QDAC-II (Compact)

Command Reference

Commented [MK1]: Could be good to add the QERP sales name Q325 to here - that is what we use in the DoC's

Start your QDAC-II (Compact) experience by first installing the latest firmware

Get the latest QDAC-II Compact pdf manual, firmware update and other information by following this link or the QR code: https://qm.quantum-machines.co/87kjeif6

Authors: Anders Kühle, Joost van der Heijden

Date: September 2024

Version: 1.1

1.1 IEEE Common commands

ABORt

Description	Aborts all functions (functions generators, sweeps etc) and returns the trigger sequences to the idle / wait for trigger state.
Syntax	ABORt
Arguments	None
Query response	N/A
Example(s)	`>SOUR:TRI:TRIG:SOUR BUS (@1:24)` *- sets the trigger source to BUS for triangle generators on all channels.* `>*TRG` *- Fires the global BUS trigger starting the ch 1-24 triangle generators* `> ABOR` *- Stops all triggered functions, including the 24 triangle generators above.*
Notes	This command will stop all waveform generators, DC:LIST or SWEeps, and current measurement. Note that if an output has not reached its set voltage due to slew rate limitations, them the output will continue ramping until the set value has been reached.

*IDN?

Description	Identification query. Returns the product name, model number, serial number etc.
Syntax	*IDN?
Arguments	None
Query response	Returns the product name, model number, serial number, and firmware version.
Example(s)	`>*IDN?` `QDevil,QDAC-II,Q310-1234,1.02`
Notes	

*RST

Description	Reset command. Bring the QDAC-II Compact into the power on state with all settings at their default values.
Syntax	*RST
Arguments	None
Query response	None. Event only.

Example(s)	`>*RST`
Notes	This command does not restart the firmware. It merely aborts all ongoing jobs, sets output voltages to zero, and resets all data structures to their power up defaults and enters the “wait for message” idle state.

*TRG

Description	Global trigger command. Triggers all functions which have BUS as their trigger source.
Syntax	*TRG
Arguments	None
Query response	N/A
Example(s)	`> SOUR1:SQU:TRIG:SOUR BUS` *- Sets the trigger source of the square wave generator on channel 1 to BUS* `>*TRG` *- Fires the global trigger starting the square wave generator on channel 1*
Notes	If no functions have their rigger source set to BUS, then the command has no effect.

*STB?

Description	Status byte query. Returns the status byte register (STBR).
Syntax	*STB?
Arguments	None
Query response	Status byte (numeric response) in the range of 0-255, representing the sum of bits in the STB register, see Table 1.

Example(s)	`>GARBage` *- undefined command.*
	`>*STB?`
	`4`

Notes	

Table 1. STB (Status Byte Register) Bits meaning.

Bit	Value	Name	Meaning
4	16	Message Available	Data is available in the instrument’s output buffer
3	8	Questionable status	One or more enabled bits in the Questionable Data Register are set.
2	4	Error Queue	One or more error messages are present in the Error Queue. These are read and cleared by the SYSTem:ERRor[:NEXT]? Command.
1	2	Not used	Value always 0.
0	1	Not used	Value always 0.

*CLS

Description Clear Status Command. Clears all status data structures and the error queue. This may also clear the status byte. Enable registers are not cleared.

Syntax *CLS

Arguments None

Query response N/A

Example(s)

```
>*STB?                    - Errors messages are available in the error queue
4
>*CLS                     - All event registers are cleared
>*STB?                    - The status byte register has been reset
0
```

Notes The data structures which are cleared are:

- Error/Event Queue

As a consequence, the STB may also be cleared.

1.2 Voltage generators

1.2.1 Output modes

The QDAC-II Compact has two voltage ranges. The purpose is to make best use of the 20 bits of resolution and to get best possible noise performance as the low voltage range exhibits lower noise than the high voltage range.

SOURce:VOLTage:RANGe

Description	Sets or reads the output voltage range for individual channels.
Syntax	SOURce[n][:VOLTage]:RANGe[?] {LOW|HIGH} [,channel_list]
Arguments	LOW: Low voltage range (±2V). HIGH: High voltage range (±10V) (default).
Query response	Returns the current output voltage range for the selected channel(s).
Example(s)	`>SOUR2:RANG LOW` *- sets the voltage range for channel 2 to LOW.* `>SOUR2:RANG?` *- returns the current voltage range of channel 2* `LOW`
Notes	If a voltage range of ±2V is sufficient for your experiments the LOW range is recommended as this mode exhibits the lowest noise level and highest resolution.

SOURce:VOLTage:RANGe:LOW

Description	Query only. Reads the calibrated MINimum or MAXimum output voltage for the LOW output voltage range for one or more channels.
Syntax	SOURce[n][:VOLTage]:RANGe:LOW:{MINimum|MAXimum}[?] [,channel_list]
Arguments	MINimum: Lowest calibrated voltage output. MAXimum: Highest calibrated voltage output.
Query response	Returns the MINimum or MAXimum calibrated output voltage (in units of Volt) for the LOW voltage range of the selected channel(s).
Example(s)	`>SOUR2:RANG:LOW:MIN?` *- returns the minimum voltage for the LOW voltage range on chan. 2.* `-2.00346`
Notes	This command can for example be used by drivers to ensure that set commands are not setting voltages outsides actual limits.

SOURce:VOLTage:RANGe:HIGH

Description	Query only. Reads the calibrated MINimum or MAXimum output voltage for the LOW output voltage range for one or more channels.
Syntax	SOURce[n][:VOLTage]:RANGe:HIGH:{MINimum|MAXimum}[?] [,channel_list]
Arguments	MINimum: Lowest calibrated voltage output. MAXimum: Highest calibrated voltage output.

Query response	Returns the MINimum or MAXimum calibrated output voltage (in units of Volt) for the HIGH voltage range of the selected channel(s).
Example(s)	`>SOUR2:RANG:HIGH:MAX?` *- returns the maximum voltage for the HIGH voltage range on chan. 2.* `10.00321`
Notes	This command can for example be used by drivers to ensure that set commands are not setting voltages outsides actual limits.

The QDAC-II Compact has three low-pass filter options, DC, MEDium and HIGH, for use in different experimental situations: The lower the bandwidth, the lower the noise level. They all two-pole filters. For outputting fast waveforms the HIGH filter option should be used. For DC, slow sweeps and slow waveforms MEDium should be used. The DC filter cut-off is around 10 Hz, so this mode is only use for changing voltages very slowly, or for keeping them at a DC level for a long time. 25 bits of resolution is default in this mode except when executing LISTs or SWEeps.

SOURce:VOLTage:FILTer:LOWPass

Description	Sets or reads the output low-pass filter frequency for individual channels.	
Syntax	SOURce[n][:VOLTage]:FILTer[:LOWPass][?] {DC|MEDium|HIGH} [,channel_list]	
Arguments	DC:	Cut-off ≈ 10 Hz, use for keeping voltage es stable and precise.
	MEDium:	Cut-off ≈ 10kHz, use for DC, slow ramps and slow waveforms.
	HIGH:	Cut-off ≈100kHz, use for fast ramps and fast waveforms.
Query response	Returns the current filter cut-off setting the selected channel(s).	
Example(s)	`>SOUR2:FILT DC`	*- sets the low-pass filter for chan. 2 to DC.*
	`>SOUR2:FILT?` `DC`	*- returns the current low-pass filter setting for chan. 2*
Notes		

1.2.2 DC generator

Not that changing any parameter SWEep parameter while SWEep is running or any LIST parameter while LIST is running, will end their current trigger cycle and thereby stop the sequence and bring the DC generator trigger state back to the *Initiated* state if it is in INITiate:CONTinuous mode or otherwise to is *Idle* state.

SOURce:DC:VOLTage:MODE

Description	Sets or reads the DC generator mode, and hence determines which commands controls the Amplitude sub-system. In FIXed mode (constant poutput voltage) the voltage is controlled by LEVel, in SWEep mode the voltage is controlled by STARt and STOP, and in LIST mode the sequence of voltages is given by the LIST:TVOLTage subsystem. When SLEWrate is <> INF the voltage will ramp from value to value at a finite rate.
Syntax	SOURce[n][:DC][:VOLTage]:MODE[?] FIXed|SWEep|LIST [,channel_list]

Arguments	FIXed:	Constant output (default).
	SWEep:	Outputs a voltage sweep according to the SWEep subsystem settings.
	LIST:	Outputs a voltage sequence according to the LIST subsystem settings.
Query response	Returns the current DC generator MODE.	
Example(s)	`>SOUR2:DC:MODE FIX`	*- sets the DC generator to FIXed mode channel 2.*
	`>SOUR2:DC:MODE?` `FIX`	*- returns the current MODE of the DC generator of ch. 2*
Notes	The IMMediate and TRIGger(ed) values in the FIXed mode are always updated with the most recent output value from the SWEep and LIST systems. Therefore, there will no voltage jumps when switching from SWEep or LIST to FIXed or vice versa. However, as soon as either a SWEep or LIST sequence are triggered there may be a jump to the start value (limited by SLEWrate and analogue filters).	

SOURce:DC:VOLTage:LEVel:IMMediate:AMPLitude

Description	Sets or reads the DC generator voltage component of the output of a selected channel(s) in FIXed mode. When SLEWrate is not INF the voltage will ramp to the set value at a finite rate. Note that this *set* value is always updated whenever a new DC value is set by the Voltage:TRIGger, LIST, or SWEep sub-systems.	
Syntax	SOURce[n][:DC]:VOLTage[:LEVel[:IMMediate[:AMPLitude]]][?] <numeric_value> [,channel_list]	
Arguments	<numeric_value>:	DC voltage in Volts or MINimum or MAXimum
	Minimum value :	-10 or -2V, depending on the range, and calibrated min/max
	Maximum value:	+10 or +2V, depending on the range, and calibrated min/max
	Default:	0
Query response	Returns the current DC component of the channel's output. If reading before the set value has been reached (due to finite slewrate), the read value may differ from the set value.	
Example(s)	`>SOUR2:VOLT 1.12`	*- sets the DC level to 1.12 for channel 2 immediately.*
	`>SOUR2:VOLT?` `1.12`	*- returns the actual temporal DC voltage component of ch.I 2*
Notes	If the DC generator is not in FIXed mode this command has no effect for setting a voltage, only for reading. If SOURce:FILTer = DC and RENHancemen= ON (default) the resolution is enhanced to 25 bit. The resolution enhancement can be disabled by the SOURce:DC:RENHancement command.	

SOURce:DC:VOLTage:LEVel:TRIGger:AMPLitude

Description Sets or reads the next triggered DC voltage output of a selected channel for FIXed mode.
Note that the value is always updated whenever a new DC value is set by the VOLTage:IMMediate sub system or when changed by the LIST, or SWEep sub-systems so that a spurious trigger will not change the output unintentionally.

Syntax SOURce[n][:DC]:VOLTage[:LEVel]:TRIGger[:AMPLitude][?] <numeric_value> [,channel_list]

Arguments

<numeric_value>:	DC voltage in Volts or MINimum or MAXimum
Minimum value :	-10 or -2V, depending on the range, and calibrated min/max
Maximum value:	+10 or +2V, depending on the range, and calibrated min/max
Default:	0

Query response Returns the value set by the most recent TRIG command or as the value always follow the IMMediate, SWEep and LIST subsystems, the most recent value set by any of them.

Example(s)

```
>SOUR2:VOLT:TRIG 3
```
- sets the next triggered DC level to 3 for channel 2.

```
>SOUR2:VOLT:TRIG?
3
```
- returns the next triggered DC voltage output of ch.l 2.

Notes When SLEWrate is not INF the voltage will ramp to the set value at a finite rate when the trigger event occurs.
See also notes for SOURce:DC:VOLTage:LEVel:IMMediate:AMPLitude.
In the current version of the firmware (13-1.57) the TRIG value is unfortunately propagated (triggered) when switching the filter for the channel to or from DC. So please avoid leaving an untriggered value in this register. (In fact it will happen if RENHancement is ON, which it is by default, but not if is off).

SOURce:DC:VOLTage:SLEW

Description Sets or reads the DC generator slew rate of a channel.
When a finite slew rate is set, the rate at which the voltage can change is limited and transients from abruptly stepping the voltage are avoided. The slew rate is active for all DC modes (FIXed, LIST, SWEep).

Syntax SOURce[n][:DC]:VOLTage:SLEW[?] <numeric_value> [,chlist]

Arguments

<numeric_value>:	Maximum voltage change rate in Volts per second (V/s)	
Minimum value :	0.01	
Maximum value:	2e7	corresponds to - *20V/1µs (same as INF)*
Default:	INF	

Query response Returns the current DC generator slew rate of the specified channel(s).

Example(s)

```
>SOUR2:VOLT:SLEW 200
```
- sets the DC slew rate for channel 2 to 200 V/s.

```
>SOUR2:VOLT:SLEW?
200
```
- returns set DC slew rate for channel 2

Notes	Depending on the magnitude of voltage steps, the set slew rate may or may not have an effect on the generated signal. If for example the difference in voltage between sub sequent values divided by the sampling interval of 1µs is smaller than the slew rate, the slew rate will have no effect. Also, if the voltage difference is for example just one DAC increment, the only observed effect would be a delay in going from the present value to the next. Ultimately the slew rate is limited by the hardware, i.e. by the choice of low pass filter and current measurement range. *A small transient (spike) must be expected to appear when changing the slew rate at nonzero voltage outputs.*

SOURce:DC:SWEep:TIME?

Description	Reads the DC generator sweep time, i.e. the duration of a single ramp.
Syntax	SOURce[n][:DC]:SWEep:TIME[?] [chlist]
Arguments	none
Query response	Returns the current DC generator SWEep TIME of the specified channel(s).
Example(s)	`>SOUR2:SWE:TIME?` *- returns the SWEep time for channel 2* `0.05`
Notes	

SOURce:DC:SWEep:DWELl

Description	Sets or reads the DC generator sweep dwell time, i.e. the duration of each level of a stepped ramp.
Syntax	SOURce[n][:DC]:SWEep:DWELl[?] <time> [,chlist]
Arguments	<time>: Ramp/sweep time in seconds Minimum value : 2e-6 (1 sample) Maximum value: 36000 (10 hours) Default: 2e-6
Query response	Returns the current DC generator SWEep DWELl time of the specified channel(s).
Example(s)	`>SOUR2:SWE:DWEL 0.01` *- sets the sweep dwell time for channel 2 to 10ms.* `>SOUR2:SWE:DWEL?` *- returns the sweep dwell time for channel 2* `0.01`
Notes	If DWELl:AUTO is ON, the query of DWELl:TIME will return the auto calculated value. Note that DWELl is simply the time between consecutive voltage steps. It is up to the user to choose DWELl times high enough for reaching a stable voltage level between voltage steps. How long it should be depends on the chosen OUTPut:FILTer, and the SLEWrate. IF DWELL is auto calculated, it may be less than the minimum value. As a step length cannot be less than a two sample periods (2 µs) the resulting sweep will have fewer steps than POINts and the step lengths (their dwell time) will be uneven. Step lengths will also be uneven if DWELl is not an integer number of sampling intervals.

SOURce:DC:SWEep:POINts

Description Sets or reads the number of points in a stepped sweep.
This parameter is not used if GENeration is set to ANALog. POINts is coupled to STEP and SPAN as:
POINts = SPAN / STEP + 1
If the value of STEP is changed POINTs will automatically be changed, but not when SPAN is changed.
POINts is also coupled to TIME

Syntax SOURce[n][:DC]:SWEep:POINts[?] <numeric_value> [,chlist]

Arguments

<numeric_value>:	Number of points
Minimum value :	1
Maximum value:	65536
Default value:	100

Query response Returns the number of points for stepped sweeps of the channel(s).

Example(s)

```
>SOUR2:SWE:POIN 101          - sets the number of sweep points for ch. 2 to 101.
                               There will be 100 levels.
>SOUR2:SWE:POIN?             - returns the DWELI:AUTO state for channel 2
101
```

Notes A sweep with only one point will thus just set the initial (STARt) voltage and then ends after one DWELl time.
For ANALog sweeps, it is easiest to set POINts = 1 and then let DWELl define the sweep TIME (in multiples of the sample rate which is 1e-6 seconds)

SOURce:DC:SWEep:GENeration

Description GENeration defines whether the sweep is stepped or smooth (analog). In case the sweep is ANALog, the parameters POINts and STEP are ignored.

Syntax SOURce[n][:DC]:SWEep:GENeration [?] STEPped|ANALog [,chlist]

Arguments

STEPped:	The sweep will be a stair case sweep
ANALog:	The sweep will be a linear ramp, only discretized by the sampling rate and the DAC precision

Query response Returns the current DC generator sweep generation type.

Example(s)

```
>SOUR2:SWE:GEN ANAL          - sets the SWEep type to be a linear ramp for channel 2.
>SOUR2:SWE:GEN?              - returns the current SWEep type for channel .2
ANAL
```

Notes The duration (TIMe) for an ANALog SWEep is given by DWELI x POINts, even though these two quantities have no individual meaning for an ANALog SWEep.
For ANALog sweeps, it is easiest to set POINts = 1 and then define the sweep TIME by setting DWELI to the requested sweep time in multiples of the sample rate, which is 1e-6 seconds.

SOURce:DC:SWEep:COUNt

Description Defines the number of repetitions of the definedsweep.

Syntax SOURce[n][:DC]:SWEep:COUNt[?] <numeric_value> [,channel_list]

Arguments	<numeric_value>: integer, number of repetitions. Minimum value : 0 Maximum value: 2^{24} – 1 (16777215), or INF Default: 1
Query response	Returns the number of periods/cycles produced when the sweep is started.
Example(s)	`>SOUR2:SWE:COUN 5` - *sets the number of repetitions to 5 for the sweep* `>SOUR2:SWE:COUN?` - returns the number of repetitions `5`
Notes	Note that SWEep and LIST are not intended to run indefinitely but can when COUNt is set to INF.

SOURce:DC:SWEep:NCLeft

Description	Queries the number of remaining cycles (repetitions) of a running SWEep.
Syntax	SOURce[n][:DC]:SWEep:NCLeft? [,channel_list]
Arguments	none
Query response	Returns the number of remaining repetitions of the sweep including an ongoing sweep, when the sweep has been triggered. Right after triggering NCLeft will return COUNt. During the last iteration it will return 1. If the generator is in its IDLE mode or is INITiated but not triggered a NCLeft value of zero is returned.
Example(s)	`>SOUR2:SWE:NCL?` - returns the number of remaining repetitions of the DC sweep on ch 2 `47`
Notes	This command is primarily intended for making it possible for a driver to re-establish the instrument state approximately, if it becomes disconnected to the instrument. But it can also serve as a means for finding out if a sweep has completed.

SOURce:DC:SWEep:VOLTage:STARt

Description Command for setting or reading the initial voltage for a sweep.

Syntax SOURce[n][:DC]:SWEep[:VOLTAGE]:STARt[?] <numeric_value> [,chlist]

Arguments

<numeric_value>:	DC voltage in Volts or MINimum or MAXimum
Minimum value :	-10 or -2V, depending on the range, and calibrated min/max
Maximum value:	+10 or +2V, depending on the range, and calibrated min/max
Default:	0

Query response Returns the current DC generator SWEep start (initial) voltage.

Example(s)

```
>SOUR2:SWE:STAR 1.1
```
- sets the SWEep start voltage to 1.1V on ch. 2.

```
>SOUR2:SWE:STAR?
1.1
```
- returns the current SWEep start voltage for ch. 2.

Notes

SOURce:DC:SWEep:VOLTage:STOP

Description Command for setting or reading the final voltage for a sweep.

Syntax SOURce[n][:DC]:SWEep[:VOLTage]:STOP[?] <numeric_value> [,chlist]

Arguments

<numeric_value>:	DC voltage in Volts or MINimum or MAXimum
Minimum value :	-10 or -2 (Volt), depending on the range, and calibrated min/max
Maximum value:	+10 or +2 (Volt), depending on the range, and calibrated min/max
Default:	0

Query response Returns the current DC generator SWEep stop (final) voltage.

Example(s)

```
>SOUR2:SWE:STOP 2.3
```
- sets the SWEep end voltage on ch. 2 to 2.3V.

```
>SOUR2:SWE:STOP?
2.3
```
- returns the current SWEep end voltage for ch. 2.

Notes

SOURce:DC:LIST:COUNt

Description Defines the number of repetitions carried out for each triggering of LIST execution.

Syntax SOURce[n][:DC]:LIST:COUNt[?] <numeric_value> [,channel_list]

Arguments

<numeric_value>:	integer, number of repetitions.
MINimum value :	0
MAXimum value:	$2^{24} – 1$ (16777215), or INF
Default:	1

Query response Returns the number of periods/cycles produced when the LIST sequence is started.

Example(s)

```
>SOUR2:LIST:COUN 5
```
- sets the number of repetitions to 5 for the list sequence

```
>SOUR2:LIST:COUN?
5
```
- returns the number of repetitions

Notes	Note that SWEep and LIST are not intended to run indefinitely but can when COUNt is set to INF.

SOURce:DC:LIST:NCLeft

Description	Queries the number of remaining repetitions of a running LIST sequence
Syntax	SOURce[n][:DC]:LIST:NCLeft? [,channel_list]
Arguments	None
Query response	Returns the number of remaining repetitions of the LIST sequence (not the number of remaining steps in the sequence) including the ongoing iteration, when the LIST has been triggered. Right after triggering NCLeft will return COUNt. During the last iteration a value of 1 will be returned. If the generator is in its IDLE mode or is INITiated but not triggered a NCLeft value of zero is returned.
Example(s)	`>SOUR2:LIST:NCL?` - returns the number of remaining repetitions of the LIST sequence on ch 2 `6`
Notes	This command is primarily intended for making it possible for a driver to re-establish the instrument state approximately, if it becomes disconnected to the instrument. But it can also serve as a means for finding out if a LIST sequence has completed.

SOURce:DC:LIST:VOLTage

Description	Command for setting or reading the voltages for a LIST sequence.
Syntax	SOURce[n][:DC]:LIST:VOLTage {<binary block>|<list of numeric_values>} [,chlist] SOURce[n][:DC]:LIST:VOLTage? [chlist]
Arguments	<list of numeric_values>: Comma separated list of voltages in units of Volts. <binary block>: IEEE 488.2 binary block (floating point values in Volts). Minimum value : -10 or -2V, depending on the range. Maximum value: +10 or +2V, depending on the range. Default: 0
Query response	Returns the current DC generator LIST voltage sequence in the format specified by the FORMat subsystem.
Example(s)	`>SOUR1:LIST:VOLT 0, 1, 2, 3` *- sets the LIST sequence to 0,1,2, and 3V on ch. 1.* `>SOUR2:LIST:VOLT #18********` *- sets the LIST sequence to the two 4 byte floating point numbers symbolized by the asterisks on ch. 2.* `>SOUR2:LIST:VOLT?` *- returns the current LIST sequence voltages of ch. .* `0.345, 1.982`
Notes	The maximum number of values which can be transferred in the comma separated format is 1024. Use the :APPend command to extend the list – or use the binary format for large lists. The maximum length of a LIST is 65536 points. A value error is produced if any of the voltages exceeds the voltage limits of the channel(s) in the currently set RANGe(s). Clipping will occur when executing the LIST if the produced voltages are outside the limits when the channel is in the LOW RANGe mode. (Voltages may have been set while in the HIGH RANGe mode).

SOURce:DC:LIST:VOLTage:APPend

Description Command for adding voltages to a LIST sequence.

Syntax SOURce[n][:DC]:LIST:VOLTage:APPend {<binary block>|<list of numeric_values>} [,chlist]

Arguments

<list of numeric_values>:	Comma separated list of voltages in units of Volts.
<binary block>:	IEEE 488.2 binary block (floating point values in Volts)
Minimum value :	-10 or -2V, depending on the range, and calibrated min/max.
Maximum value:	+10 or +2V, depending on the range, and calibrated min/max.
Default:	0

Query response None

Example(s) `>SOUR2:LIST:VOLT:APP 5, 4, 3` *- adds voltages 5, 4, 3 to the ch. 2 LIST sequence.*

Notes The maximum number of values which can be transferred in the comma separated format at a time is 1023. For more points please use this command repeatedly or use the binary format for large lists (maximum 65536 points in total).
A value error is produced if any of the voltages exceeds the voltage limits of the channel(s) in the currently set RANGe(s).
Clipping will occur when executing the LIST if the produced voltages are outside the limits when the channel is in the LOW RANGe mode. (Voltages may have been set while in the HIGH RANGe mode).

SOURce:DC:LIST:DWELl

Description Sets or reads the DC LIST generator dwell time, i.e. the duration of each voltage in a LIST sequence

Syntax SOURce[n][:DC]:LIST:DWELl[?] <numeric_value> [,chlist]

Arguments

<numeric_value>:	Time spent at each voltage level in the LIST sequence.
Minimum value :	2e-6 (1 sample)
Maximum value:	36000 (10 hours)
Default:	1e-3

Query response Returns the current DC generator LIST DWELl time of the specified channel(s).

Example(s)
`>SOUR2:LIST:DWEL 0.01` *- sets the list dwell time for channel 2 to 10ms.*
`>SOUR2:LIST:DWEL?` *- returns the list dwell time for channel 2*
`0.01`

Notes Note that DWELl is simply the time between consecutive steps in voltage. It is up to the user to choose DWELl times high enough for reaching a stable voltage level between voltage steps. How long it should be depends on the chosen OUTPut:FILTer, and the SLEWrate.

SOURce:DC:LIST:DIRection

Description Specifies the direction that the sequence list is scanned. If UP is selected, the list is scanned from the first to the last item in the sequence list. If DOWN is selected, the list is scanned from the last item to the first item in the sequence list

Syntax SOURce[n][:DC]:LIST:DIRection[?] UP|DOWN [,chlist]

Arguments	UP: LIST execution order goes from first element to last element (default). DOWN: LIST execution order goes from last element to first element.
Query response	Returns the current DC LIST generator list sequence direction.
Example(s)	`>SOUR2:LIST:DIR DOWN` - *sets the LIST direction to down for channel 2.* `>SOUR2:LIST:DIR?` `DOWN` - *returns the current LIST direction for channel .2*
Notes	

SOURce:DC:LIST:POINts?

Description	Queries the number of points I a LIST sequence. Query only command.
Syntax	SOURce[n][:DC]:LIST:POINts? [,chlist]
Arguments	None
Query response	Returns the number of points in the LIST of one or more channels.
Example(s)	`>SOUR2:LIST:POIN?` `4` - *returns the number of points in the LIST for channel 2*
Notes	

SOURce:DC:LIST:TMODe

Description	Specifies the trigger mode for the LIST sequence, whether triggering should start the automatic run of the sequence or should just advance one point.
Syntax	SOURce[n][:DC]:LIST:TMODe? {AUTO|STEPped} [,chlist]
Arguments	AUTO: (Default) Specifies that a trigger event should start the automatic run of the LIST sequence, advancing points automatically. STEPped: Every time the LIST sequence receives a trigger signal it advances to the next value in the list.
Query response	Returns the current trigger mode setting.
Example(s)	`>SOUR2:LIST:TMODE?` - *Returns the current trigger mode setting for the LIST sequence on channel 2.* `AUTO` `>SOUR2:LIST:TMODE STEP` - *Sets the trigger mode for the LIST sequence on channel 2 to STEPped.*
Notes	In both modes the sequence has to be in the INITiated state in order to respond to trigger signals.

SOURce:DC:DAC:LEVel:IMMediate:AMPLitude

Description Sets or reads the binary DAC value representing the DC voltage component of the output of a selected channel(s) in FIXed mode.
When SLEWrate is not INF the voltage will ramp to the set value at a finite rate.
Note that this *set* value is always updated whenever a new DC value is set by the Voltage:TRIGger, LIST, or SWEep sub-systems.

Syntax SOURce[n][:DC]:DAC[:LEVel[:IMMediate[:AMPLitude]]][?] <numeric_value> [,channel_list]

Arguments

<numeric_value>:	Integer representing the 20-bit DAC digital value.
MINimum value :	-524288
MAXimum value:	524287
Default:	0

Query response Returns the current DC component of the channel's output. If reading before the set value has been reached (due to finite slewrate), the read value may differ from the set value.

Example(s)

```
>SOUR2:DAC 22040
```
- sets the DC level to digital 22040 for channel 2 immediately.

```
>SOUR2:DAC?
```
- returns the actual temporal DC voltage component of channel 2 in DAC binary unconverted digital units

```
22040
```

Notes This command is primarily used during calibration of the voltage outputs of the instrument. Therefore, it is important to stop (ABORt) all waveform generators in this mode. Otherwise the result may be unexpected.
If the DC generator is not in FIXed mode this command has no effect.
As only 20 bit DAC values can be specified it makes no the different if resolution enhancement is active or not.

SOURce:DC:RENHancement

Description Enables or disables resolution DC mode (low bandwidth). When enabled the resolution in FILTer:DC mode will be 25 bit.

Syntax SOURce[n][:DC]:RENHancement][?] {ON|OFF} [,channel_list]

Arguments

ON :	25 bit resolution in DC mode (20 bits in MEDium and HIGH filter modes)
OFF:	20 bit resolution in DC mode
Default:	ON

Query response Returns the active state of resolution enhancement of the specified channel(s).

Example(s)

```
>SOUR2:RENH OFF
```
- Disables resolution enhancement on ch. 2.

```
>SOUR2:RENH?
```
- Returns the current state (ON/OFF9 of resolution enhancement on ch. 2.

```
OFF
```

Notes 25 bits resolution is by default enabled for DC (low bandwidth) mode but can be switched off.
The setting is reset upon power cycling or when executing *RST or SYSTem:RESet.

1.2.3 Sine wave generator

Note that changing any parameter, while the generator is running, will end its current trigger cycle and thereby stop the generator and bring its trigger state back to the *Initiated* state if it is in INITiate:CONTinuous mode or otherwise to is *Idle* state. To re-run, it has to be re-triggered (CONTinuous mode) or re-initiated and re-triggered.

SOURce:SINE:PERiod

Description	Defines the period for the sine wave function generator. Overrides a previously set FREQuency.
Syntax	SOURce[n]:SINE:PERiod[?] <numeric_value> [,channel_list]
Arguments	<numeric_value>: Specifies the period in seconds. Minimum value (s): 2e-6 Maximum value (s): 3600 Default (s): 0.001
Query response	Returns the period (in seconds) for the queried generator or a list of periods in case multiple channels are queried. The reported period will be the same as the value set by the user and not the actual outputted period in case it has been rounded (see notes).

Example(s)	`>SOUR2:SINE:PER 0.0001` *- sets the sine wave period on channel 2 to 0.1ms (10kHz).*
	`>SOUR:SINE:PER 0.0001, (@1:24)` *- sets the sine wave period on ch 1-24 to 0.1ms.*
	`>SOUR2:SINE:PER?` *- returns the period of the sine wave generator on ch 2.*
	`0.0001`
	`>SOUR:SINE:PER? (@2,3,10)` *- returns the sine wave period of ch 2,3 and 10.*
	`0.0001, 0.0001, 0.00002`

Notes	From a period of around 65 ms and above, the sine is not calculated for every 1µs sample, but will be interpolated. Note that not all periods, especially small periods, will give a symmetric waveform. Please see the block about ***Error! Reference source not found.*** in section **Error! Reference source not found.**. Further, the generated curve is smoothed by the low-pass filters (section **Error! Reference source not found.**).

SOURce:SINE:FREQuency

Description Defines the frequency for the sine wave function generator. Overrides a previously set PERiod.

Syntax SOURce[n]:SINE:FREQuency[?] <numeric_value> [,channel_list]

Arguments <numeric_value>: Specifies the frequency in Hz.
MINimum value (Hz): 0.00027778 (corresponding to a 1 hour period)
MAXimum value (Hz): 5e5
Default (Hz): 1000

Query response Returns the frequency (in Hz) for the queried generator or a list of frequencies in case multiple channels are queried. The reported frequency will be the same as the value set by the user and not the actual outputted frequency in case the corresponding PERiod has been rounded (see notes).

Example(s)
```
>SOUR2:SINE:FREQ 25000      - sets the sine wave frequency on channel 2 to 25kHz.
>SOUR2:SINE:FREQ?           - returns the frequency of the sine wave generator on ch 2.
25000
```

Notes From around 15Hz and below, the sine is not calculated for every 1µs sample, but will be interpolated. See note on magic periods giving a faithful reproduction of the waveform under :PERiod.

SOURce:SINE:COUNt

Description Defines the number of periods which are generated for one trigger.

Syntax SOURce[n]:SINE:COUNt[?] <numeric_value> [,channel_list]

Arguments

<numeric_value>:	integer, number of repetitions. -1 denotes infinite
INFinite:	Makes the generator run indefinately
Minimum value :	-1 same as INFinite
Maximum value:	$2^{241} - 1$ (16777215)
Default:	-1

Query response Returns the number of periods/cycles produced when the generator is started.

Example(s)
```
>SOUR2:SINE:COUN 5          - sets the number of repetitions to 5 for the sine wave
>SOUR2:SINE:COUN?           - returns the number of repetitions
5
```

Notes

SOURce:SINe:NCLeft

Description Queries the number of remaining cycles which are generated for one trigger.

Syntax SOURce[n]:SINE:NCLeft? [,channel_list]

Arguments none

Query response	Returns the number of remaining periods/cycles produced of a running generator, including the current cycle. Right after triggering NCLeft will return COUNt and will return 1 during the last cycle. If the generator is in its IDLE mode or is INITiated but not triggered a NCLeft value of zero is returned. After being triggered, if COUNt is INF (-1) then -1 is returned.
Example(s)	`>SOUR2:SINE:NCL?` - returns the number of remaining repetitions for the sine wave generator on ch 2 `3`
Notes	This command is primarily intended for making it possible for a driver to re-establish the instrument state approximately, if it becomes disconnected to the instrument. But it can also serve as a means for finding out if the generator has finished running.

SOURce:SINE:POLarity

Description	Defines the if the positive part of a cycle comes before (normal) or after the negative part.
Syntax	SOURce[n]:SINE:POLarity[?] {NORMal|INVerted} [,channel_list]
Argument	NORMal: default, the positive signal comes before the negative part INVerted: the negative portioin of a period before the positive portion
Query response	Returns the polarity of the sine wave generator.
Example(s)	`>SOUR2:SINE:POL INV` - *inverts the polarity of the sine wave generator* `>SOUR2:SINE:POL?` - returns the polarity `INV`
Notes	

SOURce:SINE:VOLTage:SPAN

Description	Defines the peak to peak voltage span of the generated sine wave, with the centre being equal to the OFFset (default zero).
Syntax	SOURce[n]:SINE[:VOLTage]:SPAN [?] <numeric_value> [,channel_list]
Arguments	<numeric_value>: peak to peak voltage, or MINimum or MAXimum Minimum value : 0V Maximum value: +20V or +4V, depending on the present range Default: 0.2 V
Query response	Returns the top to bottom voltage of the square generator.
Example(s)	`>SOUR2:SIN:SPAN 2` - *sets peak to peak amplitude to 2V* `>SOUR2:SIN:SPAN?` - returns the peak to peak amplitude `2`
Notes	If the SPAN value is set outside its MAXimum and MINimum values, a value error will be reported. However, there is no check if the resulting signal is within the actual voltage range. The signal will simply be clipped if it is.

SOURce:SINE:VOLTage:OFFSet

Description Defines the relative offset of the sine wave generator signal. If OFFSet = 0 the sine wave signal will just swing symmetrically around the source's DC value (including the sum of offsets of any other generators).

Syntax SOURce[n]:SINE[:VOLTage]:OFFSet [?] <numeric_value> [,channel_list]

Arguments

<numeric_value>:	offset in volts	
Minimum value :	-10 or -2V, depending on the range (plus half the SPAN)	
Maximum value:	+10 or +2V, depending on the range (minus half the SPAN)	
Default:	0	

Query response Returns the added offset to the center position of the sine wave signal

Example(s)

```
>SOUR2:SINE:OFFS 0.7        - sets the offset to +0.7V for the sq. wave generator on ch 2.
>SOUR2:SINE:OFFS?           - returns the relative offset
0.7
```

Notes The OFFSet is intended for compensation and not as an alternative way of setting a DC value, as it is only applied when the generator is running.
If the OFFSet value is set outside its MAXimum and MINimum values, a value error will be reported. However, there is no check if the resulting signal is within the actual voltage range. The signal will simply be clipped if it is.

SOURce:SINE:VOLTage:SLEW

Description Sets or reads the sine wave generator slew-rate of a channel.
This is mostly relevant for the initial offset applied when the generator starts (and when it ends), to avoid jumps.
If the SLEW rate is lower than the steepest part of the sine curve, the sine will become distorted and/or a phase lag will be present.

Syntax SOURce[n]:SINE[:VOLTage]:SLEW[?] <numeric_value> [,chlist]

Arguments

<numeric_value>:	Maximum voltage change rate in Volts per second (V/s)	
Minimum value :	0.01	
Maximum value:	2e7	*- 20 V/1µs (same as INF)*
Default:	INF	

Query response Returns the current sine wave generator slew-rate of the specified channel(s).

Example(s)

```
>SOUR2:SINE:VOLT:SLEW 200   - sets sine wave slew-rate for channel 2 to 200 V/s.
>SOUR2:SINE:SLEW?           - returns the set sine wave slew-rate for channel 2
200
```

Notes Depending on the amplitude of the waveform, the set slew rate may or may not have an effect on the generated signal.
Ultimately the slew rate is limited by the hardware, i.e. by the choice of low-pass filter and current measurement range.
A small transient (spike) must be expected to appear when changing the slew-rate at non zero voltage outputs.

1.2.4 Square wave generator

Not that changing any parameter, while the generator is running, will end its current trigger cycle and thereby stop the generator and bring its trigger state back to the *Initiated* state if it is in INITiate:CONTinuous mode or otherwise to is *Idle* state. To re-run, it must be re-triggered (CONTinuous mode) or re-initiated and re-triggered.

SOURce:SQUare:PERiod

Description	Defines the period for the square wave function generator. Overrides a previously set FREQuency.
Syntax	SOURce[n]:SQUare:PERiod[?] <numeric_value> [,channel_list]
Arguments	<numeric_value>: Specifies the period in seconds. Minimum value (s): 2e-6 Maximum value (s): 3600 Default (s): 0.001
Query response	Returns the period (in seconds) for the queried generator or a list of periods in case multiple channels are queried. The reported period will be the same as the value set by the user and *not* the actual outputted period in case it has been rounded (see notes).

Example(s)	`>SOUR2:SQU:PER 0.0001` *- sets the square wave period on channel 2 to 0.1ms (10kHz).*
	`>SOUR:SQU:PER 0.0001, (@1:24)` *- sets the square wave period on ch 1-24 to 0.1ms.*
	`>SOUR2:SQU:PER?` *- returns the period of the square wave generator on ch 2.*
	`0.0001`
	`>SOUR:SQU:PER? (@2,3,10)` *- returns the square wave period of channels 2,3 and 10.*
	`0.0001, 0.0001, 0.00002`

Notes	As a sample is output every micro-second it is not possible to faithfully reproduce any random square waveform. For example, for a 50% duty-cycle the minimum number of points required per period is 2. The second smallest number of points required for producing a symmetric waveform is 4, and so on adding 2 points each time. For duty-cycles different from 50% it is a bit more subtle. Please see the block about ***Error! Reference source not found.*** in section **Error! Reference source not found.**. The generated curve is smoothed by the low-pass filters (section **Error! Reference source not found.**).

SOURce:SQUare:FREQuency

Description Defines the frequency for the square wave function generator. Overrides a previously set PERiod.

Syntax SOURce[n]:SQUare:FREQuency[?] <numeric_value> [,channel_list]

Arguments <numeric_value>: Specifies the frequency in Hz.
MINimum value (Hz): 0.00027778 (corresponding to a 1 hour period)
MAXimum value (Hz): 5e5
Default (Hz): 1000

Query response Returns the frequency (in Hz) for the queried generator or a list of frequencies in case multiple channels are queried. The reported frequency will be the same as the value set by the user and not the actual outputted frequency in case the corresponding PERiod has been rounded (see notes).

Example(s)
```
>SOUR2:SQU:FREQ 25000      - sets the square wave frequency on channel 2 to 25kHz.
>SOUR2:SQU:FREQ?           - returns the frequency of the square wave generator on channel 2.
25000
```

Notes See note on magic periods giving a faithful reproduction of the waveform under :PERiod.

SOURce:SQUare:DCYCle

Description Defines the duty-cycle for the square wave function generator, i.e. the ratio of the duration of the first part of the a period divided by the period..

Syntax SOURce[n]:SQUare:DCYCLE[?] {<numeric_value>|MINimum|MAXimum} [,channel_list]

Arguments <numeric_value>: Specifies the duty cycle in percent.
Minimum value (%): 1
Maximum value (s): 99
Default (%): 50.0

Query response Returns the duty-cycle (in percent) for the queried generator or a list of duty-cycles in case multiple channels are queried. The reported duty-cycle will be the same as the value set by the user and *not* the actual outputted duty-cycle in case any of the up or down parts of the waveform is not an integer number of samples (see notes).

Example(s)
```
>SOUR2:SQU:DCYC 25          - sets the square wave duty cycle on channel 2 to 25%.
>SOUR:SQU:DCYC 20.0, (@3:24)  - sets the square wave duty cycle on ch 3-24 to 20%
>SOUR2:SQU:DCYC?           - returns the duty cycle of the square wave generator on ch 2.
25
>SOUR:SQU:DCYC? (@2,3,10)      - returns the square wave duty cycle of ch 2,3 and 10
25, 20, 20
```

Notes See note on magic periods and duty-cycles giving a faithful reproduction of the waveform under :PERiod.

SOURce:SQUare:COUNt

Description	Defines the number of periods which are generated for one trigger.
Syntax	SOURce[n]:SQUare:COUNt[?] <numeric_value> [,channel_list]
Arguments	<numeric_value>: integer, number of repetitions. -1 denotes infinite INFinite: Makes the generator run indefinately Minimum value : -1 same as INFinite Maximum value: $2^{241} - 1$ (16777215) Default: -1
Query response	Returns the number of periods/cycles produced when the generator is started.
Example(s)	`>SOUR2:SQU:COUN 5` - *sets the number of repetitions to 5 for the square wave* `>SOUR2:SQU:COUN?` - returns the number of repetitions `5`
Notes	

SOURce:SQUare:NCLeft

Description	Queries the number of remaining cycles which are generated for one trigger.
Syntax	SOURce[n]:SQUare:NCLeft? [,channel_list]
Arguments	none
Query response	Returns the number of remaining periods/cycles produced of a running generator, including the current cycle. Right after triggering NCLeft will return COUNt and will return 1 during the last cycle. If the generator is in its IDLE mode or is INITiated but not triggered a NCLeft value of zero is returned. After being triggered if COUNt is INF (-1) then -1 is returned.
Example(s)	`>SOUR2:SQU:ncl?` - returns the number of remaining repetitions for the square wave generator on ch 2 `3`
Notes	This command is primarily intended for making it possible for a driver to re-establish the instrument state approximately, if it becomes disconnected to the instrument. But it can also serve as a means for finding out if the generator has finished running.

SOURce:SQUare:POLarity

Description	Defines the if the positive part of a cycle comes before (normal) or after the negative part.
Syntax	SOURce[n]:SQUare:POLarity[?] {NORMal|INVerted} [,channel_list]
Argument	NORMal default, the positive signal comes before the negative part INVerted: the negative portioin of a period before the positive portion
Query response	Returns the polarity of the square wave generator.
Example(s)	`>SOUR2:SQU:POL INV` - *inverts the polarity of the square wave generator* `>SOUR2:SQU:POL?` - returns the polarity `INV`

Notes	

SOURce:SQUare:TYPe

Description	Defines whether the voltage SPAN is placed symmetrically around the OFFSet, or only on the positive side or the negative side, in which case the generator resembles a pulse generator
Syntax	SOURce[n]:SQUare:TYPe [?] {SYMMetric|POSitive|NEGative} [,channel_list]
Arguments	SYMMetric OFFSet defines the middle of SPAN (default). POSitive: OFFSet defines the level of the second half of the curve (positive pulses) NEGative: OFFSet defines the level of the first half of the curve (negative pulses)
Query response	Returns type/orientation of the waveform generated by the square wave generator.
Example(s)	`>SOUR2:SQU:TYP POS` - *sets the waveform to positive pulses* `>SOUR2:SQU:TYP?` - returns the current waveform type `POS`
Notes	With TYPE = SYMMetric as normal voltage symmetric square wave will be generated. If TYPE = POSitive, positive pulses will be generated. Likewise if TYPE = NEGative, negative pulses will be generated – when POLarity is NORMal.

SOURce:SQUare:VOLTege:SPAN

Description	Defines the peak to peak voltage span of the generated square wave, with the centre being equal to the OFFset (default zero).
Syntax	SOURce[n]:SQUare[:VOLTage]:SPAN [?] <numeric_value> [,channel_list]
Arguments	<numeric_value>: peak to peak voltage, or MINimum or MAXimum Minimum value : 0V Maximum value: +20V or +4V, depending on the present range Default: 0.2 V
Query response	Returns the top to bottom voltage of the square generator.
Example(s)	`>SOUR2:SQU:SPAN 2` - *sets peak to peak amplitude to 2V* `>SOUR2:SQU:SPAN?` - returns the peak to peak amplitude `2`
Notes	If the SPAN value is set outside its MAXimum and MINimum values, a value error will be reported. However, there is no check if the resulting signal is within the actual voltage range. The signal will simply be clipped if it is.

SOURce:SQUare:VOLTage:OFFSet

Description	Defines the relative offset of the square wave generator signal. If OFFSet = 0 the square wave signal will just swing symmetrically around the source's DC value (including the sum of offsets of any other generators).
Syntax	SOURce[n]:SQUare[:VOLTage]:OFFSet [?] <numeric_value> [,channel_list]

Arguments	<numeric_value>:	offset in volts
	Minimum value :	-10 or -2V, depending on the range (plus half the SPAN)
	Maximum value:	+10 or +2V, depending on the range (minus half the SPAN)
	Default:	0

Query response Returns the added offset to the center position of the square wave signal

Example(s)

```
>SOUR2:SQU:OFFS 0.7
```
- sets the offset to +0.7V for the sq. wave generator on ch 2.

```
>SOUR2:SQU:OFFS?
0.7
```
- returns the relative offset

Notes If the OFFSet value is set outside its MAXimum and MINimum values, a value error will be reported. However, there is no check if the resulting AWG signal is within the actual voltage range. The signal will simply be clipped if it is.

SOURce:SQUare:VOLTage:SLEW

Description Sets or reads the square wave generator slew-rate of a channel.
When a finite slew-rate is set, the rate at which the voltage can change is limited. Together with SPAN and OFFSet, this effectively defines the rise and fall times at the edges of the waveform.

Syntax SOURce[n]SQUare[:VOLTage]:SLEW[?] <numeric_value> [,chlist]

Arguments	<numeric_value>:	Maximum voltage change rate in Volts per second (V/s)	
	Minimum value :	0.01	
	Maximum value:	2e7	*- 20 V/1µs (same as INF)*
	Default:	INF	

Query response Returns the current square wave generator slew-rate of the specified channel(s).

Example(s)

```
>SOUR2:SQU:VOLT:SLEW 200
```
- sets square wave slew-rate for channel 2 to 200 V/s.

```
>SOUR2:SQU:VOLT:SLEW?
200
```
- returns the set square wave slew-rate for channel 2

Notes Depending on the amplitude of the waveform, the set slew-rate may or may not influence the generated signal.
Ultimately the slew-rate is limited by the hardware, i.e. by the choice of low-pass filter and current measurement range.
A small transient (spike) must be expected to appear when changing the slew-rate at non-zero voltage outputs.

1.2.5 Triangle generator

Not that changing any parameter, while the generator is running, will end its current trigger cycle and thereby stop the generator and bring its trigger state back to the *Initiated* state if it is in INITiate:CONTinuous mode or otherwise to is *Idle* state. To re-run, it has to be re-triggered (CONTinuous mode) or re-initiated and re-triggered.

SOURce:TRIangle:PERiod

Description	Defines the period for the triangle function generator. Overrides a previously set FREQuency.
Syntax	SOURce[n]:TRIangle:PERiod[?] <numeric_value> [,channel_list]
Arguments	<numeric_value>: Specifies the period in seconds. Minimum value (s): 4e-6 Maximum value (s): 3600 Default (s): 0.001
Query response	Returns the period (in seconds) for the queried generator or a list of periods in case multiple channels are queried. The reported period will be the same as the value set by the user and not the actual outputted period in case it has been rounded (see notes).

Example(s)	`>SOUR2:TRI:PER 0.0001` *- sets the triangle period on channel 2 to 0.1ms (10kHz).*
	`>SOUR:TRI:PER 0.0001,(@1:24)` *- sets the triangle period on ch 1-24 to 0.1ms.*
	`>SOUR2:TRI:PER?` *- returns the period of the triangle generator on ch 2.*
	`0.0001`
	`>SOUR:TRI:PER? (@2,3,10)` *- returns the triangle period of ch 2,3 and 10.*
	`0.0001, 0.0001, 0.00002`

Notes	As samples are output every micro-second it is not possible to faithfully reproduce any random triangle waveform. For example, for a 50% duty-cycle the minimum number of points required per period is 4. The second smallest number of points required for producing a symmetric waveform is 8, and so on adding 4 points each time. For duty-cycles different from 50% it is a bit more subtle. Please see the block about ***Error! Reference source not found.*** in section **Error! Reference source not found.**. Further, the generated curve is smoothed by the low-pass filters (section **Error! Reference source not found.**).

SOURce:TRIangle:FREQuency

Description	Defines the frequency for the triangle function generator. Overrides a previously set PERiod.
Syntax	SOURce[n]:TRIangle:FREQuency[?] <numeric_value> [,channel_list]
Arguments	<numeric_value>: Specifies the frequency in Hz. Minimum value (Hz): 0.00027778 (corresponding to a 1 hour period) Maximum value (Hz): 250e5 Default (Hz): 1000
Query response	Returns the frequency (in Hz) for the queried generator or a list of frequencies in case multiple channels are queried. The reported frequency will be the same as the value set by the user and not the actual outputted frequency in case the corresponding PERiod has been rounded (see notes).
Example(s)	`>SOUR2:TRI:FREQ 25000` *- sets the triangle frequency on channel 2 to 25kHz.* `>SOUR2:TRI:FREQ?` *- returns the frequency of the triangle generator on ch 2.* `25000`
Notes	See note on magic periods giving a faithful reproduction of the waveform under :PERiod.

SOURce:TRIangle:DCYCle

Description	Defines the duty-cycle for the triangle function generator.
Syntax	SOURce[n]:TRIangle:DCYCle[?] {<numeric_value>|MINimum|MAXimum} [,channel_list]
Arguments	<numeric_value>: Specifies the duty cycle in percent. Minimum value (%): 1 Maximum value (s): 99 Default (%): 50.0
Query response	Returns the duty-cycle (in percent) for the queried generator or a list of duty-cycles in case multiple channels are queried. The reported duty-cycle will be the same as the value set by the user and *not* the actual outputted duty-cycle in case one of the up or down parts of the waveform is *not* an integer number of samples (see notes).

Example(s)	`>SOUR2:TRI:DCYC 25` *- sets the triangle duty cycle on channel 2 to 25%.*
	`>SOUR:TRI:DCYC 20.0, (@3:24)` *- sets the triangle duty cycle on ch 3-24 to 20%*
	`>SOUR2:TRI:DCYC?` *- returns the duty cycle of the triangle generator on ch 2.*
	`25`
	`>SOUR:TRI:DCYC? (@2,3,10)` *- returns the triangle duty cycle of ch 2,3 and 10*
	`25, 20, 20`

Notes	See note on magic periods giving a faithful reproduction of the waveform under :PERiod. Each part (up, down) of the waveform needs to be an integer number of samples for faithful reproduction.

SOURce:TRIangle:COUNt

Description	Defines the number of periods which are generated for one trigger.
Syntax	SOURce[n]:TRIangle:COUNt[?] <numeric_value> [,channel_list]

Arguments	<numeric_value>:	integer, number of repetitions. -1 denotes infinite
	INFinite:	Makes the generator run indefinately
	Minimum value :	-1 same as INFinite
	Maximum value:	$2^{241} - 1$ (16777215)
	Default:	-1

Query response Returns the number of periods/cycles produced when the generator is started.

Example(s)

```
>SOUR2:TRI:COUN 5
```
- sets the number of repetitions to 5 for the triangle

```
>SOUR2:TRI:COUN?
5
```
- returns the number of repetitions

Notes

SOURce:TRIangle:NCLeft

Description Queries the number of remaining cycles which are generated for one trigger.

Syntax SOURce[n]:TRIangle:NCLeft? [,channel_list]

Arguments none

Query response Returns the number of remaining periods/cycles produced of a running generator, including the current cycle. Right after triggering NCLeft will return COUNt and will return 1 during the last cycle.
If the generator is in its IDLE mode or is INITiated but not triggered a NCLeft value of zero is returned.
If COUNt is -1 (infinite) then -1 is returned.

Example(s)

```
>SOUR2:TRI:NCL?
```
- returns the number of remaining repetitions for the triangle generator on ch 2

```
3
```

Notes This command is primarily intended for making it possible for a driver to re-establish the instrument state approximately, if it becomes disconnected to the instrument.
But it can also serve as a means for finding out if the generator has finished running.

SOURce:TRIangle:POLarity

Description Defines the if the positive part of a cycle comes before (normal) or after the negative part.

Syntax SOURce[n]:TRIangle:POLarity[?] {NORMal|INVerted} [,channel_list]

Argument	NORMal	default, the positive signal comes before the negative part
	INVerted:	the negative portioin of a period before the positive portion

Query response Returns the polarity of the triangle generator.

Example(s)

```
>SOUR2:TRI:POL INV
```
- inverts the polarity of the triangle generator

```
>SOUR2:TRI:POL?
INV
```
- returns the polarity

Notes

SOURce:TRIangle:VOLTage:SPAN

Description	Defines the peak to peak voltage span of the generated triangle, with the centre being equal to the OFFset (default zero).
Syntax	SOURce[n]:TRIangle[:VOLTage]:SPAN [?] <numeric_value> [,channel_list]
Arguments	<numeric_value>: peak to peak voltage, or MINimum or MAXimum Minimum value : 0V Maximum value: +20V or +4V, depending on the present range Default: 0.2 V
Query response	Returns the top to bottom voltage of the square generator.
Example(s)	`>SOUR2:TRI:SPAN 2` - *sets peak to peak amplitude to 2V* `>SOUR2:TRI:SPAN?` `2` - returns the peak to peak amplitude
Notes	If the SPAN value is set outside its MAXimum and MINimum values, a value error will be reported. However, there is no check if the resulting signal is within the actual voltage range. The signal will simply be clipped if it is.

SOURce:TRIangle:VOLTage:OFFSet

Description	Defines the relative offset of the triangle generator signal. If OFFSet = 0 the triangle signal will just swing symmetrically around the source's DC value (including the sum of offsets of any other generators).
Syntax	SOURce[n]:TRIangle[:VOLTage]:OFFSet [?] <numeric_value> [,channel_list]
Arguments	<numeric_value>: offset in volts Minimum value: -10 or -2V, depending on the range (plus half the SPAN) Maximum value: +10 or +2V, depending on the range (minus half the SPAN) Default: 0
Query response	Returns the added offset to the center position of the triangle signal
Example(s)	`>SOUR2:TRI:OFFS 0.7` - *sets the offset to +0.7V for the sq. wave generator on ch 2.* `>SOUR2:TRI:OFFS?` `0.7` - returns the relative offset
Notes	The OFFset is intended for compensation and not as an alternative way of setting a DC value, as it is only applied when the generator is running. If the OFFSet value is set outside its MAXimum and MINimum values, a value error will be reported. However, there is no check if the resulting AWG signal is within the actual voltage range. The signal will simply be clipped if it is.

SOURce:TRIangle:VOLTage:SLEW

Description	Sets or reads the triangle generator slew-rate of a channel. This is mostly relevant for the initial offset applied when the generator starts (and when it ends), to avoid jumps. If the SLEW rate is lower than SPAN / (PERiod x MIN[DCYCle, 100-DCYCle]/100%) then a phase lag will be introduced.
Syntax	SOURce[n]:TRIangle[:VOLTage]:SLEW[?] <numeric_value> [,chlist]

Arguments	<numeric_value>:	Maximum voltage change rate in Volts per second (V/s)	
	Minimum value :	0.01	
	Maximum value:	2e7	*- 20V/1µs (same as INF)*
	Default:	INF	
Query response	Returns the current triangle generator slew-rate of the specified channel(s).		
Example(s)	`>SOUR2:TRI:VOLT:SLEW 200`	*- sets triangle slew-rate for channel 2 to 200 V/s.*	
	`>SOUR2:TRI:SLEW?` `200`	*- returns the set triangle slew-rate for channel 2*	
Notes	Ultimately the slew rate is limited by the hardware, i.e. by the choice of low-pass filter and current measurement range. *A small transient (spike) must be expected to appear when changing the slew rate at non-zero voltage outputs.*		

1.2.6 AWG generator

Not that changing any parameter, while the generator is running, will end its current trigger cycle and thereby stop the generator and bring its trigger state back to the *Initiated* state if it is in INITiate:CONTinuous mode or otherwise to is *Idle* state. To re-run, it has to be re-triggered (CONTinuous mode) or re-initiated and re-triggered. If no TRACe is DEFined or if it does not exist an execution error will be produced.

SOURce:AWG:DEFine

Description	Sets or queries which TRACe to be used for the current AWG for the specified channel.
Syntax	SOURce[n]:AWG:DEFine[?] <quoted string> [,channel_list]
Arguments	<quoted string>: Name of the TRACe to be used for this AWG.
Query response	Returns the name of the TRACe assigned to the current AWG. If no TRACe has been assigned (by DEF) an empty quoted string is returned.
Example(s)	`>SOUR1:AWG:DEF "Mywave"` - *Assigns the TRACe named "Mywave" to the AWG on ch 1.* `>SOUR1:AWG:DEF?` - *returns name of the TRACe assigned to channel 1.* `"Mywave"`
Notes	Unless the AWG is already running, DEFine will *not* produce an error if the named TRACe is yet not present. But, when AWG is started it will check for the presence of the TRACe and will produce an execution error if the TRACe cannot be found.

SOURce:AWG:COUNt

Description	Defines or queries the number of periods which are generated for one trigger.
Syntax	SOURce[n]:AWG:COUNt[?] <numeric_value> [,channel_list]
Arguments	<numeric_value>: integer, number of repetitions. -1 denotes infinite INFinite: Makes the generator run indefinately Minimum value : -1 same as INFinite Maximum value: $2^{24} - 1$ (16777215) Default: 1
Query response	Returns the number of periods/cycles produced when the generator is started.
Example(s)	`>SOUR2:AWG:COUN 5` - *sets the number of repetitions to 5 for the AWG on ch 2* `>SOUR2:AWG:COUN?` - returns the number of repetitions `5`
Notes	

SOURce:AWG:NCLeft

Description Query only. Queries the number of remaining cycles which are generated for one trigger.

Syntax SOURce[n]:AWG:NCLeft? [,channel_list]

Arguments none

Query response Returns the number of remaining periods/cycles produced of a running generator, including the current cycle. Right after triggering NCLeft will return COUNt and will return 1 during the last cycle.
If the generator is in its IDLE mode or is INITiated but not triggered a NCLeft value of zero is returned.
After being triggered if COUNt is INF (-1) then -1 is returned.

Example(s)

```
>SOUR2:AWG:NCL?
```
- returns the number of remaining repetitions for the AWG on ch 2

```
3
```

Notes This command is primarily intended for making it possible for a driver to re-establish the instrument state approximately, if it becomes disconnected to the instrument.
But it can also serve as a means for finding out if the generator has finished running.

SOURce:AWG:VOLTage:SCALe

Description Defines or queries the scaling factor applied to amplitudes in the DEFined TRACe, which will convert the TRACe data to voltages.

Syntax SOURce[n]:AWG[:VOLTage]:SCALe[?] <numeric_value> [,channel_list]

Arguments

<numeric_value>:	Scaling factor (implicit unit V), or MINimum or MAXimum
Minimum value:	-10
Maximum value:	10
Default:	1

Query response Returns the scaling factor for the AWG(s) on the selected channel(s).

Example(s)

```
>SOUR2:AWG:SCAL 8.8
```
- applies a scaling factor of 8.8V to the TRACe for the AWG on ch 2.

```
>SOUR2:AWG:SCAL?
8.8
```
- returns the AWG scaling factor

Notes If the SCALe value is set outside its MAXimum and MINimum values, a value error will be reported. However, there is no check if the resulting signal is within the actual voltage range. The signal will simply be clipped if it is.

SOURce:AWG:VOLTage:OFFSet

Description	Defines the relative offset of the AWG signal added to the scaled TRACe amplitudes. If OFFSet = 0 AWG will just output the TRACe values multiplied by the SCALe factor.
Syntax	SOURce[n]:AWG[:VOLTage]:OFFSet [?] <numeric_value> [,channel_list]
Arguments	<numeric_value>: offset in volts Minimum value : -10 (volts), depending on the range Maximum value: +10 (volts), depending on the range Default: 0
Query response	Returns the offset added to the SCALed TRACe values on the AWG.
Example(s)	`>SOUR2:AWG:OFFS 0.7` *- sets the offset to +0.7V for AWG generator on ch 2.* `>SOUR2:AWG:OFFS?` `0.7` - returns the offset for the AWG
Notes	The OFFSet is intended for compensation and not as an alternative way of setting a DC value, as it is only applied when the generator is running. If the OFFSet value is set outside its MAXimum and MINimum values, a value error will be reported. However, there is no check if the resulting AWG signal is within the actual voltage range. The signal will simply be clipped if it is.

SOURce:AWG:VOLTage:SLEW

Description	Sets or reads the AWG generator slew-rate of a channel.
Syntax	SOURce[n]AWG[:VOLTage]:SLEW[?] <numeric_value> [,chlist]
Arguments	<numeric_value>: Maximum voltage change rate in Volts per second (V/s) Minimum value : 0.01 Maximum value: 2e7 *- 20V/1µs (same as INF)* Default: INF
Query response	Returns the current AWG generator slew-rate of the specified channel(s).
Example(s)	`>SOUR2:AWG:VOLT:SLEW 200` *- sets AWG slew-rate for channel 2 to 200 V/s.* `>SOUR2:AWG:SLEW?` `200` *- returns the set AWG slew-rate for channel 2*
Notes	Depending on the amplitude of the waveform, the set slew-rate may or may not have an effect on the generated signal. Ultimately the slew-rate is limited by the hardware, i.e. by the choice of low-pass filter and current measurement range. *A small transient (spike) must be expected to appear when changing the slew-rate at non-zero voltage outputs.*

1.2.7 Trace (AWG) curve management

TRACe:DEFine

Description	Defines and names a new trace and allocates memory for it. Up to 24 traces can be defined.
Syntax	TRACe:DEFine “quoted_name” , <size>
Arguments	“quoted_name” : Name of trace in quotes, maximum 16 ASCII characters. <size>: Size in number of data points for the trace (even positive integer), MINimum 4, MAXimum 6,291,456)
Query response	None, set only command.
Example(s)	`>TRAC:DEF “Mytrace1”,1000` - *Defines a trace named Mytrace1 and allocates 1000 datapoints in memory.*
Notes	The device will allocate trace memory corresponding to 4 bytes per data point. If the DEFine will exceed the amount of available memory an error “-225 Out of memory” is produced. If an uneven number of points are defined, the last point will be duplicated instead of generating an error.

TRACe:CATalog

Description	Returns a list of names of all traces in memory.
Syntax	TRACe:CATalog?
Arguments	None
Query response	Query returns a comma-separated list of quoted strings containing the names of all traces.
Example(s)	`>TRAC:CAT?` `“Mytrace1”,”Mytrace2”` - *Returns catalogue of traces*
Notes	If there are no <trace_name> defined, a single empty string is returned.

TRACe:REMove:ALL

Description	Deletes all traces and frees up the trace memory.
Syntax	TRACe:REMove:ALL
Arguments	None
Query response	None, set only command.
Example(s)	`>TRAC:REM:ALL` *- Deletes all traces.*
Notes	Deletes all traces and frees up the trace memory. If any trace is currently assigned to a generator, an error will be generated, and nothing happens. So, in order to delete a trace it must be disassociated with the channels' AWGs. Because of memory fragmentation, deleting traces in order to make space for longer traces may not work successfully. Use REM:ALL to reset the entire trace memory.

TRACe:DATA

Description	DATa is used for transfering trace data to the instrument.
Syntax	TRACe:DATA "quoted_name", <binary block>
Arguments	"quoted_name" : Name of defined trace in quotes. <binary block>: IEEE 488.2 binary block (floating point values) (range -1 to 1)
Query response	None, set only command.
Example(s)	`>TRAC:DAT "Mytrace1", #18**********` *- sets the TRACe data to the three 4-byte floating point numbers symbolized by the asterisks. Of course an unrealistically short trace.*
Notes	When transferring a binary block, the length has to be exactly the same as allocated using the DEFine command. Otherwise an error is produced. When the trace "played" by one of the AWGs, the data will be scaled and offset (into units of Volts) according to the settings on the specific channel's AWG. All data values must be between -1 and 1.

1.2.8 Triggering of voltage generators

SOURce:{ }:ABORt

Description Stops the specified trigger sequence (generator) and brings it to the idle state. This includes setting CONTinuous to OFF to avoid re-triggering.

Syntax SOURce[n]:{<name>|ALL}:ABORt [,channel_list]

Arguments

<name>:	DC, SQUare, SINe, TRIange, AWG
{source}:	IMMediate, BUS, HOLD, EXTernal1,… EXTernal5, INTernal1,… INTernal14

Query response None. This command is an event.

Example(s)

`>SOUR2:TRI:ABOR`	- *Stops the triangle generator on channel 2*
`>SOUR2:ALL:ABOR`	- *Stops all generators on channel 2*

Notes For stopping all generators and ongoing current measurements, the general ABORt command can be used.
The behavior of setting INIT:CONT to OFF is not according to the standard SCPI protocol, but seems to be more useful.
Note that running LIST and SWEep sequences will also be aborted and reset to their starting points. The most recent DC value will, however, stay.
Beware that if an output has not reached its set voltage due to slew rate limitations, then the output will continue ramping until the set value has been reached.

SOURce:{ }:TRIGger:SOURce

Description Sets or reads which trigger source is used for a given generator for the given channel(s).

Syntax
SOURce[n]:{<name>|ALL}:TRIGger:SOURce {source} [,channel_list]
SOURce[n]:<name>:TRIGger:SOURce? [,channel_list]

Arguments

<name>:	DC, SQUare, SINe, TRIangle, AWG
{source}:	IMMediate, BUS, HOLD, EXTernal1,… EXTernal5, INTernal1,… INTernal14
BUS:	Use the global trigger (*trg command)
HOLD:	Event detection is disabled for this sequence
EXTernal#:	Use hardware trigger no. #
INTernal#:	Use internal trigger no. #
IMMediate:	(default) If INITiate:IMMediate is executed, a single trigger event will be processed immediately.

Query response Returns the current trigger source for the specified channel(s) and generator.

Example(s)

`>SOUR2:TRI:TRIG:SOUR BUS`	- *sets the trigger source to BUS* for the triangle generator on ch 2
`>SOUR2:SIN:TRIG:SOUR INT1`	- sets the trigger source to internal trigger no. 1 for the sine generator on ch 2
`>SOUR2:SIN:TRIG:SOUR? INT1`	- *returns the trigger source for channel 2, SINe*

Notes	1) SOUR[n]:ALL:TRIG:SOUR sets or reads the trigger source for all generators 2) EXTernal1..4 are the galvanically isolated BNC trigger input. The other EXT5..10 are reserved for future use on the Digital I/O connector.

SOURce:{ }:INITiate:IMMediate

Description	Initiates the specified generator trigger sequence, making it respond to a single trigger event, where after it returns to its IDLE state (unless CONTinuos is ON). This is the simplest way to start a function generator, a sweep or a list sequence for a single run.
Syntax	SOURce[n]:{<name>|ALL}:INITiate[:IMMediate] [,channel_list]
Arguments	<name>: DC, SQUare, SINe, TRIangle, AWG.
Query response	None. This command is an event and has no readable state.
Example(s)	`>SOUR2:SIN:TRIG:SOUR IMM` - *Selects IMMediate as the trigger source for the sine generator on ch. 2.* `>SOUR2:SIN:INIT` - *Makes the sine generator on channel 2 wait for a single trigger event. As the trigger source is set to IMMediate, the generator will start when this command is issued.*
Notes	When <name> has SOURce:IMMediate, this command will start <name> immediately. The command has no effect if INIT:CONT is ON. When the channel is in FILter = DC mode, trying to INITiate waveform generators will produce an error, as waveform generators are not available in DC mode.

SOURce:{ }:INITiate:CONTinuous

Description	This command determines whether the trigger sequence for the specified generator is continuously initiated (waiting for trigger) or not. When set to ON the trigger sequence will exit its IDLE state and go to the initiated state waiting for a trigger event. Every time a trigger cycle is completed the generator will go back to the initiated state and wait for a new trigger.
Syntax	SOURce[n]:{<name>|ALL}:INITiate:CONTinuous {ON|OFF} [,channel_list]
Arguments	<name>: DC, SQUare, SINe, TRIangle, AWG. ON: CONTinuous mode is switched on. OFF: CONTinuous mode is switched off.
Query response	Returns the current INITiate:CONTinuous state of the generator.
Example(s)	`>SOUR2:TRI:INIT:CONT ON` - *Makes the triangle generator on channel 2 start waiting for trigger events. After a completed trigger cycle the generator will wait for a new trigger event.* `>SOUR2:TRI:INIT:CONT?` - *Queries the CONTinuous state of the triangle generator on channel 2.* `ON`

Notes	If TRIGger:SOURce is IMMediate the generator will be re-triggering indefinitely until an *rst or ABORt message is received or CONT is turned OFF. When set to OFF an ongoing trigger cycle is allowed to end before the generator goes to its IDLE state. When the channel is in FILter = DC mode, trying to INITiate waveform generators will produce an error, as waveform generators are not available in DC mode.

SOURce:{ }:DELay

Description	Specifies a delay from the trigger event until the generator is actually started.
Syntax	SOURce[n]:{<name>|ALL}:DELay[?] <numeric_value> [,channel_list]
Arguments	<name>: DC, SQUare, SINe, TRIangle, AWG. <numeric_value>: Delay in units of seconds Minimum value : 0 (default) Maximum value: 3600
Query response	Returns the current start delay of the generator.
Example(s)	`>SOUR2:TRI:DEL 0.02` - *Makes the triangle generator on channel 2 start 20 ms after being triggered.* `>SOUR2:TRI:DEL?` `0.02` - *Returns the start delay of the triangle generator on ch. 2.*
Notes	

1.3 Commands for controlling triggers

1.3.1 Internal triggers and markers

Internal triggers are either tripped by a generator marker event or by a manual (programmatic) trigger signal.

SOURce:{ }:MARKer:STARt:TNUMber

Description	Sets which internal trigger should be paired with the STARt MARKer event for the specified generator. The STARt MARKer occurs when the specified generator starts.
Syntax	SOURce[n]:{<name>}:MARKer:STARt[:TNUMber][?] <numeric_value>
Arguments	<name>: DC, SQUare, SINe, TRIangle, AWG. <numeric_value>: Internal trigger number (positive integer) Minimum value : 0* (default) Maximum value: 14
Query response	Returns the currently internal trigger number paired with the specified marker.
Example(s)	`>SOUR2:SQU:MARK:STAR 3` - *Pairs the STARt marker of the square generator on channel 2 with internal trigger number 3.* `>SOUR2:SQU:MARK:STAR?` - *Returns the internal trigger number which the STARt MARKer on of the square generator on ch 2 is connected to.* `3`
Notes	*Trigger numbers are 1-14. Zero means *no-trigger*. There is a one-to-one correspondence between internal triggers and marker events. However, there is (currently) no command for querying which MARKer event a specific internal trigger is connected to, only the reverse.

SOURce:{ }:MARKer:END:TNUMber

Description	Sets which internal trigger should be paired with the END MARKer event for the specified generator. The END MARKer occurs when the specified generator ENDs (after completing its specified periods, COUNt, or when ABORted).
Syntax	SOURce[n]:{<name>}:MARKer:END[:TNUMber][?] <numeric_value>
Arguments	<name>: DC, SQUare, SINe, TRIangle, AWG. <numeric_value>: Internal trigger number (positive integer) or zero Minimum value : 0* (default) Maximum value: 14
Query response	Returns the currently internal trigger number paired with the specified marker.
Example(s)	`>SOUR2:SQU:MARK:END 4` - *Pairs the END marker of the square generator on channel 2 with internal trigger number 4.* `>SOUR2:SQU:MARK:END?` - *Returns the internal trigger number which the END MARKer on of the square generator on ch 2 is connected to.* `4`

Notes	*Trigger numbers are 1-14. Zero means *no-trigger*. There is a one-to-one correspondence between internal triggers and marker events. However, there is (currently) no command for querying which MARKer event a specific internal trigger is connected to, only the reverse.

SOURce:{ }:MARKer:PSTart:TNUMber

Description	Sets which internal trigger should be paired with the PSTart MARKer event for the specified generator. The PSTart MARKer event occurs whenever a new cycle of one of the waveform generators begins.
Syntax	SOURce[n]:{<name>}:MARKer:PSTart[:TNUMber][?] <numeric_value>
Arguments	<name>: DC, SQUare, SINe, TRIangle, AWG. <numeric_value>: Internal trigger number (positive integer) or zero Minimum value : 0* (default) Maximum value: 14
Query response	Returns the currently internal trigger number paired with the specified marker.
Example(s)	`>SOUR2:DC:MARK:PSTart 5` - *Pairs the PSTart marker of the DC generator on channel 2 with internal trigger number 5.* `>SOUR2:DC:MARK:PST?` - *Returns the internal trigger number which the PSTart MARKer of the DC generator on ch 2 is connected to.* `5`
Notes	*Trigger numbers are 1-14. Zero means *no-trigger*. There is a one-to-one correspondence between internal triggers and marker events. However, there is (currently) no command for querying which MARKer event a specific internal trigger is connected to, only the reverse.

SOURce:{ }:MARKer:PEND:TNUMber

Description	Sets which internal trigger should be paired with the PEND MARKer event for the specified generator. The PEND MARKer event occurs whenever a cycle of one of the waveform generators ends.
Syntax	SOURce[n]:{<name>}:MARKer:PEND[:TNUMber][?] <numeric_value>
Arguments	<name>: DC, SQUare, SINe, TRIangle, AWG. <numeric_value>: Internal trigger number (positive integer) or zero Minimum value : 0* (default) Maximum value: 14
Query response	Returns the currently internal trigger number paired with the specified marker.
Example(s)	`>SOUR2:DC:MARK:PEND 6` - *Pairs the PEND marker of the DC generator on channel 2 with internal trigger number 6.* `>SOUR2:DC:MARK:PEND?` - *Returns the internal trigger number which the PEND MARKer of the DC generator on ch 2 is connected to.* `6`
Notes	*Trigger numbers are 1-14. Zero means *no-trigger*. There is a one-to-one correspondence between internal triggers and marker events. However, there is (currently) no command for querying which MARKer event a specific internal trigger is connected to, only the reverse.

SOURce:{ }:MARKer:SSTart:TNUMber

Description StepSTart marker. Sets which internal trigger should be paired with the SSTart MARKer event for the DC generator (Fixed, LIST and SWEep).
The SSTart MARKer event occurs when a new value of the DC generator is set and is usually used in LIST or SWEep mode.

Syntax SOURce[n]:DC:MARKer:SSTart[:TNUMber][?] <numeric_value>

Arguments

<numeric_value>:	Internal trigger number (positive integer) or zero
Minimum value :	0* (default)
Maximum value:	14

Query response Returns the currently internal trigger number paired with the specified marker.

Example(s)

```
>SOUR2:DC:MARK:SST 5
```
- *Pairs the DC generator SSTart marker on channel 2 with internal trigger number 5.*

```
>SOUR2:DC:MARK:SST?
5
```
- *Returns the internal trigger number which the SSTart MARKer of the DC generator on ch 2 is connected to.*

Notes *Trigger numbers are 1-14. Zero means *no-trigger*.
There is a one-to-one correspondence between internal triggers and marker events. However, there is (currently) no command for querying which MARKer event a specific internal trigger is connected to, only the reverse.
For ANALog SWEeps the STArt marker coincide with the PSTart marker.

SOURce:{ }:MARKer:SEND:TNUMber

Description Step END marker. Sets which internal trigger should be paired with the SEND MARKer event for the DC generator (LIST and SWEep).
The SEND MARKer event occurs when dwell time is reached for a DC generator iteration in LIST or SWEep mode.

Syntax SOURce[n]:DC:MARKer:SEND[:TNUMber][?] <numeric_value>

Arguments

<numeric_value>:	Internal trigger number (positive integer) or zero
Minimum value :	0* (default)
Maximum value:	14

Query response Returns the currently internal trigger number paired with the specified marker.

Example(s)

```
>SOUR2:DC:MARK:SEND 6
```
- *Pairs the DC generator SEND marker on channel 2 with internal trigger number 6.*

```
>SOUR2:DC:MARK:SEND?
6
```
- *Returns the internal trigger number which the SEND MARKer of the DC generator on ch 2 is connected to.*

Notes *Trigger numbers are 1-14. Zero means *no-trigger*.
There is a one-to-one correspondence between internal triggers and marker events. However, there is (currently) no command for querying which MARKer event a specific internal trigger is connected to, only the reverse.
For ANALog SWEeps the SEND marker coincide with the PEND marker.

TINT:SIGNal

Description	Fires/signals one of the 14 manual (internal) triggers.
Syntax	TINT[:SIGNal] <numeric_value>
Arguments	<numeric_value>: Internal trigger number. Integer in the range 1-14.
Query response	N/A
Example(s)	`>SOUR:SIN:TRIG:SOUR INT2,(@1:24)` - *sets the trigger source to INTernal1 for sine generators on all channels.* `>TINT 2` - *Fires internal trigger #2 starting all square wave generators*
Notes	

1.3.2 Trigger output configuration

The trigger output connectors on the front and rear panels are paired up with internal triggers and configured using the following commands.

OUTPut:TRIGger:SOURce

Description	Sets or reads which trigger is routed to the specified trigger output port.
Syntax	OUTPut:TRIGger[m]:SOURce[?] {BUS, HOLD, EXTernal1, EXTernal2,…., INTernal1, INTernal2,…} [,trigger_list]
Arguments	m: Output trigger number (1..10) {..} Specifies which trigger the output is connected to: INTernal1, INTernal2, … (INT1, INT2, …): Internal triggers EXTernal1, EXTernal2,.. (EXT1, EXT2,…): Trigger In 1..5 BUS: Responding to *trg HOLD: The output is disabled (default)
Query response	Returns the name of the internal trigger or external input trigger currently connected to the specified trigger output.

Example(s)	`>OUTP:TRIG2:SOUR?` *- Queries which trigger is connected to Trigger Out 2.*
	`HOLD`
	`>OUTP:TRIG2:SOUR INT1` *- Connects Trigger Out 2 to internal trigger no. 1*
	`>OUTP:TRIG2:SOUR?` *- Queries which trigger is connected to Trigger Out 2.*
	`INT1`

Notes	To manually send out a trigger pulse on one of the *Trigger Out* ports, first connect it to an internal trigger, and then signal the internal trigger using the TINT[:SIGNAL] command. This is for example needed when aligning the sample output on synchronized instruments.

OUTPut:TRIGger:WIDTh

Description	Sets or reads the width of the trigger pulser appearing on the specified trigger output port.
Syntax	OUTPut:TRIGger[m]:WIDTh[?] <numeric_value> [,trigger_list]
Arguments	<numeric_value>: Trigger pulse width in seconds MINimum: 1e-6 MAXimum: 3600 Default: 10e-6 (10 µs) m: Output trigger number (1..5) [,trigger_list]: List of trigger outputs (for setting multiple triggers)
Query response	Returns the width of the trigger pulse appearing on the specified trigger output(s).
Example(s)	`>OUTP:TRIG2:WIDT?` *- Reads the pulse width on Trigger Out 2.* `0.001` `>OUTP:TRIG2:WIDT 20e-6` *- Set the pulse width on Trigger Out 2 to 20µs.*
Notes	When setting the pulse width, the <numeric_value> will be rounded to the nearest number of integer micro seconds

OUTPut:TRIGger:POLarity

Description Sets or reads the polarity of the trigger pulser appearing on the specified trigger output port.

Syntax OUTPut:TRIGger[m]:POLarity {NORMal|INVerted} [,trigger_list]

Arguments

NORMal:	(default) Trigger pulse goes from low to high (positive going)
INVerted:	Trigger pulse goes from high to low (negative going)
m:	Output trigger number (1..5)
[,trigger_list]:	List of trigger outputs (for setting multiple triggers)

Query response Returns the polarity of the trigger pulse appearing on the specified trigger output(s).

Example(s)

```
>OUTP:TRIG2:POL?          - Reads the pulse width on Trigger Out 2.
NORM
>OUTP:TRIG2:POL INV       - Set the polarity on Trigger Out 2 to negative going.
```

Notes Note when the trigger signal polarity is changed, the trigger output level will immediately change. This may be interpreted as a trigger event for listening devices.
Trig Out 1-3 will in NORMal mode have in idle level of 0V and generate pulses of 5V, and opposite in INVerted mode. For Trig Out 4-5 the levels are 0 V and 3.3 V, respectively.

OUTPut:TRIGger:DELay

Description Sets or reads the delay from the source triggers until the trigger pulser appears on the specified trigger output port(s).

Syntax OUTPut:TRIGger[m]:DELay[?] <numeric_value> [,trigger_list]

Arguments

<numeric_value>:	Trigger delay in seconds
MINimum:	0
MAXimum:	3600
Default:	0
m:	Output trigger number (1..5)
[,trigger_list]:	List of trigger outputs (for setting multiple triggers)

Query response Returns the delay of the trigger pulse appearing on the specified trigger output(s).

Example(s)

```
>OUTP:TRIG2:DEL?          - Reads the pulse delay (of 0.1ms) on Trigger Out 2.
0.1e-3
>OUTP:TRIG2:DEL 50e-6     - Set the pulse delay on Trigger Out 2 to 50µs.
```

Notes When setting the pulse delay, the <numeric_value> will be rounded to the nearest number of integer micro seconds

OUTPut:SYNChronize:SIGNal

Description Makes the instrument go into SYNC Wait mode and outputs a SYNC pulse on the specified Trigger Out 4 / Sync port. Used for synchronizing multiple QDAC-II Compact units

Syntax OUTPut:SYNChronize:SIGNal

Arguments None

Query response	None. This is an event.
Example(s)	`>OUTP:SYNC:SIGN` *Sends out a sync pulse on Trigger Out 4 / Sync.*
Notes	This event sends a SYNC signal out on Trigger Out #4 (Sync). The signal will have positive polarity.

1.4 Current sensor commands

SENSe:CURRent:ABORt

Description	Stops the specified current sensor trigger sequences and brings them to their idle state, thus also setting INIT:CONTinuous to OFF*.
Syntax	SENSe[n][:CURRent]:ABORt [,channel_list]
Arguments	None
Query response	None. This command is an event.
Example(s)	`>SENS2:ABOR` - *Stops current sensor measurements on channel 2*
Notes	This command will not just abort the triggering sequence, it will also stop an ongoing already triggered measurement, which can involve several COUNts and channels. For stopping all generators and ongoing current measurements, the general ABORt command can be used. The effect for the current sensors is the same. *) The behavior of setting INIT:CONT to OFF is not according to the standard SCPI protocol but is more user friendly.

SENSe:CURRent:TRIGger:SOURce

Description	Sets or reads which trigger source is used by the current sensor for the given channel(s).
Syntax	SENSe[n][:CURRent]:TRIGger:SOURce {source} [,channel_list] SENSe[n][:CURRent]:TRIGger:SOURce? [channel_list]
Arguments	{source}: IMMediate, BUS, HOLD, EXTernal1,… EXTernal5, INTernal1,… INTernal14 BUS: Use the global trigger (*trg command) HOLD: Event detection is disabled for this sequence EXTernal#: Use hardware trigger no. # INTernal#: Use internal trigger no. # IMMediate: (default) If INITiate:IMMediate is executed, a single trigger event will be processed immediately. Automatic triggering will not occur by itself.
Query response	Returns the current trigger source for current sensor(s) of the specified channel(s).
Example(s)	`>SENS2:TRIG:SOUR BUS` - *sets the current sensor trigger source on ch 2 to BUS.* `>SENS2:TRIG:SOUR INT1` - *sets the current sensor trigger source on ch 2 to internal trigger no. 1.* `>SENS2:TRIG:SOUR?` `INT1` - *returns the current sensor trigger source for ch 2.*
Notes	The default trigger source for the current sensor is IMMediate, meaning that SENS[n]:INIT will trigger a measurements.

SENSe:CURRent:INITiate:IMMediate

Description Initiates the current sensor trigger sequence for the specified channel(s), making it respond to a single trigger event, where after it returns to its IDLE state (unless CONTinuos is ON). This command clears the measurement buffer.

Syntax SENSe[n][:CURRent]:INITiate[:IMMediate] [,channel_list]

Arguments None

Query response None. This command is an event and has no readable state.

Example(s)

```
>SENS2:TRIG:SOUR IMM
>SENS2:INIT
```

- Selects IMMediate as the trigger source for current sensor on ch 2.
- Makes the current sensor on channel 2 wait for a single trigger event. As the trigger source is set to IMMediate, the current sensor value will be captured right when this command is issued.

Notes If the trigger source is IMMediate a single measurement is performed (COUNt points).

Note that in order to record and buffer multiple measurements one should either trigger multiple times in CONT=ON mode or set COUNT > 1 as the INIT:IMM command clears the measurement buffer as the first thing.

The command has no practical effect if INIT:CONT is ON.

SENSe:CURRent:INITiate:CONTinuous

Description This command determines whether the trigger sequence for current sensors are *continuously* initiated (waiting for a trigger) or not. When set to ON the trigger sequence will exit its IDLE state and go to the initiated state waiting for a trigger event. Every time a trigger cycle is completed the sensor trigger sequence will go back to the initiated state and wait for a new trigger.
When INIT:CONT is switched from OFF to ON the measurement buffer is cleared.

Syntax SENSe[n][:CURRent]:INITiate:CONTinuous {ON|OFF} [,channel_list]

Arguments

ON:	CONTinuous mode is switched on.
OFF:	CONTinuous mode is switched off.

Query response Returns the current INITiate:CONTinuous state of the current sensor(s).

Example(s)

```
>SENS2:INIT:CONT ON
```

- Makes the current sensor on channel 2 start waiting for trigger events. After a completed trigger cycle the sensor will wait for a new trigger event.

```
>SENS2:INIT:CONT?
ON
```

- Queries the CONTinuous state of the current sensor on channel 2.

Notes If TRIGger:SOURce is IMMediate the sensor will be re-triggering indefinitely until an *rst or ABORt message is received or CONT is turned OFF.
When set to OFF an ongoing trigger cycle, i.e. an already trigged measurement, is allowed to end before the sensor trigger sequence goes to its IDLE state.
Note that if TRIGger:SOURce is IMMediate and CONT = ON then there will be only one measurement per A/D converter sample interval (0.33.. ms). If APERture is greater than the sample interval, then the integration periods will be overlapping.

SENSe:CURRent:DELay

Description Specifies a delay from the trigger event until the integrated current sensor value is captured. The applied value will be rounded to nearest multiple of sample intervals.

Syntax SENSe[n][:CURRent]:DELay[?] <numeric_value> [,channel_list]

Arguments
<numeric_value>: Delay in seconds in increments of 0.33.. ms
Minimum value : 0 (default)
Maximum value: 3600

Query response Returns the measurement delay of the current sensor. As the current sensor integrated value is updated only every 0.33.. ms, there is an uncertainty of up to half a current sensor sample interval on the returned value.

Example(s)
```
>SENS2:DEL 0.0278
```
- *Sets a delay of 27.8 ms. However, this value will be rounded to 28ms (84 x 0.33.. ms)*

```
>SENS2:DEL?
0.028
```
- *Returns the start delay of the current sensor on ch 2.*

Notes The delay mechanism can for example be used to let allow for the current sensor to integrate over an entire APERture after fx a triggered change in voltage, before the value is captured. Note that the current sensor has a digital filter with a time constant of about 1.5ms. to be on the save side minimum delay of 3 ms should be used.
As the current sensor is probed every 0.33 ms the actual delay will always me an integer number of the update period.

SENSe:CURRent:COUNt

Description Specifies how many sensor readings (each of APERture length in time) should be performed upon a single trigger event.

Syntax SENSe[n][:CURRent]:COUNt[?] <numeric_value> [,channel_list]

Arguments
<numeric_value>: Number of readings (integer).
Minimum value : 1
Maximum value: 65635

Query response Returns the number of readings to be performed upon a single trigger event.

Example(s)
```
>SENS2:COUN 5
```
- *Causes the current sensor on channel 2 to make 5 readings for each trigger event..*

```
>SENS2:COUN?
```
- *Returns the number of current readings generated per trigger event on ch 2.*

```
5
```

Notes If COUNt = 1 (default), then the present value integrated for the last APERture seconds is captured immediately. If COUNt > 1, then the first reading will be immediate, but the following readings will be spaced by APERture seconds in time.
If multiple triggers have been received and COUNt > 1 then readings will be arranged in chronological order. Say that COUNt = 2 and that 3 triggers have been received then the stored data will be:
reading1, reading1, reading2, reading2, reading3, reading3

SENSe:CURRent:NCLeft

Description	Queries how many sensor readings are remaining since a trigger event.
Syntax	SENSe[n][:CURRent]:NCLeft? [,channel_list]
Arguments	None, query only.
Query response	Returns the remaining number of readings to be performed upon a single trigger event.
Example(s)	`>SENS2:NCL?` - *Returns the number of remaining current readings for the most recent trigger events on ch 2.* `4`
Notes	Returns the number of remaining readings for the specified current sensor(s) including the present reading. Right after triggering NCLeft will return COUNt, but as the current sensor always responds with the first reading immediately, it will, unless a trigger delay is specified, return COUNt-1. While averaging the last reading in a series of COUNt, NCLeft will report 1. If the sensor is in its IDLE mode or is INITiated but not triggered a NCLeft value of zero is returned.

SENSe:CURRent:RANGe

Description	Sets or reads the current range (measurement and output range) for individual channels.
Syntax	SENSe[n][:CURRent]:RANGe[?] {LOW|HIGH} [,channel_list]
Arguments	LOW: Low current range (±200nA). HIGH: High current range (±10mA) (default).
Query response	Returns the current output voltage range for the selected channel(s).
Example(s)	`>SENS2:RANG LOW` - *sets the current range for channel 2 to LOW.* `>SENS2:RANG?` - *returns the present current range of channel 2* `LOW`
Notes	When switching range, a relay in the output path is toggled and switches in another resistor. A feedback loop will adjust the voltage in case this results in a change of current. A transient may occur. So, it is advised to do this predominantly at zero output voltage. The actual electromechanical switching of range takes about 30-40 ms. Please see the detailed information in section **Error! Reference source not found.**.

SENSe:CURRent:APERture

Description Sets or reads the current sensor integration time for the specified channel(s).
Clears the measurement buffer.

Syntax SENSe[n][:CURRent]:APERture[?] <numeric_value> [,channel_list]

Arguments

<numeric_value>:	Positive floating point number, specifies the integration time in seconds. Minimum increment is 0.00033…
MINimum value:	0.00033.. (0.33.. ms)
MAXimum value:	2 (2 seconds)
DEFault:	LFR 50Hz: 0.02 (20 ms), LFR 60 Hz: 0.0166.. (16.66.. ms).

Query response Returns the integration time for the selected channel(s) in units of seconds. The value will be a multiple of the sampling interval 0.00033.. .

Example(s)

```
>SENS2:APER 0.2
```
- sets the current sensor integration time to 200ms for ch 2 (corresponding to 10x 50Hz PLC).

```
>SENS2:APER?
```
- returns the present current sensor integration time for channel 2

```
0.2
```

Notes Integration time is the period over which samples from the current sensor analog-to-digital (A/D) converter are integrated for one measurement. The current sensor A/D converter is sampled every 1/3 ms. The result is integrated continuously according to APERture or NPLCycles.
This means that when a current measurement is triggered it is the value integrated for the past APERture seconds (integers of A/D samples) which is reported. When changing APERture one should thus wait for one integration time before capturing the measurement. Please see the detailed information in section **Error! Reference source not found.**.

SENSe:CURRent:NPLCycles

Description Sets or reads the current sensor integration time in numbers of power line cycles for the specified channel(s).
Clears the measurement buffer.

Syntax SENSe[n][:CURRent]:NPLCycles[?] <numeric_value> [,channel_list]

Arguments

<numeric_value>:	Specifies the number of power line cycles.
MINimum value:	0.0166.. (corresponding to a single sample at 50 Hz)
MAXimum value:	100
Default:	1

Query response Returns the current sensor integration time in number of power cycles for the selected channel(s). This number will be rounded to nearest multiple of 0.05.

Example(s)

```
>SENS2:NPLC 1
```
- sets the current sensor integration time to 20ms for ch 2 (when LFRequency equals 50Hz).

```
>SENS2:NPLC?
```
- returns the present number of power line cycles used in current sensor integration for channel 2

```
1
```

Notes	The current sensor A/D converter is sampled every 0.33.. ms. Therefore, the minimum incremental value is 0.0166.. at 50 Hz. If NPLCycles is set to other than a multiple of 0.0166.. at 50 Hz, it will be rounded to the nearest multiple. No error will be produced. At 60 Hz the minimum increment is 0.02. When setting NPLClycles, APERture will be set to NPLCycles / LFRequency. Please see the comments for APERture detailed information in section **Error! Reference source not found.**.

SENSe:CURRent:DATA:REMove

Description	Returns and removes previously recorded data from the current-measurement buffers of the selected channels. The data points are results of previously triggered measurements.
Syntax	SENSe[n][:CURRent]:DATA:REMove? [<num_points>] [,channel_list]
Arguments	<num_points>: Number of points to read and remove (integer) MINimum: 1 MAXimum: 65636
Query response	Returns the <num_points> oldest readings from the measurement buffer of each channel Returns all measurement data in the measurement buffer in ASCII format and deletes the points from the measurement buffer. If multiple channels are specified by channel_list, then all data from the first channel in the list is returned followed by all data from the second channel in the list etc. separated by commas. If <num_points> is greater than the number of points in the queried measurement buffers, then DATA:REMove will produce an error and will not return any data to the controlling computer. If <num_points> is not provided, then DATA:REMove will return the entire content(s) of the measurement buffer(s) and clear them.
Example(s)	`>SENS4:DATA:REM? 3` - *Returns the three oldest current readings from the current sensor measurement buffer on ch 4.* `-3.49783e-10, 1.1638535e-12, -7.4938563e-9`
Notes	Note that if multiple channels are queried, they may have different number of readings in their buffers. In that case the return result can only be deciphered my knowing the DATA:POINts of the individual channels. If there are not enough data in the measurement buffer(s) then an error should be raised (for example '-230,"Data corrupt or stale"') and the query will time out on the controlling computer as no data will be returned. Also, no data are cleared from measurement buffers in this case.

SENSe:CURRent:DATA:LAST

Description	Returns the last (newest) measurement from the measurement buffer of the selected channels. No values are cleared from the measurement buffer(s) by this command.
Syntax	SENSe[n][:CURRent]:DATA:LAST? [channel_list]
Arguments	none
Query response	Returns the most recent current measurement from the addressed channel(s) without deleting the measurement point(s). Even if measurement buffers have been cleared for example by DATa:REMove, then DATa:LAST will still report the most recent measurement. If no measurement has been performed sines *RST or power up then *NaN* (not a value) will be returned (numerical 9.91e+37).
Example(s)	`>SENS4:DATA:LAST?` - *Returns the most recent current reading from the current sensor on ch 4.* `-3.49783e-10`
Notes	

SENSe:CURRent:DATA:POINts

Description	Returns the number of data points in the measurement buffer(s) of the selected channel(s).
Syntax	SENSe[n][:CURRent]:DATA:POINts? [channel_list]
Arguments	none
Query response	Returns how many current measurement data points are available in the measurement buffer(s) of the addressed channel(s). If multiple channels are addressed, then a comma separated list of values will be returned (always in ASCII format).
Example(s)	`>SENS4:DATA:POIN?` - *Returns the number of data points in the current measurement buffer of ch 4.* `3`
Notes	

READ:CURRent

Description Performs a current measurement for the selected channels, returns the result and clears the measurement buffer. It will also stop any ongoing measurements and reset the trigger system.
This command sets the trigger source to IMMediate, and therefore will always answer promptly (unless APERture is large and COUNt > 1).

Syntax READ[n][:CURRent]? [channel_list]

Arguments None

Query response Returns all measurement data in the measurement buffer in ASCII format. If multiple channels are specified by channel_list, then all data from the first channel in the list is returned followed by all data from the second channel in the channel_list etc. separated by commas.

Example(s)

```
>READ21?                - Returns the present current sensor readout on channel 21.
-3.49783e-10
>READ? (@1:4)           - Returns the present current sensor readout on channels 1-4.
-2.978352e-10,1.1638535e-12,-7.4938563e-9, 2.97835e-10
```

Notes This command reads present integrated current sensor output(s) for the selected channels and returns the data to the controller. This command does not wait for external or internal triggers.

The READ command behind the scenes sends an ABORt message, sets the TRIGger:SOURce to IMMediate and issues a followed by an INITiate:IMMediate message to the current sensor mechanism, and sends a DATA:REMove? query message. *This implies that if INIT:CONT was on before this command was executed, it will be OFF afterwards and that the trigger source has changed if it was not already IMMediate.*
This command does not restart a current measurement, it just reports the currently integrated value. So, if the integration time is larger than the time since the most recent voltage change, the reported current will not have reached a stable value.

Only if COUNt is >1 then after capturing the first value(s) then the command will wait for a time defined by APERture before capturing the next values.

Note that if COUNt is not the same for all channels in a multi-channel query, the return result can only be deciphered my knowing the COUNt of the individual channels.

Unlike for example Keysight DMMs (3446x) the QDAC-II Compact will always enforce a temporary Immediate trigger source (TRIG:SOUR IMM) instead of reporting an error if INIT:CONT = ON. This is not standard behavior, but believed to be more user friendly.

*READ is a sequential command which may block the communication or time out, if for example COUNt x APERture represents a long time. To avoid blocking current measurements use INIT followed by a trigger command (*trg) instead and read the results using FETCh? or DATA:REMove?*

FETCh:CURRent

Description Returns all available data in the measurement buffers of the selected channels. The data points are results of previously triggered measurements. The command will wait for an ongoing measurement, which can be relevant if COUNt > 1.
The measurement buffers are not cleared but can be re-read.

Syntax FETCh[n][:CURRent]? [channel_list]

Arguments None

Query response	Returns all measurement data in the measurement buffer in ASCII format. If multiple channels are specified by channel_list, then all data from the first channel in the list is returned followed by all data from the second channel in the list etc. separated by commas.

Example(s)	`>FETC11?` - *Returns contents of the current sensor measurement buffer on channel 11. In this case three values.*
	`-3.49783e-10,1.1638535e-12,-7.4938563e-9`
	`>FETC? (@1:4)` - *Returns contents of the current sensor measurement buffers on channels 1-4.*
	`-2.978352e-10,1.1638535e-12,-7.4938563e-9,-2.97835e-10`

Notes	The FETCh command does not clear the measurement buffer. It also does not trigger measurements. If there are no data in the measurement buffer(s) then an error will be produced, and the read operation will time out. Note that if COUNt is not the same for all channels in a multi-channel query, the return result can only be deciphered my knowing the COUNt of the individual channels.

READ:ADC

Description	Same as READ:CURRent except it is the raw integer from the current sensor analog to digital converter which is reported Performs a current measurement for the selected channels, returns the result and clears the measurement buffer. This command momentarily sets the trigger source to IMMediate, and therefore will always answer promptly (unless APERture is large).
Syntax	READ[n]:ADC? [channel_list]
Arguments	None
Query response	Returns all measurement data in the measurement buffer in ascii format. If multiple channels are specified by channel_list, then all data from the first channel in the list is returned followed by all data from the second channel in the channel_list etc. separated by commas.
Example(s)	`>READ21:ADC?` - *Returns the present raw current sensor readout on channel 21.* `262199`
Notes	This command triggers the transfer of the present integrated current sensor output(s) for the selected channels and returns the data to the controller. This command does not wait for external or internal triggers. Unlike READ:CURRent? this command only returns a single value even if COUNt > 1. The reported values are the raw integers from the A/D converters before any applied scaling and offsetting (using calibration parameters, see DIAG:ICAL. The command is primarily used for calibration purposes. See also notes for READ:CURRent.

1.5 System commands

System commands are related to general instrument configuration and reading the error message queue and not controlling the primary functions of the instrument (generating voltages and measuring currents).

1.5.1 Communication setup commands

SYSTem:COMMunicate:LAN:DHCP

Description Enables, disables, or queries the instrument's use of DHCP (Dynamic Host Configuration Protocol) for getting its IP address, subnet mask and default gateway.

Syntax SYSTem:COMMunicate:LAN:DHCP[?] {OFF|0|ON|1}

Arguments

ON|1:	The instrument tries to obtain an IP address from a DHCP server at power on (default).
OFF |0:	The instrument uses the static IP address, Subnet Mask, and Default Gateway. The same happens in the absence of a DHCP server.

Query response Returns the current setting, 0 (OFF) or 1 (ON).

Example(s)

```
>SYST:COMM:LAN:DHCP ON   - Enables DHCP.
>SYST:COMM:LAN:UPD       - Stores any changed settings.
>SYST:REST               - Restarts the firmware including the LAN interface.
>SYST:COMM:LAN:DHCP?     - Queries the DHCP state.
1
```

Notes To store this setting a :LAN:UPDate must be executed. For changes to take effect a SYSTem:RESTart (or power cycling) is required.

Configuration of the LAN interface is often done via the USB interface, as LAN may not be functional.

Note that if the instrument does not have a permanent entry in the DHCP server's map of clients (static DHCP), the retrieved IP address may change between power on events or other events leading to new requests.

If no DHCP server is available or DHCP is OFF, the instrument will use the stored IP address, Subnet Mask, and Default Gateway.

SYSTem:COMMunicate:LAN:HOSTname

Description Sets or queries the host name of the instrument. This is the name under which the instrument appears on the local network. The factory set host name is identical to the serial number of the unit.

Syntax SYSTem:COMMunicate:LAN:HOSTname "<hostname>"
SYSTem:COMMunicate:LAN:HOSTname?

Arguments

"<hostname>"	Quoted string containing the new name, max 16 characters.

Query response Returns the host name as a quoted string.

Example(s)

```
>SYST:COMM:LAN:HOST?     - Queries the unit's LAN host name
"Q310-12345"
```

Notes	The hostname is the host portion (left most) of the domain name, which is translated into an IP address. When DHCP is activated, the hostname is registered with the Dynamic Domain Name System (Dynamic DNS) service at power-on. To store this setting a :LAN:UPDate must be executed. For changes to take effect a SYSTem:RESTart (or power cycling) is required.

SYSTem:COMMunicate:LAN:MAC?

Description	Queries the instrument's Media Access Control (MAC) address
Syntax	SYSTem:COMMunicate:LAN:MAC?
Arguments	None
Query response	Returns the 12-character hexadecimal MAC address surrounded by quotes.
Example(s)	`>SYST:COMM:LAN:MAC?` - *Queries the instrument's MAC address.* `"70B3D5921000"`
Notes	

SYSTem:COMMunicate:LAN:IPADdress

Description	Sets or queries the instrument's IP address.
Syntax	SYSTem:COMMunicate:LAN:IPADdress "<address>" SYSTem:COMMunicate:LAN:IPADdress? [{CURRent|STATIC}]
Arguments	"<address>": IP address surrounded by quotes. [{CURRent|STATIC}]: Optional argument. If present, it determines if it is the stored address or the actual address in use which is queried. CURRent is default.
Query response	Returns either the stored (static) IP address or the actually used IP address, which may be different from the stored one if DHCP is ON.
Example(s)	`>SYST:COMM:LAN:IPAD "192.168.14.178"` *-Sets the static IP address* `>SYST:COMM:LAN:IPAD? STATIC` *-Queries the static IP address.* `"192.168.14.178"`
Notes	Assigns a static Internet Protocol (IP) address for the instrument. If DHCP is enabled (SYSTem:COMMunicate:LAN:DHCP ON), the specified static IP address is not used. As with other LAN setting commands to store this setting a :LAN:UPDate must be executed. For changes to take effect a SYSTem:RESTart (or power cycling) is required. Further it is not possible to use the short form of the keyword "STATIC", which would be "STAT" .

SYSTem:COMMunicate:LAN:GATeway

Description Sets or queries the default gateway for the instrument.

Syntax
SYSTem:COMMunicate:LAN:GATeway "<address>"
SYSTem:COMMunicate:LAN:GATeway? [{CURRent|STATic}]

Arguments

"<address>":	IP address surrounded by quotes.
[{CURRent|STATic}]:	Optional argument. If present, it determines if it is the stored gateway or the current gateway in use which is queried. CURRent is default.

Query response Returns either the stored gateway (STATic) or the actually used gateway (CURRent), which may be different from the stored one if DHCP is ON.

Example(s)

```
>SYST:COMM:LAN:GAT “192.168.1.1”      -Sets the static default gateway.
>SYST:COMM:LAN:GAT? STAT              -Queries the static default gateway.
“192.168.1.1”
```

Notes Assigns a static default gateway for the instrument. If DHCP is enabled (SYSTem:COMMunicate:LAN:DHCP ON), the specified static gateway is not used.

As with other LAN setting commands to store and activate changes to this setting a SYSTem:COMMunicate:LAN:UPDate must be executed.

Further a SYSTem:RESTart is required in order for changes to take effect.

SYSTem:COMMunicate:LAN:SMASk

Description Sets or queries the instrument’s sub-net mask.

Syntax
SYSTem:COMMunicate:LAN:SMASk "<mask>"
SYSTem:COMMunicate:LAN:SMASk? [{CURRent|STATic}]

Arguments

"<mask>":	Sub-net address surrounded by quotes.
{CURRent|STATic}:	Optional argument. If present, it determines if it is the stored mask or the current mask in use which is queried. CURRent is default.

Query response Returns either the stored (static) subnet mask or the actually used mask, which may be different from the stored one if DHCP is ON.

Example(s)

```
>SYST:COMM:LAN:SMASK “255.255.255.0”  -Sets the static sub-net mask.
>SYST:COMM:LAN:SMASK? STAT            -Queries the static sub-net mask..
“255.255.255.0”
```

Notes Assigns a static sub-net mask for the instrument. If DHCP is enabled (SYSTem:COMMunicate:LAN:DHCP ON), the specified static sub-net mask is not used.

As with other LAN commands a :LAN:UPDate is required to store the setting in non-volatile memory and a SYSTem:RESTart or power cycling is required in order for changes to take effect.

Implementation notes

SYSTem:COMMunicate:LAN:UPDate

Description	Stores the current LAN settings including any changes in non-volatile memory.
Syntax	SYSTem:COMMunicate:LAN:UPDate
Arguments	None
Query response	None. This is an event.
Example(s)	`>SYST:COMM:LAN:UPD` *- Store LAN settings to non-volatile..*
Notes	A SYSTem:RESTart is required in order for changes to take effect

1.5.2 Error system

SYSTem:ERRor:ALL

Description	Reads all error and event messages from the error queue
Syntax	SYSTem:ERRor:ALL?
Arguments	None
Query response	Returns a comma separated list of error codes and descriptions and empties the queue. If the queue is already empty '0, "No error"' is returned. The return format is (oldest first): <error code>,<error string>,<error code>,<error string> ….

Example(s)	`>SYST:ERR:ALL?` - *Reads all messages in the error queue and clears the queue.*
	`-114,"Header suffix out of range;SOUR36",-113,"Undefined header;SOYR"`
	`>SYST:ERR:ALL?`
	`0, "No error"`

Notes	

SYSTem:ERRor:NEXT

Description	Reads the oldest error message from the error/event queue
Syntax	SYSTem:ERRor[:NEXT]?
Arguments	None
Query response	Returns a comma separated pair of error codes and descriptions and empties the queue. If the queue is empty - *0, "No error"* - is returned. The return format is: <error code>,<error string>
Example(s)	`>SYST:ERR?` - *Reads and removes the oldest message from the error queue* `-114,"Header suffix out of range;SOUR36"`
Notes	If the error queue is filled and more errors occur up the most recent error in the queue will be replaced by - *350,"Queue overflow"*.

SYSTem:ERRor:COUNt

Description	Reports the number of error messages in the error/event queue
Syntax	SYSTem:ERRor:COUNt
Arguments	None
Query response	Returns an positive integer equal to the number of entries in the error/event queue.<error
Example(s)	`>SYST:ERR:COUN?` - *Queries the number of messages in the error/event queue* `2`

1.5.3 Other commands

SYSTem:BEEPer:STAT

Description	Sets or reads the current state of the built-in error buzzer.
Syntax	SYSTem:BEEPer:STAT[?]{ON|OFF}
Arguments	{ON|OFF}: ON: The buzzer sounds when an error occur (default). OFF: The buzzer is silent when errors occur.
Query response	Reports if the buzzer is enabled of disabled for error reporting.
Example(s)	`>SYST:BEEP:STAT OFF` - *Causes the buzzer not to sound when errors occur.* `>SYST:BEEP:STAT?` `OFF` - *Queries the buzzer state.*
Notes	Sometimes one gets tired of hearing a buzz very time an error occurs. In such cases the buzzer can disabled.

SYSTem:BEEPer:IMMediate

Description	Makes the built-in buzzer sound.
Syntax	SYSTem:BEEPer[:IMMEdiate]
Arguments	None
Query response	No query version. Event only.
Example(s)	`>SYST:BEEP` - *Makes the instrument produce a sound.*
Notes	

SYSTem:TEMPerature

Commented [Jv2]: revised section

Description	Reads the on board temperature sensors
Syntax	SYSTem:TEMPerature? <board_no>, <sensor_no>
Arguments	<board_no>: 1, 2, or 3. (1: outputs 1-8, 2: outputs 9-16, 3: outputs 17-24). <sensor_no>: 1, 2, or 3. (1: left, 2: middle, 3: right).
Query response	Query only command. Returns the temperature (in degree Celcius) at the specified location on the specified output printed circuit board (outputs 1-8, 9-16, 17-24)
Example(s)	`>SYST:TEMP? 1,3` - *Reads the temperature of the right most sensor on the first board.* `38.91`
Notes	Used very rarely, primarily for diagnostic purposes.

SYSTem:CLOCk:SOURce

Description	Sets or reads if the internal or an external clock is use for sample output.
Syntax	SYSTem:CLOCk:SOURce[?] {INTernal|EXTernal}
Arguments	{INTernal|EXTernal}: Specifies whether to use the internal clock (default) or an external 10MHz clock on the clock input connector (same as TRIG-IN 4).
Query response	Returns the selected currently used sample clock source.
Example(s)	`>SYST:CLOC:SOUR EXT` *- Makes the instrument use an externally supplied 10MHz clock for sample output synchronization.* `>SYST:CLOC:SOUR?` *- Queries which clock is currently used* `EXT`
Notes	This command is used for synchronizing multiple instruments. Note that the sample output rate is 1MHz, so an externally supplied 10MHz clock will be subdivided. The peak-to-peak amplitude of the signal should be at least 2.2 V and positive. Signals below 5 MHz are ignored. The QDAC-II Compact can also generate a 10MHz clock on the TRIG Out 5 port.

SYSTem:CLOCk:EXTernal:STATus

Description	Sets or reads if the internal or an external clock is use for sample output.
Syntax	SYSTem:CLOCk:External:STATus?
Arguments	None
Query response	Returns 1 (true) if the external 10 MHz clock signal is used, otherwise 0 (false).
Example(s)	`>SYST:CLOC:EXT:STAT?` *- Check if an externally supplied 10MHz clock for sample output synchronization is being used.* `1`
Notes	This command is used for reassuring that an external 10 MHz clock signal (on Trigger In #4) is being used. This is useful, because when setting SYST:CLOC:SOUR to EXT, the external signal is only used if the quality (amplitude) of the signal is acceptable.

SYSTem:CLOCk:SYNChronize:IMMediate

Description	Aligns the (1MHz) sample clock to the next external trigger event on the *Sync In* port (same as *Trigger In #3*).
Syntax	SYSTem:CLOCk:SYNChronize[:IMMediate]
Arguments	None
Query response	None. This is an event.
Example(s)	`>SYST:CLOC:SYNC` *- Makes the instrument align its 1 MHz output sample clock to the next trigger event on Sync In.*

Notes This command is used for synchronizing multiple instruments sharing the same 10 MHz clock. The :CLOC:SYNC command is basically a “phase alignment” so that the samples output at a 1MHz rate appear simultaneously for all synchronized units.

If there is no 10MHz signal available on the clock input connector, the instrument will use its internal clock. In that case, after SYNC’ing the units may drift apart causing time shifts between the samples out from the different instruments.

When the :CLOC:SYNC command is issued, the instrument will go into *Sync Wait* mode for 2 seconds and wait for a positive flank on the *Sync In* BNC port (*Trigger In #3*). After the two seconds the instrument will go into normal operation mode.

Note that while in *Sync Wait* mode all outputs are paused.

The synchronization pulse is usually received from the *Sync* out port (*Trigger Out 4*) on the master QDAC-II Compact unit.

SYSTem:CLOCk:SEND:ENABle

Description Enables/disables sending out a 10MHz clock signal on *Trigger Out 5*.

Syntax SYSTem:CLOCk:SEND[:ENABle][?] {ON|OFF}

Arguments {ON|OFF}: ON: *Trigger Out 5* is used to send out the 10MHz signal
OFF: *Trigger Out 5* is used as a normal trigger output

Query response Returns whether sending out a 10MHz clock on *Trigger Out 5* is enabled or not.

Example(s)

```
>SYST:CLOC:SEND ON      - Makes the instrument send out a 10MHz clock on the clock output port (Trigger Out 5) for synchronizing multiple instruments.
>SYST:CLOC:SEND?        - Queries if Trigger Out 5 is currently used to send out a 10MHz clock (ON) or not (OFF)
1
```

Notes This command is used for synchronizing multiple instruments to the clock of one unit.
When CLOCk:SEND is ON, *Trigger Out 5* is blocked for use as a standard trigger output port.
To align the sample output across units a trigger pulse should be sent to the *Sync In* port (*Trigger In 3*).

SYSTem:LFRequency

Description Sets or reads line frequency used by the instrument to convert NPLCycles to APERture for the current sensor and vice versa.

Syntax SYSTem:LFRequency[?] <numeric_value>

Arguments <numeric_value>: 50 or 60. Any other provided value will be converted to the nearest of the two.

Query response Returns the currently used line frequency for NPLCycles <> APERture conversion

Example(s)

```
>SYST:LFR 50             - Sets the used line frequency to 50 Hz
>SYST:LFR?               - Queries the currently used line frequency
50
```

Notes	Used very rarely, primarily at factory or before starting to use the instrument. Because the instrument does not have mains input it cannot faithfully detect the line frequency by itself.

SYSTem:RESTart

Description	Restarts the firmware including LAN interface, and resets all registers and data structures to power up conditions.
Syntax	SYSTem:RESTart
Arguments	None
Query response	N/A
Example(s)	`>SYST:REST`
Notes	Used very rarely, and typically only by the firmware updater.

1.6 Diagnostic commands

Diagnostic commands are related to service and maintenance of the instrument, and are very specific for the QDAC-II Compact.

DIAGnostic:VCALibration:{HIGH|LOW}:A

Description	Sets or queries the voltage multiplication factor used for transforming a desired voltage to the DAC digital value for the specified channel(s) for the specified voltage range.
Syntax	DIAGnostic:VCALibration[n]:{HIGH|LOW}:A[?] <numeric_value> [,channel_list]
Arguments	HIGH, LOW: Specifies the high or low voltage range A: Specifies the multiplication factor in DAC units per Volt. MINimum: 1 MAXium 1e6 Default HIGH: 52428.8 Default LOW: 262144
Query response	Returns the multiplication factor used for converting a desired voltage in units of Volts to DAC units for the specified voltage range.
Example(s)	`>DIAG:VCAL3:LOW:A 476684.5` - *Sets the voltage calibration multiplication factor in the LOW voltage range for ch 3.* `>DIAG:VCAL3:LOW:A?` - *Returns the present voltage calibration multiplication factor for the LOW voltage range of ch 3.* `4.766845e5`
Notes	Each channel has individual linear calibration constants for its voltage output in both ranges. With these command, new calibration constants can be set, or the existing ones can be read out. The raw DAC digital value is calculated as: DAC_value = VOLTage:Level:AMPLitude x A + B In order to persist new calibration parameters between power down and ups, the parameters have to be stored using the CALibration:UPDate command. Note that there is no command for restoring factory settings.

DIAGnostic:VCALibration:{HIGH|LOW}:B

Description	Sets or queries the DAC offset used when transforming a desired voltage to the DAC code for the specified channel(s) for the specified voltage range.
Syntax	DIAGnostic:VCALibration[n]:{HIGH|LOW}:B[?] <numeric_value> [,channel_list]
Arguments	HIGH, LOW: Specifies the high or low voltage range <numeric_value>: Integer. Specifies the calibration offset in DAC code units. MINimum: -524288 MAXimum: 524287 Default HIGH: 0 Default LOW: 0
Query response	Returns the DAC offset used for converting a desired voltage in units of Volts to DAC code for the specified voltage range.

Example(s)	`>DIAG:VCAL3:LOW:B 231` *- Sets the voltage calibration offset voltage in the LOW voltage range for ch 3.* `>DIAG:VCAL3:LOW:B?` *- Returns the present voltage calibration offset voltage in the LOW voltage range for ch 3.* `231` `>DIAG:VCAL3:HIGH:A 476684;B 231` *– Sets both voltage calibration factors for ch 3 in the HIGH voltage range*
Notes	Each channel has individual linear calibration constants for its voltage output in both ranges. With these command, new calibration constants can be set, or the existing ones can be read out. In order to persist new calibration parameters between power down and ups, the parameters have to be stored using the CALibration:UPDate command. Note that there is no command for restoring factory settings. The raw DAC digital code is calculated as: DAC_code = VOLTage:Level:AMPLitude x A + B

DIAGnostic:ICALibration:{HIGH|LOW}:A

Description	Sets or queries the current sensor multiplication factor used when transforming a current sensor ADC code to Amperes for the specified channel(s) in the specified current range.
Syntax	DIAGnostic:ICALibration[n]:{HIGH|LOW}:A[?] <numeric_value> [,channel_list]
Arguments	HIGH, LOW: Specifies the high or low current range <numeric_value>: Specifies the calibration multiplication factor in units of Ampere/LSB. MINimum: -1 MAXium 1 Default HIGH: 2.622605e-9 Default LOW: 2.622605e-14
Query response	Returns the multiplication factor used when converting an ADC sensor code to current in units of Ampere/LSB for the specified range.
Example(s)	`>DIAG:ICAL3:HIGH:A 2.62262e-9` *- Sets the current calibration multiplication factor in the HIGH voltage range for ch 3.* `>DIAG:ICAL3:HIGH:A?` *- Returns the present current calibration multi–plication factor in the HIGH voltage range for ch 3.* `2.62262e-9`
Notes	Each channel has individual linear calibration constants for its current sensors in both current ranges. With these command, new calibration constants can be set, or the existing ones can be read out. In order to persist new calibration parameters between power down and ups, the parameters have to be stored using the CALibration:UPDate command. Note that there is no command for restoring factory settings. The measured current in Amperes is calculated as: Current(Amps) = ADC_code x A + B

DIAGnostic:ICALibration:{HIGH|LOW}:B

Description Sets or queries the current sensor offset used when transforming a current sensor ADC code to Amperes for the specified channel(s) in the specified current range.

Syntax DIAGnostic:ICALibration[n]:{HIGH|LOW}:B[?] <numeric_value> [,channel_list]

Arguments

HIGH, LOW:	Specifies the high or low current range
<numeric_value>:	Specifies the calibration offset in units of Amperes.
MINimum:	-1
MAXimum	1
Default HIGH:	0
Default LOW:	0

Query response Returns the offset used when converting an ADC sensor digital code to current in units of Ampere for the specified current range.

Example(s)

```
>DIAG:ICAL3:HIGH:B 0.01
```
- Sets the current calibration offset in the HIGH voltage range for ch 3.

```
>DIAG:ICAL3:HIGH:B?
```
- Returns the present current calibration offset in the HIGH voltage range for ch 3.

```
0.01
```

Notes Each channel has individual linear calibration constants for its current sensors in both current ranges. With these command, new calibration constants can be set, or the existing ones can be read out.
In order to persist new calibration parameters between power down and ups, the parameters have to be stored using the CALibration:UPDate command. Note that there is no command for restoring factory settings.
The measured current in Amperes is calculated as:
Current(Amps) = ADC_code x A + B

DIAGnostic:CALibration:UPDate

Description Writes the actual DAC and current sensor calibration factors for all channels to the flash memory on the output circuit boards.

Syntax DIAGnostic:CALibration:UPDate

Arguments None

Query response None. This is an event.

Example(s)

```
>DIAG:CAL:UPD
```
- Writes calibration factors to flash memory.

Notes Writing calibration factors to the memory on the output boards is an operation which is usually carried out after calibration and calculation of new calibration factors. This always happens before the instrument leaves the factory.

DIAGnostic:CCHannel

Commented [Jv3]: revised section

Description	Routes the Monitor port on the front panel to the selected channel.
Syntax	DIAGnostic:CCHannel <channel_no>
Arguments	<channel_no>: Number (0, 1-24) of the channel which should be routed to the Monitor output. Channel no zero means that the Monitor connector will not be connected to any of the sources. Default: 0
Query response	Returns the number of the channel which is currently routed to the Monitor outpout
Example(s)	`>DIAG:CCH 14` - *Connects channel 14 to the Monitor output* `>DIAG:CCH?` - *Returns the number of the channel currently connected to the Monitor output.* `14`
Notes	The Monitor connecter on the front panel is a convenience feature used to probe the voltage output of a single channel and also when calibrating the instrument, as one can connect for example connect a voltmeter to this output and then one by one programmatically connect each channel to this connecter and perform the calibration.