QDAC-II Operation Manual

24-channel DC voltage source with:

- Integrated waveform generators
- DC current sensors on each channel
- Up to 25-bit resolution in DC mode

Please start your QDAC-II experience by first installing the latest firmware

Get the latest QDAC-II pdf manual, firmware update and other information

by following this link or the QR code: https://qm.quantum-machines.co/87kjeif6

Author *AK & JvdH*
Date *2025-06-20*
Doc version: *2.4*

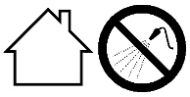
Table of contents

1 Safety and regulatory information

This device has been tested and has been supplied in a safe condition. This manual contains some information and warnings which must be followed by the user to ensure safe operation and to retain the instrument in a safe condition.

This device has been designed for indoor use in the temperature range 15°C to 30°C, **preferably 20-25°C**, 20% - 80% RH (non-condensing). Do not operate while condensation is present. When bringing the device from a cold environment to a warm environment, please allow the unit to thermalize (2-4 hours) before taking it out of its shipping box and attempting to power it on.

Use of this instrument in a manner not specified by these instructions may impair the safety protection provided. Do not operate the instrument outside its rated supply voltages or environmental range.

	Keep this device away from children and unauthorized users.
	**Indoor use only.** Keep this device away from rain, moisture, splashing and dripping liquids. Never put objects filled with liquids on top of or close to the device.
	**DO NOT** disassemble or open the cover without first getting advice from QM Technologies ApS.
	**Caution:** Device heats up during use. Make sure that the ventilation openings at the bottom and rear plates of the instrument are clear at all times. Place the device on a flat, heat resistant surface, do not place the device on carpets, fabrics etc. To avoid over heating allow some spacing to other heat generating devices (e.g. other QDAC-IIs).
	**Please** only connect the instrument to the included power supply or other approved power sources using the cable(s) delivered with the instrument.
	Keep this device away from dust and extreme temperatures.
	Protect this device from shocks and abuse. Avoid brute force when operating the device.
	Do not use the device when damage to housing or cables is noticed. Do not attempt to service the device yourself but contact QM Technologies ApS.
	Not for general waste, recover and recycle as electronic waste. Return to vendor for disposal.[1]

[1] Contact QM Technologies ApS for instructions.

2 Manual overview

Please see Appendix A for operational guidelines for the current version (14-1.70) of the firmware.

1 Safety and regulatory information
Important precautions and requirements to the environment in which this instrument is used.

2 Manual overview
This section.

3 Introduction to the QDAC-II
To get to know the instrument it is highly recommended starting by reading this. A feature overview is provided mentioning the unique features of the QDAC-II as well some of the limitations of the instrument.

4 Getting started
This section provides the information required to get the instrument hooked up so that you can try out some of the examples in the “5 Features and functionality” section.

5 Features and functionality
Describes the essential functionality and the associated SCPI commands. Examples are also provided. It does not give a full description of the command syntax. For that please see the Command Reference document on https://qm.quantum-machines.co/87kjeif6.

6 Operation

Introduces the SCPI protocol, command sequencing and status and errors.

7 The IEEE 488.2 binary block format
Contains a description of the binary block format used for transferring AWG traces and long DC LISTs.

8 Specifications
Contains a specification overview and data sheet curves.

9 Bibliography
Is a list of relevant references.

Appendix A
Important operational guidelines (firmware version 14-1.70)

Appendix B
Using laboratory power supplies

Appendix C
Open-source licenses

3 Introduction to the QDAC-II

The QDAC-II is a 24 channel 20-bit voltage source with a 1 million samples per second output rate, where each BNC output has five voltage generators working in parallel, plus a current monitor:

- One DC voltage source with sweep and list capabilities.
- One sine wave generator.
- One triangle / saw tooth generator.
- One square wave / pulse generator.
- One arbitrary wave form generator.
- Current sensing at 3 kHz speed (with ≈1.5 ms integration time).

Each channel has two voltage ranges and two current ranges and three low pass filter options. In its low bandwidth “DC” mode, the resolution is enhanced to 25 bits (see section 8) when setting fixed DC levels (in DC:FIXed mode).

The bandwidth of the waveform generators is limited by the sample rate and the low pass filters (see below).

An advanced triggering system with internal as well as external routing makes it possible to generate advanced signals and sequencies with accurate timing and external synchronization.

The number of channels can be enhanced by synchronizing multiple QDAC-II units so that voltage changes and waveforms are coordinated with sub-microsecond precision over several units.

3.1 Basic description

Front panel

On the front panel the 24 voltage outputs are located. They are standard BNC connectors with their shield connected to the common chassis ground. The output series resistance is 50 ohms.

Figure 1. Front plate of the QDAC-II. Here, all voltage outputs and three isolated trigger outputs are found.

In addition, three trigger outputs (no. 1-3) are in the right most column. These have their shield and signal galvanically isolated from the rest of the QDAC, so that ground loops are avoided when connecting these

outputs to other instruments. The voltage level is 4.8V with an output impedance of approximately 50 Ω (±20 Ω) and will provide around 3.3 V in 50 load. This means that it is safe to connect Trig. Out 1-3 to a terminated 50Ω, 3.3V input.

Rear panel

Most important on the rear panel are the Power connector and the USB and Ethernet connectors. The power is supplied by the QDevil QPSU which is a linear power supply with galvanic isolation to mains ground. Both the LAN and USB connectors feature galvanic isolation for both ground and signals, so that there is no risk of creating ground loops that way, or of introducing noise to the QDAC galvanically.

Figure 2. Rear plate of the QDAC-II. On the rear panel the power, USB, and ethernet connections are found plus additional trigger in and out connectors, including a 44 pin sub-D connector for additional future functionality.

In addition to the trigger outputs on the front panel, there are two more trigger outputs on the rear panel. Note that these two are not galvanically isolated. These outputs have a 3.3V level in a high impedance. The output impedance is not specified, but they will give up to around 2.1V in a 50 Ω load suggesting an effective output impedance of around 30 Ω. The same two outputs are used when the unit is the master in a stack of synchronized QDAC-II units.

The four trigger *inputs* (no. 1-4) are all galvanically isolated and used for triggering the voltage generators or current sensors in the QDAC-II from external devices. No. 3 and 4 are also used as inputs for clock and synchronization when the unit is a slave in a stack of synchronized units. The input impedance is 10 kΩ, the maximum voltage is 3.8 V, and the minimum voltage ÷0.5 V. The inputs react on positive going edges. The minimum guaranteed trigger level is +2.2 V and should preferably be +2.5 V.

The Digital I/O connector (44 pin high density D-sub) is a multi-purpose connector for future expansion. It includes 3.3V and 5V power lines for external electronics.

Architecture

Very simplified the QDAC-II can be considered to be made up by five functional blocks:

- Channel outputs
- Waveform generators
- Trigger inputs
- Trigger outputs
- Trigger system

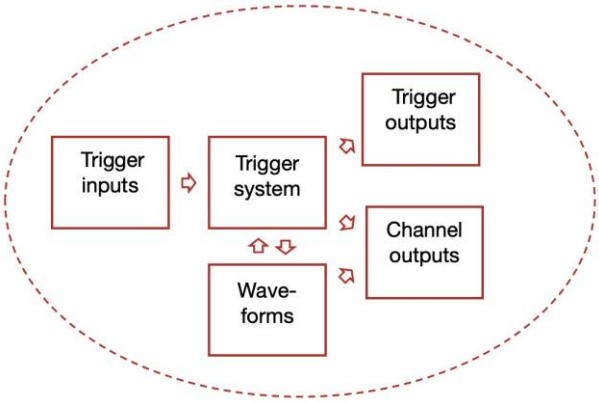


Figure 3. Functional architecture of QDAC-II

Channel outputs

Overview

Each of the 24 channels consists of a high precision DAC (digital to analog converter) working in two selectable voltage ranges (±10 V and ±2 V). The standard resolution is 20 bits.

The signal chain has two stages of lowpass filtering, one of them appearing at the final output. The two stages are hard-linked so that they switch simultaneously. The combined filter has three selectable cut-off frequencies, HIGH ≈300 kHz, medium ≈10 kHz, DC ≈10 Hz.

In the “DC” filter mode the DAC resolution is enhanced to 25 bits when setting fixed DC levels (in DC:FIXed mode).

In between the two lowpass filter stages is a current sensor with two ranges, a coarse range (HIGH) used for currents up to 10mA[1], and a fine range (LOW) for very small currents (up to 200nA), useful for detecting leaks. In the fine range there are some limitations to the bandwidth of the voltage output, which are a little bit subtle please see section 5.1.4.

Figure 4. Key function blocks of each channel on QDAC-II: D/A converter with two ranges, low-pass filtering with three cut-offs, and current sensing with two ranges.

Output impedance

The low pass (LP) filter closest to the BNC is an RC filter with a 50 Ω output resistor. The capacitance is determined by which low pass filter is invoked and the capacitance of the load, see more in section 5.1.2.

Low noise and drift

Detailed drift and noise data are provided in section 8.1. After switching on the instrument please allow a warm-up of 3 hours, and preferably 24 hours to achieve temperature stability. After that the output is stable to within about ±2 µV at low voltages and currents, if the ambient temperature is kept stable within ±0.5 ˚C.

[1] 10mA nominally as default, but can be tweaked up to 20 mA on a few channels.

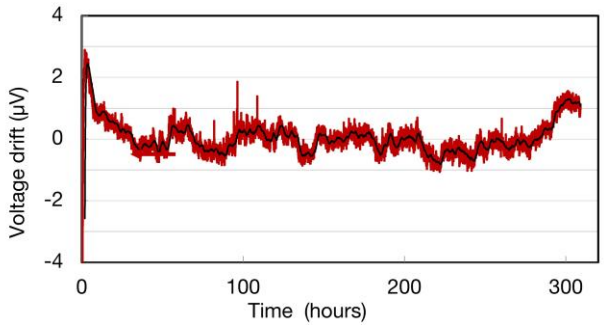


Figure 5. Change in output voltage recorded from 2-3 minutes after power on at 1V output in DC mode and 2V range.

Low transients, device protection

All DAC chips working by the resistor ladder principle (R2R) exhibits a transient – *glitch* – at certain transitions (when major carry in the binary code take place). In QDAC-II this effect has been minimized so that the maximum glitch transient amplitude is around 1.2 mV, and is only observable in the HIGH lowpass filter mode. In the MEDium and DC filter modes the glitch transients are not observable.

The lowpass filters are analogue RC filters which inherently will cause a transient to be observed at the output when switched in and out. However, a key feature of the QDAC-II is that a special circuit (patent pending) allows switching lowpass filter range with a minimal transient appearing at the output. The maximum transient to appear, when switching filters at a DC output voltage of ±9.9V, has been measured to maximum ±1.5 mV in a 350 MHz bandwidth, see details in section 8.3.

To assist the user in not accidentally applying steep voltage steps to the device under test all channels can be individually slew rate limited – at the generator level. The analogue lowpass filters will of course have a comparable function.

Current sensing resolution

The current sensor signal to noise ratio (resolution) allows detection of currents steps down to 50pA or better, and often better, (at 1PLC integration time) in LOW current mode and down to 10 µA or better in HIGH current mode. Note that this on the time scale of seconds or minutes. At longer time scales 1/f noise and drift plays a role making the long-term signal to noise ratio a bit worse

Note that the signal to noise ratio is, perhaps a bit surprising, higher in FILter=HIGH mode than in FILTer=MEDium or DC.

In addition, in both FILTer=DC and FILTer= MEDium modes the measured current will have a long settling time due to the slow relaxation of trapped charges in the output capacitors. This is possible to observe in the LOW current mode, especially following large voltage steps (see warning below).

So it is a tradeoff: with low voltage noise follows high current noise and very long response times. Please see section 8.3 in "*8 Specifications*" for details and examples.

The accuracy is limited by the non-linearities described in section 5.1.4, but can be partly accounted for.

> Note, that when FILTer=DC the time constant of the current sensor is of the order of minutes. Hence, it can not be recommended using current sensing in DC mode for other than very slow monitoring of the current at steady voltages. For getting the most precise readings FILTer=HIGH mode should be used for measuring current.

Waveform generators

Each output channel is driven by five signal generators coupled in parallel as illustrated in Figure 6.

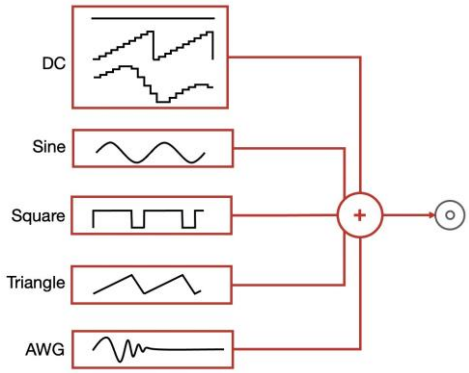


Figure 6. Each output channel has five signal generators coupled in parallel.

Each channel has its individual set of generators, so that all channels work independently from each other. In addition, their operation can be synchronized using the *trigger system*.

The DC generator can produce a DC signal, a sweep of equally spaced DC values or a list of arbitrary chosen DC values.

The Sine, Square and Triangle generators output waveforms according to their name with the possibility of adding a compensating DC offset. The Square signal generator is also useful as a pulse generator and has special settings for easily setting pulse width etc.

The arbitrary waveform generator gets its waveform data (traces) from a shared memory pool, meaning that multiple output channels can share the same waveform, but output it with individual amplitude and dc-offset for each channel.

> Naturally the actual voltage output will depend on which low-pass filter cut-off has been chosen. In addition, the chosen maximum slew-rate will also modify the output.

3.2 Communication

The QDAC-II is controlled through its USB/serial interface or via ethernet – or both.

The USB/serial port speed is 1 Mbits/sec. It is recommended to use ethernet port, especially for trace uploads. For first time use (setting up LAN options etc) and for updating the firmware it is however necessary to use the USB/serial port.

The ethernet port has giga bit speed. For transferring traces for the AWGs, ethernet should preferably be used. It is a single TCP/IP socket connection and do as such not support IEEE488.2 status byte polling and service request generation. The instrument allows several clients to control the instrument. It is not recommended for multiple users to use the instrument simultaneously, though this is possible, as global commands (such as *RST or ABORt) may disturb other users unintendedly – just think of what a *RST will do. The multiuser feature is more intended for being able to share channels in multiple setups without having to disconnect and move the instrument around.

To communicate with the instrument, make sure to send command sequences one line at a time ending with a <LF> (Line Feed) termination character, hexadecimal code ‘x0A’.

USB/serial setup

The USB connector located on the rear panel of the instrument. The communication works through a virtual serial port over the USB interface, as there is a USB-to-serial adapter inside the instrument (opto-coupled). Since the USB connection galvanically isolated from the circuits inside the instrument opto-coupled, no transients in the output voltages will occur when the USB cable is plugged in or unplugged.

When communicating with the QDAC it is necessary to specify the correct serial port to the software. See how to find the correct port number or name in the Getting started section, where the appropriate communication port settings are listed (section 4.3.2).

LAN setup

From factory the LAN interface is set up to acquire an IP address, gateway and subnet mask using DHCP (Dynamic Host Configuration Protocol). The instrument will present itself with its hostname which by default is the same as its serial number.

It is also possible to manually specify set the LAN options by setting DHCP to OFF.

After making changes to LAN settings, the :LAN:UPDate command must be sent to the instrument in order for the modified settings to take effect and to store the settings in non-volatile memory.

Command	Description
SYSTem:COMMunicate:LAN:DHCP	Determines whether automatic LAN configuration is on or off (default ON)
SYSTem:COMMunicate:LAN:IPADdress	Sets or queries the instrument’s IP address.
SYSTem:COMMunicate:LAN:HOSTname	Sets or queries the instrument’s LAN name
SYSTem:COMMunicate:LAN:GATeway	Sets or queries the IP address of the gateway (router).
SYSTem:COMMunicate:LAN:SMASk	Sets or queries the sub-net mask.
SYSTem:COMMunicate:LAN:MAC?	Queries MAC address of the instrument.
SYSTem:COMMunicate:LAN:UPDate	Stores the LAN settings and restarts the LAN interface.

Note that with the current firmware version (14-1.70), to make the instrument acquire an IP address via DHCP, the ethernet cable should be plugged in before powering up the unit, or it should be restarted after plugging in the cable (SYSTem:REStart).

SCPI Commands

The command set of the QDAC-II adheres to the SCPI standard with a few deviations.

One deviation is that the QDAC-II protocol does not support explicit units. Instead, all values are assumed to be in SI units (volts, amperes, seconds, hertz, etc.). This makes it simpler to write drivers. Some functionalities such as command synchronization is, however, not available in the first versions of the firmware.

3.3 Firmware update

Firmware updates are distributed as executables for multiple platforms. To perform the firmware update the instrument must be connected to the host computer via the USB/serial port. Please disconnect any program which may be connected to the device over USB/Serial before updating.

On MAC-OS and Linux systems the updater file must have its attribute changed to “executable” by using the “chmod +x” command. In addition, on MAC-OS it might be required to grant the updater file access in the Mac settings under *Privacy & Security -> Security -> Allow accessories to connect*.

The firmware update program will automatically identify the instrument and start updating. It lasts a couple of minutes. Please avoid interrupting the instrument and computer during the update.

> Before starting the firmware updater program, make sure that you have disconnected any USB/Serial interface on your computer, which might be connected to the QDAC-II, for example a terminal program or Python code.

Example – executing the firmware installer on MAC-OS

```
> chmod +x qdac2-fw-update-macos
> ./qdac2-fw-update-macos

Updating QDAC-II firmware...
Deploying firmware version 4-0.9.20
14:25:15: Using serial device /dev/cu.usbserial-14240
14:25:15: Rebooting device...
14:25:23: Uboot version 1
14:25:24: Transfering APU0... (ETA 14:25:28)
14:25:39: Erasing Flash...
14:25:41: Writing Flash...
14:25:42: Transfering APU1... (ETA 14:25:46)
14:25:56: Erasing Flash...
14:25:58: Writing Flash...
14:25:59: Transfering PL... (ETA 14:28:11)
14:28:11: Erasing Flash...
14:28:21: Writing Flash...
14:28:33: Number of updates: 13
14:28:36: Rebooting device
(Press Enter to close)

Update finished
>
```

> Please remember to push <enter> when finished, if prompted. Otherwise, a message will appear in your terminal/program when connecting over the USB/serial interface afterwards.

Web address for firmware updates

Updates, when available, can be downloaded from this web site: https://qm.quantum-machines.co/87kjeif6

Use of open-source libraries

The QDAC-II firmware incorporates open-source code, see Appendix C.

4 Getting started

As the first thing when receiving a new QDAC-II and after powering up, please check for new firmware updates and install the latest firmware (see

Firmware update in section 3).

To avoid ground loops or other spurious circulating currents, it is highly recommended to use the enclosed rack-mount isolators when mounting the QDAC-II in a metallic rack. Avoid making the QDAC-II cabinet touch other instruments in the rack.

> Please pay attention when rack-mounting the instrument using the included isolators, as the isolation on the 6 mm screws may move when pressed through the plastic “ears”. Please test the isolation before connecting the QDAC-II to anything else.

4.1 Powering up QDAC-II using the QDevil Power Supply

It is recommended to install the instrument in a normal laboratory environment (i.e. at non-extreme temperatures and humidity) with a stable temperature in the range of 18-25 °C. In order to achieve minimal noise (e.g. 50/60 Hz), the instrument does not have a built-in power supply. Hence, an external power supply is required. In all cases place the power supply as far away from the instrument as the cable permits and preferably not on the same vertical axis. Due to the relatively high current draw and in order to avoid significant voltage drop, the power cable delivered with the instrument is only 3 meters long. Avoid grounding the power supply to racks etc. through its cabinet. Only ground it through the mains ground.

Please first read the instructions included with the QDevil Power Supply and please make sure that the Input Voltage Range selectors have been set correctly. After inserting the mains power cable to the QDAC -II instrument and to a grounded mains outlet, make sure that the power switch on the back side of the power supply is in its off position. Then connect the QDAC to the power supply using the enclosed power cable with the 4 pole Amphenol connectors. Finally switch on the power supply. All three green LEDs should light up and the QDAC LED should start flashing red/green.

To shut down the QDAC-II, simply press the switch on the back side of the power supply to its off position. **Please wait at least 2 minutes after powering off before powering on again.**

> Please allow a warm-up time of at least 3 hours, preferably 6, for the instrument to settle any drift to within specification.

QDevil Power Supply. Rear panel enlarged on the right. Please consult the QDevil Power Supply manual for details.

4.2 Connecting the QDAC-II to your experiment

It is advised not to connect your quantum devices to the instrument before power-up. You may even want to perform your own initialization after power up. After power-up all output channels will be in the 10 Volt range and will after about 15 seconds be initialized to zero volts using the internal calibration. In the first 15 seconds there may be up to a few milli-Volts on the outputs. As the temperature in your lab may differ from the temperature during calibration, you may observe a small offset (some micro-Volts), even after the first 15 seconds.

ALWAYS disconnect your experiment from the QDAC before powering off, as the outputs may drift a little bit up and down when power is switched off – not much (NOT to ±10V), but perhaps enough to cause damage to the connected devices.

4.3 Communicating with the instrument

As mentioned in section 4.3, command strings should be terminated by a *New Line / Line Feed* character (x0A). This does not apply to binary blocks (see section 7). A *Carriage Return* (x0D) before the New Line character will be ignored and therefore does no harm.

Likewise, all responses, except for binary blocks sent by the instrument are terminated by a New Line character, according to the IEEE 488.2 standard.

4.3.1 Ethernet communication

When connecting the instrument to a local network it will try to retrieve an IP address via DHCP. The network administrator should preferably lock an IP address to the instrument in the router. Alternatively, a fixed IP address can be set using the IPADdress command (see 4.3.1 and the Command Reference document). If it is not possible to discover the instrument's IP address on the local network, it will be necessary first to connect via the USB/serial interface.

When opening a connection to the instrument it is important to use port 5025, otherwise the instrument will not respond. The instrument can have up to 8 concurrent TCP/IP connections.

> From firmware version 14-1.70, when a TCP/IP connection is made, while all 8 connections are in use, the QDAC-II Compact will remove the oldest connection to allow for the new connection to be made.

> There is a latency of up to 10 ms when sending messages to the instrument over the ethernet port. Consequently, messages may become delayed up to 10 ms (and bundled with subsequent messages) even when connecting a computer directly to the instrument's ethernet port using no router or switch.

Ethernet via a terminal

The easiest way to test the connection to the instrument is to use a simple Telnet client.

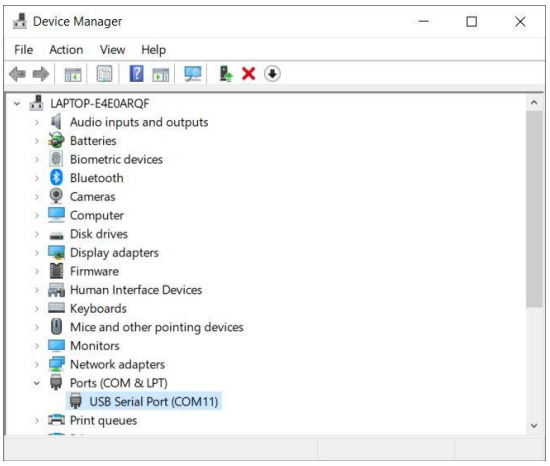
4.3.2 USB/serial communication

The instrument will appear as a serial device on the host computer. The communication settings are fixed and cannot be changed:

Serial port setting	Value
Communication rate	921600
Number of bits	8
Parity	None
Stop bits	1
Flow control	None

Windows

On Windows systems the port can be found by going the Device Manager and look under Ports (COM & LPT). In the example below it is seen that USB serial port is found at “COM11”.



The driver is usually pre-installed, or Windows will download the driver when the QDAC usb cable is plugged into the PC. If this does not happen the VCP driver can be downloaded from ftdi: https://www.ftdichip.com/Drivers/VCP.htm.

> On Windows computers the port number may change when the USB cable is unplugged and plugged in again, if the computer is rebooted, or a when changing USB port - especially if a large number of serial devices are connected to the computer.

Control using a terminal program (Windows)

Several terminal programs are available for Windows for example CoolTerm (see *MAC OS and Unix/Linux* below). Another one is HTerm. The only requirement is that the program can send a NewLine character (also known as LF – linefeed) at the end of every message. An example of a program which will not do that is Putty; so it is recommended **not** trying to use that.

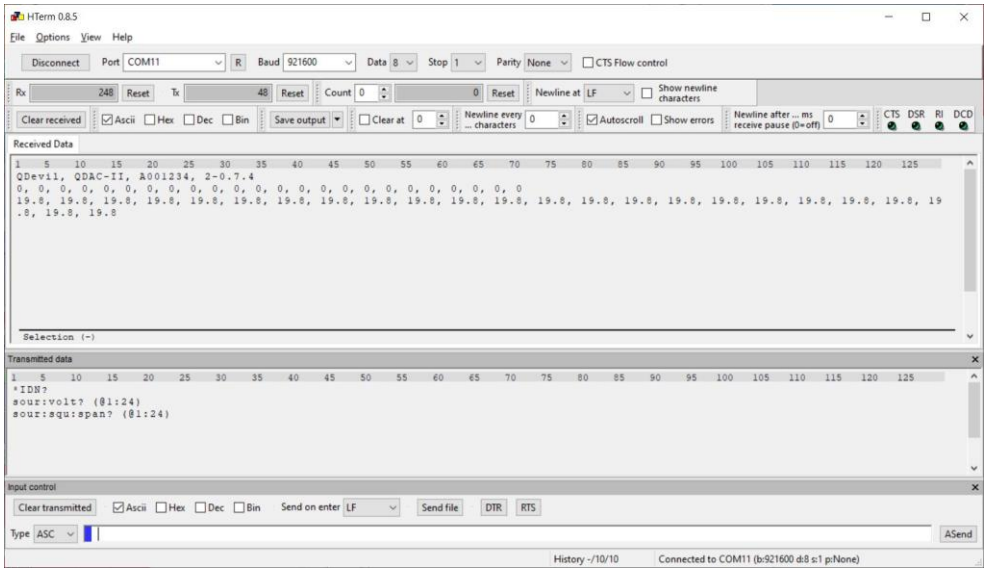
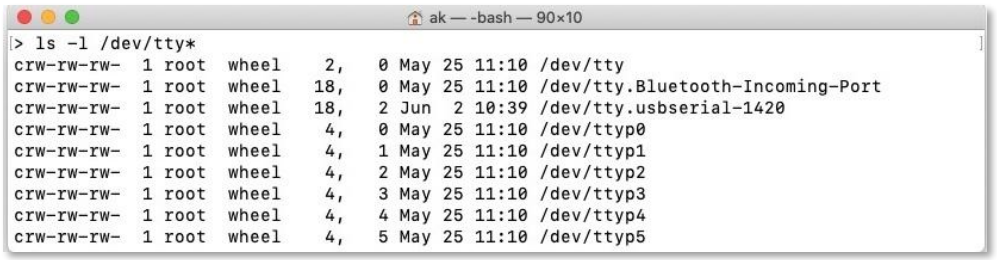


Figure 7. Example of sending a few commands using HTerm on a Windows PC..

MAC OS and Unix/Linux

Open a command prompt (terminal) and browse the /dev folder for all “tty” devices and look for USB/serial devices. Use the ‘ls’ command to do so (do not type the “>” character). The QDAC-I will appear as ‘cu.usbserial-####’, where the number #### varies from instrument to instrument:

Alternatively, the instrument may be found under /dev/tty/:



If the usb-serial device does not appear in the list, it may be necessary to install it. The “VCP” driver can be found here: https://www.ftdichip.com/Drivers/VCP.htm.

Control using a terminal program

Many users find the built-in terminal program “screen” difficult to use. So, alternatives such as the app “serial” found in App Store, or the free *Cool Term* (see macupdate.com) or *picocom* are recommended.

Picocom

To install picocom please make sure that you have the Homebrew package manager installed (see https://brew.sh). Then install picocom using the command line

```
>brew -install
```

Starting picocom with the correct configuration. Please replace '/dev/cu.usbserial-1420' with the device address found above.

```
>picocom -b 921600 -c --omap crlf /dev/cu.usbserial-1440
```

CoolTerm (MAC OS, Linux, or Windows)

CoolTerm (GUI program) can be downloaded from *macupdate.com* and is very easy to use.

Before connecting to the instrument, the Connection-Options have to be adjusted. Please update the port number with the previously found port ID:

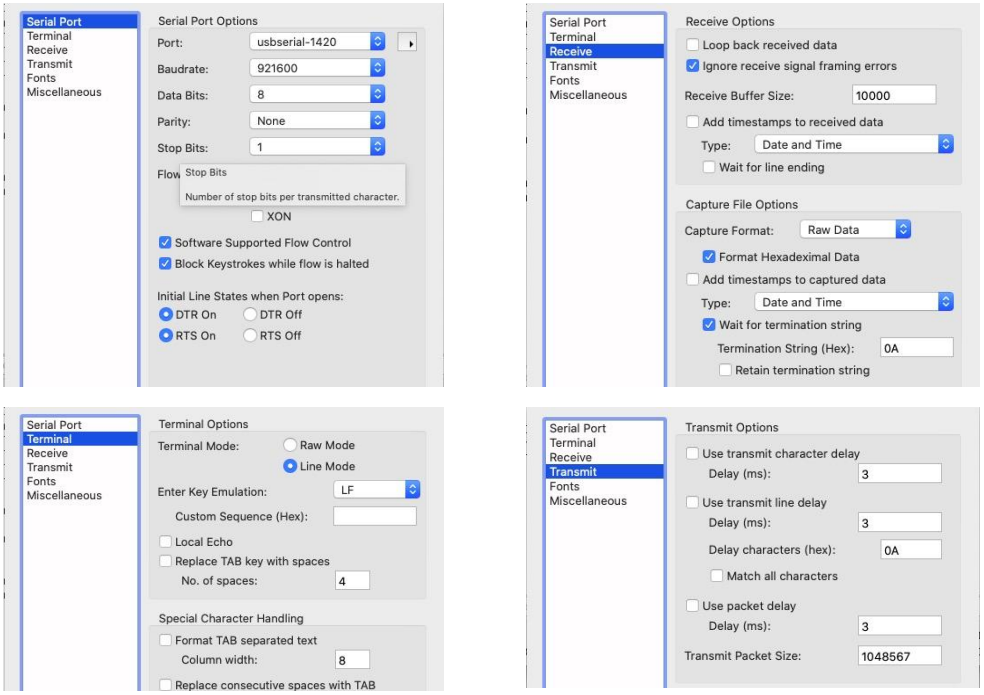


Figure 8. Connection options to be used with CoolTerm or any other serial communications program.

Figure 9. Example of sending a few commands via CoolTerm on a Mac.

4.4 Python

QDAC-II is intended to be controlled by a higher-level instrument control program. A driver is available for QCoDeS, which is Python based. Controlling the instrument directly from Python is also straight forward. One can connect to the USB/serial port using the *pyserial* package or to the TCP/IP ethernet connection using the *socket* package. However, often it is easier just to use the VISA library. To do so, the *pyvisa* package needs to be installed and imported. Further, either the ivi back-end (part of NI-VISA from National Instruments) or the pyvisa.py backend from the *pyvisa* consortium needs to be installed.

> Especially when using the USB port, QDevil recommends using the IVI (NI-VISA) backend, as data may get garbled when big chunks are read back from the QDAC-II via USB with the pyvisa.py backend, see warning in section 5.2.3.

Once the above is all set a simple Python program, here using a Jupyter Notebook, may look like this:

```
import pyvisa as visa
class QDAC_II():
    def __init__(self, visa_addr="ASRL15:INSTR", lib=''):
        rm = visa.ResourceManager(lib) # To use pyvisa-py backend, use argument lib='@py'
        self._visa = rm.open_resource(visa_addr)
        self._visa.write_termination = '\n'
        self._visa.read_termination = '\n'
        # Set baudrate and stuff for serial communication only
        if (visa_addr.find("ASRL") != -1):
            self._visa.baud_rate = 921600
            self._visa.send_end = False
    def query(self, cmd):
        return self._visa.query(cmd)
    def write(self, cmd):
        self._visa.write(cmd)
    def write_binary_values(self, cmd, values):
        self._visa.write_binary_values(cmd, values)
    def __exit__(self):
        self.close()
```

```
rm = visa.ResourceManager('')  # To use pyvisa-py backend, use argument '@py'
# List connected instruments. Instruments on LAN are often not shown.
rm.list_resources()
```

```
('ASRL15::INSTR',)
```

```
q = QDAC_II(visa_addr = "ASRL15::INSTR", lib = '')
# To use the ethernet port please replace visaAddress by the appropriate
# address  in the form of: visaAddress = "TCPIP::192.168.8.197::5025::SOCKET")
```

```
print(q.query('*IDN?'))
print(q.query("syst:err:all?"))
```

```
QDevil, QDAC-II, 48762, 2-0.8.6
0, "No error"
```

```
# Start a 20kHz sine with 1V pp on ch1
q.write("sour1:sine:freq 20000")
q.write("sour1:sine:span 1")
q.write("sour1:sine:count inf")
q.write("sour1:sine:trig:sour IMM")
q.write("sour1:sine:init")
```

```
# Stop the sine generator
q.write("sour1:sine:abort")
# Set a DC voltage of 0.2 V on ch2 - ch5
q.write("sour:volt 0.2,(@2:5)")
```

4.4.1 QCoDeS (Python)

QDevil has developed a QDAC-II (“QDAC2”) driver for QCoDeS, which is included in the Qcodes_contrib_drivers package in the official QCoDeS repository. Example Jupyter notebooks can be in the docs/examples sub folder. Documentation can be found here: https://qcodes.github.io/Qcodes_contrib_drivers/examples/QDevil/index.html

4.4.2 Labber (Python)

At the date of publishing this manual a minimalistic Labber driver, is provided. The driver supports only DC voltages, including hardware ramping, and current measurement. Improvements are planned. Please find a link on the QDevil download page (https://qm.quantum-machines.co/87kjeif6).

This driver consists of a driver definition file, QDevil QDAC-II.ini, and a Python file with just a few chunks of code: QDevil QDAC-II.py.

These two files should be placed in a subfolder called for example “QDevil_QDAC-II” in the Labber driver directory in your user folder or in the driver folder in the Labber installation folder. Please see the Labber documentation for details.

5 Features and functionality

5.1 Channel configuration

Each channel has global settings controlling its overall properties: Voltage range, current range and bandwidth:

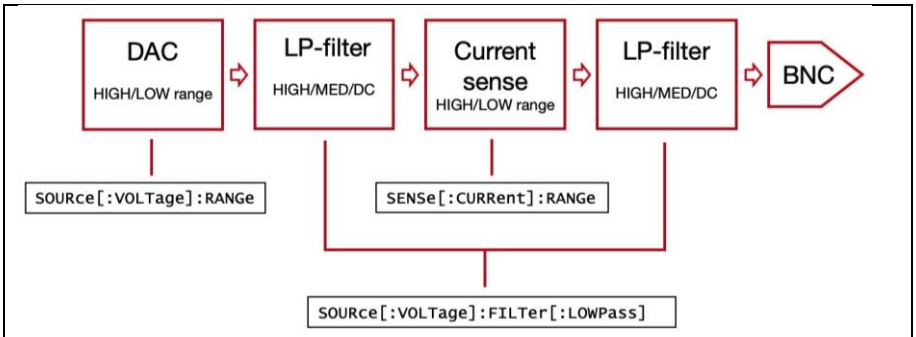


Figure 10. Each channel has its individual configuration regarding voltage range, current range, current limits, slew rate and analogue low pass filtering.

Except for the current sensor, each channel is addressed by the keyword “SOURce” with the channel number appended. Addressing channel 14 this becomes “SOURce14”. For addressing multiple channels the “channel” SCPI Syntax can be used. Here we address channels 14 to 16: “SOURce<:other keywords> (@14:16)”. For more about SCPI syntax, see (see section 6.1). In the following we omit the channel(s) in the command descriptions. The current sensor is addressed using the “SENSe” keyword.

5.1.1 Voltage range selection

The voltage range can be changed separately on each channel on the QDAC-II. On the standard product the ranges are ±10V (HIGH) and ±2V (LOW). The voltage range is set or queried by the RANGe command. Additional commands for querying the minimum and maximum values are also provided, primarily for use in drivers.

The advantage of using the LOW voltage range, if possible, is that it reduces the noise floor of the output, as the LOW range is generated as a voltage division after the digital to analogue conversion. In addition, the voltage resolution becomes 5 times higher, from nominally 19.1 µV to 3.8 µV at 20 bits DAC resolution.

Command	Description
SOURce[:VOLTage]	Node
:RANGe[?] {LOW|HIGH}	Sets the output voltage range to HIGH (±10V) or LOW (±2V), or queries the currently selected range.
:RANGE:LOW:MINimum?	Queries the minimum achievable voltage in the LOW range.
:RANGE:HIGH:MINimum?	Queries the minimum achievable voltage in the HIGH range.
:RANGE:LOW:MAXimum?	Queries the maximum achievable voltage in the LOW range.
:RANGE:HIGH:MAXimum?	Queries the maximum achievable voltage in the HIGH range.

5.1.2 Low pass filters, output impedance

Each channel has low-pass filters with three choices of cut-off frequencies: DC, MEDium, HIGH.

Filter Setting	Nominal cut-off frequency	Resolution (bits)	Output capacitance
DC	10 Hz	25	1.0 µF
MEDium	10 kHz	21	0.22 µF
HIGH	230 kHz	20	10 nF

Table 1. Nominal RC low pass filter cut-off frequencies for the three selectable bandwidths. In addition, the effective bit resolutions and the RC filter output capacitances are listed. The series resistance in the RC filters is 50 Ω.

Note that there are two stages of low pass filtering on each channel (see Figure 4). Both stages are switched when changing the filter setting. The two filter stages have slightly different cut-offs and it is the lowest of the two which is given in Table 1. In addition to the application of the analogue low pass filtering, the filter setting also controls how much resolution enhancement is provided.

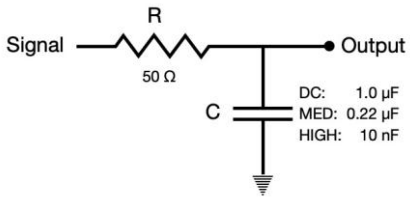


Figure 11. Output stage RC filter. The RC low pass to the BNC connector has R = 50 Ω. The capacitance is determined by the which low pass filter is invoked. Note that the circuits connected to the BNC output may add capacitance thus lowering the effective low pass filter cut-off.

Command	Description
SOURce[:VOLTage]	Node
:FILTer[:LOWpass] {DC|MEDium|HIGH}	Sub-command for setting the lowpass filter cut-off for a channel

Example

```
>SOUR:FILT MED,(@1:24)          # Set the low-pass filter to MEDium for all channels.
```

5.1.3 Slew rate

In order to avoid sudden jumps in voltage, for example caused by mis-typing a DC value or a sweep span, the QDAC-II offers *slew-rate* limitation in hardware. That is a limit for how fast the voltage can change. So, the unit is Volts per second.

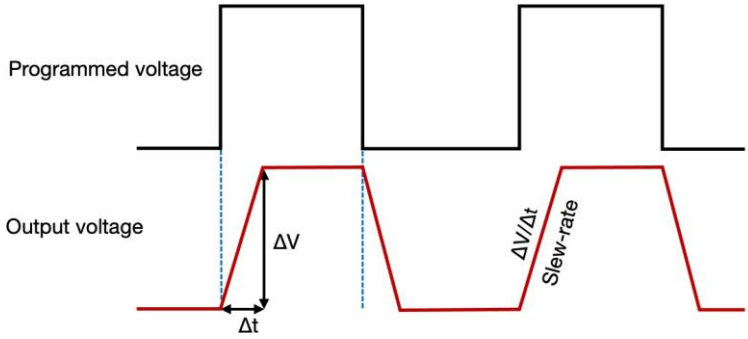


Figure 12. Slew-rate limitation sets an upper limit of how fast the voltage can change on the output of a channel.

Each channel has its own slew-rate settings. Further each voltage generator has its own individual setting. Often a low slew rate is requested on the DC generator, whereas a higher one must be used on the waveform generators, for example when overlaying a kilohertz sine. Note that the analogue low-pass filters will ultimately pose some limits for the highest obtainable slew-rate.

It must be noted that as generator signals are added the maximum slew rate will be the sum of slew-rates if high enough simultaneous voltages steps are applied on several generators.

> Note that the slew-rate for waveform generators is the same for the AC part of the signal and for the DC offset.
> *A small transient (spike, negative or positive) must be expected to appear when changing the slew rate at non-zero voltage outputs.*

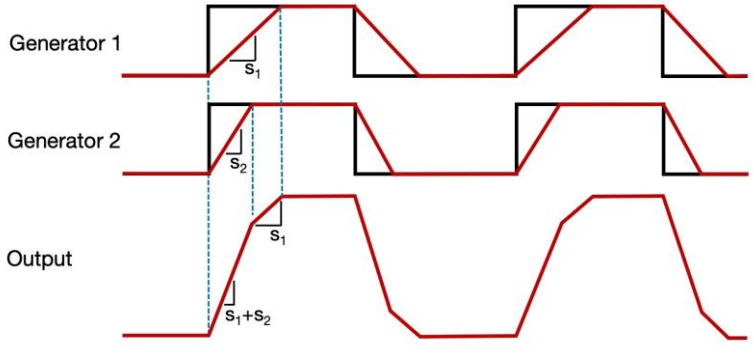


Figure 13. As the output of the five generators for each channel are added, the maximum possible slew-rate is a sum of the generator's individual slew-rate limits. This means that if fast voltage changes occur simultaneously on several generators the resulting voltage change rate may exceed the individually set limits.

Note that SLEW rate limitation should not be considered to be a substitute for using DC:SWEep (fx in ANAlog mode), as the SWEep function has more options, in particular the ability of being stopped by the ABORt command. SLEWrate limitation works as if it was an integrating filter.

Command	Description
SOURce[:DC]:VOLTage:SLEW[?]	Sets/queries the slew-rate of the DC generator.
SOURce:{}[:VOLTage]:SLEW[?]	Sets/queries the slew-rate of waveform generators.

{} = Sine, SQUare, TRIangle, or AWG. Note that setting the slew-rate for the DC generator implies that the is the same for all three modes (FIXed, SWEep, LIST) of the DC generator.

Examples

```
>SOUR2:VOLT:SLEW 115        # Sets the slew-rate of the DC generator on ch. 2 to 115 V/s
>SOUR24:SQU:SLEW 100        # Sets the slew-rate of the square wave generator on ch. 24 to 100 V/s
```

Minimum and maximum limits for SLEWrate

The lowest slewrate possible is 0.01 V/s. The highest is 2e7 V/s (or *inf*), corresponding to a full range step of 20V in the time of one sample which is 1 µs.

When FILTer = DC, *and* 25 bit resolution enhancement (RENHancement) is active, which it is by default, voltage generation is only possible using the DC generator which then has to be in FIXed mode. In this situation the lowest possible SLEWrate is 40 V/s. If the SLEWrate has been set lower than that, it will be clamped to 40 V/s until resolution enhancement is switched off, fx by exiting FILT = DC mode. A query will return the *set* value, not the *clamped* value.

5.1.4 Current range and current limitations

The QDAC-II has two current ranges. The measurement ranges are slightly lower than the actual sourcing limits. Nominally, the HIGH current range provides currents up to 10mA, and the LOW range up to 200nA. But, in a short circuit the instrument can actually deliver up to around +43 mA at +10 V and -51 mA at -10 V in the HIGH current range and around +4.3 mA at +10 V and -11 mA at -10 V in the LOW current range. *However, sourcing more than 10mA on just a few channels is not recommended*, as the internal power supply circuits are designed for all channels maximum sourcing 10 mA simultaneously. Further, to avoid stressing the current sensors one should preferentially avoid sourcing much more than ≈ ±20 µA in the LOW current range.

The current sensing resolution is considerably higher in SENSe:RANGe:LOW range compared to the HIGH range. So LOW current mode is recommended for detecting leaks and measuring small current changes precisely. However, for fast signal generation the HIGH current range mode is recommended. The signal to noise level can be reduced when increasing the integration time (see section 5.3), though it should be noted that low frequency components (like 1/f) will remain.

Command	Description
SENSe[:CURRent]:RANGe	(Defines the current sensing range, HIGH (default) or LOW.

5.2 Voltage generation

As introduced in section 2, each channel is comprised of five voltage generators added together. The DC generator is considered the main generator, as the instrument is primarily a voltage source, where the four waveform generators are thought of as adding perturbations to the DC signal. So large sweeps are intended to be performed by the DC generator using either its SWEEP or LIST functionality. Except for setting an immediate DC voltage, the signal generators are all started using the trigger system, please see section 5.4 for details.

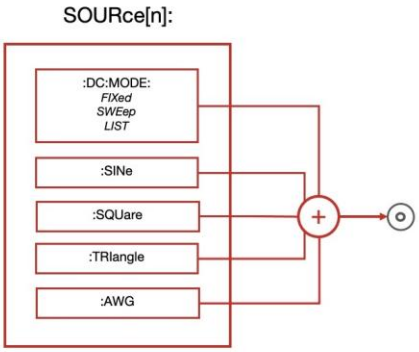


Figure 14. Each output channel (SOURce) has five added signal generators.

Note that the final sum of generator outputs may differ from the actual output at the BNC connector as both the analogue lowpass filters and the load connected to the output will affect the practical slew-rate and rise time.

5.2.1 DC signal generation

The DC generator has three modes: FIXed, SWEep, and LIST. In FIXed mode (default) the generator simply outputs a DC voltage. In SWEep mode the DC generator produces staircase ramps where all steps have identical height and run (dwell). In LIST mode the DC generator outputs a user defined list of voltages either equally spaced in time or advanced one by one by trigger events. All three modes share the same slew-ate setting.

Note that FIXed mode can be thought of as a SWEep or LIST with just a single point. Further, a SWEep can be thought of as a special case of a LIST. Looking at it that way it is not so surprising that these modes share some commands, for example for reading the output of the DC generator.

Command	Description
SOURce[:DC]	(top node)
[:VOLTage]:MODE	Sets or queries the functional mode of the DC generator (FIXed, SWEep or LIST). Default is FIXed mode.
:VOLTage:SLEW	Sets or queries the maximum slew-rate for the DC generator.

:VOLTage[:LEVel[:IMMediate[AMPLitude]]?	Queries the actual generated DC value at the time of the query. This may be slightly different from the set value due to the finite resolution of the D/A converters and if a finite slew-rate is set.
:VOLTage [:LEVel[:IMMediate[AMPLitude]]:LAST?	Queries the last set DC value, see example under FIXed mode.

FIXed mode

A DC output can be set immediately or at the next trigger event from the DC generator's trigger source. This way it is possible to simultaneously alter the DC output of multiple channels or synchronize the DC voltage change with external equipment such as for example RF generators, detectors, or sensors.

Command	Description
SOURce[:DC]:VOLTage	(top node)
[:LEVel[:IMMediate[AMPLitude]]	Sets an IMMediate DC voltage in FIXed mode. For query, see above.
[:LEVel]:TRIGger[:AMPLitude][?]	Sets or queries the DC value which will be set at the next trigger event in FIXed mode.
SOURce[:DC]:DAC:[:LEVel[:IMMediate[AMPLitude]][?]	Rarely used command for setting the DC output to a specific DAC code in FIXed mode. This command is primarily used in the calibration procedure. Likewise, the DAC code can be queried.

Example – setting a voltage immediately or triggered

```
>SOUR:VOLT 0,(@1:24)          # Sets the voltage of all 24 channels to zero.
>SOUR:VOLT:TRIG 1,(@1:8)      # Prepares ch. 1-8 for stepping to 1V output at next trigger.
>SOUR:DC:INIT (@1:8)          # Triggers ch. 1-8 to change their output voltage to the triggered value.
```

SWEep mode

In SWEep mode the instrument generates staircase sweeps between two voltages defined by STARt and a STOP. Besides the start and end voltages the staircase is defined by the time dwelled at each step, and the number of steps. The number of steps, equal to the number of voltage levels is given by POINTs.

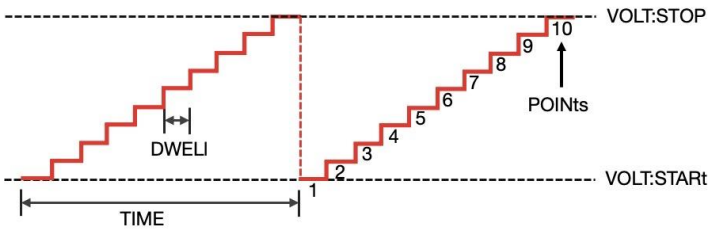


Figure 15. Main parameters for a stepped (staircase) DC generator sweep.

The time it takes for a single stepped sweep to complete is POINTs x DWELl. The height of each step is given by (STOP - STARt) / (POINts – 1).

The sweep is repeated COUNt times. During the sweeping it is possible query the number of remaining sweeps using the NCLeft (number of counts left) command.

In addition to generating a staircase, the SWEep option can also generate a continuous ramp, when GENeration is set to ANALog. Due to the finite sample speed and the finite bit resolution of the DA converters even an ANALog sweep will have finite, however small, steps. The duration (TIMe) for an ANALog SWEep is, as for STEPped SWEeps, given by DWELl x POINts, even though these two quantities separately are ambiguous for ANALog SWEep. The actual dwell (step length) is always equal to the instrument's sample update time of 1 µs.

> Be sure to set the DC slew-rate high enough for a stable level to be reached at every step, and in case of repeated sweeps for the STARrt level to be reached comfortably when returning from STOP and starting the next sweep (> |STOP-STARt| / DWELl). Also set the DWELl time several times higher than the time constant of the used low-pass filter.

Command	Description
SOURce[:DC]:SWEep	(top node)
:TIME?	Read the duration of a single sweep.
:DWELl	The dwell time at each voltage level in the sweep.
:DWELl:AUTO	Enables auto calculation of the dwell time.
:POINts	Number of voltage levels in the sweep, including the first and the last.
:COUNt	Number of repeats of the sweep.
:NCLeft	Number of repetitions left including the ongoing sweep.
[:VOLTage]:STARt	The voltage level of the first step.
[:VOLTage]:STOP	The voltage level of the last step.
:GENeration	As default SWEeps are staircases with a STEPped structure. However, they can also be continuous when GENeration is set to ANALog.

Example – setting up and starting a stepped sweep

```
>SOUR8:SWE:VOLT:STAR -0.1      # Sets the start voltage to -0.1 V for a sweep on channel 8.
>SOUR8:SWE:VOLT:STOP 0.2       # Sets the end voltage to 0.3 V.
>SOUR8:SWE:DWEL 0.001          # Sets the dwell time to 1ms.
>SOUR8:SWE:COUN 1              # Just make a single sweep.
>SOUR8:DC:SWE:GEN STEP         # Set the sweep generator to stepped mode. Not really required as this is the default.
>SOUR8:MODE SWE                # Set the DC generator to SWEep mode
>SOUR8:SWE:INIT                # Start the sweep (assuming that trigger source is set to IMMediate).
```

LIST mode

In LIST mode a channel will, when triggered, output a user defined sequence of absolute voltages. The sequence can be repeated, however not indefinitely. SWEep is a special case of LIST, where all voltage steps (voltage difference between consecutive points) are identical. LIST is used when the autogenerated SWEep sequence is not sufficient for the application. One example could be when non-linear, e.g. logarithmic, sequences are required. Another case could be when it is necessary to overlay a slow varying background to series of staircases, e.g. when compensating for cross coupling in a 2D gate-electrode scan of a quantum dot device (virtual gates).

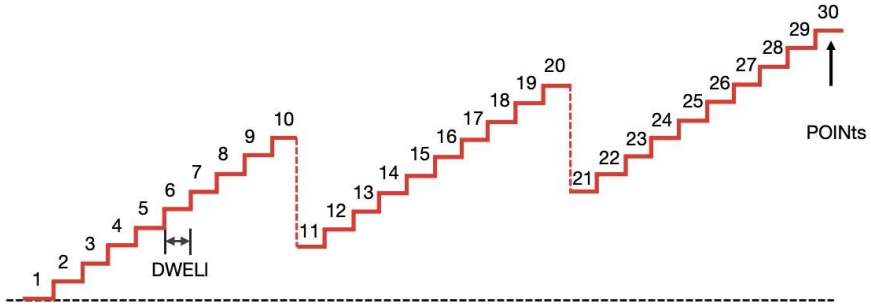


Figure 16. Example of 3 staircases in one LIST sequence each with an additional offset.

In firmware version 14-1.70 the maximum length of lists has been increased to 2.097.152 points.

There also some similarities between LIST and the arbitrary waveform generator (AWG), but also some differences of which some a given in the table below. The AWG will typically be used for overlaying semi-fast curve shapes on to a DC signal, whereas LIST is used to produce a sequenced DC signal.

LIST	AWG
Variable playback speed	Fixed fast playback speed
One LIST per channel	Waveforms can be shared between channels
Absolute voltages	Scalable/offset-able numbers
Markers at each point, repetiion, and start/end	Markers at start/end

As LIST sequences can be long, it is possible to choose between two formats when transferring the sequence of voltages using the SOURce:DC:LIST:VOLTage and :VOLTage:APPend commands: Ascii or binary block. The binary block format is described in section 7, and examples are provided in the command descriptions in the Command Reference document.

Command	Description
SOURce[:DC]:LIST	(top node)
:COUNt	Number of iterations of the LIST, primarily relevant when TMODe = AUTO.
:DIRection	Determines the direction of traversing through the LIST sequence.

:DWELl	The dwell time at each voltage level in the LIST sequence (when TMODe = AUTO).
:NCLeft	Number of repetitions left including the ongoing iteration.
:VOLTage	Defines the voltages in the sequence (erases a previous sequence), in ascii mode only 1023 values are allowed. Use APPend for longer lists.
:VOLTage:APPend	Appends more points to the sequence (max. 1024 values at a time).
:VOLTage:POINts	Number of voltage levels in the LIST, including the first and the last.
:TMODe	Defines whether the complete LIST should run when triggered or if a trigger should just advance to the next point (first trigger “advances” to the first point).

Example – setting up and starting a list

```
>SOUR8:LIST:VOLT 0,0.1,0.2,0.3,0.4,0.5,0.6   # Defines a sequence of voltages for the LIST on channel 8.

>SOUR8:LIST:VOLT:APP 0.7,0.8,0.9,1           # Adds additional points.

>SOUR8:LIST:DWEL 0.01                        # Sets the dwell time to 10ms

>SOUR8:LIST:COUN 5                           # Iterate the LIST sequence 5 times.

>SOUR8:LIST:TMOD AUTO                        # Sets the trigger mode such as to run the entire sequence on a single
                                             trigger.
>SOUR8:VOLT:MODE LIST                        # Sets the DC generator on channel 8 to LIST mode.

>SOUR8:DC:TRIG:SOUR IMM                      # Sets the trigger source to IMMediate.

>SOUR8:DC:INIT                               # Initiates the DC generator and thereby starts it.
```

Example – triggering the LIST point by point

```
>SOUR8:LIST:TMOD STEP                        # Sets the trigger mode to STEPped so that each point in the sequence
                                             is advanced to by a separate trigger event.
For I = 1 to 5                               # Pseudo code. Iterate through the 11 points in the LIST
    for j = 1 to 11                          # sequence 5 times. Here INIT is used to make the IMM trigger
        >SOUR8:DC:INIT                       # trig.
        # wait for settling
        # measure something
    end j
end i

>SOUR8:DC:TRIG:SOUR BUS                      # The universal and preferred way is to use a “real” trigger event
>SOUR8:DC:INIT:CONT ON                       # (which IMM is not) and fire that multiple times while INIT:CONT is
for i = 1 to 5                               # ON. The global BUS trigger could be one example. But internal
    for j = 1 to 11                          # triggers as well as external triggers are probably closer to
        >*trg                                # real life situations
        # wait for settling
        # measure something
    end j
end i
>SOUR8:DC:INIT:CONT OFF
```

5.2.2 Fixed waveform generators

Each channel has 3 fixed waveforms generators: SINe, SQUare, and TRIangle.

The waveform generators have a lot of similar characteristics and commands which are described in the following paragraphs. In addition, all their commands are listed in separate paragraphs.

> Note that waveform generators including AWG are only available in MEDium and HIGH filter modes, and *not* in DC filter mode.

Common properties of waveform generators

For convenience it is possible to set a generator's frequency (FREQuency) as well as its period (PERiod). The most recently set property which will overwrite the other.

Waveform generators are by default applied symmetrically around the channel's DC value, as they are intended to serve as (small) perturbations to the DC generator signal. Their peak-to-peak amplitude is defined by the SPAN property. If a DC offset is required, this can be applied using the OFFSet property.

POLarity specifies if the first part of a generator's period (or cycle) is positive (POS) or negative (NEG). If less than a 180 degrees phase shift is required this must be set using the DELay property for the assigned trigger source.

The maximum slew rate (SLEW) can be set on an individual generator basis (default is INF). Note that it may be overruled by the channel's low-pass filter. The purpose of having finite slew rates for waveform generators is to restrict the edge-slopes for especially the square wave generator, but also to smooth the step applied when a generator starts or stops and its OFFSet is non zero.

By default a waveform generator continues indefinitely when started, until it is stopped by an ABORt command or when a property is changed. However, using the COUNt property it is also possible to set a specific number of cycles to run after starting a generator. Another property, NCLeft, (number of cycles left) can be queried to find out how many cycles are remaining once the generator has been started.

Generators are started using the trigger system, please see section 5.4.1. Interesting waveforms can be generated overlaying or coupling waveforms (and DC steps) using the MARKer and TRIGger systems as the signals for all generators are added (see for example Figure 24). If the added signals extend the current voltage range, they will be clipped. No warning or error is produced when clipping occurs.

> In order to offer as much flexibility as possible the allowable parameter ranges are not adjusted according to chosen slew-rates, low-pass filter, and the hardware imposed slew-rate limitation in for example the LOW current mode. A mismatch in parameters or exceeding the instrument capability may result in unexpected curve shapes.
> **Warning: Asymmetric waveforms with frequencies above the filter cut-offs will impose a DC offset because the filter will average the AC signal! Note that square waves of only a few µs period will be highly asymmetric due to the asymmetric slew-rate of the output circuitry.**

Waveform distortion and magic periods and duty-cycles

Discrete frequencies and periods

Waveforms are generated by outputting a sample (a new voltage) every 1 µs. This implies that only PERiods of an integer number of microseconds are possible. In other words, not all frequencies are allowed. The highest frequency which can be generated is 500 kHz (2 samples), then 333.33.. kHz (3 samples), then 250 kHz (4 samples) etc.

> In the current version of the firmware (14-1.70) no error or warning is produced when trying to set the FREQuency or PERiod to an intermediate value. Instead, the period, or corresponding period when the frequency has been set, is rounded to the nearest integer number of microseconds. When queried, it will be the value set by the user which is returned and not the rounded value.

PERiod	FREQuency
2 µs	500.00 kHz
3 µs	333.33.. kHz
4 µs	250.00 kHz
..	
33 µs	30.30… kHz
..	
99 µs	10.10.. kHz
100 µs	10.00 kHz

Examples of supported periods and corresponding frequencies.
Note: For TRIangle waveforms the minimum period is 4 µs.

Magic periods and duty-cycles for accurate waveform generation

The fact that a period must be an integer number of samples and due to the way that the fixed waveforms are implemented, a given waveform cannot be faithfully reproduced for all choices of periods and duty-cycles (square wave and triangle). Even for the sine waveform an integer number of samples are required per period, because otherwise it would not be possible to synchronize with the triggering system, which has the same 1 µs update rate as the channel outputs.

The square wave can be defined by as little as two samples (each lasting 1µs). The output at that high frequency will look like a sinusoid with some DC offset, heavily damped by the low-pass filters. The next period giving a symmetric waveform is 4, and so on +2. For duty-cycles different from 50% it becomes a bit more subtle. A square wave with a period of 3 µs will de facto have a duty cycle of 66.66.. % even if DCYCle is set to 50 % (and hence is reported as 50%). For long periods the deviation will be small.

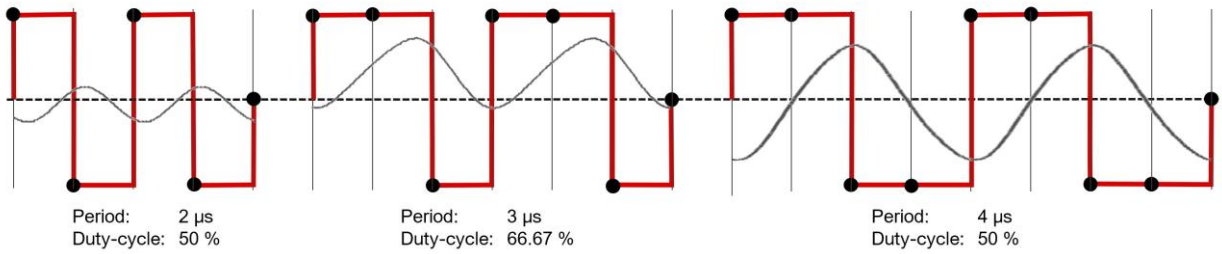


Figure 17. The three minimum periods for the SQUare wave generator. Note that for the 3 µs period, the duty-cycle will be 66.67 % even if it is set to 50 %! The red curves are theoretical, the black dots representing every time a sample is output (every 1 µs). The actual output curves (HIGH filter mode) will be damped, and phase shifted by the low-pass filters, capacitive loads, and distorted/offset by the asymmetric finite driving force of the output amplifiers.
Actual recorded curves are shown in grey color. Here it is clearly seen that asymmetric curves (e.g. 3 µs period) will result in a net DC offset when the frequency is below or around the cut-off frequency of the low-pass filter.

The triangle waveform requires at least 4 samples (PERiod = 4 µs) for defining the two extrema and two zero points. The zero points (having start and ending points halfway between the extrema) are required for implementation purposes (and triggering). The next magic number of points is 8. So “perfect” symmetric triangles with 50 % duty-cycle are only available with periods which are multiples of 4 µs.

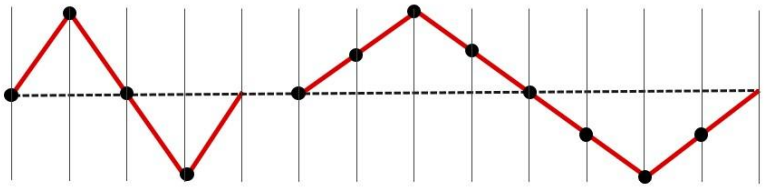


Figure 18. Number of points for generating the two shortest but faithfully reproduced triangle waveforms: Left: 4 points. Right: 4+4 points.

If the period is not a multiple of 4 µs, the triangle will be approximately symmetric, but will end prematurely, meaning that there will be a small kink between repetition of periods, see Figure 19. For duty-cycles different from 50% it becomes a bit more subtle – each of the two waveform parts (up, down) should in addition be an integer number of microseconds for perfect waveform output.

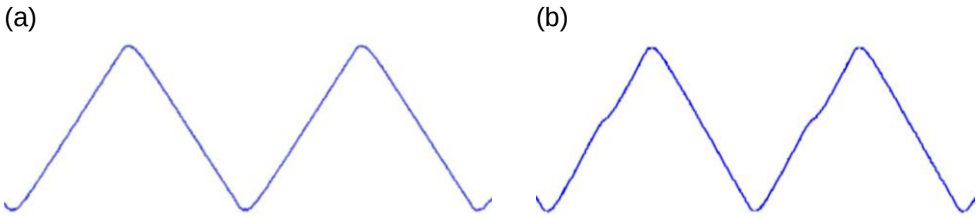


Figure 19. (a) Example of a “perfect” triangle waveform with a multiple of 4 points (28), and (b) one with a non “magic” number of points (26). A small kink is observed between repetitions of periods.

Note that regardless of these effect, **fast varying signals will be distorted due to the low-pass filter stages** and the finite (asymmetric) driving force of the output amplifiers, see section 8.2. Therefore, it may be difficult to observe the effect of magic/non-magic number of points for high frequencies / short periods. In the opposite regime, for *very* long periods (low frequencies), the relative distortion may be so small that it can hardly be observed.

SINe generator

The sine wave generator is controlled in full by the common waveform generator commands.

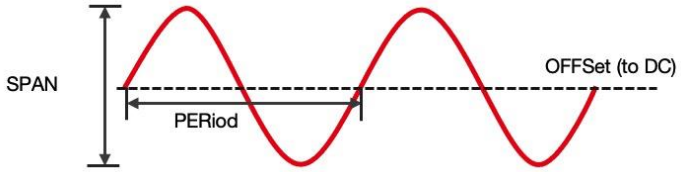


Figure 20. Basic properties of sine waves.

Command	Description
SOURce:SINE	(top node)
:PERiod	Sets or gets the waveform period in units of seconds (overwrites FREQuency)
:FREQuency	Sets or gets the frequency (unit of Hz) of the waveform (overwrites PERiod)
:COUNt	Determines the number waveform periods to be produced. Can be INFinite.
:NCLeft?	Number of repetitions left including the ongoing iteration.
:POLarity	Determines if the first half period is positive (NORMal) (default) or negative (INVerted).
[:VOLTage]:SPAN	Sets or gets the peak-to-peak voltage of the waveform.
[:VOLTage]:OFFSet	A (small) offset relative to DC can be applied.
[:VOLTage]:SLEW	Sets or queries the maximum slew-rate for this waveform generator.

Sine wave generator limitations.

Each period of the sine is calculated with up to 65536 points and hence a very faithful reproduction of a sine is achieved, except at very long periods: As samples are output every 1µs this means that sines with periods longer than ≈ 65.5 ms, corresponding to frequencies smaller than about 15.38 Hz, will be interpolated. To produce a well-shaped sine waveform a certain minimum number of points (each 1 µs) are needed. Together with the low pass filter this sets a limit for the upper usable frequency, e.g. ~ 30.3 kHz. Note that a PERiod of 2 µs will result in a flat curve, as the two calculated amplitudes will be the sine's two zero-crossing points. Also note that "magic periods" apply, see above, in order to have extrema and zero crossings represented. However, distortions are less evident than on the triangle.

Example – program and starting a sine generator immediately

```
>sour8:filt high                    # First set the filter, voltage range, and current sensor range for ch8 to HIGH
>sour8:rang high                    # in order to get the highest bandwidth. Note that, this is not necessary after
>sens8:rang high                    # power up as these are default settings.

>sour8:sine:freq 5000               # Then set the frequency to 5kHz, peak to peak amplitude to 4 volt.
>sour8:sine:span 4                  # Set number of repetitions to infinite (default).
>sour8:sine:count inf

>sour8:sine:trig:sour imm           # Set the trigger source to IMMediate (default).
>sour8:sine:init                    # Start the sine generator on ch8.
```

SQUare wave generator

Both the square wave generator and the triangle wave generator share the concept of *duty-cycle*. For the square wave, duty cycle describes the percentage of each period taken up by the positive (first part) of the waveform, see Figure 21. In case POL = INVerted, then it is the percentage of the negative part. With a 1 µs sample output rate it is not possible to generate all combinations of periods and duty cycles. Please see the block about *Waveform distortion and magic periods and duty-cycles* above.

> To make sure that the resulting curve looks as expected, always set the PERiod (or FREQuency) is set so that each part (low and high) contains an integer number of samples (of 1µs).

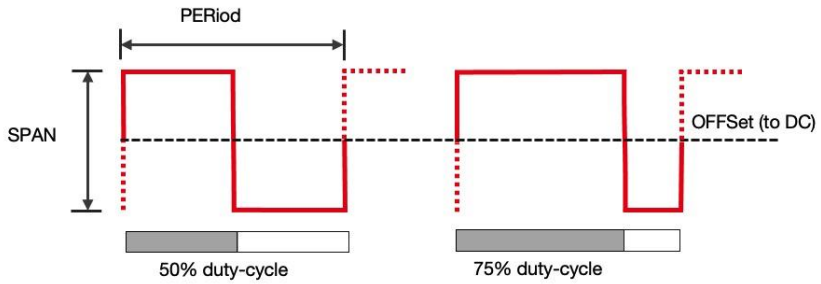


Figure 21. Basic properties of the square waveform. Note that the waveform always starts and ends with a voltage equal to OFFSet, defining a period. If the SLEWrate is low or MEDium filter is enganged, the flanks will be somwhat skewed.

Command	Description
SOURce:SQUare	(top node)
:PERiod	Sets or gets the waveform period in units of seconds (overwrites FREQuency)
:FREQuency	Sets or gets the frequency (unit of Hz) of the waveform (overwrites PERiod)
:DCYCle	Duty cycle, the duration of the first half of the period divided by the entire period (default 50%).
:COUNt	Determines the number waveform periods to be produced (repetitions). Can be INFinite.
:NCLeft?	Number of repetitions left including the ongoing iteration.
:POLarity	Determines if the first half period is positive (NORMal) (default) or negative (INVerted).
[:VOLTage]:SPAN	Sets or gets the peak-to-peak voltage of the waveform.
[:VOLTage]:OFFSet	A (small) offset relative to DC can be applied.
[:VOLTage]:SLEW	Sets or queries the maximum slew-rate for this waveform generator.

TRIangle generator

For the triangle wave *duty cycle* describes the percentage of each period taken up by the positive going (first part) of the waveform. In case POL = INV, then it is the percentage of the negative going part. Please see the block about

Waveform distortion and magic periods and duty-cycles in the beginning of this section.

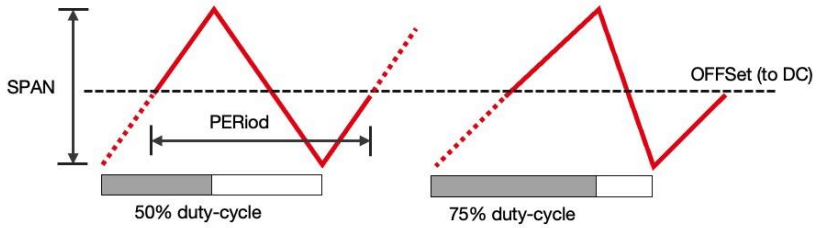


Figure 22. Basic properties of the triangle waveform. Note that the waveform always starts and ends with a voltage equal to OFFSet, defining a period. The duty cycle is best visualized by considering the period from one extremum to the next same type of extremum, e.g. from minimum to minimum.

Command	Description
SOURce:TRIangle	(top node)
:PERiod	Sets or gets the waveform period in units of seconds (overwrites FREQuency)
:FREQuency	Sets or gets the frequency (unit of Hz) of the waveform (overwrites PERiod)
:DCYCle	Duty cycle, the duration of the first half of the period divided by the entire period (default 50%).
:COUNt	Determines the number waveform periods to be produced. Can be INFinite.
:NCLeft?	Number of repetitions left including the ongoing iteration.
:POLarity	Determines if the first half period is positive (NORMal) (default) or negative (INVerted).
[:VOLTage]:SPAN	Sets or gets the peak-to-peak voltage of the waveform.
[:VOLTage]:OFFSet	A (small) offset relative to DC can be applied.
[:VOLTage]:SLEW	Sets or queries the maximum slew-rate for this waveform generator.

Example – setting up and starting a square wave with a triangle on top

```
>sour8:squ:freq 1000;span 2;trig:sour int1      # Defines square wave with e frequency of 1kHz and peak to peak amplitude of 2V on channel 8. The trigger source is set to INTernal1.

>sour8:tri:freq 1e4;span0.2;trig:sour int1      # Defines triangle wave with e frequency of 10kHz and peak to peak amplitude of 2V on channel 8. The trigger source is set to INTernal1.

>sour8:squ:init                                 # Both generators are initiated and started simultaneously by activating internal trigger 1.
>sour8:tri:init
>tint 1
```

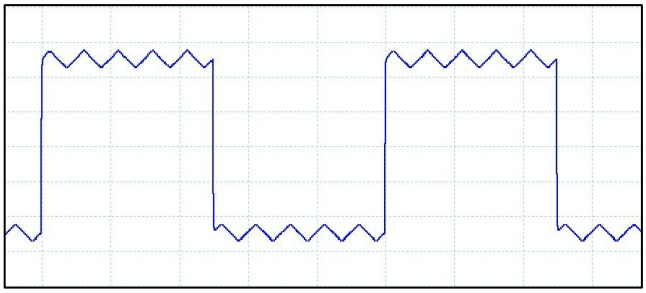


Figure 23. Scope trace of the added square and triangle wave signals. Note that the triangle starts and ends at its center value. If one wants to align the lowest point in the triangle with the positive edge of the square a TRIangle:DELay of a quarter of a triangle period can be specified.

Example – Smearing the plateaus in a stepped sweep

```
>sour4:dc:volt:mode sweep                       # Define and start a stepped sweep (with just a few steps and let it run indefinately.
>sour4:swe:start -0.5;stop 0.5;dwel 1e-3;poin 6;count inf
>sour4:dc:init

>sour4:dc:marker:sstart:tnumber 7               # Make internal trigger no. 7 fire at the beginning of every step.

>sour4:tri:per 50e-6;span 0.1;count 10          #Define a fast triangle with a just a few periods.

>sour4:tri:delay 0.2e-3;trig:sour int7          # Trigger the waveform when internal trigger 7 is fired after a delay of 0.2 ms

>sour4:tri:init:cont on                         # Initialise the triangle generator in continuous mode so that it can be retriggered indefinitely.
```

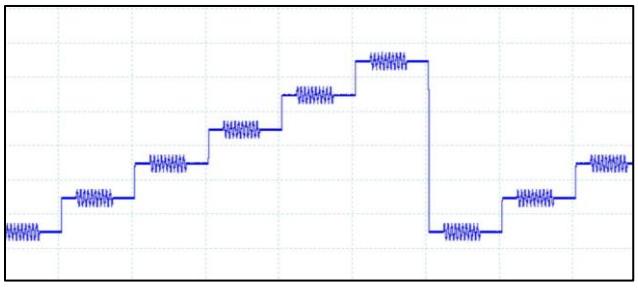


Figure 24. Scope trace of a stepped sweep with triangle waveform superimposed on every step. This is done in order to smear the applied voltage during the time of the step so that any measurements will be averaging over a small voltage range.

5.2.3 Arbitrary waveform generator (AWG)

The arbitrary waveform generator, of which there is one per channel, instead of generating waveforms on the fly, reads userdefined waveformes, TRACes, form QDAC-II volatile memory. The user uploads waveforms as *TRACes*, which can be “played” by the AWGs with individual scaling and DC offsets.

TRACes

Traces are blocks of normalized voltage data where each point corresponds to a sample of 1 µs. Hence 1 million voltages correspond to a sequence of one second. The maximum length of a single trace is 6 million points (6,291,456). The length of traces should be an even number.

> Note that the length of traces should be an even number. When uploading traces with uneven number of points, no error is produced, but the last point is duplicated.

Trace values should be in the interval [-1;1]. Their scaling into actual volts is handled by the AWG generator of the relevant channels. The instrument has 1GB of memory for traces. In order to avoid timeouts, it is recommended to use TCP-IP communication (ethernet) when traces are more than around 40,000 points, especially when using the pyvisa framework.

A trace has three properties: a name, a length and its data points. The name can be a maximum of 15 characters long, length as mentioned maximum 6 mio. Datapoints are transferred as a binary block of 32 bit floating point values, see section 7. A maximum of 24 traces can be defined.

Command	Description
TRACe:	(top node)
:CATalog?	Returns a list of names of all traces.
:DEFine	Defines a trace by its name and number of data points.
:DATA	Transfers trace data to the instrument in binary format.
:REMove:ALL	Clears the trace memory.

When traces become larger than a few tens of thousands points it is recommended to use TCP-IP communication rather than USB/serial. In both cases, when using frameworks with short default timeouts, it may be necessary to increase the timeout. For example in NI-VISA the timeout is by default just 2 seconds.

> When communicating using VISA in Python, QDevil has good experience using both the pyvisa.py backend and the ivi backend included in NI-VISA.
>
> However, when reading big amounts of data from the QDAC-II via USB, the data will often get garbled when using the pyvisa.py backend! So for USB the ivi backend is recommended.
>
> On the other hand, when it comes to transferring traces the ivi backend (NI-VISA) has in rare cases been observed to time out unexpectedly.

Example – Defining traces and viewing the trace catalog

```
>trace:remove:all                          # Deletes all traces in trace memory

>trace:define “PulsRCos20us”,40            # Defines a trace of 40 points (40 µs
>trace:define “Ring1ms”,1000               # Defines a trace of 40 points (1 ms)
>trace:define “Ramp_nonlin_1s”,1e6         # Defines a trace of 40 points (1 s)

>trac:cat?                                 # Returns a catalogue (list) of all trace names.
“PulsRCos20us”,“Ring1ms”,“Ramp_nonlin_1s”
```

Example – Uploading trace data using pyvisa (Python)

```
traclen= 1000                              # Generating a simple not very long trace as a simple Python list
p = 100/(2*math.pi)
values = []
for i in range(traclen):
    values.append(math.sin(i/p)*(1-i/traclen))

qdac2.write_binary_values \                # Transferring the data as a binary block using the pyvisa
("trace:data \"Ring1ms\",",values)         “write_binary_values()” convenience function.
```

The arbitrary waveform generator (AWG)

To “play” a trace on a channel, its arbitrary waveform should first be linked to the trace and, and the scaling and offset of the trace should be set (default 1 V and 0 V, respectively). In addition, the number of repetitions can be set as for the fixed waveform generators. Thereafter the AWG is started like any other generator using the trigger system. The same trace can be used on multiple channels simultaneously each with its own scaling and dc offset and trigger delay (start delay after triggering).

Example – Assigning a trace to a channel’s AWG

```
>sour3:awg:define "Ring1ms "               # Setup the AWG on channel 3 to play the “Ring1ms” trace with a voltage
>sour3:awg:scale 1;offset 0                scale of 1V and a zero dc offset, without repeating.
>sour3:awg:count 1

>sour3:awg:trig:sour imm                   # Set the trigger source to IMMediate and make an IMMediate INITialization
>sour3:awg:init                            in order to start the AWG on ch. 3 immediately.
```

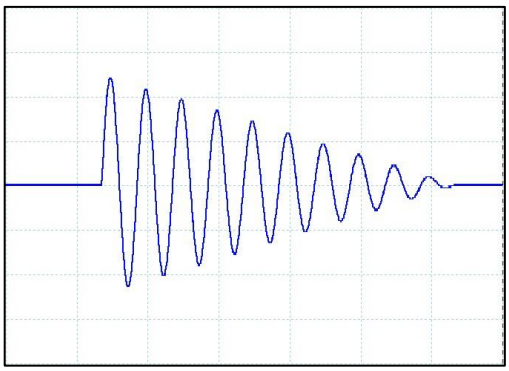


Figure 25. Scope trace of the “Ring 1ms” trace.

5.3 Current sensing

Each channel has an individually operating current sensor, each with two possible measurement ranges. See section 5.1.4 for commands.

5.3.1 Current measurement

Update rate and integration time

The current sourced out of a channel is probed by an IV converter. The voltage is converted into units of current at sampling intervals of 0.33.. milliseconds. However, the A/D converter applies a lowpass filter with a cut-off around 600 Hz, so the actual measurement speed is closer to 1-2 ms than to 0.33 ms.

To suppress noise, the sampled values are continuously integrated, and a running average calculated. The integration time is given by APERture. The most significant noise component is almost always induced 50 or 60 Hz mains power cycle noise. Therefore, it is wise to integrate over an integer number of power line cycles (PLC). APERture can be set indirectly by instead setting the number of PLCs to integrate over, using the NPLCycles command. The default (*RST) number of PLCs is one, meaning that when the LFRequency is 50 Hz, APERture will be 20 ms and for 60 Hz, 16.66.. ms. The maximum integration time is 2 seconds.

Whenever a measurement is triggered the currently integrated value is immediately transferred to the *measurement buffer*. This means that there is no wait time from a current measurement is requested until a value is delivered by the instrument. **It is up to the user to wait "long enough" between reads to accommodate changes in current caused by changes in the output voltages.**

> The recommended minimum required waiting time is 3 ms for the shortest integration times (APERtures). For longer APERtures one should wait at least 3 ms + APERture. This also means that when triggering current measurements (see further down) for example on a DC voltage change, one should use sufficiently long a trigger DELay.

Also, if the integration time (APERture or NPLC) is changed it is up to the user to wait sufficiently long time before reading out a new current measurement. **One needs to wait according to the difference in old and new integration time**.

If COUNt > 1 the instrument will space the values with APERture seconds in time, but without waiting for the very first. Consequently, the reading will last (COUNt-1) x APERture seconds.

Range switching time

When switching between the HIGH and LOW ranges, an electro-mechanical relay is toggling between two measurement resistors. This takes 30-40 milli-seconds. The SCPI command used to change the measurement range "SENSe:RANGe" will for that reason block the command execution for about 30 milli seconds. However, this does not prevent the user from queueing up more commands in the meantime. If the current range is changed for N channels in a *compound command* [6.2], then there will be a delay of N x 30 ms before the next command is executed! This must be taken into account before doing the next current reading. In addition, one should wait for the duration of one APERure after the range switching has completed. A simple way of checking when the RANGe command is complete is to send a query to the instrument, for example a short one such as *STB?.

Example – changing ranges and apertures

```
>sens:rang low,(@1:5)                    # Set the current range to low for channels 1 to 5.
>*stb?                                   # Make a simple query to test when the range switching is complete.
0                                        # Alternatively one can wait 5 x 30ms: sleep(0.15).
>sens:aper 0.2,(@1:5)
sleep(0.2)                               # Change the APERture to 0.2 sec for channel 1-5.
>read? (@1:5)                            # Pseudo code. Assuming that the APERture was 20 ms before, sleep 0.2 s.
5.56377e-10,1.02779e-09,-5.01892e-11,    # Readout all 5 channels
4.50601e-10,6.00684e-10
```

Note that, in addition to the above, it takes some time for any relay contact to stabilize completely when switched from *off* to *on*. Therefore, when switching from LOW to HIGH where there is a very small resistor value in series with the relay, it is recommended to wait a couple seconds (or more) before taking current readings in order to achieve stable readings.

> Note that sending a command is not the same as it will be executed immediately. When using LAN communication there can be up to 10 ms delay from a command is present at the LAN connector until it is executed, which should be considered.
>
> The safest way to ensure that the instrument has executed all sent commands is to perform a small query, for example *STB?, and then wait for the response.

Current range: Implied voltage generation bandwidth

The in-line IV converter utilizes a sensing resistor of which there are two, one for the HIGH current mode and one for the LOW current mode. Due to the high value used in LOW current mode and due to the inherent non-linearities in the active circuits the effective bandwidth of the instrument in LOW current mode is not a simple RC filter behavior but is AC amplitude and DC voltage dependent. At DC voltages and AC peak-to-peak amplitudes smaller than about 3V, the behavior is linear, but at higher amplitudes or DC offsets the behavior in non-linear, limiting the available dV/dt (slew). The curves shown in section 8.2 in "*8 Specifications"* illustrate this non-linear behavior.

Accuracy and precision

Nonlinearity, and precision

Due to in particular the nonlinear output voltage dependence of the IV conversion (common mode error) the accuracy is limited by integral nonlinearity which is of the order of:

$$\textit{Max int. nonlin.}\ (\text{LOW}) \approx 1\ \text{nA/V} \times (\ V_{out} - I_{out} \times 0.04\ \text{V/nA}\ ) \pm 3.5\text{nA}$$

$$\textit{Max int. nonlin.}\ (\text{HIGH}) \approx 0.1\ \text{mA/V} \times (\ V_{out} - I_{out} \times 0.001\ \text{V/mA}\ ) \pm 0.35\ \text{mA}$$

Where V_{out} is the output voltage (as set) and I_{out} is the output current (measured). The first term can actually be used for compensating the recorded currents for the first order common mode effect.

This means that in LOW current mode, if no post correction is performed the maximum error is around 1nA/V x (10V-0nA x 0.04V/nA) + 3.5nA ≈ 14nA, assuming resistive loads. In HIGH current mode the corresponding number for is about 1.4mA. Future firmware versions may incorporate this compensation.

As described in section 3 (Current sensing resolution), the resolution is considerably better in LOW current mode than in HIGH current mode. Please see section 8.3 in *8 Specifications* for details and examples.

Offset error

The current sensors have a non-negligible offset dependence on temperature, which is not compensated for in the current version of the firmware. Future versions may include some compensation. This means that if the temperature of the environment where the instrument is used is different from the temperature in the laboratory where the instrument was calibrated a non-zero value will be read out even when a channel is unloaded, and the voltage is zero. This error is typically around 1nA in LOW range and 0.1mA in HIGH current range but can be up to 5 times higher. It is therefore recommended to record the “zero current” before measurements, if the absolute value is of importance.

Commands

The easiest way to measure current is to use the READ? query command, as it will initiate and trigger a single measurement (of COUNt values) and transfer the result to the output buffer. Note that if COUNt is large or APERture is larger than the READ? query may time out, leaving values in the output buffer.

The measurement buffer can hold up to 65536 values. If more values are added, a memory overflow error will be produced.

To avoid timeouts or to synchronize current measurements with other events or equipment measurements can instead be triggered and held in the measurement buffer until they are queried. The trigger commands are listed below, but please see section 5.4 for details on the trigger system.

Commands for configuring, reading, and triggering current measurements.

Command	Description
SENSe[:CURRent]	(top node)
:ABORt	Stops on going measurements and trigger sequencies.
:APERture	Defines the averaging time in seconds (default one power line cycle).
:NPLCycles	Defines the averaging time in units of power line cycles (default 1).
:COUNt	Defines the number or measurements to perform on per trigger (default 1).
:NCLeft?	When performing more than one measurement (COUNt > 1) NCLeft will report how many measurements are left at a given time for the ongoing trigger cycle.
:DELay	Specifies the delay after a trigger signal is received until the measurement is performed
:INITiate[:IMMediate]	Makes the sensor ready to receive a trigger and run a single trigger cycle, where after it will go back into idle mode. If the trigger source is IMMediate, then a measurement is performed immediately.
:INITiate:CONTinuous	Switches continuous mode on or off. When set to ON the sensor will react to every trigger received until it is set to OFF.

:TRIGger:SOURce	Defines the trigger source (INTernal#, EXTernal#, BUS, IMMediate). Default is IMMediate
:RANGe	Defines the current sensing range, HIGH (default) or LOW.
:DATA:POINTs?	Queries the number of measurement points in the measurement buffer.
:DATA:REMove?	Queries and deletes a specified number, or all measurement points in the measurement buffer.
:DATA:LAST?	Queries the most recent *triggered* measurement point. This can be done even if the measurement buffer has been emptied.
FETch[:CURRent]?	Reads all data points in the measurement buffer, without clearing the buffer.
READ[:CURRent]?	Runs a single trigger cycle and returns the (COUNt) measurement datapoints. If the trigger sequence is initiated and waiting for a trigger, READ? will assume priority*.
READ:ADC?	Reads the raw code from the analog to digital converter used for current sensing. Mostly used during factory calibration. Reports a single value also if COUNT > 0.

*)Note that the READ? Command will behind the scenes set the trigger source of the current sensor to IMMediate, INIT:CONT to OFF and issue a single INIT:IMMediate command. After the read command INIT:CONT will be off and TRIG:SOUR will be IMMediate. So, any triggered operation will as a consequence be aborted.

Example – read a single current value

```
>READ8?                          # Captures the and returns the current value of the running average on
2.53123e-3                       # channel 8 (2.53. mA).
```

Example – trigger and read 3 measurements on ch4

```
>sens4:coun 3                    # Set the number of readings to 3
>sens4:trig:sour imm             # Set the trigger source to immediate (default).
>sens4:init                      # Start one trigger cycle.
>sens4:data:poin?                # Read the number of points in the measurement buffer.
3
>fetch4?                         # Read the entire measurement buffer for the channel with our clearing.
-0.0010562,-0.0010562,-0.0010563
>sens4:data:rem?                 # Read and remove all datapoints in the measurement buffer.
-0.0010562,-0.0010562,-0.0010563
>sens4:data:poin?
0
```

5.4 Trigger system

The QDAC-II has an advanced triggering system allowing synchronization of programmed changes in for example DC steps and waveforms.

Except for a few global commands there is trigger system with one sequence per channel functionality: Under each source:<source name> node there will be an independent trigger sequence. In addition, there is a trigger sequence for the current measurement on each channel; please see section 5.3.1 for details.

The trigger model is a very simplified version of the SCPI model, omitting the arm layer.

Starting and stopping generators is done by the generator's trigger system. All generators have an identical trigger system (see next section). and all generators can be addressed using the source name "ALL", e.g. "SOUR:ALL:INIT".

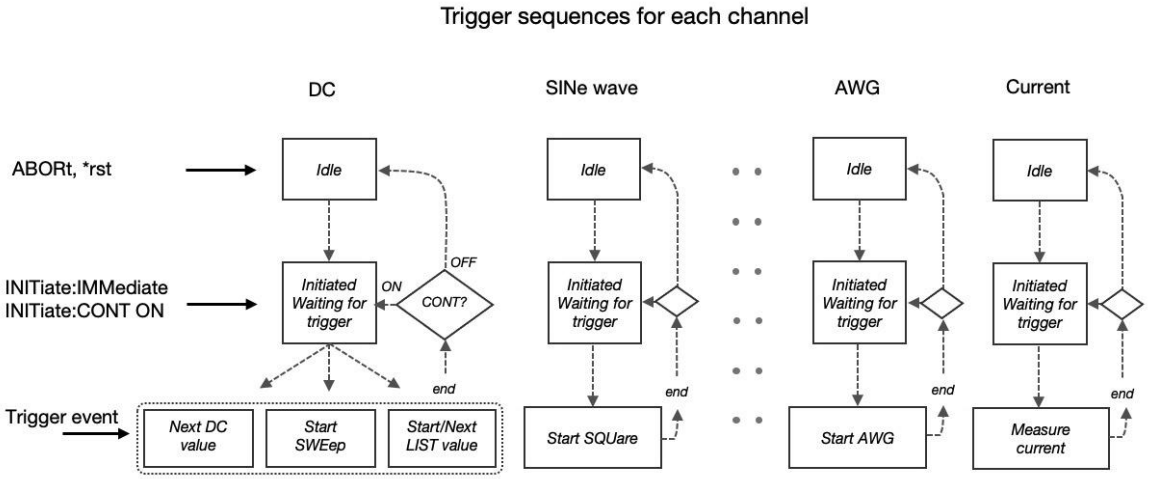


Figure 26. Trigger model. Every channel has a trigger sequence for each of its sources and for the current measurements.

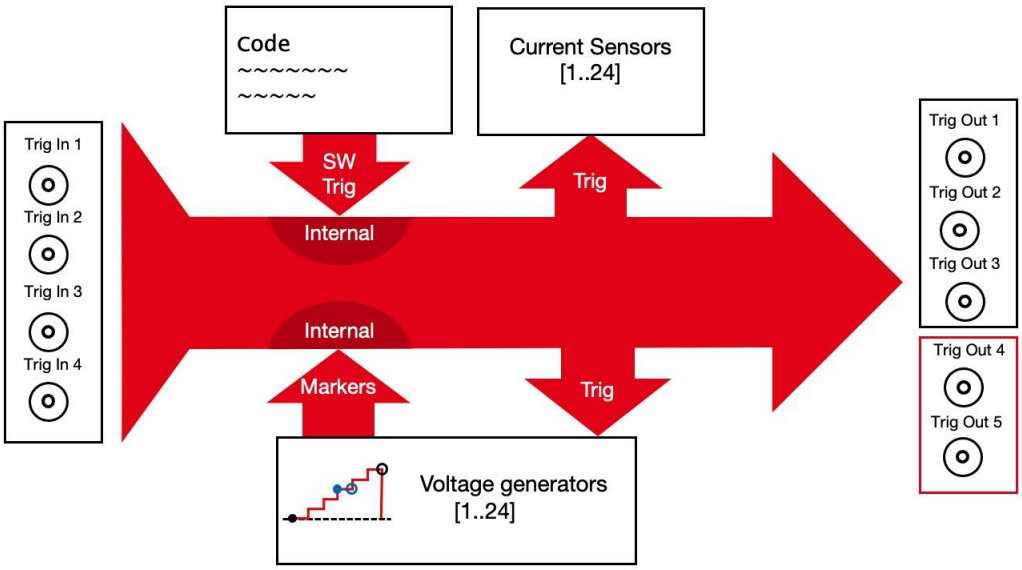


Figure 27. The QDAC-II trigger system allows synchronization between channels and waveforms and between multiple QDAC-II units. There are 14 internal triggers available.

5.4.1 Starting (triggering) and stopping voltage generators

Except for the simple case of just setting a DC voltage all generators need to be started by a trigger event. The reason is that each generator has several parameters which one would like to set correctly before run. Further triggering provides many ways of configuring coordinated waveform generation and measurements.

It is, however, very simple to trigger a generator as all have their default trigger source set to IMMediate, meaning that a trigger signal is always assumed to be present; all what is needed is to INITiate the generator, and it will start. Trigger sources can also come from the external trigger inputs the internal BUS or the *internal triggers* which can be connected to signal generator *markers*-

For examples of starting/triggering generators, please see section 5.2.

Command	Description
SOURce:{DC|SINe|SQUare|TRIangle|AWG|ALL}	(top node)
:ABORt	Stops the specified generator.
:DELay	Specifies a delay from the trigger signal until the generator actually starts.
:INITiate	Makes the generator wait for a trigger signal (if the trigger source is Immediate, which is the default, this command will start the generator).
:TRIGger:SOURce	Specifies the trigger source.

5.4.2 Triggering of current sensors

Current sensors are triggered in the same way as voltage generators with the commands described in section 5.3.1, where examples can also be found. The current sensors themselves cannot generate trigger signals. As their timing in known it is not necessary.

5.4.3 Trigger Outputs and Inputs

The QDAC-II has three galvanically isolated ≈5 V trigger outputs on the front panel and two *non-isolated* 3.3 V outputs on the rear panel. The output impedance of the outputs on the front panel is nominally 50 Ω (±20 Ω uncertainty). The output impedance of Trig Out 4 and 5 is around 30 Ω .

The five Trigger Outputs are either controlled by internal triggers, external trigger inputs or software triggers and commands.

Note that the trigger output pulses appear approximately 1µs *before* the marker position of the voltage generator output due to different delays (phase shifts) in the analogue circuits. To compensate, a suitable trigger DELay can be added (1 µs increments).

The QDAC-II has 4 galvanically isolated trigger BNC connector trigger inputs on the rear panel. These are used for synchronizing the instrument to external devices or other QDAC-II units. The trigger pulses must be at least +2.2 V (preferably > +2.5 V) high. Negative voltages (below -0.5 V) should be avoided not to damage the input circuits. Further, they should not exceed +3.8 V. This is particularly important when connecting an external clock signal to Clock input (*Trigger In 4*) because the *on*-time will typically be 50%, and not just occasional pulses.

> For the 3.3 V trigger outputs to drive these inputs it is recommended to use 75 Ω cabling for interconnection multiple QDAC-IIs. This will give a slightly higher signal compared to using 50 Ω cabling, as it is highly recommended to terminate the interconnecting cable according to the impedance.

There are two reasons for terminating the cables:

1. Standing waves will alter the signal quality seen by the respective instruments in the chain.
2. Standing waves may increase the signal height above the allowed 3.8V or make it lower than -0.5 V.

Commands for configuring and connecting Trigger Outputs to internal or external triggers

Command	Description
OUTPut:TRIGger	Top node
:SOURce	Specifies which input or internal trigger should control the Trigger Out signal
:WIDTh	Specifies the width of the trigger output pulse, minimum 1µs.
:POLarity	Specifies if the pulse is positive or negative
:DELay	Specifies a delay from receipt of a signal from tegh SOURce until the trigger pulse is output
:SOURce	Specifies the source (INTernal1..14, EXTernal1..4, BUS, HOLD)

Example – Output a 10 µs pulse on Trig Out 1 when a sweep starts

```
>sour1:dc:swe:poin 4096                  # Set up the sweep on ch1
>sour1:dc:swe:dwell 0.0001
>sour1:dc:swe:count 1
>sour1:dc:swe:star -0.2;stop 0.6
>sour1:dc:volt:mode sweep
>sour1:dc:trig:sour IMM
>sour1:dc:marker:start:tnum 1           # Connect the sweep period start marker to internal trigger 1
>outp:trig1:sour int1                   # Make Trig Out 1 listen to internal trigger 1
>outp:trig1:width 0.0001                # Set the pulse duration or Trig Out 1
>sour1:dc:init                          # Start the dc generator onch1 by initializing it (trigger source = immediate)
```

5.4.4 Internal triggers and Markers

Internal triggers

Internal triggers can start a generator or trigger the next step in a sweep or list sequence or trigger a current measurement. Internal triggers can also trigger external devices when it is routed to one of the Trigger Out connectors. *Internal triggers* can be fired programmatically using a scpi command (`TINT`) for example for starting several generators simultaneously on the same or on multiple channels, or they can be fired by *marker* events, see below. There are 14 internal triggers available.

All generators have a configurable trigger *DELay* so that starting can be delayed until an arbitrary chosen time (in 1µs increments) after the trigger event.

The standard scpi global `*trg` command is an internal “master” trigger which, however, cannot be activated by markers.

Markers

Markers are events on waveforms and DC sequencies which can activate *internal triggers*.

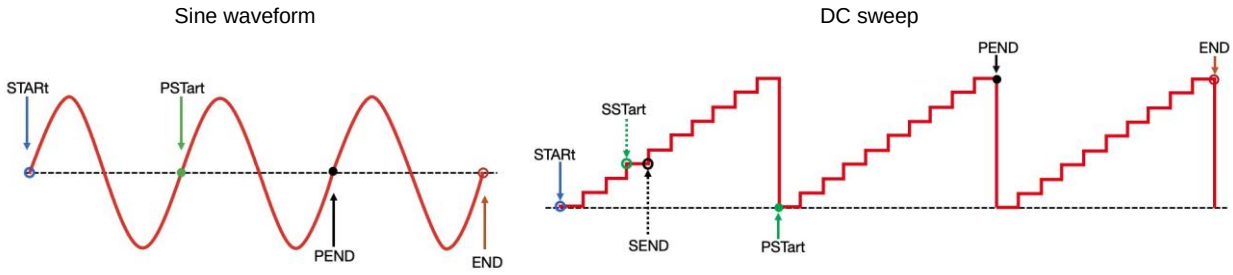


Figure 28. Waveforms and DC sequencies have *markers* associated with them. One pair marking the start and end of the waveform or sequence, and one pair marking the start and end of every period. The PSTart and Pend markers shown above are actually called SSTart and SEND for DC LIST and SWEep and mark the beginning and the end of a voltage step.

There are four markers for each generator, six for the DC generator. For the sine, square, triangle and AWG there are markers at the start and at the end of the signal (STARt, END), and at the start and end of each period (PSTart, PEND). The end of period marker is in general only used in those rare cases where a delay of one period from the start is needed.

For the DC generator there are also step-start and step-end (SSTart, SEND) markers for the LIST and SWEep modes generating markers at the start and end of each step. For ANALog SWEeps these markers coincide with the PSTart/PEND markers.

Note that the STARt and PSTart markers will both be present at the beginning of the waveform. In the same way the END and PEND markers will both be present at the end of the waveform. For the DC generator the same counts for the SSTart and SEND markers.

Commands connecting generator markers to internal triggers

Command	Description
SOURce:{ }:MARKer { } = DC:|SINE|TRIangle|SQUare|AWG|ALL	Top node for markers
:STARt:TNUMber	Specifies which internal trigger is linked to the STARt marker.
:END:TNUMber	Specifies which internal trigger is linked to the END marker.
:PSTart:TNUMber	Specifies which internal trigger is linked to the *period start* (PSTart) marker.
:PEND:TNUMber	Specifies which internal trigger is linked to the *period end* (PEND) marker.
SOURce:DC:SSTart:TNUMber	Specifies which internal trigger is linked to the SSTart (step start) marker.
SOURce:DC:SEND:TNUMber	Specifies which internal trigger is linked to the SEND (step end) marker.
TINT[:SIGNal]	Immediately fires the specified internal trigger.

Example – Start sine generator on ch11 when the sine on ch3 ends.

```
>sour:sine:freq 7500,(@3,11)          # Set the frequency of the sine generators on ch3 and ch11 to 7.5 kHz.
>sour:sine:span 0.2,(@3,11)           # Set peak-to-peak amplitude of the sine on ch3 and ch11 to 200 mV.
>sour:sine:coun 75,(@3,11)            # Set the number of periods per run for each sine to 75.
>sour3:sine:mark:END:TNUM 1           # Connect internal trigger 1 to the END marker for the sine wave on ch3.
>sour11:trig:sour INT1                # Set the trigger source for the sine on ch11 to internal trigger 1
>sour11:sine:init                     # Initiate the sine generator on ch11 so it is ready to receive a trigger.
>sour3:trig:sour IMM                  # Make sure that the trigger source for the sine generator on ch3 is
                                      immediate.
>sour3:sine:init                      # Start the sine generator on ch3
```

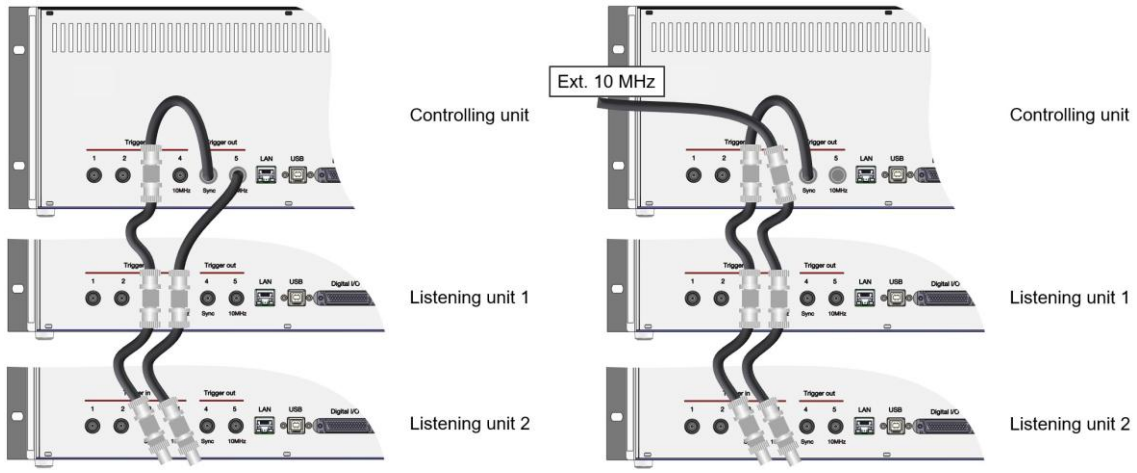
Example – Start sweeps on ch1 and ch2 simultaneously using internal triggering

```
>sour:dc:swe:poin 128,(@1,2)          # Set number of ramp steps on ch1 and 2 to 128
>sour:dc:swe:dwell 0.05,(@1,2)        # Set duration of each level to 50 ms
>sour:dc:swe:count 1,(@1,2)           # Set number of ramps to one
>sour1:dc:swe:star -0.1               # Set start and stop voltages for the two sweeps
>sour1:dc:swe:stop 0.3
>sour2:dc:swe:star 0
>sour2:dc:swe:stop 1.2
>sour:dc:volt:mode sweep,(@1,2)       # Set the dc generator on ch1 and ch2 to sweep mode
>sour:dc:trig:sour INT1,(@1,2)        # Set the trigger source on both channels to internal trigger 1
>sour:dc:init (@1,2)                  # Initialise the dc generator on both channels
>tint 1                               # Activate internal trigger 1 to start the two seeps.
```

5.5 Synchronization of multiple QDAC-II units

Multiple QDAC-IIs can be inter-connected so that samples are output synchronously, meaning that ramps, waveforms and DC values can be triggered to appear simultaneously to within fractions of a microsecond. Note that all cables should be terminated according to their characteristic impedance, typically 50 Ω or 75 Ω.

It is possible to use an external 10 MHz clock for all units, but one QDAC-II unit can also act as *reference* and generate a shared 10 MHz clock signal. 10 MHz frequency is chosen for compatibility with other instruments.



(a) Internal clock reference (b) External clock reference

Figure 29. Wiring up three synchronized QDAC-IIs, with (a) use of 10 MHz clock generated by the master unit, and (b) external 10 MHz clock. Note that both cables need to be terminated according to their impedance.

Voltage samples are output at a rate of 1 MHz. The 1 MHz clock is generated from the 10 MHz internal or external clock. To make sure that the 1 MHz clocks on connected units sharing the same 10 MHz clock are in phase, they need to be aligned. Otherwise, they may be phase shifted by any multiple <9 of 100 ns.

Alignment is carried out by a one-time *synchronization* of the units upon a positive edge at the *Sync In* port (*Trigger In #3*). One QDAC unit needs to be be the controller and output a Sync pulse on its *Sync Out* (*Trigger Out #4*) port which should therefore be routed to the *Sync In* ports of all units including the *controller* itself. An external pulse can also be used (see section 3 “Rear panel” for appropriate voltage levels).

When waiting for the signal on *Sync In* the instruments will enter *Sync Wai*t mode and will freeze their outputs for a short period of time. There is a time-out limit of 2 seconds in waiting for the *Sync In* signal.

Note that the use of terminated 75 Ω cabling is recommended in order to ensure that the *On* thresholds of the Trigger Inputs are exceed. Also, note that BNC cables have a group delay of about 5 ns/m, which may influence the timing. Typical timings are better than 50 ns, often in the range of 20-30 ns using short impedance matched cables.

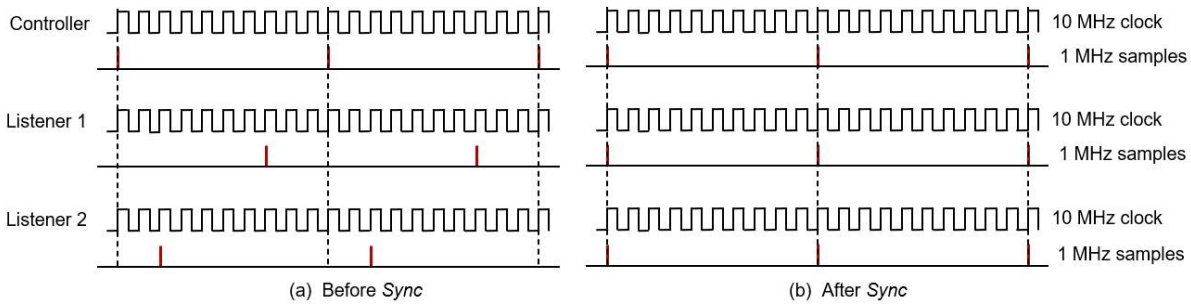


Figure 30. When multiple QDAC-II units are sharing a 10 MHz reference it is necessary to synchronize the voltage outputs by sending a trigger signal from one QDAC-II (the controller) to the other units (listeners) so that the 1 MHz sample output comes *in phase* across the units. After the synchronization pulse is sent from the controller, all units will output samples with inter delays less than 50 ns depending on the cabling. All instruments need to enter *Sync Wait* mode to react on the sync pulse.

Command	Description
SYSTem:CLOCk:SOURce	Defines the source of the main 10 MHz clock, either INTernal or EXTernal. In case of EXTernal clock, a 10 MHz square wave must be present on the Trigger In #4 (10MHz) BNC connector. The peak-to-peak amplitude should be at least 2.2 V and positive. Signals below 5 MHz are ignored.
SYSTem:CLOCk:EXTernal:STATus?	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]	This command (event) tells a listening unit to enter *Sync Wait* mode and listen for the next rising edge at *Sync In* (*Trigger In #3*) BNC connector. At that moment shift its 1 MHz sample clock to align with the next rising edge of the 10 MHz clock. The instrument will leave *Sync Wait* mode after 2 seconds if a signal has not been received.
SYSTem:CLOCk:SEND	Makes the instrument (the controller) output a 10 MHz clock on Trigger Out #5 (“10 MHz”)
OUTPut:SYNChronize:SIGNal	Makes the instrument enter Sync Wait mode (see above) and thereafter sends a 10 µs pulse out on the *Sync Out* (*Trigger Out #4*) BNC port. The signal will have positive polarity and will always be preceded by a 0V output of 10 µs duration.

Example – synchronizing using the master unit’s clock

Controller:*

```
>SYST:CLOC:SEND ON          # Send 10 MHz clock out signal out on 10 MHz Out (Trigger Out #5).
```

Listeners:

```
>SYST:CLOC:SOUR EXT         # Use external 10 MHz clock signal present on 10 MHz In (Trigger In #4).
>SYST:CLOC:SYNC             # On next rising edge on Sync In (Trigger In #3) align 1 MHz sample.
```

Controller – less than 2 seconds after

```
>SYST:CLOC:SYNC             # Enter Sync Wait mode and send pulse on Sync out (Trigger Out #4).
>OUTP:SYNC:SIGN
```

*)If an external clock is used the Controller unit should have this command issued instead:

```
>SYST:CLOC:SOUR EXT         # Use external 10 MHz clock signal present on “10 MHz In” (Trigger In #4).
```

Example – start the square wave generator simultaneously on synchronized units

```
ctrller>OUTP:TRIG4:SOUR INT1        # Prepare Trigger Out #4 to react on Internal Trigger #1 on the controller.
ctrller>SOUR4:SQU:TRIG:SOUR EXT4    # Make the square wave gen. on ch04 on the controller start on Ext. Trig. Input #4
ctrller>SOUR4:SQU:INIT              # Initialise the square wave generator on ch04 on the controller

listen1>SOUR6:SQU:TRIG:SOUR EXT4    # Make the square wave gen. on ch06 on listening unit start on Ext. Trig. Input #4
listen1>SOUR6:SQU:INIT              # Initialise the square wave generator on ch06 on the listening unit #1

ctrller>TINT 1                      # Fire Internal Trigger #1 on the controller. This will fire External trigger #4.
```

6 Operation

The QDAC-II command set adheres to the IEEE 488.2 (IEEE Standards Board, 1992) and SCPI standards (SCPI Consortium, 1999), with those adaptions required for exploiting the advanced functionalities of the instrument and omissions of those parts of the standards which do not contribute to instrument usability.

6.1 SCPI Command groups

In SCPI a command is represented by a *program header* consisting of one or more *instrument control headers,* also referred to as *program mnemonics*, or *keywords*, separated by colon characters ":". So each command is a hierarchy of colon separated keywords; a command tree, where the keywords are branches and the colons are nodes. In SCPI terminology each branch are called a *sub-system*. Below is a list of the top level keywords – or *sub-systems* – for QDAC-II:

Command group (sub-system)	Description
*xxx	xxx represents one of the IEEE488.2 mandatory commands .
ABORt	Stops all triggered actions.
SOURce	Controls the voltage outputs.
SENSe	Controls the current sensors.
READ	Shortcut for read out of the current sensors.
FETCH	Reads measured current values.
OUTPut	Setup of Controls trigger outputs.
TINT	Direct control of internal triggers.
TRACe	Management for arbitrary waveform generation traces
STATus	Instrument status register readout and configuration.
SYSTem	Error read out, non-measurement related settings (e.g. LAN settings), synchronization.
DIAGnostic	Functionality diagnostics commands, calibration management etc.

6.2 SCPI Command syntax

Except for the IEEE 488.2 common command headers all starting with a "*" character, most *instrument control header mnemonics* have a long form and a short form. The short form is the first four letters of the long form, except if the 4th letter is a vowel and the long form consists of more than 4 letters. Only the short form *or* the long form of command is accepted, not anything in between. However, short and long form mnemonics/keywords can be mixed in a program header. For example `syst:comm:lan:ipad?` for querying the instrument's IP address can also be written as `system:comm:lan:ipaddress?`.

In the Command Reference document, the short form of the mnemonics (keywords) are written in uppercase though the instrument does not distinguish between uppercase and lowercase characters. So uppercase and lower case can be mixed within a keyword.

Command lines should always end with a newline character <nl> (0x0A) – also known as line feed character. The instrument will accept carriage return <cr> (0x0D) character preceding the newline character.

Command syntax

```
Level1:level2:level3 parameter1,parameter2,…
```

Query syntax

```
Level1:level2:level3? parameter1, parameter2,…
```

Compound command lines

A single command line can contain multiple program headers (commands) separated by a semicolon.

Keywords after a semicolon inherits the command tree up to the right most keyword of the first command, unless the keywords start with a colon, then it is starting at the root.

Principle of the compound command syntax

```
Cmd-1_Level1:cmd-1_level2:cmd-1_level3;cmd-2_level3;cmd-3_level3
```

Compound command syntax using colon (:) to reset the command tree

```
Cmd-1_Level1:cmd-1_level2:cmd-1_level3;:Cmd-2_Level1:cmd-2_level2:cmd-2_level3
```

Example – setting the parameters and triggering a generator as a compound command

```
>sour1:squ:freq 5000;span 0.2;dcyc 25;count 1000;trig:sour bus;:*trg
```

Channel list syntax

Individual channels are addressed by adding a channel number suffix to the relevant keyword. Keywords which can have a channel number suffix:

Command/keyword
SOURce
SENSe
READ
FETCh
STATus:OPERation:{}:CHANnel
STATus: QUEStionable:{}:CHANnel
DIAGnostics:VCALibration
DIAGnostics:ICALibration

Other commands which can have a numeric suffix:

Command/keyword
OUTPut:TRIGger
SYSTEM:TEMPerature

Example – setting the dc (FIXed mode) output for a single channel

```
>sour10:volt 0.54
```

However, the instrument supports addressing multiple channels using the so-called channel list syntax. Instead of adding a numeric suffix to the relevant keyword a channel list may be added as the last parameter in a command. Channels lists can specify a list of individual channels or ranges of channels, or both in combination. Channel lists starts with "(@" and ends with ")".

When the command is executed it is repeated for every channel in the list. Hence, the command is not executed in parallel for the channels but sequentially one channel at a time.

Channel list syntax	Description
(@1,3,5,6)	Comma separated. Addresses channels 1,3,5, and 6.
(@1:24)	Range, addresses all channels.
(@1:3,9,17)	Combined range and comma separated. Addresses channels 1,2,3,9, and 17.

Example – setting all dc (FIXed mode) output to zero

```
>sour:volt 0,(@1:24)
```

Note that channel lists are not fully supported in compound messages. They only work for the last command in the compound program header.

6.3 Command synchronization

6.3.1 Sequential versus overlapped commands

There are two major categories of commands in a SCPI instrument as QDAC-II:

- **Sequential commands**
- **Overlapped commands**

Sequential commands are commands or functions which need to be fully executed before the next command can be executed. Hence, when sending a series of sequential commands to the instruments they will be executed in the order they are received one after the other. All commands except those mentioned below are sequential.

Note that sending a command is not the same as it will be executed immediately. When using LAN communication there can be up to 10 ms delay from a command is present at the LAN connector until it is executed, which should be taken into account. The safest way to ensure that the instrument has executed all sent commands is to perform a small query, for example *STB?, and then wait for the response.

Overlapped commands are commands which execute while following commands are interpreted and executed. An overlapped command does not really have to be a command, it is in fact most often a trigger (internally or externally) which causes some action.

All triggered operations, and immediate setting of a DC voltage are overlapped commands. In order to check if a waveform generator, a DC:SWEep or DC:LIST is completed, their NCL (number of counts left) property can be checked. The same is true for the current sensors. When setting an immediate DC voltage one can test the LAST? property against a VOLT? query to see when the requested voltage has been reached. This may not be immediately due to finite slew rate; delays caused by the analogue low-pass filters are not accounted for.

6.3.2 Hardware Synchronization

The QDAC-II offers an advanced internal triggering system which is coupled to the outside world by the *Trig In* and *Trig Out* physical ports, see sections 5.4 and 5.5.

6.4 Status and events

For monitoring the status of various operations and events, the QDAC-II Compact has a SCPI compliant status register structure, which is also used for reporting over current events. In the current version of the firmware (14-1.70) only the status byte register (STB) has been implemented.

The Status Byte Register (STB) is a summary register. It contains summary information about occurred errors, the Error/Event Queue and the Output Queue. Bit 2 in the STB is raised when the Error queue is non-empty.

Table 2. STB (Status Byte Register).

Bit	Value	Name	Meaning
4	16	Message Available	Data is available in the instrument's output buffer
3	8	Not used	Value always 0.
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.

6.5 Errors and events reporting

Error messages follow the scpi numbering with relevant device specific information added, giving relevant hints to the details of what has failed.

For a general description of scpi errors see the SCPI Vol. 2 Command Reference (SCPI Consortium, 1999) section 21.8.

The error numbers fall in different categories and number ranges

Error number range	Group description
-499 .. -400	Query errors (standard)
-399 .. -300	Device errors (standard)
-299 .. -200	Execution errors (standard)
-199 .. -100	Command errors (standard
300 .. 399	Device errors (specific)

Table 3 There are four standard SCPI error number (negative) ranges for different error categories. Positive error numbers are device specific.

6.5.1 Buzzer and LED

When an error occurs (including over current ILIMit), the buzzer will produce a beep (if not disabled), and the LED will start flashing a pattern indicating the type of error/event.

When an error or event is present the LED will only signal one error/event according to the priority shown in the table below (lowest number, highest priority). This means if both an error message is available and a current limit event has occurred, then it is the current limit event pattern which is signalled. Once the current limit event has been processed by the user, the LED will signal “SCPI error message available” until the error queue is empty, only interrupted if communication is taking place, or the instrument is restarted.

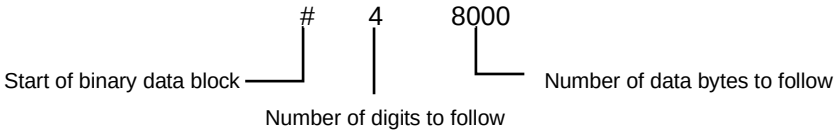
LED blink pattern (approx.)	State / error	Priority
	(Re)starting firmware	1
	Communicating	2
	Current limit event occurred	3
	SCPI error message available	4
	Waiting for command (idle)	5

7 The IEEE 488.2 binary block format

For commands and queries transferring large amounts of data, there is the option for using a block format. The format is described in IEEE-488.2. For this instrument only a small fraction of the options described in the standard are used.

Write

For writing data to the instrument, the binary data is preceded by a block header of this format:



The binary block header is followed by the actual data in IEEE 754 single precision floating point format (*binary32*). As an example of 8000 data bytes will give 2000 floating point numbers.

Read

For reading large chunks of data it is necessary to instruct the instrument to send the data in binary format. This is done on the sub-subsystem (command) level and not as a general setting (which is the intended way by the standard), as only LIST and AWG commands support the binary block format. The IEEE-488.2 standard is implemented in a simplified way using the FORMat sub-system:

:FORMat:READings:DATA <type> [,<length>]

<type>: ASCii or REAL

<length>: 32 or 64. Length is only used for REAL.

In REAL mode data will be transferred as described above for *write* – in a “IEEE-488.2 definite length block”, that is with the #<*digits*><*bytes*> header. In IEEE488.2 default length is 64 bits. This instrument uses 32 bits as default.

8 Specifications and performance

Number of channels	24
Voltage ranges	±10 V and ±2 V, configurable by software on each channel individually
Voltage resolution	20 bits: 1 LSB = 3.8µV (2V range), nominally 1 LSB = 19.1µV (10V range), nominally 25 bits DC mode[1)]: 1 LSB = 0.1µV (2V range), nominally 1 LSB = 0.6µV (10V range), nominally
Output resistance	50 Ohms resistive
Output capacitance	1 µF / 0.22 µF / 10 nF (DC / MED / HIGH filter modes)
Temperature coefficient	Not measured. But data indicates < 5 µV/°C @ 23°C, 0.1V output (10V range)
Stability	Less than ±2 µV fluctuations in 14 days (2V range, 1V output, zero current) after minimum 3 hours warm-up (room temperature stable within ±0.5°C)
Maximum integral non-linearity	1.7 LSB (typical)
Maximum differential non-linearity	0.7 LSB (typical)
Current output, nominally[2)]	±10 mA / ±150 nA (high and low current range respectively)
Sample rate	1 MHz
Rise time (1-99% open)	4 µs (in high bandwidth and high current mode))
Predefined waveform generators	Sine, Square, Triangle on every channel.
Arbitrary waveform generator	One on each channel (channels can share waveform definitions)
Traces	16 each with 6 million points (6 seconds) - to be increased in future updates
Trigger outputs (BNC)	3x isolated 5V on front, output resistance 50 ±20 Ω 2x non-isolated 3.3V on rear, output resistance not specified but is effectively around 30 Ω
Trigger inputs	4x isolated 3.3V on rear (minimum -0.5V, max 3.8V), input current, typ. +0.01 µA, max. ±10µA. Trigger minimum level ≈ 2. V, safe level 2.5V.
Current resolution	Minimum detectable current step: ±10mA range: ≈ 10µA ±150nA range: ≈ 50pA
Power requirements[3]	±15.4V (1.5A). Min-max range: ± (15.2-15.6) V 5.5V (1.5A). Min-max range: 5.2-5.7V
Dimensions (incl. feet)	45 cm x 37 cm x 18.5 cm
Weight	6.4 kg

[1)] 25 bits resolution is available in FIXed DC mode, that is only steady DC output is supported, thus not including sweeps and lists which are still 20 bits.

[2)] The instrument can actually source up to around 18 mA in the high current range on a few channels simultaneously, but not on all. Further, specifications are for ±10 mA. In the LOW current range the sourcing ±10V short circuit currents are about +4.3mA and -11 mA, respectively. In LOW current (sense) mode the current sensor is only guaranteed to measure up to ±150 nA. However, in most cases it will actually measure up to ±200 nA before saturating.

[2)] The requirements are satisfied by the QDevil Power Supply. If for some reason a third-party power supply is required, it is possible to use one when the mentioned requirements are fulfilled.

8.1 Noise and drift

Noise

Noise spectra have been derived from multiple time series each of about 34 seconds. The QDAC-II signal was amplified by an AC coupled Stanford Research SR560 low noise pre-amplifier (0.03 Hz high-pass filter) set to a gain of 10^4 and a 1 MHz bandwidth. The signal was digitized at 4 MS/s using an AC coupled 8 bit USB oscilloscope (PicoScope 2205A-MSO). The spectra have been binned into approximately 1 Hz wide bins and normalized to power spectral density. 5-10 power spectra were averaged to achieve reasonably smooth curves. To ease interpretation, the power spectra are presented as amplitude spectra. Reference measurements were performed by substituting the QDAC BNC output with a 50 ohm termination. All measurements have been carried out in the QDevil office building *without* access to clean ground, good shielding, temperature control etc. The QDAC-II was powered by a by the QDevil Power Supply (QPS) placed approx. 1.5 m away from the QDAC-II and the SR560.

Measurements were performed both in the 10V range (HIGH) and in the 2V output range (LOW) with the output set to 1V. No load other than that of the SR560 was connected to the QDAC-II output.

The results are shown in Figure 31 and a summarized in Table 4. The 1/*f* bend is observed around 60 Hz. Above the cut-off frequency of the built-in low pass filters the noise floor settles at the noise level of the setup which is around 3.5V/√Hz. The upwards bending of the spectra above 100kHz is an artifact of the measurement setup.

Filter range	10 V range	2V range
HIGH	15 nV/√Hz	8 nV/√Hz
MEDium	15 nV/√Hz	8 nV/√Hz
LOW	4.5 nV/√Hz	4.5 nV/√Hz

Table 4. Noise floor for combinations of filter range and voltage range.

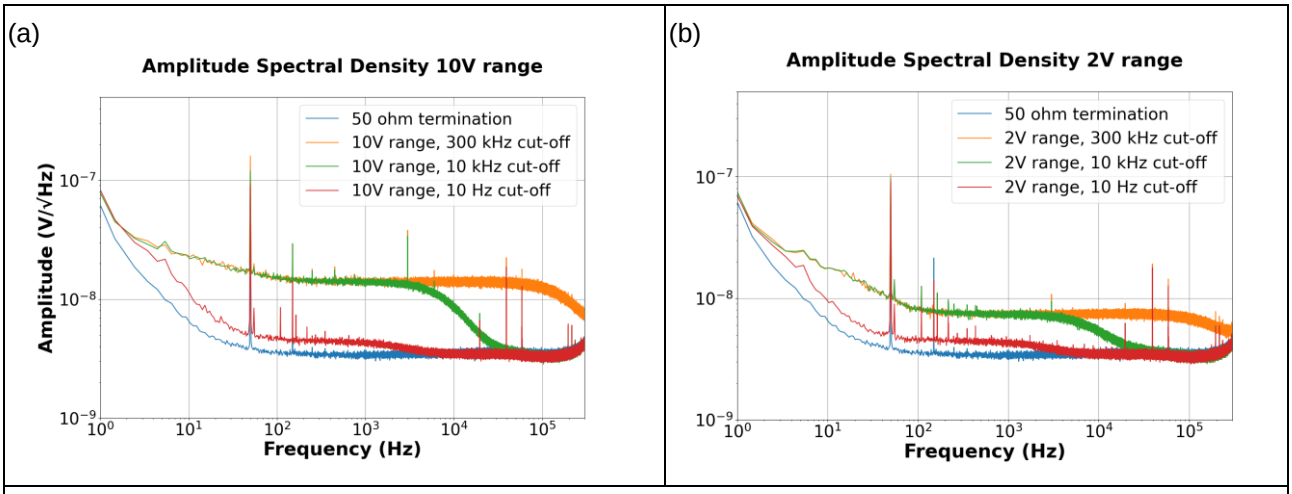


Figure 31. Left: Noise spectra in the 10V output range with a DC voltage output of 1V. Curves are shown for the 3 low-pass filter settings, including DC mode which has 25 bits DAC resolution and 10 Hz cut-off. The blue curve shows the background noise measured in a 50 Ω termination. Right: The same data measured in the 2V output range, also with a DC voltage output of 1V.

Drift

Drift is another word for low frequency noise, however studied in the time domain. To measure the low frequency components of a signal, extremely stable equipment is required and full control over the laboratory temperature, or the influence thereof. Therefore, QDevil has engaged with the Danish National Metrology institute (the Danish NIST) for measuring the variation in output when a fixed DC value is set on one of the QDAC-II channels.

After switching on the QDAC-II, the filter mode was immediately switched to DC (without resolution enhancement active), the output range was set to LOW (2V) and the actual output (CH04) to around 1V. The difference between a traceable Zener reference (Fluke 732) and the QDAC-II output was then logged using a high accuracy 8.5-digit DMM (Solartron 7081) for a period of about 12 days. The voltage was averaged and logged in 1 minute time intervals. The QDAC-II channel was only loaded by the input of the DMM.

The temperature in the laboratory was kept stable at nominally 23.0 ± 0.5 ˚C. The temperature 20 cm from the QDAC-II was logged during the measurement. The laboratory was not free from people traffic, but a small room divider was placed next to the table with the setup in order to shield any draft coming from people passing by. So, a quite realistic experimental situation.

A summary of the measurements can be seen in Figure 32. After an initial relaxation period of about 3 hours the output is stable to within about ±2 µV at low voltages and currents if the ambient temperature, as here, is kept stable within ±0.5 ˚C. As most drift from experience is due to the internal voltage reference, it is expected that the drift is approximately the same in the 10V (HIGH) output range. Some correlation between the close-by ambient temperature and the QDAC-II output voltage can be observed, however, the correlation is a bit ambiguous.

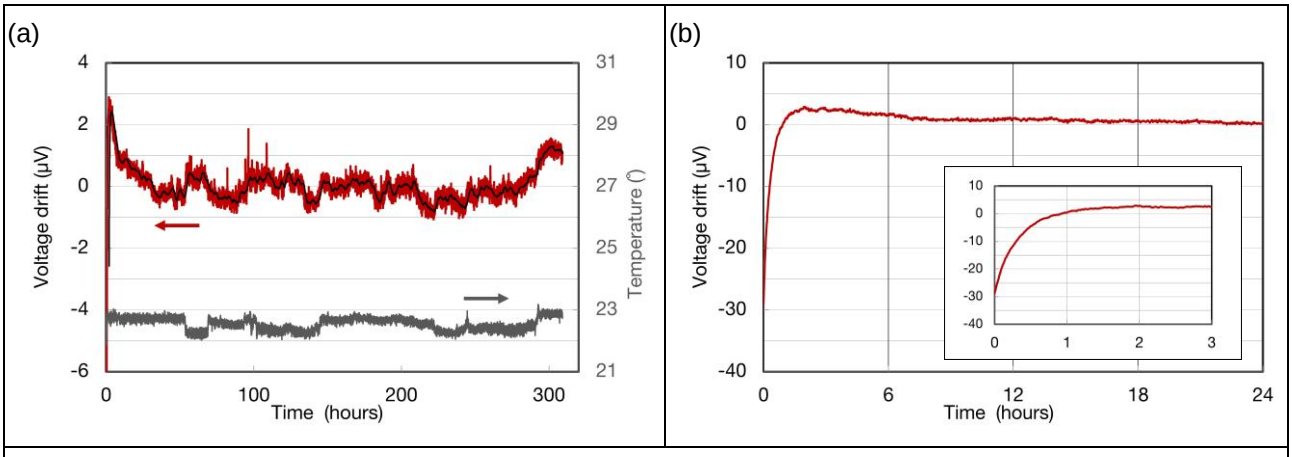


Figure 32. (a) Change in output voltage recorded from 2-3 minutes after powe- on at 1V output in DC mode in the 2V range. Recorded by the Danish National Metrology institute, DFM, using a traceable Zener reference and a precision nano voltmeter. The voltmeter has an integration time of about 1 minute, so the highest frequency is around 30 mHz. The temperature was logged close to the instrument and can be seen to be around 22.5˚C ±0.5˚.

(b) The first 24 hours of drift after power on. It is seen that after about 3 hours the output is close to its final value. But allowing a warm-up of 24 hours will settle the output to vary less than ±2 µV, approximately.

8.2 Non-linear output characteristics

8.2.1 LOW current mode

The nonlinearity of the IV convertor used for current sensing causes the frequency response of the voltage generators in the instrument to be non-linear too. This is primarily in the LOW current mode, and in particular for negative going fast signals. Effectively the bandwidth becomes limited the closer one approaches the negative voltage limit (-10 V) where a slew rate limit of the order of 15 mV/V. This corresponds to the maximum slope of a 2Vpp sine at 2.4KHz.

*Note that in the HIGH current mode these limits are **not** observed.*

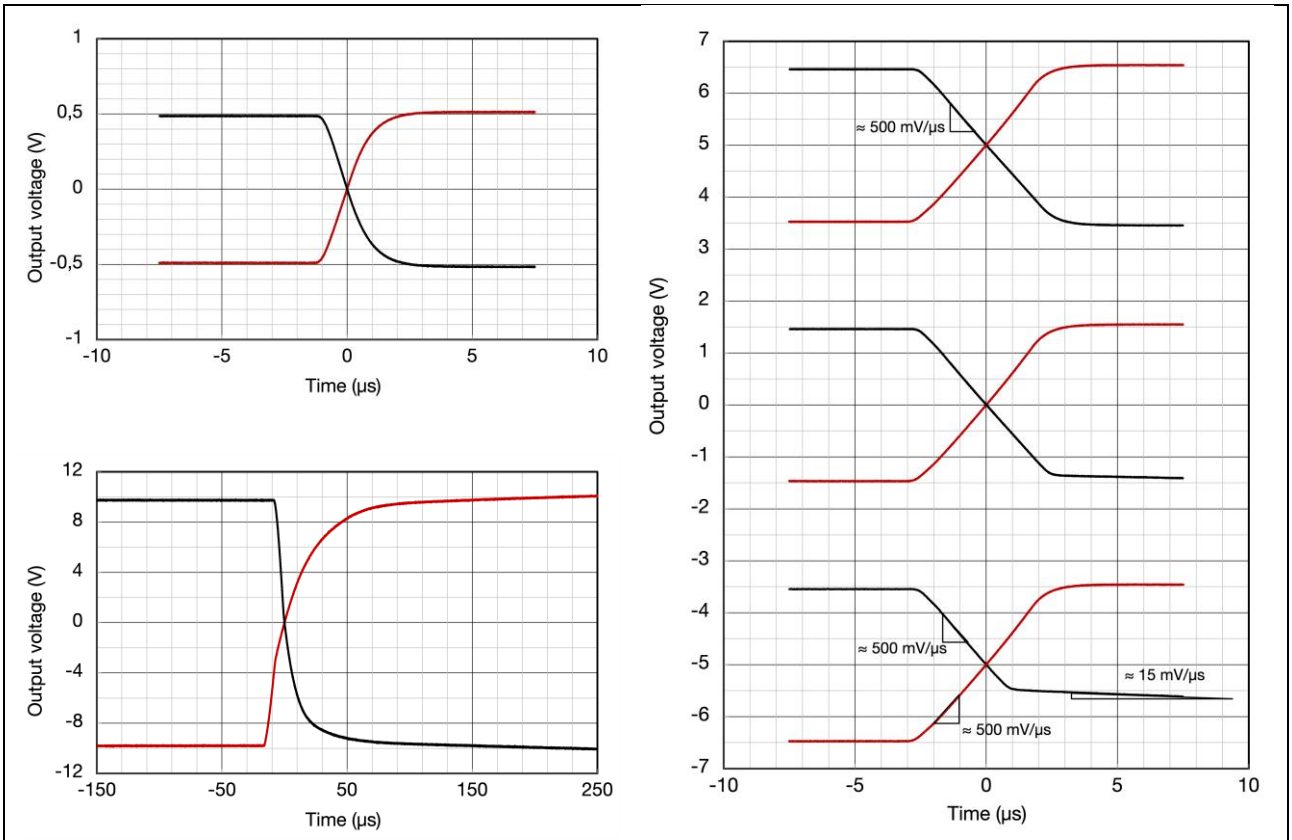


Figure 33. Step responses in CURRent:RANGe = LOW mode.

Left: Small step and large step symmetrically around zero **HIGH filter mode** (300 KHz cut-off) in the LOW current mode with no external load connected to the channel. It is seen that for low voltages (e.g. 1 V) a ≈99% rise time of 3-4 µs is achieved whereas for a 20V step it is more like 400 µs with an initial steep slope but falling off a few volts.

Right: 3 V steps recorded at different DC offsets also in **HIGH filter mode**. It is seen that the response is slew rate limited at around 500 mV/µs for positive going steps and for negative going steps starting at high voltages. But at negative going steps larger than ≈2V the slew rate settles at around 15 mV/µs for negative voltages.

8.2.2 HIGH current mode

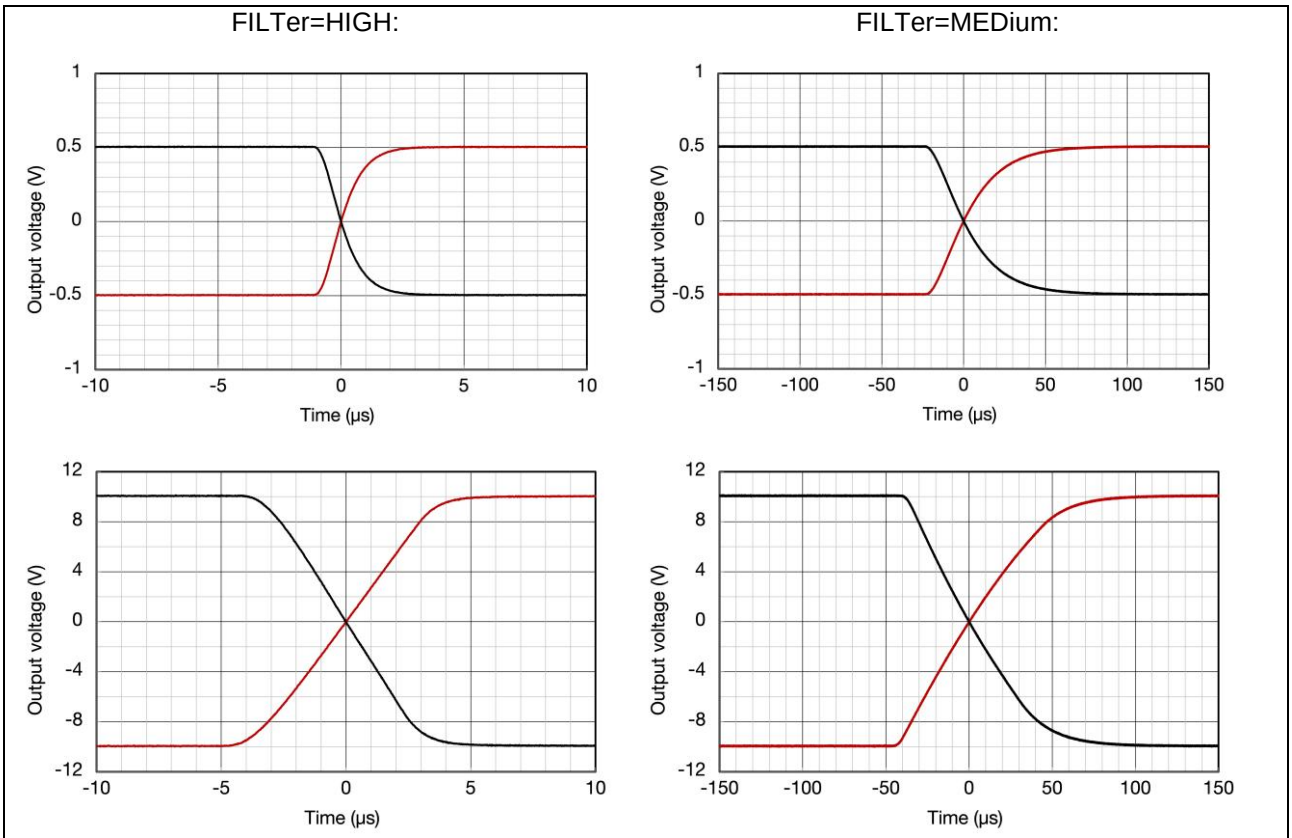


Figure 34. Step responses in CURRent:RANGe = HIGH mode with no external load connected to the channel

Left: Small step and large step symmetrically around zero in **HIGH filter mode** (≈300 kHz cut-off) in the **HIGH current mode**. It is seen that for low voltages (e.g. 1 V) an *RC filter like* ≈99% rise time of 3-4 µs, is achieved whereas for a 20V step it is more like 8µs partially limited by the output amplifier slew rate.

Right: Small step and large step symmetrically around zero in **MEDium filter mode** (≈10 kHz cut-off) in the **HIGH current mode**. Both for low and high steps an *RC filter like* response is observed with a ≈99% rise time of ≈80 µs for small steps and 120 µs for the 20V step, slightly limited by the output amplifier slew rate.

8.3 Filter switch transients

When switching almost anything it is unavoidable that some transient signal will appear somewhere. That is also true when switching the low-pass filters in the QDAC-II. However, a special circuit (patent pending) developed for avoiding transients ensures that there are no noticeable, or vanishing, voltage steps occurring at the output when switching the filters. However, some remaining transients induced by the relays remain. There are *step signatures* and *ringing signatures* present, depending on the actual transition (HIGH -> LOW, HIGH -> MED, etc.).

Figure 35 shows examples of the characteristic transient signatures, in this case from a HIGH→DC filter switch with a DC output at +9.9 V. For the measurements the QDAC-II output was AC coupled (via an RC high-pass filter, cutoff ≈1 kHz) to a cascaded SR445 amplifier. The measurements were performed at a 350 MHz bandwidth but sampled at 60 MHz by an oscilloscope.

In Figure 35(a) it is seen that the voltage step at the switch is smaller than 100µV. The noise which can be observed both before and after the switch is primarily from the cascaded SR445 and not from the QDAC-II itself. In Figure 35(b) an example of a *ringing signature* is shown, actually one of the clearest ones measured. These are characterized by a short (≈0.1µs) very high frequency oscillation and a slow (≈0.5µs) decaying oscillation at around 10 MHz.

For normal use cases low pass filtering will be present in the signal chain, e.g. a QFilter. This will effectively take care of the high frequency transient signals. In the curves below, the grey lines show the data filtered by a digital implementation of an RC filter with a 1MHz cut-off frequency.

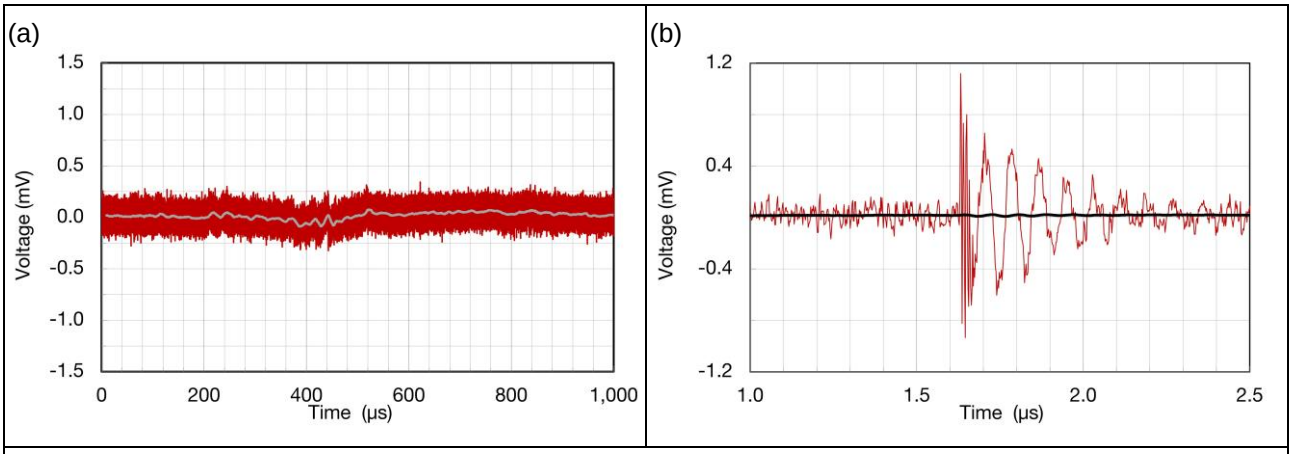


Figure 35**.** (a) The actual remaining step (change) in voltage when switching from HIGH to DC filter mode at an output level of +9.9 V. (b) One of the clearest (worst) examples of the characteristic *ringing signature* observed during switching from HIGH to DC mode.

The red lines are the recorded data. The grey lines are after filtering the signals with a digital implementation of an RC filter with a 1MHz cut-off frequency.

8.4 Current measurement signal to noise ratio

The current sensor's signal to noise ratio reading depends on the magnitude of the current and on the integration time (APERture), but in particular on the used lowpass FILTer (DC, MEDium, HIGH, as suppression of voltage noise at the output RC stages is converted to current noise. The following graphs show series of current readings recorded over 20 seconds using different filters and integration times. The channel was loaded by a 40 MΩ resistor. For each trace the voltage was stepped by ≈0.8 mV corresponding to an offset of current of nominally 20 pA.

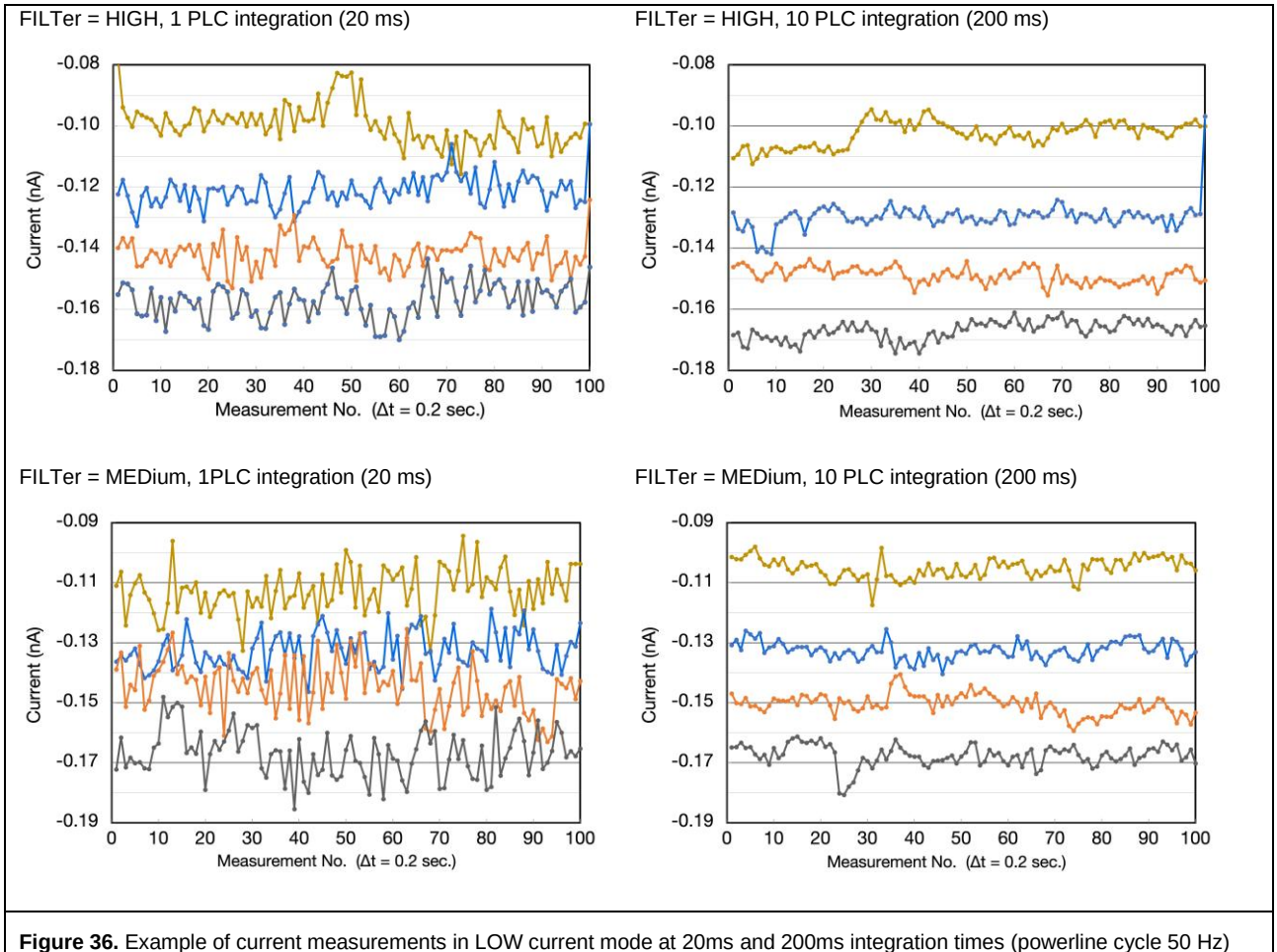


Figure 36. Example of current measurements in LOW current mode at 20ms and 200ms integration times (powerline cycle 50 Hz) over 20 seconds. The output is loaded by a 40 MΩ resistor and the voltages changed in ≈ 0.8 mV steps, corresponding to 20 pA steps. The low pass filter is in HIGH mode. It this seen that 20pA changes may be resolved consistently, at least on short time scales. However, in MEDium filter mode the current noise requires more integration than in HIGH filter mode.
Note that increasing the integration time (APERture) will reject more high frequency noise and possibly more 50/60 Hz noise, but will not suppress low frequency noise (drift).

9 Bibliography

IEEE Standards Board. (1992). *IEEE Standard Codes, Formats, Protocols, and Common Commands for Use With IEEE Std 488.1-1987, IEEE Standard Digital Interface for Programmable Instrumentation.* New York, USA: The Institute of Electrical and Electronics Engineers, Inc.

SCPI Consortium. (1999). *Standard Commands for Programmable Instruments (SCPI).* Retrieved from https://ivifoundation.org/scpi

Appendix A Important operational guidelines (firmware version 14-1.70)

Below is a list of important operational guidelines for the QDAC-II, operated with firmware version 14-1.70. Quantum Machines publishes new firmware versions frequently and notify any changes to the operational guidelines in the release notes. We appreciate any reports of additional bugs which may be found, please contact us on www.quantum-machines.co/contact-us/.

Communication

- To make the instrument acquire an IP address via DHCP, the ethernet cable should be plugged in before powering up the unit, or it should be restarted after plugging in the cable (SYSTem:REStart).
- In addition to executing a SYSTem:COMMunicate:LAN:UPDate when LAN settings are changed a SYSTem:REStart is also required in order for changes to take effect.
- When uploading vast amount of binary data over LAN, for example multiple large (4-6 mio points) arbitrary waveform traces, instabilities in the local network may occasionally cause the instrument to report an error and the upload will fail. So always check for errors after uploading a trace, and in case of errors, re-upload the trace.
- Channel list notation only works for the last command in a compound command.

Functionality

- Please do not change the voltage range (RANGe) at non-zero output voltages, and always immediately explicitly set the DC voltage to zero (again) after changing the range, as some mechanisms required when changing ranges are not fully automatic yet.
- Triggering of DC voltages (SOUR:VOLT:TRIG level) in FILTER = DC (resolution-enhanced) mode is not working.
- Please avoid setting SOUR:VOLT:TRIG to a new value immediately before switching in and out of DC mode while resolution enhancement (RENH) is enabled (which it is by default). The TRIG value will be erroneously applied when switching the filter, potentially causing sample damage.
- Current limitation is not implemented. Please do not try to source more current than around 10mA for each channel.
- Trigger outputs are appearing 1-4 µs *before* voltage outputs triggered by the same sources, e.g. waveform markers or trigger inputs, due to delays in the analogue circuits. To compensate (to best sub-µs level) please add a trigger output delay.
- Setting the voltage output of a channel using the raw SOURce:DAC command will not set a 25 bit value in DC mode (where 25 bit resolution enhancement is active). A fractional DAC value will be rounded to its nearest integer.
- Queried voltages are slightly off (by less than 20µV; or 4µV in the 2V range) due to internal rounding errors. The same is true for queried raw DAC values.
- NCLeft is reporting “-1“ for generators which have been ABORted if they have COUNt set to Inf. But they *should* return”0”.
- Continuous triggering (INIT:CONT = ON) of the DC generator in DC filter mode (when resolution enhancement (RENH) is on, does not work. One time triggering following an INIT command works fine.

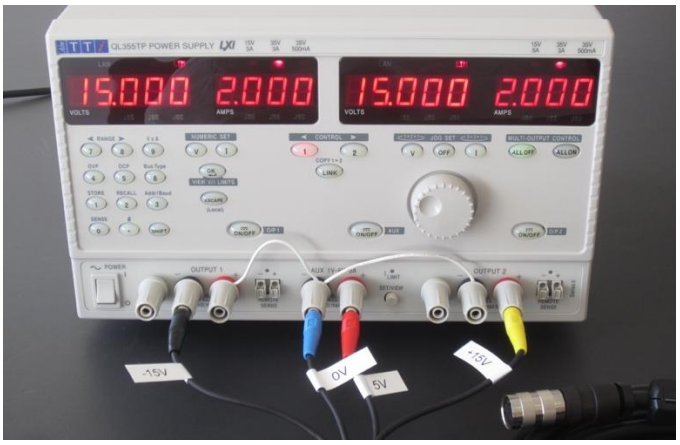
Appendix B Using laboratory power supplies

Though it *is* recommended to use the QDevil Power Supply for powering the QDAC-II it is also possible to use a third party power supply of good quality, which can switch all voltages on simultaneously.

Appendix B1. Procedure for configuring a QL355TP power supply

1. Before starting, ensure that the QDAC-II is NOT connected to the power supply. Then switch on the power supply using its POWER button.
2. Establish a common reference by connecting "+" on OUTPUT1 to "-" on AUX and to "-" on OUTPUT2. Use a jumper wire with conductor area ~1mm^2 (not included).
3. Set Output 1 and Output 2 to 15.5V and a 2.00A current limit.
4. Press the SET/VIEW button and set the AUX output to 5.50V and a 3.00A current limit.
5. Press SHIFT # 33 on the keyboard. This will lock the settings of the power supply.
6. Verify that the output voltages and currents cannot be changed when turning the knobs of the power supply.
7. Check that the ALL OFF button is lit.
8. Connect all 4 bananas from the QDAC-II power cord to the power supply outlets as shown below. Note the voltages printed on the flags on the wires:

Black 	-15.5 V	OUTPUT1 -
Blue 	0 V (Common)	AUX -
Red 	+5.5 V	AUX +
Yellow 	+ 15.5 V	OUTPUT2 +



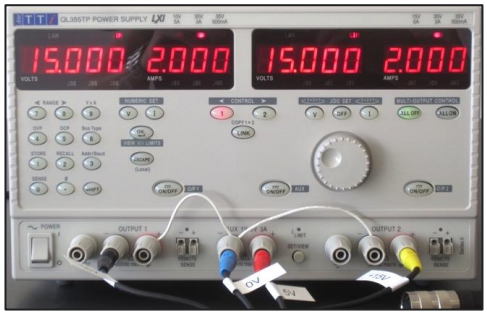
9. Ensure that the cables are firmly mounted, so that they will not fall off.
10. Connect the power cord to the backside of the QDAC-II and lock the connector by turning its shield clock-wise so that the plug goes all the way into the socket and so that it will not fall off easily.

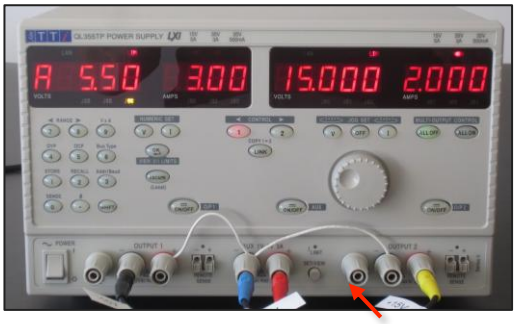
If you have difficulties setting up the power supply or the QDAC-II unit, or if you have other questions, please contact QDevil at info@qdevil.com.

Appendix B2. Procedure for turning ON the QDAC-II using a QL355TP

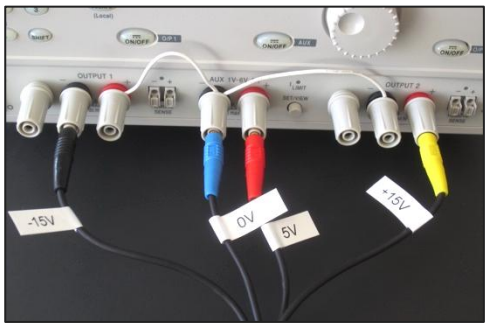
> **WARNING**
>
> The QDAC-II instrument is a sensitive laboratory apparatus. Always follow this checklist before turning on power, and carefully read the manual. The instrument may be damaged if the start-up checklist is not followed.

1. If the power supply has not yet been configured, you must first follow the checklist for configuring the power supply (next page).
2. Turn on the QL355TP power supply
3. Check that the ALL OFF button is lit.
4. Check that Output 1 and Output 2 have been set to 15.000V and 2.000A.





5. Press the SET/VIEW button and check that the AUX has been set to 5.50V and 3.00A.



6. Verify that the power wires (**and 0V jumper wires)** are mounted firmly as shown above.
7. Verify that the power cord is firmly inserted into the back of the QDAC-II. Never disconnect or re-mount power cords unless the **ALL OFF** button is lit.
8. Press the **ALL ON** button (always allow at least 2 seconds before power-on after power-off).
9. Notice that the LED on the front panel of the QDAC-II is blinking. For the first 10 seconds during initialization it will blink red-green, then it should start blinking green only (idle mode).
10. To switch off the QDAC-II unit, press the **ALL OFF** button on the power supply **before** disconnecting the plug from the back side of the QDAC-II.

> Please wait minimum 5 seconds after switching off the instrument until you switch it on again.

Appendix C Open-source licenses

This is a list of open-source licenses used in the firmware.

Xilinx firmware libraries (SPDX: MIT):

```
/******************************************************************************
*
* Copyright (C) 2007 - 2018 Xilinx, Inc.  All rights reserved.
*
* Permission is hereby granted, free of charge, to any person obtaining a copy
* of this software and associated documentation files (the "Software"), to deal
* in the Software without restriction, including without limitation the rights
* to use, copy, modify, merge, publish, distribute, sublicense, and/or sell
* copies of the Software, and to permit persons to whom the Software is
* furnished to do so, subject to the following conditions:
*
* The above copyright notice and this permission notice shall be included in
* all copies or substantial portions of the Software.
*
* THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR
* IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,
* FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL
* XILINX  BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY,
* WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF
* OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE
* SOFTWARE.
*
* Except as contained in this notice, the name of the Xilinx shall not be used
* in advertising or otherwise to promote the sale, use or other dealings in
* this Software without prior written authorization from Xilinx.
*
******************************************************************************/

/* print.c -- print a string on the output device.
 *
 * Copyright (c) 1995 Cygnus Support
 *
 * The authors hereby grant permission to use, copy, modify, distribute,
 * and license this software and its documentation for any purpose, provided
 * that existing copyright notices are retained in all copies and that this
 * notice is included verbatim in any distributions. No written agreement,
 * license, or royalty fee is required for any of the authorized uses.
 * Modifications to this software may be copyrighted by their authors
 * and need not follow the licensing terms described here, provided that
 * the new terms are clearly indicated on the first page of each file where
 * they apply.
 *
 */
```

LwIP TCP/IP stack (SPDX: BSD-3-Clause):

```
/*
* Copyright (c) 2001-2004 Swedish Institute of Computer Science.
* All rights reserved.
*
* Redistribution and use in source and binary forms, with or without modification,
* are permitted provided that the following conditions are met:
*
* 1. Redistributions of source code must retain the above copyright notice,
*    this list of conditions and the following disclaimer.
* 2. Redistributions in binary form must reproduce the above copyright notice,
*    this list of conditions and the following disclaimer in the documentation
*    and/or other materials provided with the distribution.
* 3. The name of the author may not be used to endorse or promote products
*    derived from this software without specific prior written permission.
*
* THIS SOFTWARE IS PROVIDED BY THE AUTHOR ``AS IS'' AND ANY EXPRESS OR IMPLIED
* WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF
* MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT
* SHALL THE AUTHOR BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL,
* EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT
* OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS
* INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN
* CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING
* IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY
* OF SUCH DAMAGE.
*
* This file is part of the lwIP TCP/IP stack.
*
* Author: Adam Dunkels <adam@sics.se>
* Author: Simon Goldschmidt
* Improved by Marc Boucher <marc@mbsi.ca> and David Haas <dhaas@alum.rpi.edu>
*/

/*
* Copyright (c) 2001-2004 Leon Woestenberg <leon.woestenberg@gmx.net>
* Copyright (c) 2001-2004 Axon Digital Design B.V., The Netherlands.
* All rights reserved.
*
* Redistribution and use in source and binary forms, with or without modification,
* are permitted provided that the following conditions are met:
*
* 1. Redistributions of source code must retain the above copyright notice,
*    this list of conditions and the following disclaimer.
* 2. Redistributions in binary form must reproduce the above copyright notice,
*    this list of conditions and the following disclaimer in the documentation
*    and/or other materials provided with the distribution.
* 3. The name of the author may not be used to endorse or promote products
*    derived from this software without specific prior written permission.
*
* THIS SOFTWARE IS PROVIDED BY THE AUTHOR ``AS IS'' AND ANY EXPRESS OR IMPLIED
* WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF
```

```
* MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT
* SHALL THE AUTHOR BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL,
* EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT
* OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS
* INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN
* CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING
* IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY
* OF SUCH DAMAGE.
*
* This file is part of the lwIP TCP/IP stack.
* The Swedish Institute of Computer Science and Adam Dunkels
* are specifically granted permission to redistribute this
* source code.
*
* Author: Leon Woestenberg <leon.woestenberg@gmx.net>
*/
```