float] = driver.multiEval.listPy.acp.wpow.average.fetch()
```

Returns narrowband and wideband (statistical) power values in the ACP measurement for all active list mode segments. The narrowband filter is 1.23 MHz, the wideband filter is 8 MHz wide. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return wideband_power: float Comma-separated list of values, one per active segment.
Range: -100 dBm to 50 dBm, Unit: dBm

7.4.7.6.7.3 Maximum

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:ACP:WPOW:MAXimum
CALCulate:EVDO:MEASurement<Instance>:MEValuation:LIST:ACP:WPOW:MAXimum
```

class Maximum

Maximum commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

calculate() → List[RsCmwEvdoMeas.enums.ResultStatus2]

```
# SCPI: CALCulate:EVDO:MEASurement<instance>:MEValuation:LIST:ACP:WPOW:MAXimum
value: List[enums.ResultStatus2] = driver.multiEval.listPy.acp.wpow.maximum.
↪calculate()
```

Returns narrowband and wideband (statistical) power values in the ACP measurement for all active list mode segments. The narrowband filter is 1.23 MHz, the wideband filter is 8 MHz wide. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return wideband_power: float Comma-separated list of values, one per active segment.
Range: -100 dBm to 50 dBm, Unit: dBm

fetch() → List[float]

```
# SCPI: FETCh:EVDO:MEASurement<instance>:MEValuation:LIST:ACP:WPOW:MAXimum
value: List[float] = driver.multiEval.listPy.acp.wpow.maximum.fetch()
```

Returns narrowband and wideband (statistical) power values in the ACP measurement for all active list mode segments. The narrowband filter is 1.23 MHz, the wideband filter is 8 MHz wide. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return wideband_power: float Comma-separated list of values, one per active segment.
Range: -100 dBm to 50 dBm, Unit: dBm

7.4.7.6.7.4 Minimum

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:ACP:WPOW:MINimum
CALCulate:EVDO:MEASurement<Instance>:MEValuation:LIST:ACP:WPOW:MINimum
```

class Minimum

Minimum commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

calculate() → List[RsCmwEvdoMeas.enums.ResultStatus2]

```
# SCPI: CALCulate:EVDO:MEASurement<instance>:MEValuation:LIST:ACP:WPOW:MINimum
value: List[enums.ResultStatus2] = driver.multiEval.listPy.acp.wpow.minimum.
↪calculate()
```

Returns narrowband and wideband (statistical) power values in the ACP measurement for all active list mode segments. The narrowband filter is 1.23 MHz, the wideband filter is 8 MHz wide. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return wideband_power: float Comma-separated list of values, one per active segment.
Range: -100 dBm to 50 dBm, Unit: dBm

fetch() → List[float]

```
# SCPI: FETCh:EVDO:MEASurement<instance>:MEValuation:LIST:ACP:WPOW:MINimum
value: List[float] = driver.multiEval.listPy.acp.wpow.minimum.fetch()
```

Returns narrowband and wideband (statistical) power values in the ACP measurement for all active list mode segments. The narrowband filter is 1.23 MHz, the wideband filter is 8 MHz wide. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return wideband_power: float Comma-separated list of values, one per active segment.
Range: -100 dBm to 50 dBm, Unit: dBm

7.4.7.6.7.5 StandardDev

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:ACP:WPOW:SDEViation
CALCulate:EVDO:MEASurement<Instance>:MEValuation:LIST:ACP:WPOW:SDEViation
```

class StandardDev

StandardDev commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

calculate() → List[RsCmwEvdoMeas.enums.ResultStatus2]

```
# SCPI: CALCulate:EVDO:MEASurement<instance>
↪:MEValuation:LIST:ACP:WPOW:SDEViation
value: List[enums.ResultStatus2] = driver.multiEval.listPy.acp.wpow.standardDev.
↪calculate()
```

Returns narrowband and wideband (statistical) power values in the ACP measurement for all active list mode segments. The narrowband filter is 1.23 MHz, the wideband filter is 8 MHz wide. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return wideband_power: float Comma-separated list of values, one per active segment. Range: -100 dBm to 50 dBm, Unit: dBm

fetch() → List[float]

```
# SCPI: FETCh:EVDO:MEASurement<instance>:MEValuation:LIST:ACP:WPOW:SDEViation
value: List[float] = driver.multiEval.listPy.acp.wpow.standardDev.fetch()
```

Returns narrowband and wideband (statistical) power values in the ACP measurement for all active list mode segments. The narrowband filter is 1.23 MHz, the wideband filter is 8 MHz wide. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return wideband_power: float Comma-separated list of values, one per active segment. Range: -100 dBm to 50 dBm, Unit: dBm

7.4.7.6.8 Current

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:ACP:CURRent
CALCulate:EVDO:MEASurement<Instance>:MEValuation:LIST:ACP:CURRent
```

class Current

Current commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Seg_Reliability: List[int]: 0 | 3 | 4 | 8 The segment reliability indicates whether one of the following exceptions occurred in this segment: 0: No error 3: Signal overflow 4: Signal low 8: Synchronization error If a combination of exceptions occurs, the most severe error is indicated.
- Acpm_10: List[enums.ResultStatus2]: No parameter help available
- Acpm_9: List[enums.ResultStatus2]: No parameter help available
- Acpm_8: List[enums.ResultStatus2]: No parameter help available
- Acpm_7: List[enums.ResultStatus2]: No parameter help available
- Acpm_6: List[enums.ResultStatus2]: No parameter help available
- Acpm_5: List[enums.ResultStatus2]: No parameter help available
- Acpm_4: List[enums.ResultStatus2]: No parameter help available
- Acpm_3: List[enums.ResultStatus2]: No parameter help available
- Acpm_2: List[enums.ResultStatus2]: No parameter help available
- Acpm_1: List[enums.ResultStatus2]: No parameter help available
- Acp_Carrier: List[enums.ResultStatus2]: No parameter help available
- Acpp_1: List[enums.ResultStatus2]: No parameter help available
- Acpp_2: List[enums.ResultStatus2]: No parameter help available
- Acpp_3: List[enums.ResultStatus2]: No parameter help available
- Acpp_4: List[enums.ResultStatus2]: No parameter help available
- Acpp_5: List[enums.ResultStatus2]: No parameter help available
- Acpp_6: List[enums.ResultStatus2]: No parameter help available
- Acpp_7: List[enums.ResultStatus2]: No parameter help available
- Acpp_8: List[enums.ResultStatus2]: No parameter help available
- Acpp_9: List[enums.ResultStatus2]: No parameter help available
- Acpp_10: List[enums.ResultStatus2]: No parameter help available
- Ms_Power_Wide: List[enums.ResultStatus2]: No parameter help available
- Ms_Power_Narrow: List[enums.ResultStatus2]: No parameter help available
- Out_Of_Tol_Count: List[enums.ResultStatus2]: float Out of tolerance result, i.e. percentage of measurement intervals of the statistic count (see [CMDLINK: CONFigure:EVDO:MEASi:MEValuation:SCOunt:SPECtrum CMDLINK]) exceeding the specified limits, see 'Limits (Spectrum) '. Range: 0 % to 100 %, Unit: %
- Cur_Stat_Count: List[enums.ResultStatus2]: decimal Number of evaluated valid slots in this segment.

`class FetchStruct`

Response structure. Fields:

- Reliability: int: No parameter help available
- Seg_Reliability: List[int]: 0 | 3 | 4 | 8 The segment reliability indicates whether one of the following exceptions occurred in this segment: 0: No error 3: Signal overflow 4: Signal low 8: Synchronization error If a combination of exceptions occurs, the most severe error is indicated.
- Acpm_10: List[float]: No parameter help available

- Acpm_9: List[float]: No parameter help available
- Acpm_8: List[float]: No parameter help available
- Acpm_7: List[float]: No parameter help available
- Acpm_6: List[float]: No parameter help available
- Acpm_5: List[float]: No parameter help available
- Acpm_4: List[float]: No parameter help available
- Acpm_3: List[float]: No parameter help available
- Acpm_2: List[float]: No parameter help available
- Acpm_1: List[float]: No parameter help available
- Acp_Carrier: List[float]: No parameter help available
- Acpp_1: List[float]: No parameter help available
- Acpp_2: List[float]: No parameter help available
- Acpp_3: List[float]: No parameter help available
- Acpp_4: List[float]: No parameter help available
- Acpp_5: List[float]: No parameter help available
- Acpp_6: List[float]: No parameter help available
- Acpp_7: List[float]: No parameter help available
- Acpp_8: List[float]: No parameter help available
- Acpp_9: List[float]: No parameter help available
- Acpp_10: List[float]: No parameter help available
- Ms_Power_Wide: List[float]: No parameter help available
- Ms_Power_Narrow: List[float]: No parameter help available
- Out_Of_Tol_Count: List[float]: float Out of tolerance result, i.e. percentage of measurement intervals of the statistic count (see [CMDLINK: CONFigure:EVDO:MEASi:MEValuation:SCOunt:SPECtrum CMDLINK]) exceeding the specified limits, see 'Limits (Spectrum) '. Range: 0 % to 100 %, Unit: %
- Cur_Stat_Count: List[int]: decimal Number of evaluated valid slots in this segment.

calculate() → CalculateStruct

```
# SCPI: CALCulate:EVDO:MEASurement<instance>:MEValuation:LIST:ACP:CURRent
value: CalculateStruct = driver.multiEval.listPy.acp.current.calculate()
```

Returns all ACP value results in list mode. To define the statistical length for AVERage, MAXimum, MINimum calculation and enable the calculation of the results use the command method RsCmwEvdoMeas.Configure.MultiEval.ListPy.Segment. Spectrum.set. The ranges indicated below apply to all results except standard deviation results. The minimum for standard deviation results equals 0. The maximum equals the width of the indicated range divided by two. Exceptions are explicitly stated. The values listed below in curly brackets {} are returned for each active segment: {...}seg 1, {...}seg 2, ..., {. ..}seg n. The number of active segments n is determined by method RsCmwEvdoMeas.Configure.MultiEval.ListPy.count. The number to the left of each result parameter is provided for easy identification of the parameter position within the result array. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for

each result listed below. For the out of tolerance and code channel filter match ratio, results retrieved via the CURRent, AVERage and MAXimum commands are identical.

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

fetch() → FetchStruct

```
# SCPI: FETCh:EVDO:MEASurement<instance>:MEValuation:LIST:ACP:CURRent
value: FetchStruct = driver.multiEval.listPy.acp.current.fetch()
```

Returns all ACP value results in list mode. To define the statistical length for AVERage, MAXimum, MINimum calculation and enable the calculation of the results use the command method RsCmwEvdoMeas.Configure.MultiEval.ListPy.Segment. Spectrum.set. The ranges indicated below apply to all results except standard deviation results. The minimum for standard deviation results equals 0. The maximum equals the width of the indicated range divided by two. Exceptions are explicitly stated. The values listed below in curly brackets {} are returned for each active segment: {...}seg 1, {...}seg 2, ..., {. ..}seg n. The number of active segments n is determined by method RsCmwEvdoMeas.Configure.MultiEval.ListPy.count. The number to the left of each result parameter is provided for easy identification of the parameter position within the result array. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below. For the out of tolerance and code channel filter match ratio, results retrieved via the CURRent, AVERage and MAXimum commands are identical.

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

7.4.7.6.9 Average

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:ACP:AVERage
CALCulate:EVDO:MEASurement<Instance>:MEValuation:LIST:ACP:AVERage
```

class Average

Average commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Seg_Reliability: List[int]: 0 | 3 | 4 | 8 The segment reliability indicates whether one of the following exceptions occurred in this segment: 0: No error 3: Signal overflow 4: Signal low 8: Synchronization error If a combination of exceptions occurs, the most severe error is indicated.
- Acpm_10: List[enums.ResultStatus2]: No parameter help available
- Acpm_9: List[enums.ResultStatus2]: No parameter help available
- Acpm_8: List[enums.ResultStatus2]: No parameter help available
- Acpm_7: List[enums.ResultStatus2]: No parameter help available
- Acpm_6: List[enums.ResultStatus2]: No parameter help available
- Acpm_5: List[enums.ResultStatus2]: No parameter help available
- Acpm_4: List[enums.ResultStatus2]: No parameter help available
- Acpm_3: List[enums.ResultStatus2]: No parameter help available

- Acpm_2: List[enums.ResultStatus2]: No parameter help available
- Acpm_1: List[enums.ResultStatus2]: No parameter help available
- Acp_Carrier: List[enums.ResultStatus2]: No parameter help available
- Acpp_1: List[enums.ResultStatus2]: No parameter help available
- Acpp_2: List[enums.ResultStatus2]: No parameter help available
- Acpp_3: List[enums.ResultStatus2]: No parameter help available
- Acpp_4: List[enums.ResultStatus2]: No parameter help available
- Acpp_5: List[enums.ResultStatus2]: No parameter help available
- Acpp_6: List[enums.ResultStatus2]: No parameter help available
- Acpp_7: List[enums.ResultStatus2]: No parameter help available
- Acpp_8: List[enums.ResultStatus2]: No parameter help available
- Acpp_9: List[enums.ResultStatus2]: No parameter help available
- Acpp_10: List[enums.ResultStatus2]: No parameter help available
- Ms_Power_Wide: List[enums.ResultStatus2]: No parameter help available
- Ms_Power_Narrow: List[enums.ResultStatus2]: No parameter help available
- Out_Of_Tol_Count: List[enums.ResultStatus2]: float Out of tolerance result, i.e. percentage of measurement intervals of the statistic count (see [CMDLINK: CONFigure:EVDO:MEASi:MEValuation:SCOunt:SPECtrum CMDLINK]) exceeding the specified limits, see 'Limits (Spectrum) '. Range: 0 % to 100 %, Unit: %
- Cur_Stat_Count: List[enums.ResultStatus2]: decimal Number of evaluated valid slots in this segment.

class FetchStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Seg_Reliability: List[int]: 0 | 3 | 4 | 8 The segment reliability indicates whether one of the following exceptions occurred in this segment: 0: No error 3: Signal overflow 4: Signal low 8: Synchronization error If a combination of exceptions occurs, the most severe error is indicated.
- Acpm_10: List[float]: No parameter help available
- Acpm_9: List[float]: No parameter help available
- Acpm_8: List[float]: No parameter help available
- Acpm_7: List[float]: No parameter help available
- Acpm_6: List[float]: No parameter help available
- Acpm_5: List[float]: No parameter help available
- Acpm_4: List[float]: No parameter help available
- Acpm_3: List[float]: No parameter help available
- Acpm_2: List[float]: No parameter help available
- Acpm_1: List[float]: No parameter help available
- Acp_Carrier: List[float]: No parameter help available
- Acpp_1: List[float]: No parameter help available

- Acpp_2: List[float]: No parameter help available
- Acpp_3: List[float]: No parameter help available
- Acpp_4: List[float]: No parameter help available
- Acpp_5: List[float]: No parameter help available
- Acpp_6: List[float]: No parameter help available
- Acpp_7: List[float]: No parameter help available
- Acpp_8: List[float]: No parameter help available
- Acpp_9: List[float]: No parameter help available
- Acpp_10: List[float]: No parameter help available
- Ms_Power_Wide: List[float]: No parameter help available
- Ms_Power_Narrow: List[float]: No parameter help available
- Out_Of_Tol_Count: List[float]: float Out of tolerance result, i.e. percentage of measurement intervals of the statistic count (see [CMDLINK: CONFigure:EVDO:MEASi:MEValuation:SCOunt:SPECtrum CMDLINK]) exceeding the specified limits, see 'Limits (Spectrum) '. Range: 0 % to 100 %, Unit: %
- Cur_Stat_Count: List[int]: decimal Number of evaluated valid slots in this segment.

calculate() → CalculateStruct

```
# SCPI: CALCulate:EVDO:MEASurement<instance>:MEValuation:LIST:ACP:AVERage
value: CalculateStruct = driver.multiEval.listPy.acp.average.calculate()
```

Returns all ACP value results in list mode. To define the statistical length for AVERage, MAXimum, MINimum calculation and enable the calculation of the results use the command method RsCmwEvdoMeas.Configure.MultiEval.ListPy.Segment. Spectrum.set. The ranges indicated below apply to all results except standard deviation results. The minimum for standard deviation results equals 0. The maximum equals the width of the indicated range divided by two. Exceptions are explicitly stated. The values listed below in curly brackets {} are returned for each active segment: {...}seg 1, {...}seg 2, ..., {. ..}seg n. The number of active segments n is determined by method RsCmwEvdoMeas.Configure.MultiEval.ListPy.count. The number to the left of each result parameter is provided for easy identification of the parameter position within the result array. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below. For the out of tolerance and code channel filter match ratio, results retrieved via the CURRent, AVERage and MAXimum commands are identical.

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

fetch() → FetchStruct

```
# SCPI: FETCh:EVDO:MEASurement<instance>:MEValuation:LIST:ACP:AVERage
value: FetchStruct = driver.multiEval.listPy.acp.average.fetch()
```

Returns all ACP value results in list mode. To define the statistical length for AVERage, MAXimum, MINimum calculation and enable the calculation of the results use the command method RsCmwEvdoMeas.Configure.MultiEval.ListPy.Segment. Spectrum.set. The ranges indicated below apply to all results except standard deviation results. The minimum for standard deviation results equals 0. The maximum equals the width of the indicated range divided by two. Exceptions are explicitly stated. The values listed below in curly brackets {} are returned for each active segment: {...}seg 1,

{...}seg 2, ..., {. ..}seg n. The number of active segments n is determined by method RsCmwEvdoMeas.Configure.MultiEval.ListPy.count. The number to the left of each result parameter is provided for easy identification of the parameter position within the result array. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below. For the out of tolerance and code channel filter match ratio, results retrieved via the CURRent, AVERage and MAXimum commands are identical.

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

7.4.7.6.10 Maximum

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:ACP:MAXimum
CALCulate:EVDO:MEASurement<Instance>:MEValuation:LIST:ACP:MAXimum
```

class Maximum

Maximum commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Seg_Reliability: List[int]: 0 | 3 | 4 | 8 The segment reliability indicates whether one of the following exceptions occurred in this segment: 0: No error 3: Signal overflow 4: Signal low 8: Synchronization error If a combination of exceptions occurs, the most severe error is indicated.
- Acpm_10: List[enums.ResultStatus2]: No parameter help available
- Acpm_9: List[enums.ResultStatus2]: No parameter help available
- Acpm_8: List[enums.ResultStatus2]: No parameter help available
- Acpm_7: List[enums.ResultStatus2]: No parameter help available
- Acpm_6: List[enums.ResultStatus2]: No parameter help available
- Acpm_5: List[enums.ResultStatus2]: No parameter help available
- Acpm_4: List[enums.ResultStatus2]: No parameter help available
- Acpm_3: List[enums.ResultStatus2]: No parameter help available
- Acpm_2: List[enums.ResultStatus2]: No parameter help available
- Acpm_1: List[enums.ResultStatus2]: No parameter help available
- Acp_Carrier: List[enums.ResultStatus2]: No parameter help available
- Acpp_1: List[enums.ResultStatus2]: No parameter help available
- Acpp_2: List[enums.ResultStatus2]: No parameter help available
- Acpp_3: List[enums.ResultStatus2]: No parameter help available
- Acpp_4: List[enums.ResultStatus2]: No parameter help available
- Acpp_5: List[enums.ResultStatus2]: No parameter help available
- Acpp_6: List[enums.ResultStatus2]: No parameter help available
- Acpp_7: List[enums.ResultStatus2]: No parameter help available

- Acpp_8: List[enums.ResultStatus2]: No parameter help available
- Acpp_9: List[enums.ResultStatus2]: No parameter help available
- Acpp_10: List[enums.ResultStatus2]: No parameter help available
- Ms_Power_Wide: List[enums.ResultStatus2]: No parameter help available
- Ms_Power_Narrow: List[enums.ResultStatus2]: No parameter help available
- Out_Of_Tol_Count: List[enums.ResultStatus2]: float Out of tolerance result, i.e. percentage of measurement intervals of the statistic count (see [CMDLINK: CONFigure:EVDO:MEASi:MEValuation:SCOunt:SPECtrum CMDLINK]) exceeding the specified limits, see 'Limits (Spectrum) '. Range: 0 % to 100 %, Unit: %
- Cur_Stat_Count: List[enums.ResultStatus2]: decimal Number of evaluated valid slots in this segment.

`class FetchStruct`

Response structure. Fields:

- Reliability: int: No parameter help available
- Seg_Reliability: List[int]: 0 | 3 | 4 | 8 The segment reliability indicates whether one of the following exceptions occurred in this segment: 0: No error 3: Signal overflow 4: Signal low 8: Synchronization error If a combination of exceptions occurs, the most severe error is indicated.
- Acpm_10: List[float]: No parameter help available
- Acpm_9: List[float]: No parameter help available
- Acpm_8: List[float]: No parameter help available
- Acpm_7: List[float]: No parameter help available
- Acpm_6: List[float]: No parameter help available
- Acpm_5: List[float]: No parameter help available
- Acpm_4: List[float]: No parameter help available
- Acpm_3: List[float]: No parameter help available
- Acpm_2: List[float]: No parameter help available
- Acpm_1: List[float]: No parameter help available
- Acp_Carrier: List[float]: No parameter help available
- Acpp_1: List[float]: No parameter help available
- Acpp_2: List[float]: No parameter help available
- Acpp_3: List[float]: No parameter help available
- Acpp_4: List[float]: No parameter help available
- Acpp_5: List[float]: No parameter help available
- Acpp_6: List[float]: No parameter help available
- Acpp_7: List[float]: No parameter help available
- Acpp_8: List[float]: No parameter help available
- Acpp_9: List[float]: No parameter help available
- Acpp_10: List[float]: No parameter help available
- Ms_Power_Wide: List[float]: No parameter help available

- Ms_Power_Narrow: List[float]: No parameter help available
- Out_Of_Tol_Count: List[float]: float Out of tolerance result, i.e. percentage of measurement intervals of the statistic count (see [CMDLINK: CONFigure:EVDO:MEASi:MEValuation:SCOunt:SPECtrum CMDLINK]) exceeding the specified limits, see 'Limits (Spectrum) '. Range: 0 % to 100 %, Unit: %
- Cur_Stat_Count: List[int]: decimal Number of evaluated valid slots in this segment.

calculate() → CalculateStruct

```
# SCPI: CALCulate:EVDO:MEASurement<instance>:MEValuation:LIST:ACP:MAXimum
value: CalculateStruct = driver.multiEval.listPy.acp.maximum.calculate()
```

Returns all ACP value results in list mode. To define the statistical length for AVERage, MAXimum, MINimum calculation and enable the calculation of the results use the command method RsCmwEvdoMeas.Configure.MultiEval.ListPy.Segment. Spectrum.set. The ranges indicated below apply to all results except standard deviation results. The minimum for standard deviation results equals 0. The maximum equals the width of the indicated range divided by two. Exceptions are explicitly stated. The values listed below in curly brackets {} are returned for each active segment: {...}seg 1, {...}seg 2, ..., {. ..}seg n. The number of active segments n is determined by method RsCmwEvdoMeas.Configure.MultiEval.ListPy.count. The number to the left of each result parameter is provided for easy identification of the parameter position within the result array. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below. For the out of tolerance and code channel filter match ratio, results retrieved via the CURRent, AVERage and MAXimum commands are identical.

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

fetch() → FetchStruct

```
# SCPI: FETCh:EVDO:MEASurement<instance>:MEValuation:LIST:ACP:MAXimum
value: FetchStruct = driver.multiEval.listPy.acp.maximum.fetch()
```

Returns all ACP value results in list mode. To define the statistical length for AVERage, MAXimum, MINimum calculation and enable the calculation of the results use the command method RsCmwEvdoMeas.Configure.MultiEval.ListPy.Segment. Spectrum.set. The ranges indicated below apply to all results except standard deviation results. The minimum for standard deviation results equals 0. The maximum equals the width of the indicated range divided by two. Exceptions are explicitly stated. The values listed below in curly brackets {} are returned for each active segment: {...}seg 1, {...}seg 2, ..., {. ..}seg n. The number of active segments n is determined by method RsCmwEvdoMeas.Configure.MultiEval.ListPy.count. The number to the left of each result parameter is provided for easy identification of the parameter position within the result array. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below. For the out of tolerance and code channel filter match ratio, results retrieved via the CURRent, AVERage and MAXimum commands are identical.

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

7.4.7.6.11 Minimum

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:ACP:MINimum
CALCulate:EVDO:MEASurement<Instance>:MEValuation:LIST:ACP:MINimum
```

class Minimum

Minimum commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Seg_Reliability: List[int]: 0 | 3 | 4 | 8 The segment reliability indicates whether one of the following exceptions occurred in this segment: 0: No error 3: Signal overflow 4: Signal low 8: Synchronization error If a combination of exceptions occurs, the most severe error is indicated.
- Acpm_10: List[enums.ResultStatus2]: No parameter help available
- Acpm_9: List[enums.ResultStatus2]: No parameter help available
- Acpm_8: List[enums.ResultStatus2]: No parameter help available
- Acpm_7: List[enums.ResultStatus2]: No parameter help available
- Acpm_6: List[enums.ResultStatus2]: No parameter help available
- Acpm_5: List[enums.ResultStatus2]: No parameter help available
- Acpm_4: List[enums.ResultStatus2]: No parameter help available
- Acpm_3: List[enums.ResultStatus2]: No parameter help available
- Acpm_2: List[enums.ResultStatus2]: No parameter help available
- Acpm_1: List[enums.ResultStatus2]: No parameter help available
- Acp_Carrier: List[enums.ResultStatus2]: No parameter help available
- Acpp_1: List[enums.ResultStatus2]: No parameter help available
- Acpp_2: List[enums.ResultStatus2]: No parameter help available
- Acpp_3: List[enums.ResultStatus2]: No parameter help available
- Acpp_4: List[enums.ResultStatus2]: No parameter help available
- Acpp_5: List[enums.ResultStatus2]: No parameter help available
- Acpp_6: List[enums.ResultStatus2]: No parameter help available
- Acpp_7: List[enums.ResultStatus2]: No parameter help available
- Acpp_8: List[enums.ResultStatus2]: No parameter help available
- Acpp_9: List[enums.ResultStatus2]: No parameter help available
- Acpp_10: List[enums.ResultStatus2]: No parameter help available
- Ms_Power_Wide: List[enums.ResultStatus2]: No parameter help available
- Ms_Power_Narrow: List[enums.ResultStatus2]: No parameter help available

- Out_Of_Tol_Count: List[enums.ResultStatus2]: float Out of tolerance result, i.e. percentage of measurement intervals of the statistic count (see [CMDLINK: CONFigure:EVDO:MEASi:MEValuation:SCOunt:SPECtrum CMDLINK]) exceeding the specified limits, see 'Limits (Spectrum) '. Range: 0 % to 100 %, Unit: %
- Cur_Stat_Count: List[enums.ResultStatus2]: decimal Number of evaluated valid slots in this segment.

class FetchStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Seg_Reliability: List[int]: 0 | 3 | 4 | 8 The segment reliability indicates whether one of the following exceptions occurred in this segment: 0: No error 3: Signal overflow 4: Signal low 8: Synchronization error If a combination of exceptions occurs, the most severe error is indicated.
- Acpm_10: List[float]: No parameter help available
- Acpm_9: List[float]: No parameter help available
- Acpm_8: List[float]: No parameter help available
- Acpm_7: List[float]: No parameter help available
- Acpm_6: List[float]: No parameter help available
- Acpm_5: List[float]: No parameter help available
- Acpm_4: List[float]: No parameter help available
- Acpm_3: List[float]: No parameter help available
- Acpm_2: List[float]: No parameter help available
- Acpm_1: List[float]: No parameter help available
- Acp_Carrier: List[float]: No parameter help available
- Acpp_1: List[float]: No parameter help available
- Acpp_2: List[float]: No parameter help available
- Acpp_3: List[float]: No parameter help available
- Acpp_4: List[float]: No parameter help available
- Acpp_5: List[float]: No parameter help available
- Acpp_6: List[float]: No parameter help available
- Acpp_7: List[float]: No parameter help available
- Acpp_8: List[float]: No parameter help available
- Acpp_9: List[float]: No parameter help available
- Acpp_10: List[float]: No parameter help available
- Ms_Power_Wide: List[float]: No parameter help available
- Ms_Power_Narrow: List[float]: No parameter help available
- Out_Of_Tol_Count: List[float]: float Out of tolerance result, i.e. percentage of measurement intervals of the statistic count (see [CMDLINK: CONFigure:EVDO:MEASi:MEValuation:SCOunt:SPECtrum CMDLINK]) exceeding the specified limits, see 'Limits (Spectrum) '. Range: 0 % to 100 %, Unit: %
- Cur_Stat_Count: List[int]: decimal Number of evaluated valid slots in this segment.

calculate() → CalculateStruct

```
# SCPI: CALCulate:EVDO:MEASurement<instance>:MEValuation:LIST:ACP:MINimum
value: CalculateStruct = driver.multiEval.listPy.acp.minimum.calculate()
```

Returns all ACP value results in list mode. To define the statistical length for AVERage, MAXimum, MINimum calculation and enable the calculation of the results use the command method RsCmwEvdoMeas.Configure.MultiEval.ListPy.Segment. Spectrum.set. The ranges indicated below apply to all results except standard deviation results. The minimum for standard deviation results equals 0. The maximum equals the width of the indicated range divided by two. Exceptions are explicitly stated. The values listed below in curly brackets {} are returned for each active segment: {...}seg 1, {...}seg 2, ..., {. ..}seg n. The number of active segments n is determined by method RsCmwEvdoMeas.Configure.MultiEval.ListPy.count. The number to the left of each result parameter is provided for easy identification of the parameter position within the result array. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below. For the out of tolerance and code channel filter match ratio, results retrieved via the CURRent, AVERage and MAXimum commands are identical.

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

fetch() → FetchStruct

```
# SCPI: FETCh:EVDO:MEASurement<instance>:MEValuation:LIST:ACP:MINimum
value: FetchStruct = driver.multiEval.listPy.acp.minimum.fetch()
```

Returns all ACP value results in list mode. To define the statistical length for AVERage, MAXimum, MINimum calculation and enable the calculation of the results use the command method RsCmwEvdoMeas.Configure.MultiEval.ListPy.Segment. Spectrum.set. The ranges indicated below apply to all results except standard deviation results. The minimum for standard deviation results equals 0. The maximum equals the width of the indicated range divided by two. Exceptions are explicitly stated. The values listed below in curly brackets {} are returned for each active segment: {...}seg 1, {...}seg 2, ..., {. ..}seg n. The number of active segments n is determined by method RsCmwEvdoMeas.Configure.MultiEval.ListPy.count. The number to the left of each result parameter is provided for easy identification of the parameter position within the result array. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below. For the out of tolerance and code channel filter match ratio, results retrieved via the CURRent, AVERage and MAXimum commands are identical.

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

7.4.7.6.12 StandardDev

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:ACP:SDEViation
CALCulate:EVDO:MEASurement<Instance>:MEValuation:LIST:ACP:SDEViation
```

class StandardDev

StandardDev commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: No parameter help available

- Seg_Reliability: List[int]: 0 | 3 | 4 | 8 The segment reliability indicates whether one of the following exceptions occurred in this segment: 0: No error 3: Signal overflow 4: Signal low 8: Synchronization error If a combination of exceptions occurs, the most severe error is indicated.
- Acpm_10: List[enums.ResultStatus2]: No parameter help available
- Acpm_9: List[enums.ResultStatus2]: No parameter help available
- Acpm_8: List[enums.ResultStatus2]: No parameter help available
- Acpm_7: List[enums.ResultStatus2]: No parameter help available
- Acpm_6: List[enums.ResultStatus2]: No parameter help available
- Acpm_5: List[enums.ResultStatus2]: No parameter help available
- Acpm_4: List[enums.ResultStatus2]: No parameter help available
- Acpm_3: List[enums.ResultStatus2]: No parameter help available
- Acpm_2: List[enums.ResultStatus2]: No parameter help available
- Acpm_1: List[enums.ResultStatus2]: No parameter help available
- Acp_Carrier: List[enums.ResultStatus2]: No parameter help available
- Acpp_1: List[enums.ResultStatus2]: No parameter help available
- Acpp_2: List[enums.ResultStatus2]: No parameter help available
- Acpp_3: List[enums.ResultStatus2]: No parameter help available
- Acpp_4: List[enums.ResultStatus2]: No parameter help available
- Acpp_5: List[enums.ResultStatus2]: No parameter help available
- Acpp_6: List[enums.ResultStatus2]: No parameter help available
- Acpp_7: List[enums.ResultStatus2]: No parameter help available
- Acpp_8: List[enums.ResultStatus2]: No parameter help available
- Acpp_9: List[enums.ResultStatus2]: No parameter help available
- Acpp_10: List[enums.ResultStatus2]: No parameter help available
- Ms_Power_Wide: List[enums.ResultStatus2]: No parameter help available
- Ms_Power_Narrow: List[enums.ResultStatus2]: No parameter help available
- Out_Of_Tol_Count: List[enums.ResultStatus2]: float Out of tolerance result, i.e. percentage of measurement intervals of the statistic count (see [CMDLINK: CONFigure:EVDO:MEASi:MEValuation:SCOunt:SPECtrum CMDLINK]) exceeding the specified limits, see 'Limits (Spectrum) '. Range: 0 % to 100 %, Unit: %
- Cur_Stat_Count: List[enums.ResultStatus2]: decimal Number of evaluated valid slots in this segment.

class FetchStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Seg_Reliability: List[int]: 0 | 3 | 4 | 8 The segment reliability indicates whether one of the following exceptions occurred in this segment: 0: No error 3: Signal overflow 4: Signal low 8: Synchronization error If a combination of exceptions occurs, the most severe error is indicated.
- Acpm_10: List[float]: No parameter help available
- Acpm_9: List[float]: No parameter help available

- Acpm_8: List[float]: No parameter help available
- Acpm_7: List[float]: No parameter help available
- Acpm_6: List[float]: No parameter help available
- Acpm_5: List[float]: No parameter help available
- Acpm_4: List[float]: No parameter help available
- Acpm_3: List[float]: No parameter help available
- Acpm_2: List[float]: No parameter help available
- Acpm_1: List[float]: No parameter help available
- Acp_Carrier: List[float]: No parameter help available
- Acpp_1: List[float]: No parameter help available
- Acpp_2: List[float]: No parameter help available
- Acpp_3: List[float]: No parameter help available
- Acpp_4: List[float]: No parameter help available
- Acpp_5: List[float]: No parameter help available
- Acpp_6: List[float]: No parameter help available
- Acpp_7: List[float]: No parameter help available
- Acpp_8: List[float]: No parameter help available
- Acpp_9: List[float]: No parameter help available
- Acpp_10: List[float]: No parameter help available
- Ms_Power_Wide: List[float]: No parameter help available
- Ms_Power_Narrow: List[float]: No parameter help available
- Out_Of_Tol_Count: List[float]: float Out of tolerance result, i.e. percentage of measurement intervals of the statistic count (see [CMDLINK: CONFigure:EVDO:MEASi:MEValuation:SCOunt:SPECtrum CMDLINK]) exceeding the specified limits, see 'Limits (Spectrum) '. Range: 0 % to 100 %, Unit: %
- Cur_Stat_Count: List[int]: decimal Number of evaluated valid slots in this segment.

calculate() → CalculateStruct

```
# SCPI: CALCulate:EVDO:MEASurement<instance>:MEValuation:LIST:ACP:SDEViation
value: CalculateStruct = driver.multiEval.listPy.acp.standardDev.calculate()
```

Returns all ACP value results in list mode. To define the statistical length for AVERage, MAXimum, MINimum calculation and enable the calculation of the results use the command method RsCmwEvdoMeas.Configure.MultiEval.ListPy.Segment. Spectrum.set. The ranges indicated below apply to all results except standard deviation results. The minimum for standard deviation results equals 0. The maximum equals the width of the indicated range divided by two. Exceptions are explicitly stated. The values listed below in curly brackets {} are returned for each active segment: {...}seg 1, {...}seg 2, ..., {. ..}seg n. The number of active segments n is determined by method RsCmwEvdoMeas.Configure.MultiEval.ListPy.count. The number to the left of each result parameter is provided for easy identification of the parameter position within the result array. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below. For the out of tolerance and code channel filter match ratio, results retrieved via the CURRent, AVERage and MAXimum commands are identical.

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

fetch() → FetchStruct

```
# SCPI: FETCh:EVDO:MEASurement<instance>:MEValuation:LIST:ACP:SDEViation
value: FetchStruct = driver.multiEval.listPy.acp.standardDev.fetch()
```

Returns all ACP value results in list mode. To define the statistical length for AVERage, MAXimum, MINimum calculation and enable the calculation of the results use the command method RsCmwEvdoMeas.Configure.MultiEval.ListPy.Segment. Spectrum.set. The ranges indicated below apply to all results except standard deviation results. The minimum for standard deviation results equals 0. The maximum equals the width of the indicated range divided by two. Exceptions are explicitly stated. The values listed below in curly brackets {} are returned for each active segment: {...}seg 1, {...}seg 2, ..., {. ..}seg n. The number of active segments n is determined by method RsCmwEvdoMeas.Configure.MultiEval.ListPy.count. The number to the left of each result parameter is provided for easy identification of the parameter position within the result array. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below. For the out of tolerance and code channel filter match ratio, results retrieved via the CURRent, AVERage and MAXimum commands are identical.

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

7.4.7.7 Obw

class Obw

Obw commands group definition. 18 total commands, 5 Sub-groups, 0 group commands

Cloning the Group

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

Subgroups

7.4.7.7.1 Frequency

class Frequency

Frequency commands group definition. 10 total commands, 6 Sub-groups, 0 group commands

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.multiEval.listPy.obw.frequency.clone()
```

Subgroups

7.4.7.7.1.1 Current

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:OBW:FREQuency:CURRent
CALCulate:EVDO:MEASurement<Instance>:MEValuation:LIST:OBW:FREQuency:CURRent
```

class Current

Current commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

calculate() → List[RsCmwEvdoMeas.enums.ResultStatus2]

```
# SCPI: CALCulate:EVDO:MEASurement<instance>
↪:MEValuation:LIST:OBW:FREQuency:CURRent
value: List[enums.ResultStatus2] = driver.multiEval.listPy.obw.frequency.
↪current.calculate()
```

Returns the occupied bandwidth (statistical) values for all active list mode segments. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return obw: float Comma-separated list of values, one per active segment. Range: 0 MHz to 16 MHz (SDEViation 0 MHz to 8 MHz) , Unit: Hz

fetch() → List[float]

```
# SCPI: FETCh:EVDO:MEASurement<instance>:MEValuation:LIST:OBW:FREQuency:CURRent
value: List[float] = driver.multiEval.listPy.obw.frequency.current.fetch()
```

Returns the occupied bandwidth (statistical) values for all active list mode segments. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return obw: float Comma-separated list of values, one per active segment. Range: 0 MHz to 16 MHz (SDEViation 0 MHz to 8 MHz) , Unit: Hz

7.4.7.7.1.2 Average

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:OBW:FREQuency:AVERage
CALCulate:EVDO:MEASurement<Instance>:MEValuation:LIST:OBW:FREQuency:AVERage
```

class Average

Average commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

calculate() → List[RsCmwEvdoMeas.enums.ResultStatus2]

```
# SCPI: CALCulate:EVDO:MEASurement<instance>
↪:MEValuation:LIST:OBW:FREQuency:AVERage
value: List[enums.ResultStatus2] = driver.multiEval.listPy.obw.frequency.
↪average.calculate()
```

Returns the occupied bandwidth (statistical) values for all active list mode segments. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return obw: float Comma-separated list of values, one per active segment. Range: 0 MHz to 16 MHz (SDEViation 0 MHz to 8 MHz) , Unit: Hz

fetch() → List[float]

```
# SCPI: FETCh:EVDO:MEASurement<instance>:MEValuation:LIST:OBW:FREQuency:AVERage
value: List[float] = driver.multiEval.listPy.obw.frequency.average.fetch()
```

Returns the occupied bandwidth (statistical) values for all active list mode segments. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return obw: float Comma-separated list of values, one per active segment. Range: 0 MHz to 16 MHz (SDEViation 0 MHz to 8 MHz) , Unit: Hz

7.4.7.7.1.3 Maximum

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:OBW:FREQuency:MAXimum
CALCulate:EVDO:MEASurement<Instance>:MEValuation:LIST:OBW:FREQuency:MAXimum
```

class Maximum

Maximum commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

calculate() → List[RsCmwEvdoMeas.enums.ResultStatus2]

```
# SCPI: CALCulate:EVDO:MEASurement<instance>
↪:MEValuation:LIST:OBW:FREQuency:MAXimum
value: List[enums.ResultStatus2] = driver.multiEval.listPy.obw.frequency.
↪maximum.calculate()
```

Returns the occupied bandwidth (statistical) values for all active list mode segments. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return obw: float Comma-separated list of values, one per active segment. Range: 0 MHz to 16 MHz (SDEViation 0 MHz to 8 MHz) , Unit: Hz

fetch() → List[float]

```
# SCPI: FETCh:EVDO:MEASurement<instance>:MEValuation:LIST:OBW:FREQuency:MAXimum
value: List[float] = driver.multiEval.listPy.obw.frequency.maximum.fetch()
```

Returns the occupied bandwidth (statistical) values for all active list mode segments. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return obw: float Comma-separated list of values, one per active segment. Range: 0 MHz to 16 MHz (SDEViation 0 MHz to 8 MHz) , Unit: Hz

7.4.7.7.1.4 StandardDev

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:OBW:FREQuency:SDEViation
CALCulate:EVDO:MEASurement<Instance>:MEValuation:LIST:OBW:FREQuency:SDEViation
```

class StandardDev

StandardDev commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

calculate() → List[RsCmwEvdoMeas.enums.ResultStatus2]

```
# SCPI: CALCulate:EVDO:MEASurement<instance>
↪:MEValuation:LIST:OBW:FREQuency:SDEViation
value: List[enums.ResultStatus2] = driver.multiEval.listPy.obw.frequency.
↪standardDev.calculate()
```

Returns the occupied bandwidth (statistical) values for all active list mode segments. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return obw: float Comma-separated list of values, one per active segment. Range: 0 MHz to 16 MHz (SDEViation 0 MHz to 8 MHz) , Unit: Hz

fetch() → List[float]

```
# SCPI: FETCh:EVDO:MEASurement<instance>
↪:MEValuation:LIST:OBW:FREQuency:SDEViation
value: List[float] = driver.multiEval.listPy.obw.frequency.standardDev.fetch()
```

Returns the occupied bandwidth (statistical) values for all active list mode segments. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return obw: float Comma-separated list of values, one per active segment. Range: 0 MHz to 16 MHz (SDEViation 0 MHz to 8 MHz) , Unit: Hz

7.4.7.7.1.5 Lower

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:OBW:FREQuency:LOWer
```

class Lower

Lower commands group definition. 1 total commands, 0 Sub-groups, 1 group commands

fetch() → List[float]

```
# SCPI: FETCh:EVDO:MEASurement<instance>:MEValuation:LIST:OBW:FREQuency:LOWer
value: List[float] = driver.multiEval.listPy.obw.frequency.lower.fetch()
```

Returns lower and upper OBW frequencies for all active list mode segments. The values are taken from the 'CURRent' measurement. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return obw_lower: No help available

7.4.7.7.1.6 Upper

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:OBW:FREQuency:UPPer
```

class Upper

Upper commands group definition. 1 total commands, 0 Sub-groups, 1 group commands

fetch() → List[float]

```
# SCPI: FETCh:EVDO:MEASurement<instance>:MEValuation:LIST:OBW:FREQuency:UPPer
value: List[float] = driver.multiEval.listPy.obw.frequency.upper.fetch()
```

Returns lower and upper OBW frequencies for all active list mode segments. The values are taken from the 'CURRent' measurement. The values described below are returned by FETCh commands. CALCulate commands return limit check results instead, one value for each result listed below.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

return obw_upper: No help available

7.4.7.7.2 Current

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:OBW:CURRent
CALCulate:EVDO:MEASurement<Instance>:MEValuation:LIST:OBW:CURRent
```

class Current

Current commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: decimal 'Conventions and General Information'. In list mode, a zero reliability indicator indicates that the results in all measured segments are valid. A non-zero value indicates that an error occurred in at least one of the measured segments.
- Seg_Reliability: List[int]: 0 | 3 | 4 | 8 The segment reliability indicates whether one of the following exceptions occurred in this segment: 0: No error 3: Signal overflow 4: Signal low 8: Synchronization error If a combination of exceptions occurs, the most severe error is indicated.
- Obw: List[enums.ResultStatus2]: float Occupied bandwidth Range: 0 MHz to 16 MHz (SDEViation 0 MHz to 8 MHz) , Unit: MHz
- Lower_Freq: List[float]: No parameter help available
- Upper_Freq: List[float]: No parameter help available

class FetchStruct

Response structure. Fields:

- Reliability: int: decimal 'Conventions and General Information'. In list mode, a zero reliability indicator indicates that the results in all measured segments are valid. A non-zero value indicates that an error occurred in at least one of the measured segments.
- Seg_Reliability: List[int]: 0 | 3 | 4 | 8 The segment reliability indicates whether one of the following exceptions occurred in this segment: 0: No error 3: Signal overflow 4: Signal low 8: Synchronization error If a combination of exceptions occurs, the most severe error is indicated.
- Obw: List[float]: float Occupied bandwidth Range: 0 MHz to 16 MHz (SDEViation 0 MHz to 8 MHz) , Unit: MHz
- Lower_Freq: List[float]: No parameter help available
- Upper_Freq: List[float]: No parameter help available

calculate() → CalculateStruct

```
# SCPI: CALCulate:EVDO:MEASurement<instance>:MEValuation:LIST:OBW:CURRent
value: CalculateStruct = driver.multiEval.listPy.obw.current.calculate()
```

Returns occupied bandwidth (OBW) results in list mode. To define the statistical length and to enable the calculation of the results, use the command method RsCmwEvdoMeas.Configure.MultiEval.ListPy.Segment.Modulation.set. The values listed below in curly brackets {} are returned for each active segment: {...}seg 1, {...}seg 2, ..., {...}seg n. The number of active segments n is determined by method RsCmwEvdoMeas.Configure.MultiEval.ListPy.count. 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. The values described below are returned by the

FETCh command. The CALCulate command returns limit check results instead, one value for each result listed below. For details, refer to 'Multi-Evaluation Measurement Results'.

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

fetch() → FetchStruct

```
# SCPI: FETCh:EVDO:MEASurement<instance>:MEValuation:LIST:OBW:CURRent
value: FetchStruct = driver.multiEval.listPy.obw.current.fetch()
```

Returns occupied bandwidth (OBW) results in list mode. To define the statistical length and to enable the calculation of the results, use the command method RsCmwEvdoMeas.Configure.MultiEval.ListPy.Segment.Modulation.set. The values listed below in curly brackets {} are returned for each active segment: {...}seg 1, {...}seg 2, ..., {...}seg n. The number of active segments n is determined by method RsCmwEvdoMeas.Configure.MultiEval.ListPy.count. 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. The values described below are returned by the FETCh command. The CALCulate command returns limit check results instead, one value for each result listed below. For details, refer to 'Multi-Evaluation Measurement Results'.

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

7.4.7.7.3 Average

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:OBW:AVERage
CALCulate:EVDO:MEASurement<Instance>:MEValuation:LIST:OBW:AVERage
```

class Average

Average commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Seg_Reliability: List[int]: 0 | 3 | 4 | 8 The segment reliability indicates whether one of the following exceptions occurred in this segment: 0: No error 3: Signal overflow 4: Signal low 8: Synchronization error If a combination of exceptions occurs, the most severe error is indicated.
- Obw: List[enums.ResultStatus2]: float Occupied bandwidth Range: 0 MHz to 16 MHz (SDEViation 0 MHz to 8 MHz) , Unit: MHz

class FetchStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Seg_Reliability: List[int]: 0 | 3 | 4 | 8 The segment reliability indicates whether one of the following exceptions occurred in this segment: 0: No error 3: Signal overflow 4: Signal low 8: Synchronization error If a combination of exceptions occurs, the most severe error is indicated.
- Obw: List[float]: float Occupied bandwidth Range: 0 MHz to 16 MHz (SDEViation 0 MHz to 8 MHz) , Unit: MHz

calculate() → CalculateStruct

```
# SCPI: CALCulate:EVDO:MEASurement<instance>:MEValuation:LIST:OBW:AVERage
value: CalculateStruct = driver.multiEval.listPy.obw.average.calculate()
```

Returns occupied bandwidth (OBW) results in list mode. To define the statistical length for AVERage, MAXimum and to enable the calculation of the results, use the command method RsCmwEvdoMeas.Configure.MultiEval.ListPy.Segment.Modulation. set. The ranges indicated below apply to all results except standard deviation results. The minimum for standard deviation results equals 0. The maximum equals the width of the indicated range divided by two. Exceptions are explicitly stated. The values listed below in curly brackets {} are returned for each active segment: {...}seg 1, {...}seg 2, ..., {. ..}seg n. The number of active segments n is determined by method RsCmwEvdoMeas.Configure.MultiEval.ListPy.count. 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:EVDO:MEASurement<instance>:MEValuation:LIST:OBW:AVERage
value: FetchStruct = driver.multiEval.listPy.obw.average.fetch()
```

Returns occupied bandwidth (OBW) results in list mode. To define the statistical length for AVERage, MAXimum and to enable the calculation of the results, use the command method RsCmwEvdoMeas.Configure.MultiEval.ListPy.Segment.Modulation. set. The ranges indicated below apply to all results except standard deviation results. The minimum for standard deviation results equals 0. The maximum equals the width of the indicated range divided by two. Exceptions are explicitly stated. The values listed below in curly brackets {} are returned for each active segment: {...}seg 1, {...}seg 2, ..., {. ..}seg n. The number of active segments n is determined by method RsCmwEvdoMeas.Configure.MultiEval.ListPy.count. 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.

7.4.7.7.4 Maximum

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:OBW:MAXimum
CALCulate:EVDO:MEASurement<Instance>:MEValuation:LIST:OBW:MAXimum
```

class Maximum

Maximum commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Seg_Reliability: List[int]: 0 | 3 | 4 | 8 The segment reliability indicates whether one of the following exceptions occurred in this segment: 0: No error 3: Signal overflow 4: Signal low 8: Synchronization error If a combination of exceptions occurs, the most severe error is indicated.
- Obw: List[enums.ResultStatus2]: float Occupied bandwidth Range: 0 MHz to 16 MHz (SDEViation 0 MHz to 8 MHz) , Unit: MHz

class FetchStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Seg_Reliability: List[int]: 0 | 3 | 4 | 8 The segment reliability indicates whether one of the following exceptions occurred in this segment: 0: No error 3: Signal overflow 4: Signal low 8: Synchronization error If a combination of exceptions occurs, the most severe error is indicated.
- Obw: List[float]: float Occupied bandwidth Range: 0 MHz to 16 MHz (SDEViation 0 MHz to 8 MHz) , Unit: MHz

calculate() → CalculateStruct

```
# SCPI: CALCulate:EVDO:MEASurement<instance>:MEValuation:LIST:OBW:MAXimum
value: CalculateStruct = driver.multiEval.listPy.obw.maximum.calculate()
```

Returns occupied bandwidth (OBW) results in list mode. To define the statistical length for AVERage, MAXimum and to enable the calculation of the results, use the command method RsCmwEvdoMeas.Configure.MultiEval.ListPy.Segment.Modulation. set. The ranges indicated below apply to all results except standard deviation results. The minimum for standard deviation results equals 0. The maximum equals the width of the indicated range divided by two. Exceptions are explicitly stated. The values listed below in curly brackets {} are returned for each active segment: {...}seg 1, {...}seg 2, ..., {. ..}seg n. The number of active segments n is determined by method RsCmwEvdoMeas.Configure.MultiEval.ListPy.count. 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:EVDO:MEASurement<instance>:MEValuation:LIST:OBW:MAXimum
value: FetchStruct = driver.multiEval.listPy.obw.maximum.fetch()
```

Returns occupied bandwidth (OBW) results in list mode. To define the statistical length for AVERage, MAXimum and to enable the calculation of the results, use the command method RsCmwEvdoMeas.Configure.MultiEval.ListPy.Segment.Modulation. set. The ranges indicated below apply to all results except standard deviation results. The minimum for standard deviation results equals 0. The maximum equals the width of the indicated range divided by two. Exceptions are explicitly stated. The values listed below in curly brackets {} are returned for each active segment: {...}seg 1, {...}seg 2, ..., {. ..}seg n. The number of active segments n is determined by method RsCmwEvdoMeas.Configure.MultiEval.ListPy.count. 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.

7.4.7.7.5 StandardDev

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:MEValuation:LIST:OBW:SDEViation
CALCulate:EVDO:MEASurement<Instance>:MEValuation:LIST:OBW:SDEViation
```

class StandardDev

StandardDev commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

class CalculateStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Seg_Reliability: List[int]: 0 | 3 | 4 | 8 The segment reliability indicates whether one of the following exceptions occurred in this segment: 0: No error 3: Signal overflow 4: Signal low 8: Synchronization error If a combination of exceptions occurs, the most severe error is indicated.
- Obw: List[enums.ResultStatus2]: float Occupied bandwidth Range: 0 MHz to 16 MHz (SDEViation 0 MHz to 8 MHz) , Unit: MHz

class FetchStruct

Response structure. Fields:

- Reliability: int: No parameter help available
- Seg_Reliability: List[int]: 0 | 3 | 4 | 8 The segment reliability indicates whether one of the following exceptions occurred in this segment: 0: No error 3: Signal overflow 4: Signal low 8: Synchronization error If a combination of exceptions occurs, the most severe error is indicated.
- Obw: List[float]: float Occupied bandwidth Range: 0 MHz to 16 MHz (SDEViation 0 MHz to 8 MHz) , Unit: MHz

calculate() → CalculateStruct

```
# SCPI: CALCulate:EVDO:MEASurement<instance>:MEValuation:LIST:OBW:SDEViation
value: CalculateStruct = driver.multiEval.listPy.obw.standardDev.calculate()
```

Returns occupied bandwidth (OBW) results in list mode. To define the statistical length for AVERage, MAXimum and to enable the calculation of the results, use the command method RsCmwEvdoMeas.Configure.MultiEval.ListPy.Segment.Modulation. set. The ranges indicated below apply to all results except standard deviation results. The minimum for standard deviation results equals 0. The maximum equals the width of the indicated range divided by two. Exceptions are explicitly stated. The values listed below in curly brackets {} are returned for each active segment: {...}seg 1, {...}seg 2, ..., {. ..}seg n. The number of active segments n is determined by method RsCmwEvdoMeas.Configure.MultiEval.ListPy.count. 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:EVDO:MEASurement<instance>:MEValuation:LIST:OBW:SDEViation
value: FetchStruct = driver.multiEval.listPy.obw.standardDev.fetch()
```

Returns occupied bandwidth (OBW) results in list mode. To define the statistical length for AVERage, MAXimum and to enable the calculation of the results, use the command method RsCmwEvdoMeas.Configure.MultiEval.ListPy.Segment.Modulation. set. The ranges indicated below apply to all results except standard deviation results. The minimum for standard deviation results equals 0. The maximum equals the width of the indicated range divided by two. Exceptions are explicitly stated. The values listed below in curly brackets {} are returned for each active segment: {...}seg 1, {...}seg 2, ..., {. ..}seg n. The number of active segments n is determined by method RsCmwEvdoMeas.Configure.MultiEval.ListPy.count. 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.

7.5 Oltr

SCPI Commands

```
INITiate:EVDO:MEASurement<Instance>:OLTR
ABORt:EVDO:MEASurement<Instance>:OLTR
STOP:EVDO:MEASurement<Instance>:OLTR
```

class Oltr

Oltr commands group definition. 15 total commands, 2 Sub-groups, 3 group commands

abort() → None

```
# SCPI: ABORt:EVDO:MEASurement<instance>:OLTR
driver.oltr.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.

abort_with_opc() → None

```
# SCPI: ABORt:EVDO:MEASurement<instance>:OLTR
driver.oltr.abort_with_opc()


    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.

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

initiate() → None

```
# SCPI: INITiate:EVDO:MEASurement<instance>:OLTR
driver.oltr.initiate()


    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.

initiate_with_opc() → None

```
# SCPI: INITiate:EVDO:MEASurement<instance>:OLTR
driver.oltr.initiate_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 initiate, but waits for the operation to complete before continuing further. Use the RsCmwEvdoMeas.utilities.opc_timeout_set() to set the timeout value.

stop() → None

```
# SCPI: STOP:EVDO:MEASurement<instance>:OLTR
driver.oltr.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() → None

```
# SCPI: STOP:EVDO:MEASurement<instance>:OLTR
driver.oltr.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 RsCmwEvdoMeas.utilities.opc_timeout_set() to set the timeout value.

Cloning the Group

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

Subgroups

7.5.1 State

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:OLTR:STATe
```

class State

State commands group definition. 2 total commands, 1 Sub-groups, 1 group commands

fetch() → RsCmwEvdoMeas.enums.ResourceState

```
# SCPI: FETCh:EVDO:MEASurement<instance>:OLTR:STATe
value: enums.ResourceState = driver.oltr.state.fetch()
```

Queries the main measurement state. Use FETCh:...:STATe:ALL? to query the measurement state including the substates. Use INITiate..., STOP..., ABORt... to change the measurement state.

return meas_state: OFF | RUN | RDY OFF: measurement switched off, no resources allocated, no results available (when entered after ABORt...) RUN: measurement running (after INITiate..., READ...) , synchronization pending or adjusted, resources active or queued RDY: measurement has been terminated, valid results are available

Cloning the Group

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

Subgroups

7.5.1.1 All

SCPI Commands

```
FETCh:EVDO:MEASurement<Instance>:OLTR:STATe:ALL
```

class All

All commands group definition. 1 total commands, 0 Sub-groups, 1 group commands

class FetchStruct

Response structure. Fields:

- Main_State: enums.ResourceState: OFF | RDY | RUN OFF: measurement switched off, no resources allocated, no results available (when entered after STOP...) RDY: measurement has been terminated, valid results are available RUN: measurement running (after INITiate..., READ...) , synchronization pending or adjusted, resources active or queued
- Sync_State: enums.ResourceState: PEND | ADJ | INV PEND: waiting for resource allocation, adjustment, hardware switching ('pending') ADJ: all necessary adjustments finished, measurement running ('adjusted') INV: not applicable because MainState: OFF or RDY ('invalid')
- Resource_State: enums.ResourceState: QUE | ACT | INV QUE: measurement without resources, no results available ('queued') ACT: resources allocated, acquisition of results in progress but not complete ('active') INV: not applicable because MainState: OFF or RDY ('invalid')

fetch() → FetchStruct

```
# SCPI: FETCh:EVDO:MEASurement<instance>:OLTR:STATe:ALL
value: FetchStruct = driver.oltr.state.all.fetch()
```

Queries the main measurement state and the measurement substates. Both measurement substates are relevant for running measurements only. Use FETCh:...:STATe? to query the main measurement state only. Use INITiate..., STOP..., ABORt... to change the measurement state.

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

7.5.2 Sequence<Sequence>

RepCap Settings

```
# Range: Nr1 .. Nr5
rc = driver.oltr.sequence.repcap_sequence_get()
driver.oltr.sequence.repcap_sequence_set(repcap.Sequence.Nr1)
```

class Sequence

Sequence commands group definition. 10 total commands, 1 Sub-groups, 0 group commands Repeated Capability: Sequence, default value after init: Sequence.Nr1

Cloning the Group

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

Subgroups

7.5.2.1 Trace

class Trace

Trace commands group definition. 10 total commands, 2 Sub-groups, 0 group commands

Cloning the Group

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

Subgroups

7.5.2.1.1 Up

SCPI Commands

```
READ:EVDO:MEASurement<Instance>:OLTR:SEQuence<Sequence>:TRACe:UP
FETCh:EVDO:MEASurement<Instance>:OLTR:SEQuence<Sequence>:TRACe:UP
```

class Up

Up commands group definition. 5 total commands, 1 Sub-groups, 2 group commands

fetch(*sequence*=<*Sequence.Default: -1*>) → List[float]

```
# SCPI: FETCh:EVDO:MEASurement<instance>:OLTR:SEQuence<Sequence>:TRACe:UP
value: List[float] = driver.oltr.sequence.trace.up.fetch(sequence = repcap.
↪Sequence.Default)
```

Returns the values of the OLTR traces. For each sequence, DOWN/UP commands return the results of the 100 ms interval following the power up/down step.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

param sequence optional repeated capability selector. Default value: Nr1 (settable in the interface 'Sequence')

return up_power: No help available

read(*sequence*=<*Sequence.Default: -1*>) → List[float]

```
# SCPI: READ:EVDO:MEASurement<instance>:OLTR:SEQuence<Sequence>:TRACe:UP
value: List[float] = driver.oltr.sequence.trace.up.read(sequence = repcap.
↪Sequence.Default)
```

Returns the values of the OLTR traces. For each sequence, DOWN/UP commands return the results of the 100 ms interval following the power up/down step.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

param sequence optional repeated capability selector. Default value: Nr1 (settable in the interface 'Sequence')

return up_power: No help available

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.oltr.sequence.trace.up.clone()
```

Subgroups

7.5.2.1.1.1 State

SCPI Commands

```
READ:EVDO:MEASurement<Instance>:OLTR:SEQuence<Sequence>:TRACe:UP:STATe
FETCh:EVDO:MEASurement<Instance>:OLTR:SEQuence<Sequence>:TRACe:UP:STATe
CALCulate:EVDO:MEASurement<Instance>:OLTR:SEQuence<Sequence>:TRACe:UP:STATe
```

class State

State commands group definition. 3 total commands, 0 Sub-groups, 3 group commands

calculate(*sequence=<Sequence.Default: -1>*) → List[RsCmwEvdoMeas.enums.ResultStatus2]

```
# SCPI: CALCulate:EVDO:MEASurement<instance>:OLTR:SEQuence<Sequence>
↪:TRACe:UP:STATe
value: List[enums.ResultStatus2] = driver.oltr.sequence.trace.up.state.
↪calculate(sequence = repcap.Sequence.Default)
```

For each sequence, state commands return limit violation results 'State Down/Up Power' in 20 equidistant parts of the 100 ms interval following the power down/up step. 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.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

param sequence optional repeated capability selector. Default value: Nr1 (settable in the interface 'Sequence')

return state_up_power: No help available

fetch(*sequence=<Sequence.Default: -1>*) → List[RsCmwEvdoMeas.enums.StatePower]

```
# SCPI: FETCh:EVDO:MEASurement<instance>:OLTR:SEQuence<Sequence>:TRACe:UP:STATe
value: List[enums.StatePower] = driver.oltr.sequence.trace.up.state.
↪fetch(sequence = repcap.Sequence.Default)
```

For each sequence, state commands return limit violation results 'State Down/Up Power' in 20 equidistant parts of the 100 ms interval following the power down/up step. 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.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

param sequence optional repeated capability selector. Default value: Nr1 (settable in the interface 'Sequence')

return state_up_power: No help available

read(*sequence*=<*Sequence.Default: -1*>) → List[RsCmwEvdoMeas.enums.StatePower]

```
# SCPI: READ:EVDO:MEASurement<instance>:OLTR:SEQuence<Sequence>:TRACe:UP:STATe
value: List[enums.StatePower] = driver.oltr.sequence.trace.up.state.
↪read(sequence = repcap.Sequence.Default)
```

For each sequence, state commands return limit violation results 'State Down/Up Power' in 20 equidistant parts of the 100 ms interval following the power down/up step. 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.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

param sequence optional repeated capability selector. Default value: Nr1 (settable in the interface 'Sequence')

return state_up_power: No help available

7.5.2.1.2 Down

SCPI Commands

```
READ:EVDO:MEASurement<Instance>:OLTR:SEQuence<Sequence>:TRACe:DOWN
FETCh:EVDO:MEASurement<Instance>:OLTR:SEQuence<Sequence>:TRACe:DOWN
```

class Down

Down commands group definition. 5 total commands, 1 Sub-groups, 2 group commands

fetch(*sequence*=<*Sequence.Default: -1*>) → List[float]

```
# SCPI: FETCh:EVDO:MEASurement<instance>:OLTR:SEQuence<Sequence>:TRACe:DOWN
value: List[float] = driver.oltr.sequence.trace.down.fetch(sequence = repcap.
↪Sequence.Default)
```

Returns the values of the OLTR traces. For each sequence, DOWN/UP commands return the results of the 100 ms interval following the power up/down step.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

param sequence optional repeated capability selector. Default value: Nr1 (settable in the interface 'Sequence')

return down_power: No help available

read(*sequence*=<*Sequence.Default: -1*>) → List[float]

```
# SCPI: READ:EVDO:MEASurement<instance>:OLTR:SEQuence<Sequence>:TRACe:DOWN
value: List[float] = driver.oltr.sequence.trace.down.read(sequence = repcap.
↪Sequence.Default)
```

Returns the values of the OLTR traces. For each sequence, DOWN/UP commands return the results of the 100 ms interval following the power up/down step.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

param sequence optional repeated capability selector. Default value: Nr1 (settable in the interface 'Sequence')

return down_power: No help available

Cloning the Group

```
# Create a clone of the original group, that exists independently
group2 = driver.oltr.sequence.trace.down.clone()
```

Subgroups

7.5.2.1.2.1 State

SCPI Commands

```
READ:EVDO:MEASurement<Instance>:OLTR:SEQuence<Sequence>:TRACe:DOWN:STATe
FETCh:EVDO:MEASurement<Instance>:OLTR:SEQuence<Sequence>:TRACe:DOWN:STATe
CALCulate:EVDO:MEASurement<Instance>:OLTR:SEQuence<Sequence>:TRACe:DOWN:STATe
```

class State

State commands group definition. 3 total commands, 0 Sub-groups, 3 group commands

calculate(*sequence*=<*Sequence.Default: -1*>) → List[RsCmwEvdoMeas.enums.ResultStatus2]

```
# SCPI: CALCulate:EVDO:MEASurement<instance>:OLTR:SEQuence<Sequence>
↪:TRACe:DOWN:STATe
value: List[enums.ResultStatus2] = driver.oltr.sequence.trace.down.state.
↪calculate(sequence = repcap.Sequence.Default)
```

For each sequence, state commands return limit violation results 'State Down/Up Power' in 20 equidistant parts of the 100 ms interval following the power down/up step. 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.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

param sequence optional repeated capability selector. Default value: Nr1 (settable in the interface 'Sequence')

return state_down_power: No help available

fetch(*sequence*=<*Sequence.Default: -1*>) → List[RsCmwEvdoMeas.enums.StatePower]

```
# SCPI: FETCh:EVDO:MEASurement<instance>:OLTR:SEQuence<Sequence>
↪:TRACe:DOWN:STATe
value: List[enums.StatePower] = driver.oltr.sequence.trace.down.state.
↪fetch(sequence = repcap.Sequence.Default)
```

For each sequence, state commands return limit violation results 'State Down/Up Power' in 20 equidistant parts of the 100 ms interval following the power down/up step. 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.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

param sequence optional repeated capability selector. Default value: Nr1 (settable in the interface 'Sequence')

return state_down_power: No help available

read(*sequence*=<*Sequence.Default: -1*>) → List[RsCmwEvdoMeas.enums.StatePower]

```
# SCPI: READ:EVDO:MEASurement<instance>:OLTR:SEQuence<Sequence>:TRACe:DOWN:STATe
value: List[enums.StatePower] = driver.oltr.sequence.trace.down.state.
↪read(sequence = repcap.Sequence.Default)
```

For each sequence, state commands return limit violation results 'State Down/Up Power' in 20 equidistant parts of the 100 ms interval following the power down/up step. 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.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

param sequence optional repeated capability selector. Default value: Nr1 (settable in the interface 'Sequence')

return state_down_power: No help available

7.6 RpInterval

class RpInterval

RpInterval commands group definition. 4 total commands, 1 Sub-groups, 0 group commands

Cloning the Group

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

Subgroups

7.6.1 Sequence<Sequence>

RepCap Settings

```
# Range: Nr1 .. Nr5
rc = driver.rpInterval.sequence.repcap_sequence_get()
driver.rpInterval.sequence.repcap_sequence_set(repcap.Sequence.Nr1)
```

class Sequence

Sequence commands group definition. 4 total commands, 1 Sub-groups, 0 group commands Repeated Capability: Sequence, default value after init: Sequence.Nr1

Cloning the Group

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

Subgroups

7.6.1.1 Trace

class Trace

Trace commands group definition. 4 total commands, 2 Sub-groups, 0 group commands

Cloning the Group

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

Subgroups

7.6.1.1.1 Up

SCPI Commands

```
READ:EVDO:MEASurement<Instance>:RPINterval:SEQuence<Sequence>:TRACe:UP
FETCh:EVDO:MEASurement<Instance>:RPINterval:SEQuence<Sequence>:TRACe:UP
```

class Up

Up commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

fetch(*sequence*=<*Sequence.Default: -1*>) → List[float]

```
# SCPI: FETCh:EVDO:MEASurement<instance>:RPINterval:SEQuence<Sequence>:TRACe:UP
value: List[float] = driver.rpInterval.sequence.trace.up.fetch(sequence =␣
↪repcap.Sequence.Default)
```

Returns the values of the reference power traces. For each sequence, DOWN/UP commands return the results of the reference power interval for the power up/down step.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

param sequence optional repeated capability selector. Default value: Nr1 (settable in the interface 'Sequence')

return up_power: No help available

read(*sequence*=<*Sequence.Default: -1*>) → List[float]

```
# SCPI: READ:EVDO:MEASurement<instance>:RPINterval:SEQuence<Sequence>:TRACe:UP
value: List[float] = driver.rpInterval.sequence.trace.up.read(sequence = repcap.
↪Sequence.Default)
```

Returns the values of the reference power traces. For each sequence, DOWN/UP commands return the results of the reference power interval for the power up/down step.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

param sequence optional repeated capability selector. Default value: Nr1 (settable in the interface 'Sequence')

return up_power: No help available

7.6.1.1.2 Down

SCPI Commands

```
READ:EVDO:MEASurement<Instance>:RPINterval:SEQuence<Sequence>:TRACe:DOWN
FETCh:EVDO:MEASurement<Instance>:RPINterval:SEQuence<Sequence>:TRACe:DOWN
```

class Down

Down commands group definition. 2 total commands, 0 Sub-groups, 2 group commands

fetch(*sequence*=<*Sequence.Default: -1*>) → List[float]

```
# SCPI: FETCh:EVDO:MEASurement<instance>:RPINterval:SEQuence<Sequence>
↪:TRACe:DOWN
value: List[float] = driver.rpInterval.sequence.trace.down.fetch(sequence =␣
↪repcap.Sequence.Default)
```

Returns the values of the reference power traces. For each sequence, DOWN/UP commands return the results of the reference power interval for the power up/down step.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

param sequence optional repeated capability selector. Default value: Nr1 (settable in the interface 'Sequence')

return down_power: No help available

read(*sequence*=<*Sequence.Default: -1*>) → List[float]

```
# SCPI: READ:EVDO:MEASurement<instance>:RPINterval:SEQuence<Sequence>:TRACe:DOWN
value: List[float] = driver.rpInterval.sequence.trace.down.read(sequence =␣
↪repcap.Sequence.Default)
```

Returns the values of the reference power traces. For each sequence, DOWN/UP commands return the results of the reference power interval for the power up/down step.

Use RsCmwEvdoMeas.reliability.last_value to read the updated reliability indicator.

param sequence optional repeated capability selector. Default value: Nr1 (settable in the interface 'Sequence')

return down_power: No help available

INDEX

F

R

S

T