

QSwitch

User Manual



Remote controlled breakout box:

- 24 pin Fischer connectors for input/output.
- 24 switchable signal lines pairwise twisted.
- 8 BNC break-out lines.
- Individual soft grounding for each signal line.
- Automatic soft grounding on power failure.
- USB and LAN interface.



Get the latest QSwitch pdf manual, firmware update and other information by following the QR code or this link: <https://docs.quantum-machines.co/latest/docs/Hardware/QSwitch/>

Table of contents

1	SAFETY AND REGULATORY INFORMATION	3
2	MANUAL OVERVIEW	4
3	INTRODUCTION TO THE QSWITCH	5
3.1	Description.....	5
3.2	Architecture and connections.....	7
3.3	Device safety features	9
4	GETTING STARTED	10
4.1	Grounding the QSwitch	10
4.2	Powering up the QSwitch	10
4.3	Connecting the QSwitch to a device or fridge	11
4.4	Communication setup	11
4.4.1	USB/serial setup	12
4.4.2	LAN setup.....	15
4.4.3	Python	16
5	FUNCTIONALITY AND OPERATION.....	19
5.1	Quick introduction to SCPI.....	19
5.2	Opening and closing relays	21
5.3	Autosave and restart.....	23
5.4	Error reporting.....	24
6	COMMAND REFERENCE.....	26
6.1	IEEE Common commands.....	26
6.2	Signal routing.....	28
6.3	System commands	30
6.3.1	Communication commands.....	30
6.3.2	Error system.....	34
6.3.3	Other commands.....	34
7	SPECIFICATIONS AND PERFORMANCE	36
8	BIBLIOGRAPHY	38
APPENDIX A	FIRMWARE UPDATE	39
APPENDIX B	MAC AND LINUX USB/SERIAL DRIVER INFORMATION.....	41
	Appendix B1. Linux driver instructions.....	41
	Appendix B2. MAC OS driver instructions	42

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 35°C, preferably 20–25°C, 20% – 80% RH (non-condensing), see section 7 *Specifications and performance*. 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. The instrument should be supplied inside its rated supply levels using the designated mains adapter. Further, the instrument should be operated only inside the rated environmental range, and the signal voltages should be kept inside the specified safe limits: **AC: 30 Vrms, 42 Vpp, DC: 60 V**.

	For device- and for human protection from any applied high signal voltages, always make sure to ground the instrument, see section 4.1.
	Only connect the instrument to the included power supply using the cable delivered with the instrument.
	Keep this device away from dust, moisture, and extreme temperatures.
	Do not use the device when damage to housing or cables is noticed. Do not attempt to service the device yourself but contact Quantum Machines.
	Keep this device away from children and unauthorized users.
	Keep away from strong magnetic fields, as the device contains sensitive magneto-mechanical components.
	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 Quantum Machines.
	Protect this device from shocks and abuse. Avoid brute force when operating the device.
	Not for general waste, recycle as electronic waste.

2 Manual overview

1 Safety and regulatory information

Important precautions and requirements to the environment in which this instrument is to be used.

2 Manual overview

This section.

3 Introduction to the QSwitch

To get to know the instrument it is highly recommended starting by reading this section. An overview is provided mentioning the unique features of the QSwitch.

4 Getting started

Instructions in powering up, connecting and setting up communication.

5 Functionality and operation

Description and examples of the core features and commands of the QSwitch, including a short introduction to the SCPI protocol and list of errors.

6 Command Reference

A complete reference for all commands.

7 Specifications and performance

Contains a specification overview and data sheet curves.

8 Bibliography

A list of relevant references.

3 Introduction to the QSwitch

The intended use of the QSwitch from Quantum Machines is control and measurement of quantum computing science and quantum computing production systems. The target users are physicists, engineers, trained technicians, or supervised university students – in short people who are aware of good and safe practice in electro-technical physics laboratories, in particular for sensitive quantum devices. For safety reasons the instrument must not be used with signal voltages in excess of 30 V_{rms}, 42 V_{pp}, or 60 V DC.

3.1 Description

The QSwitch is a remote-controlled 8 channel breakout box for 24-line Fischer (or Micro-D) connector terminated cables. The QSwitch is designed for being placed between for example a 24-channel DC voltage source such as the QDAC-II Compact or a 24-channel manual breakout box such as the QBox (both from Quantum Machines) and the fridge wiring, so that fully automated breakout of any of the 24 DC/low frequency lines from the fridge – the *signal lines* – is possible.

The QSwitch makes it possible to automate/remote control experiments where instruments connected to the breakouts need to be shared between device terminals or switched in and out.

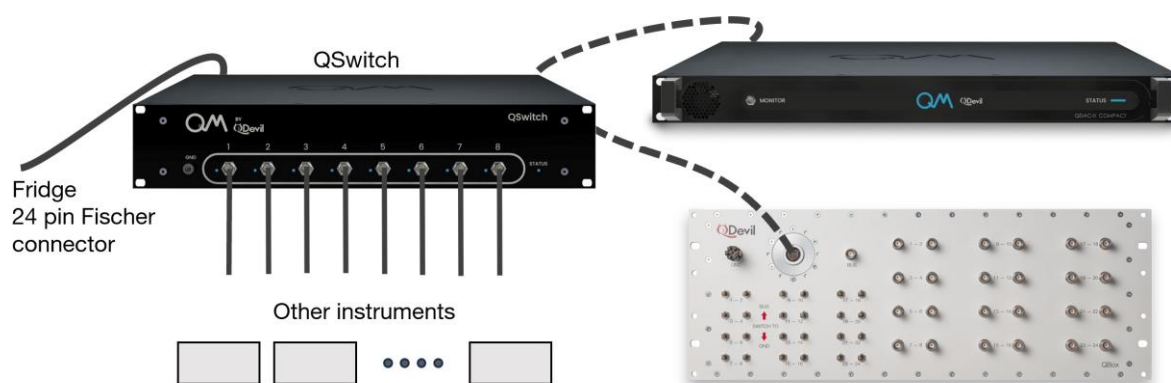


Figure 1. The QSwitch features remote controlled breakout to 8 BNC connectors of the wiring of a standard 24-channel DC/low frequency fridge. It also features individual soft grounding of each signal line and breaking of the further signal path to the apparatus connected at its input (QDAC-II Compact, QBox or any 24-pin Fischer connector terminated instrument).

The QSwitch offers break-out of any of the 24 signal lines to any of the 8 BNC connectors. The QSwitch has two 24-pin Fischer connectors, one for connecting to the device under test (OUT/DUT) and one for connecting to a manual breakout box (IN) – or any 24-channel instrument, for example a low-frequency/DC source, featuring a compatible 24-pin connector.

The 24 *signal lines* are wired from the input side (IN) to the output side (OUT).

To be able to connect the DUT (OUT) to the source at the input (IN) in a controlled way, all input lines can be switched in and out individually. This is useful for protecting the DUT and the source and

supports the case where the DUT should be sourced through one of the breakout BNCs instead of from the input (IN).

Every signal line can be tapped out to one or more of the eight breakout lines (BNC).

The internal microcontroller used for controlling the relays is EMI shielded from the signal lines and uses an external power adapter to minimize noise from the mains power outlet.

The QSwitch is operated through a USB/serial interface or through a LAN interface using SCPI commands. The QSwitch implements a subset of SCPI commands appropriate for its operation. Python programming examples and a QCoDeS driver with examples are available.

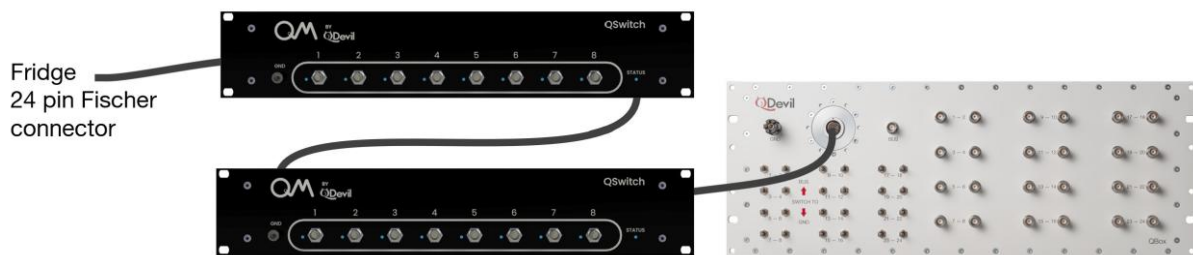


Figure 2. To automate access to more than 8 out of 24 lines, several QSwitches can be daisy chained. Obviously, with three QSwitches, all 24 lines can be accessed. In that case a 24-channel manual breakout box becomes obsolete.

Multiple QSwitches can be cascaded to be able to access more lines (Figure 2). They can also be coupled in parallel to access lines from multiple Fischer connector cables (Figure 3), for example to share instruments between multiple devices.

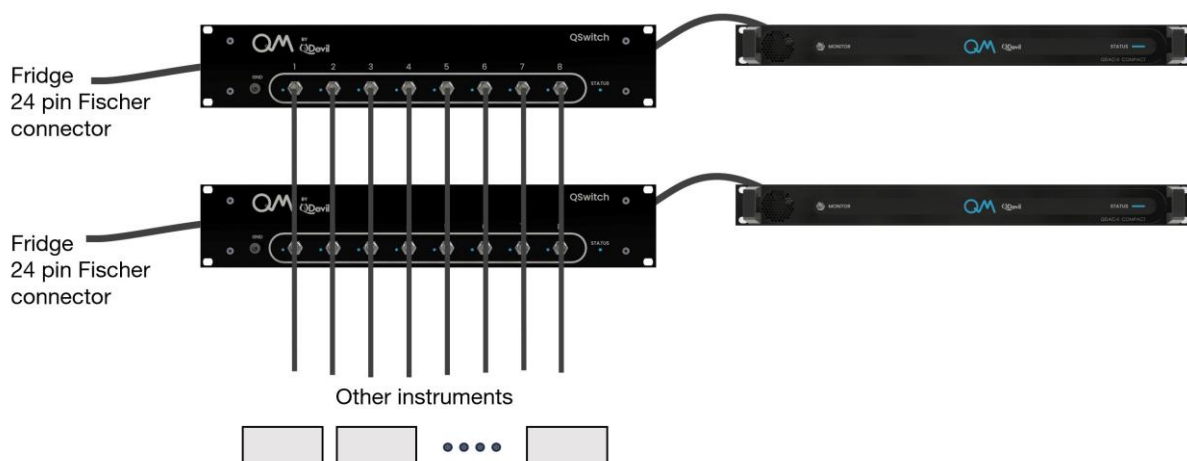


Figure 3. To share instruments between multiple devices or devices with more than 24 terminals, several QSwitches can be coupled in parallel by connecting the breakout channels on all units as shown.

To protect the DUT when connecting the QSwitch to the fridge, each signal line is at the output side by default grounded through a 1 M Ω resistor. The 1 M Ω resistors ensure that any charge on the device's

terminals is removed in a safe and well controlled way. In addition, grounding through 1 M Ω rather than 0 Ω reduces the risk of damaging a connected signal source (this would not damage the QDAC-II, though).

Should *hard* ground be desired, one can short circuit one of the breakouts to use as common ground.

3.2 Architecture and connections

Schematic / signal routing

Inside the QSwitch there are 10x24 relays which can be toggled individually. Eight of the relay groups are to break out to the BNC connectors. One group (!0) is for individual (soft) grounding of the device signal lines. The last group (!9) is to connect the instrument at the input (IN) to the internal signal lines.

No more than 40 breakout line relays (for the BNC connectors) can be in the closed state simultaneously in order not to overload on the power management circuits.

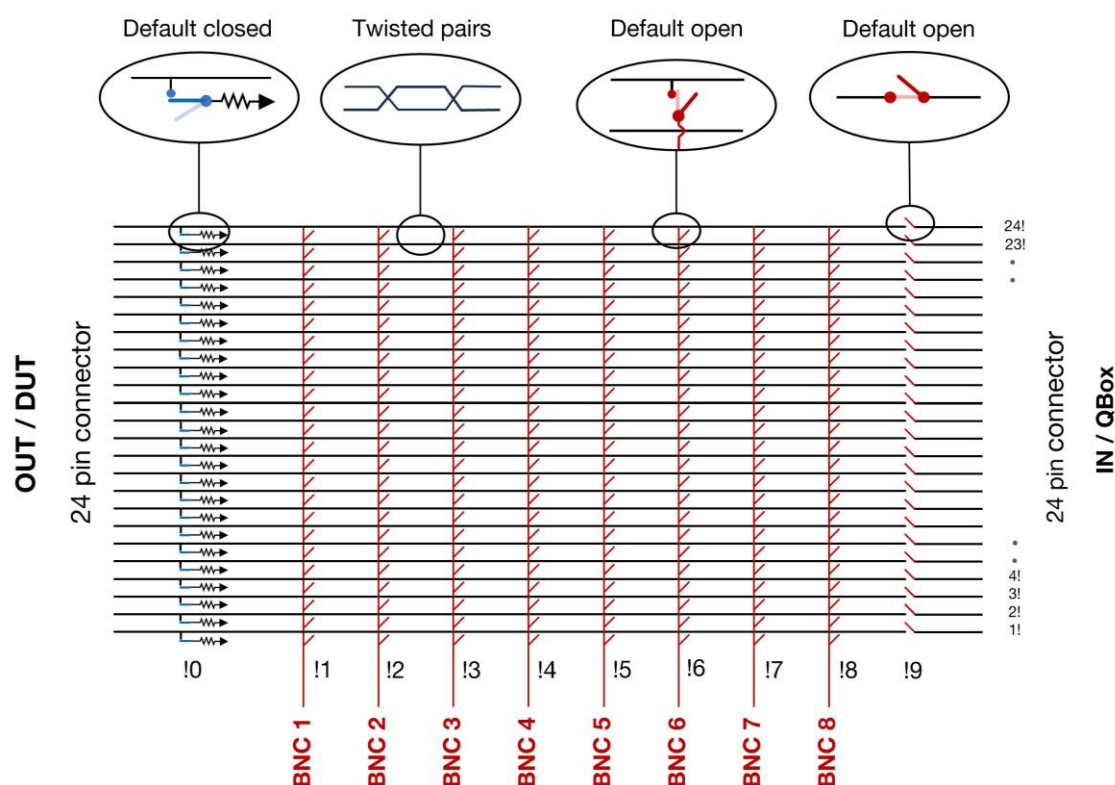


Figure 4. Principle schematic of the QSwitch. All switches (relays) are open by default, except for the 24 soft ground relays. “Default” means: when the power is off or just turned on (unless autosave/restore is enabled). The exclamation marks (!) after the signal line numbers and before the breakout line numbers refer to the syntax for addressing relays, e.g., “23!4” refers to the relay connecting signal line 23 to breakout line number 4 (BNC 4). See *SCPI “Channel list” syntax* in section 5.1

For optimum noise immunity the 24 signal lines are pairwise twisted like they are in typical fridge wiring (1-2, 3-4, 5-6, etc...) and in the QM Fischer connector cable.

The distance between neighboring breakout lines is about 35.5 mm and more for non-neighbors. This implies that if two BNC connectors are connected through an instrument to for example a pair of twisted signal lines, there is a certain area for electromagnetic pickup existing between the breakout lines; smallest when the breakouts are connected to signal lines close to the BNC connectors (1!, 2! ..), and highest when connected to signal lines with high numbers (23!, 24!, ..).

Front panel

The 8 BNC breakouts are on the front panel. They are standard BNC connectors with their shield connected to the common chassis ground. Each BNC connector has an LED indicating if it is connected to any of the 24 signal lines. There are no LEDs for the grounding and input relay groups.

The unit is intended to be rack-mounted using rack mount isolators to avoid ground loops in the setup. Grounding of the unit can be via the shield of any of the BNC lines or the shield of one of the two signal cables connected to the rear panel or via the ground banana socket (GND) on the front panel. The intended grounding scheme is via the shield of the 24-pin signal cable connecting the QSwitch to the fridge (device).



Figure 5. Front panel of the QSwitch.

Rear panel

To reduce cable clutter on the front of the rack in which the QSwitch is mounted, the 24-pin input and output connectors are located on the rear panel. Here, also the power connector and the USB and ethernet connectors are found. The power is supplied by an included low noise isolated 9V adapter. To avoid ground loops and additional noise, both the LAN and USB connectors are galvanically isolated from the signal ground.

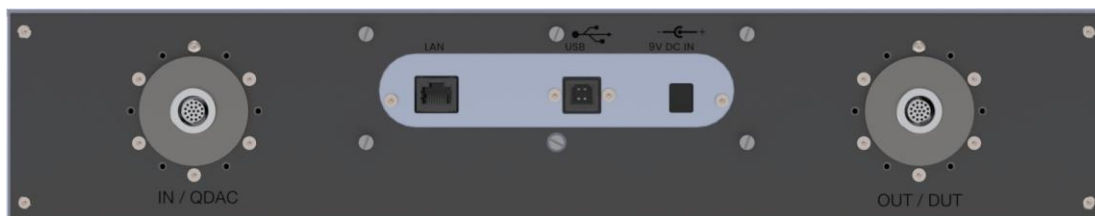


Figure 6. Rear panel of the QSwitch.

Left: 24-pin connector going to equipment like a QBox (typically wired to a voltage source, such as a QDAC-II).

Right: 24-pin connector for connecting to the fridge / device under test (DUT), or any cascaded QSwitch devices.

Middle: LAN and USB connectors connections for communication, and 9 V DC input from the included adapter.

3.3 Device safety features

When the QSwitch is unpowered, each signal line is connected to soft ground (through $1\text{ M}\Omega$) and disconnected from the input and all BNC breakout lines, so that it is safe to connect a device to the output (*OUT*).

In case of power loss, the relays connecting the signal lines to soft ground will close within a few milliseconds after the power loss has been detected, whereafter the relays connecting to the input side and to the breakouts will open within 200 ms in random order.

Transients

To avoid transients, it is recommended to open or close relays only when voltages and currents are zero. See section 4.3.

4 Getting started



Figure 7. Included in the shipping box: (1) QSwitch rack instrument, (2) Power supply, (3) USB cable, (4) Rack mount isolators.

4.1 Grounding the QSwitch

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

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

The QSwitch enclosure is intended to be grounded via the signal ground of the 24-way (Fischer) cable connecting the fridge to the OUT connector on the rear panel, or alternatively through the IN connector. The QSwitch can also function as a signal grounding star point if grounded through the GND banana socket on the front panel or via the shield of one of the BNC connectors. Note that the QSwitch has low signal voltage limits, see specifications. So, grounding is *not* for safety of persons but for integrity of signals and safety of connected sensitive devices.

It is very important to connect the QSwitch enclosure to ground. Otherwise, the soft-ground will be floating and the device not protected against common mode voltages.

4.2 Powering up the QSwitch

The instrument is intended to be installed in a normal laboratory environment.

There is no *on/off* switch on the QSwitch. It will power up when the enclosed power adapter is connected to the instrument and to mains. The 9V barrel plug is secured to the socket in the QSwitch by two small tabs which must be aligned with the socket before pushing in the plug. Note that the tabs are placed

asymmetrically meaning that there is only one correct orientation of the barrel plug. When pushed-in, the plug should be turned about 15 degrees clockwise to lock.

Be careful not to pull out the mains cable from the wall or from the adapter plug during operation. Make sure that the plugs are inserted firmly.
Use only the included adapter, otherwise the instrument may not meet specifications and regulatory requirements.

4.3 Connecting the QSwitch to a device or fridge

When the QSwitch is off, all relays are open, except for the soft-ground relays, which will connect the device side (DUT/OUT) to ground through 1 M Ω resistors. This state is un-altered after powering up the QSwitch, unless AUTosave is enabled, see section 5.3 *Autosave and restart*. So, it is safe to connect your device both in the OFF state and after powering ON. Soft grounding ensures that any charge on the terminals of the device, e.g. gate electrodes on a quantum dot device, is slowly conducted to ground.

However, if the instrument has been in use, it is advised either to reset the instrument (*RST command) or power cycle it before connecting it to the device. Note that *RST (reset) will not reload an autosaved state but put all relays in their default position (see schematic in [Figure 4](#)).

Always return signal voltages to 0 V before switching in or switching out devices or instruments, as the combined inductance and capacitance inside the QSwitch and connected cables may cause unwanted transients.

Because of the finite mutual capacitance between signal lines, especially neighbors, one can observe small transients even on passive lines, if voltages are coupled in or out on other lines.

4.4 Communication setup

The QSwitch is controlled through its USB/serial interface or via ethernet (LAN) – or both.

For first time use (setting up LAN etc.) and for updating the firmware it is necessary to use the USB port.

The command set of the QSwitch is a sub-set of the commands described in the SCPI standard (SCPI Consortium, 1999). See section 5.1 about SCPI command syntax, and limitations.

Messages to the instrument should always end with either a newline character <nl> (0x0A) – also known as line feed character (SCPI standard) or a carriage return <cr> (0x0D) character.

Note that a single QSwitch command line must *not* be longer than 127 characters.
Please allow a delay between sub-sequent commands of minimum 75 milliseconds, except for the short *OPC? query, which is allowed immediately after commands.
Otherwise, an error may be generated, and the second command skipped. Further, query command responses may come out of sync.

4.4.1 USB/serial setup

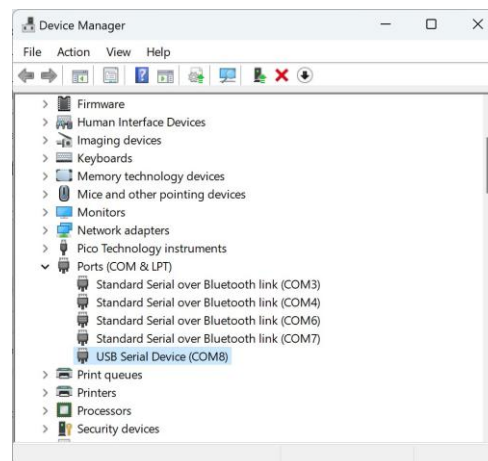
Communication through the USB port works through a virtual serial port over the USB interface, as there is a USB-to-serial adapter inside the instrument (opto-coupled). Since the USB connection is galvanically isolated from the circuits inside the instrument, no transients in the output voltages will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

When communicating with the QSwitch it is necessary to specify the correct serial port to the software. See how to find the correct port number or name below. 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	9600
Number of bits	8
Parity	None
Stop bits	1
Flow control	None

Windows

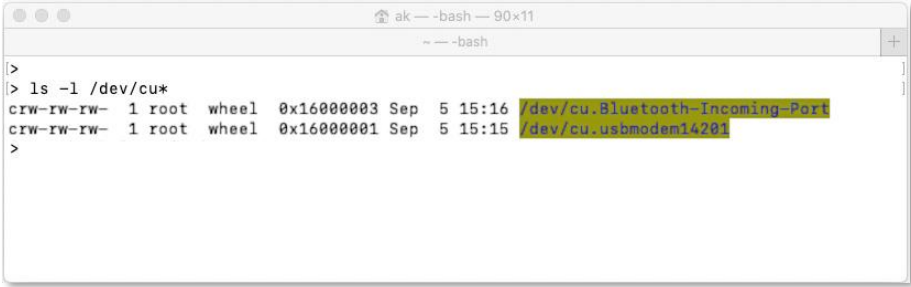
On Windows systems the port can be found by going to the Device Manager and looking under Ports (COM & LPT). In the example below it is seen that USB serial port is found at "COM8".



The driver is usually pre-installed, or Windows will download the driver when the QSwitch USB cable is plugged into the PC. If this does not happen the driver can be downloaded from Microchip:
<https://www.microchip.com/en-us/product/MCP2221>.

MAC OS and Linux

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

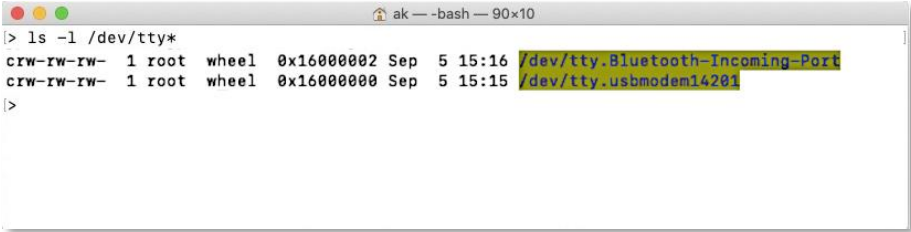


```

>
> ls -l /dev/cu*
crw-rw-rw-  1 root  wheel  0x16000003 Sep  5 15:16 /dev/cu.Bluetooth-Incoming-Port
crw-rw-rw-  1 root  wheel  0x16000001 Sep  5 15:15 /dev/cu.usbmodem14201
>

```

Alternatively, the instrument may be found as “tty*” device:



```

>
> ls -l /dev/tty*
crw-rw-rw-  1 root  wheel  0x16000002 Sep  5 15:16 /dev/tty.Bluetooth-Incoming-Port
crw-rw-rw-  1 root  wheel  0x16000000 Sep  5 15:15 /dev/tty.usbmodem14201
>

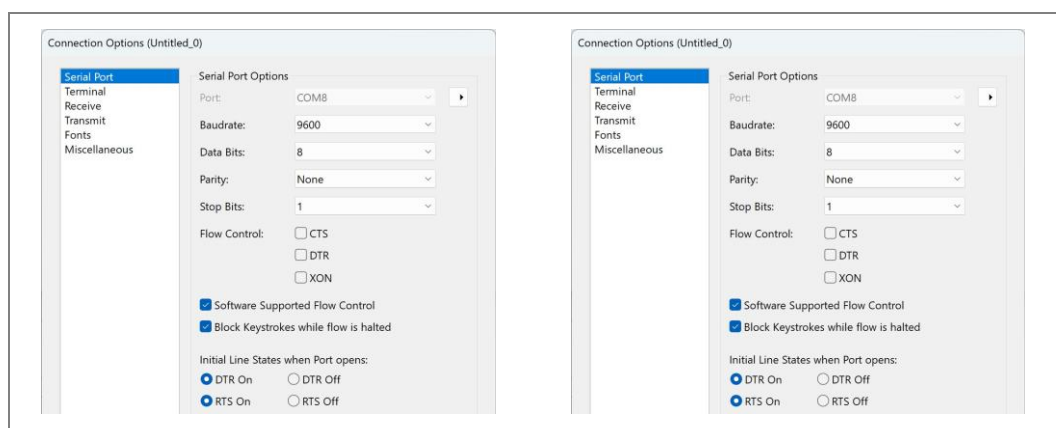
```

If the usb-serial device does not appear in the list, it may be necessary to install the device driver, see [Appendix B](#).

Control using a terminal program via USB

An example of a terminal program which runs on both Windows, Mac, and Linux is CoolTerm (<https://freeware.the-meiers.org/>). The screen shots below is from CoolTerm on a Windows PC.

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



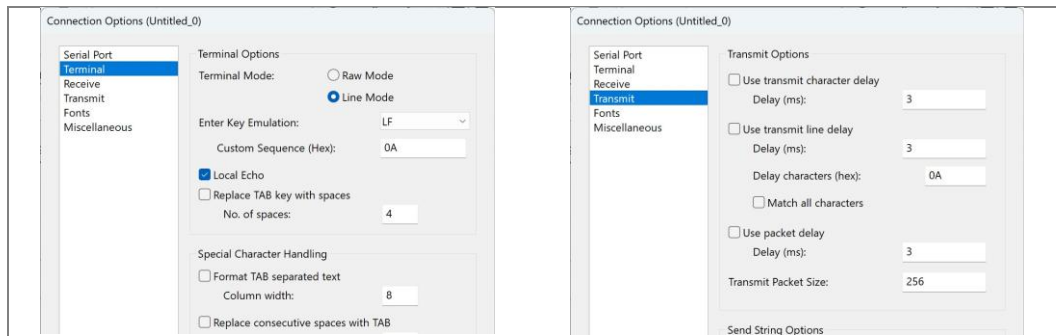


Figure 8. Connection options to be used with CoolTerm or any other serial communications program.

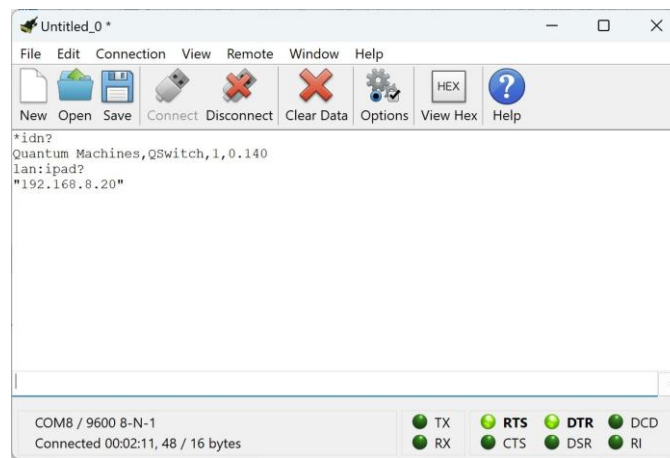


Figure 9. Example of sending a few commands via CoolTerm on a Mac.

4.4.2 LAN setup

From the 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 the LAN settings. To use the specified (STATic) settings, it is required to set DHCP to OFF.

Note that, when setting DHCP back ON, the IP address (LAN:IPAD) should be set to "0.0.0.0" – otherwise DHCP will not work.

Also note, if the instrument does not have a permanent entry in the DHCP server's map of clients (static DHCP), the retrieved IP address may change between power-on events or restart.

Changes to LAN settings are automatically stored in non-volatile memory. However, to effectuate changes, the instrument needs to be restarted either by power cycling or by sending the SYSTem:REStart command to the instrument.

Command	Description
[[SYSTem:]COMMunicate:]	Node (optional)
LAN:DHCP[?] [<Boolean>]	Determines whether automatic LAN configuration is on or off (default ON)
LAN:IPADdress[?] [<IP address in quotes>]	Sets or queries the instrument's IP address.
LAN:HOSTname[?] [<name in quotes>]	Sets or queries the instrument's LAN name
LAN:GATeway[?] [<IP address of gateway in quotes>]	Sets or queries the IP address of the gateway (router).
LAN:SMASK[?] [<sub-net mask>]	Sets or queries the sub-net mask, in CIDR format.
LAN:MAC?	Queries MAC address of the instrument.

Please see section 5.1 for a brief introduction to the SCPI protocol, and the Command Reference section 6.3.1 for the details of the above commands.

Example – sets the instrument static IP address and queries the current IP address:

```
>LAN:IPAD "192.168.14.178"      -Sets the static IP address
>LAN:IPAD? CURRENT              -Queries the current IP address.
"192.168.14.170"
```

Connecting via the LAN / ethernet port (UDP)

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 6.3.1). 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 QSwitch uses UDP communication instead of TCP/IP communication to ensure a stable connection (implemented in firmware version 2.0 and above). UDP works without handshaking. To make sure that no relay (ROUTE) commands are lost it is advised to check that the commands have been well received and executed. A wrapper and drivers for the QSwitch that include automatic checks to ensure the communication succeeded are available on the documentation page (<https://docs.quantum-machines.co/latest/docs/Hardware/QSwitch/>). In fact, in this manner the UDP protocol is more tolerant against short congestions and other network interruptions than the usual TCP/IP protocol.

Note, a consequence of using the UDP communication protocol is that the VISA library cannot be used for ethernet communication to the QSwitch, as VISA does not support UDP. The USB/serial interface is still supported by VISA.

4.4.3 Python

The instrument is designed to be controlled by a high-level instrument control program. A driver is available for QCoDeS, which is Python based. However, 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 UDP ethernet connection using the *socket* package. In many laboratory environments instrument communication relies on using the VISA library, which is available for the USB connection (UDP is not supported by VISA). To do so the *pyvisa* package needs to be installed and imported, and either the NI VISA backend from National Instruments has to be installed or the *pyvisa-py* backend from the *pyvisa* consortium.

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

```
import pyvisa as visa
class QSwitch():
    def __init__(self, visa_addr="ASRL15:INSTR", lib=''):
        rm = visa.ResourceManager(lib) # To use pyvisa-py backend, use argument '@py'
        self._visa = rm.open_resource(visa_addr)
        self._visa.write_termination = '\n'
        self._visa.read_termination = '\n'
        self._visa.baud_rate = 9600
        self._visa.send_end = False
    def query(self, cmd):
        return self._visa.query(cmd)
    def write(self, cmd):
        self._visa.write(cmd)
    def close(self):
        self._visa.close()
    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_usb = QSwitch(visa_addr = "ASRL15::INSTR", lib = '')
# To use pyvisa-py backend, use argument lib='@py'
```

```
import time
print(q_usb.query('*IDN?'))
print(q_usb.query("sys:err:all?"))
```

```
Quantum Machines,QSwitch,123,0.140
0, "No error"
```

```
q_usb.close()
```

A simple Python program to connect via the UDP ethernet connection, may look like this, here using a Jupyter Notebook. *Note that this is without any handshaking, see above:*

```
import socket
import time
class QSwitch():
    def __init__(self, ip_addr="192.168.8.20"):
        self._sock = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
        self._sock.settimeout(2)
        self._ip = ip_addr
    def query(self, cmd):
        self._sock.sendto(f"{cmd}\n".encode(), (self._ip, 5025))
        time.sleep(0.01)
        data, _ = self._sock.recvfrom(1024)
        answer = data.decode().strip()
        return answer
    def write(self, cmd):
        self._sock.sendto(f"{cmd}\n".encode(), (self._ip, 5025))
    def close(self):
        self._sock.close()
    def __exit__(self):
        self.close()
```

```
q_lan = QSwitch(ip_addr = "192.168.8.20")
```

```
print(q_lan.query('*IDN?'))
print(q_lan.query("sys:err:all?"))
```

```
Quantum Machines,QSwitch,123,0.140
0, "No error"
```

```
q_lan.close()
```

In order to include *handshaking* for the UDP connection, every essential write command (especially for closing and opening relays) should be confirmed with a query command. Please find the links to a wrapper and driver on <https://docs.quantum-machines.co/latest/docs/Hardware/QSwitch/> that have this functionality included.

The following example shows how to import and use the wrapper with handshaking for the UDP connection:

```
from qswitch import QSwitch, Udpconfig
from common import connection

qswitch_addr = '192.168.8.101'
device = UdpConfig(ip = qswitch_addr)
qswitch = QSwitch(device)
print(qswitch.query("*IDN?"))
```

QCoDeS (Python)

QM has developed a QCoDeS driver for the QSwitch. It is included in the Qcodes_contrib_drivers package in the official QCoDeS repository on GitHub, and can be installed as a normal package in Python. You can find a link to the latest QCoDeS driver on the QSwitch documentation website:

<https://docs.quantum-machines.co/latest/docs/Hardware/QSwitch/>

5 Functionality and operation

The QSwitch supports a small sub-set of the SCPI standard (SCPI Consortium, 1999) only including commands strictly necessary for the intended use, thus making it very easy to program the instrument. The instrument does not live up to all requirements of the SCPI standard.

5.1 Quick introduction to SCPI

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 is called a *sub-system*. Below is a list of the top-level keywords – or *sub-systems* – for the QSwitch:

Command group (sub-system)	Description
*xxx	xxx represents one of the IEEE488.2 mandatory commands. Just *RST and *IDN? Are implemented in QSwitch.
ROUTe	Controls the voltage outputs.
SYSTem	Error read out, non-measurement related settings (e.g. LAN settings), synchronization.

SCPI syntax in short

Except for the special command headers starting with an asterisk (*), most *instrument control header mnemonics* have a long form and a short form which can be mixed, see (SCPI Consortium, 1999) section 6.1.2. For example `syst:comm:lan:ipad?` for querying the instrument’s IP address can also be written as `system:comm:lan:ipaddress?`.

In addition, some mnemonics are optional. Therefore, commands can often be shortened significantly, for example `system:communicate:lan:ipaddress?` can for the QSwitch be shortened down to: `lan:ipad?`.

In the Command Reference (section 6), the short form of the mnemonics (keywords) are written in uppercase though the instrument does not distinguish between uppercase and lowercase characters. Optional keywords are surrounded by square brackets in the documentation.

Note that this instrument does not support compound (semicolon separated) commands, e.g. `syst:comm:lan:ipad?;hostname?;*rst`

Command syntax

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

Query syntax

All queries end with a question mark. Some queries take one or more parameters, but most don't.

Level1:level2:level3? parameter1, parameter2,...

SCPI "Channel list" syntax

Relays are addressed using the so-called *channel list* syntax. Channel lists start with "(" and ends with ")". Inside the parentheses individual relays are addressed as "x!y", where x designates the *signal line* number (1-24) and y designates the *breakout line* number (0-9, where 0 and 9 designate the grounding relays and the input relays, respectively).

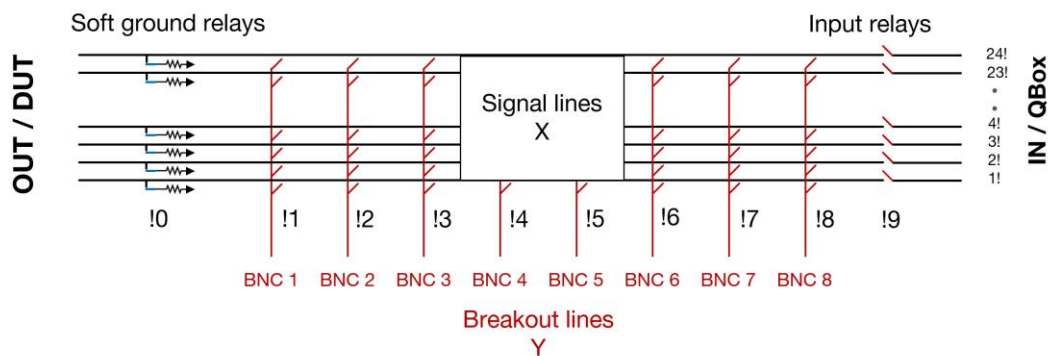


Figure 10. When addressing the relays/switches using channel list notation, one provides the signal line number followed by the breakout line number separated by an exclamation mark. The soft ground relays are addressed as breakout line number "0", and the input relays as breakout line number "9". Example: "19!2" will address the relay connecting signal line 19 and breakout line number 2.

Intervals (ranges) of relays can more compactly be addressed in *range notation* by "x_{n1}!y:x_{n2}!y". Only the signal line relays "x" support ranges.

Channel list syntax	Description
(@!1!3,5!3)	Comma separated. Addresses multiple relays either individually or by range notation.
(@!1!0:24!0)	Range notation for addressing multiple relays with consecutive numbers.

Example – ask the instrument any errors messages and open all soft grounding relays

```
>err:all?
0,"No error"
>open (@!1!0:24!0)
```

5.2 Opening and closing relays

There are three commands for opening and closing relays: OPEN, CLOSe and *RST. Though *RST is an IEEE 488.1 common command, effectively it is a “set all relays to default position” command. Likewise, there is a single command which will query the state of all relays: CLOSe:STATe?

Command	Description
[R OUTe:]	Node (optional)
CLOSe[?] <channel_list>	Closes the specified relays or queries their close status (0 = open, 1 = closed.).
OPEN[?] <channel_list>	Opens the specified relays or queries their open status (1 = open, 0 = closed.).
[CLOSe:]STATe?	Returns a list of all closed relays in channel list notation.
*RST	Sets all relays to their default positions (see Figure 4), meaning that all soft grounding relays will close, and all other relays will open (IEEE 488.2 common command).

Examples of using the open and close commands

```
>c!close (@12!3,8!4)           # Connect signal line 12 to BNC 3 and signal line 8 to BNC 4.
>c!close (@1!9:24!9)           # Connect all 24 signal lines to the input.
>open (@1!0:24!0)              # Unground all 24 signal lines.
>c!os:stat?                     # Returns all closed relays.
(@1!9:24!9,12!3,8!4)
```

Timing

All QSwitch commands are sequential commands, meaning that every command is fully executed before the next command in queue is processed.

The USB/Serial communication speed of 9600 baud *and the required minimum 75 ms delay between transmitted command*, see [section 4.4](#), set the limit for how fast commands can be executed.

The OPEN, CLOSe, or *RST commands take up to around 25 ms to execute due to the low-noise internal digital communication and the actual electro-mechanical switching of the relays. This applies no matter how many relays are addressed in one command. The order of switching is not defined. Some relays may switch already within 5 ms and the last ones close after up to 25 ms.

If AUTosave is enabled there will be an additional delay related to OPEN and CLOSe for writing the configuration to non-volatile memory. This delay is of the order of 45 ms. So, ~70 ms in total internally.

Both because of the limited input buffer length of 127 bytes of the instrument and because of these timing constraints it is recommended to use range notation whenever possible, for example when opening all *grounding relays* or closing all *through relays*.

Command synchronization (*OPC?)

To make sure that an `OPEN`, `CLOSE` or `*RST` command – or `RESTART` – has completed, one must wait at least 75 ms before issuing the next command.

However, it is recommended instead to perform an `*OPC?` query, as this command will respond with a “1” when the previous command is fully executed. This short command is allowed to immediately follow a command, irrespective of the recommended minimum delay of 75 ms.

As queries block communication until the reply is returned to the computer, a delay (or `*OPC?`) is not required after queries.

5.3 Autosave and restart

The QSwitch features autosave and auto recovery of relay states. When autosave is enabled the total relay configuration will be saved to non-volatile memory every time one or more relays are switched. In case of a power failure, the relays will go to their default states in the order described in 3.3 *Device safety features*. However, if AUTosave is ON, the relays will thereafter immediately go back to their positions before power was lost.

Relay states are also persisted the moment AUTosave is enabled. Should it be required to restart the instrument, for example to instate changed communication settings, this can be done without changing the relay states by first enabling AUTosave. After REStart one can disable AUTosave again.

Note that *reset* (*RST) and REStart differ in the way that *RST does *not* restart the firmware, it only resets the relays to their default (factory) positions, while at the same time disabling AUTosave. REStart on the other hand restarts the firmware and in case AUTosave is ON effectively leaves the relay states unchanged. If AUTosave is OFF, REStart will reset the relays to their default (factory) positions.

Command	Description
[SYSTem:]	Node (optional)
AUTosave[?] [<Boolean>]	Specifies or queries if autosave is on (1) or off (0). If AUTosave is ON, then the on/off state of all relays will be recovered upon power-cycling, and will be unaltered during REStart, unlike *RST (reset) which will disable AUTosave and set all relays to their default (un-pulled) states.
REStart	Restarts the firmware including the LAN interface (effectuating any changes to the LAN settings), and resets the device to its power up condition. If AUTosave is enabled, the states of all relays will be unaltered.

Examples of using AUTosave and REStart

```

>*rst                                # Resets all relay positions to their default positions
>*opc?                               # Wait until the instrument is ready to receive the next command
1
>close (@1!9:24!9)                   # Close all 24 input relays.
>*opc?
1
>open (@1!0:24!0)                     # Open all soft grounding relays
>*opc?
1
>close (@12!3,8!4)                   # Connect signal line 12 to BNC 3 and signal line 8 to BNC 4
>*opc?
1
>autosave on                         # Turn autosave ON and save the current relay states
>*opc?
1
>restart                             # Restart the firmware
>*opc?
1
>close:state?                        # Check that all relays are still at their recent positions
(1!9:24!9,12!3,8!4)

```

5.4 Error reporting

Error messages follow the SCPI numbering with 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.

For simplicity, the QSwitch has just a single error query command, and does not feature any of the status and event registers which are usually a part of SCPI and IEEE 488.2.

Command	Description
[SYSTEM:]	Node (optional)
ERRor:ALL?	Returns list of all errors (or 0,"No error"), thus resetting the queue

Example – ask for the list of errors after sending an illegal command

```
>b1ab1a
>SYST:ERR:ALL?
-113,"Undefined header"
>SYST:ERR:ALL?
0,"No error"
```

The error queue is emptied every time the `system:error:all?` query is performed.

Below is a list of all possible errors. Note that the error queue buffer has a finite length. Therefore, there is also an error for error queue overflow.

Error number and description	Meaning of error
-108,"Parameter not allowed"	A command with an illegal number of parameters has been received by the instrument.
-109,"Missing parameter"	A command is missing a parameter.
-110,"Command header error"	Something is wrong with a received command.
-113,"Undefined header"	An unknown (e.g. misspelled) command keyword has been received.
-120,"Numeric data error"	A number provided as parameter (e.g. relay number) is out of range or wrong format.
-151,"Invalid string data"	A parameter in the form of a quoted string is wrong in some way. For example, an IP address containing letters or a too long LAN hostname.
-200,"Execution error"	Internal firmware error
-240,"Hardware error"	A hardware error has been detected. Rare.
-240,"Hardware error; USB framing"	Internal firmware communication interface error. Rare.
-240,"Hardware error; USB overrun"	Internal firmware communication interface error. Rare.

-240,"Hardware error; USB general"	Internal firmware communication interface error. Rare.
-310,"System error; out of memory"	The internal controller has run out of memory. Rare.
-310,"System error; internal software error"	Something went wrong inside the firmware. Rare.
-350,"Error queue overflow"	The error queue is full.

Table 1. List of errors and their meanings. The ones in grey ink are not likely to ever occur.

Buzzer and LED

When an error occurs, an error message is appended to the error queue. In addition, if enabled by the user, the internal buzzer will produce a single beep and the front (power on) LED will start flashing. When the error queue has been emptied, the LED will return to its steady state.

In order to reduce electronic and electro-acoustic noise potentially coupling to the signal lines, the beeper and LED flashing pattern is disabled by default and needs to be enabled to produce the described behavior.

LED blink pattern (approx.)	State / error
	Instrument is on and ready
	SCPI error message available (requires BEEP:STAT ON)

6 Command Reference

6.1 IEEE Common commands

Only selected IEEE 488.2 common (“mandatory”) commands are implemented.

*IDN?

Description	Identification query.
Syntax	*IDN?
Arguments	None
Query response	Returns the product name, model number, serial number, and firmware version.
Example(s)	<code>>*IDN?</code> <code>Quantum Machines,QSwitch,123,1.6</code>
Notes	

*OPC?

Description	Operation Complete Query. Use this query to wait for completion of previous commands (software synchronization).
Syntax	*OPC?
Arguments	None
Query response	Returns “1” when operations (preceeding command(s) are complete.
Example(s)	<code>>close (@12!3)</code> - close some relays <code>>*opc?</code> - Wait until preceeding command is complete <code>1</code>
Notes	The *OPC? query is a blocking command as it waits for pending operations to be completed. The computer, however, will not wait longer than the timeout set in the communication protocol (e.g. VISA), which is typically 2 seconds. Usually not a problem for QSwitch. Just beware not to set the timeout too low (see section 5.2).

*RST

Description	Reset command. Bring the QSwitch into the power on state with all relays at their default positions. It also switches AUTOSave OFF.
Syntax	*RST
Arguments	None
Query response	None. Event only.
Example(s)	>*RST
Notes	This command does not <i>restart</i> device. This means that an active LAN connection is kept alive and that all LAN settings are unaltered, even though they may have been set to new values by the user. For this the REStart command should be used.

*TST?

Description	Runs a self-test and reports if any hardware problem has been detected.
Syntax	*TST?
Arguments	None
Query response	0 No errors 1 A problem has been detected.
Example(s)	>*TST? 0
Notes	Checks that internal communication between the switch matrix board and the micro-controller works. Note that relay switching is not tested.

6.2 Signal routing

All relays are controlled using the same ROUTe:{OPEN|CLOSE} commands.

The syntax for selecting a specific relay is:

`<signal line no.>!<breakout line no.>`

To select a range of relays [$n1:n2$] the syntax is:

`<signal line no. n1>!<breakout line no. >:<signal line no. n2>!<breakout line no.>`

where

`<signal line no.>` = 1-24 specifies Fischer connector pin number 1-24
`<breakout line no.>` = 1-8 specifies the BNC connector number 1-8 with two special cases:
`<breakout line no.>` = 0 designates the soft grounding relays (default closed)
`<breakout line no.>` = 9 designates the input relays

Note that the soft-ground “default closed” relays are controlled the same as all the other relays which are “default open”. Their default state is just different.

To reset all relays to their default state (not AUTosave state), use the *RST command.

ROUTe:CLOSE

Description	Closes one or more specified relay(s) establishing connection between signal lines and breakout lines, soft ground relays and the input connector.	
Syntax	[ROUTe:]CLOSE[?] [<channel_list>]	
Arguments	<channel_list>	(@<range_1>,<range_2> ...) Where range_n specifies a single relay x!y or an interval of signal line relays x:[$n1:n2$]: $x_{n1}!y:x_{n2}!y$ Where x = signal lines, y = breakout lines (incl ground an input), and $x_n = 1..24$, $y = 0..9$
Query response	Returns the close-state of the specified relay(s): 0 = open, 1 = closed.	
Example(s)	>close (@12!3) - connect signal line #12 to BNC 3. >close (@1!9:24!9) - connect all 24 inputs to the signal lines. >close? (@12!3) - queries the close-state of the line #12 to BNC#3 relay. 1	
Notes	Note that the order of closing the relays is not specified and not guaranteed to be as in the channel list. Note that only the signal line relays “x” can be specified as an interval, whereas the breakout relays “y” cannot.	

ROUTe:CLOSe:STATe?

Description	Query only. Returns a list of all closed relays.	
Syntax	[[ROUTe:]CLOSe:]STATe?	
Arguments	None	
Query response	Returns a list of all closed relays in SCPI channel list notation.	
Example(s)	<code>>close:stat?</code> - When the command is executed after power up or reset (*rst) it will return a list of all the closed soft-ground relays (assuming AUTosave off). <code>(@1!0:24!0)</code>	
Notes	To minimize the number of bytes to be transferred, the response is compacted using channel list notation. A snippet of Python code is available for "unpacking" such a reply to a more code friendly format.	

ROUTe:OPEN

Description	Opens one or more specified relay(s) disconnecting signal lines from breakout lines, soft ground relays or the input.	
Syntax	[ROUTe:]OPEN[?] <channel_list>	
Arguments	<channel_list>	See description for CLOSe
Query response	Returns the open-state of the specified relay(s): 1 = open, 0 = closed.	
Example(s)	<code>>open (@12!3)</code> - disconnect signal line #12 to BNC 3. <code>>open (@1!9:24!9)</code> - disconnect all 24 inputs from the signal lines.	
Notes	Note that the order of opening the relays is not specified and not guaranteed to be as in the channel list.	

6.3 System commands

System commands are related to general instrument configuration and reading the error message queue.

6.3.1 Communication commands

SYSTem:COMMunicate:LAN:DHCP

Description	Enables, disables or queries the instrument's use of DHCP (Dynamic Host Configuration Protocol) for getting its IP address, subnet mask and default gateway.	
Syntax	[[SYSTem:]COMMunicate:]LAN:DHCP {OFF 0 ON 1} [[SYSTem:]COMMunicate:]LAN:DHCP[?] [{CURRENT STATIC}]	
Arguments	ON 1:	The instrument tries to obtain an IP address from a DHCP server at power-on or REStart (default).
	OFF 0:	The instrument uses the static IP address, Subnet Mask, and Default Gateway. The same happens in the absence of a DHCP server.
	[{CURRENT STATIC}]: Optional argument. If present, it determines if it is the stored setting or the actual state in use (current) which is queried. CURRENT is default.	
Query response	Returns the current or saved (STATic) setting, 0 (OFF) or 1 (ON).	
Example(s)	<pre>>SYST:COMM:LAN:DHCP ON - Enables DHCP. >SYST:REST >SYST:COMM:LAN:DHCP? - Queries the current DHCP state. 1</pre>	
Notes	Changes to LAN settings are automatically saved in non-volatile memory. To effectuate a new setting, the device needs to be restarted (system:restart) or power cycled. First time configuration of the LAN interface is often done via the USB interface, as the instrument's IP address may be unknown at this stage. Note, that if the instrument does not have a permanent entry in the DHCP server's map of clients (static DHCP), the retrieved IP address may change between power-on events or other events leading to new DHCP requests. When setting DHCP to ON, in order to actually work the IP address (SYSTem:COMMunicate:LAN:IPADdress) should be set to "0.0.0.0" . If no DHCP server is available or DHCP is OFF, the instrument will use the stored IP address, Subnet Mask, and Default Gateway.	

SYSTem:COMMunicate:LAN:HOSTname

Description	Sets or queries the host name of the instrument on the network. This is the name under which the instrument appears on the local network. The factory-set host name is identical to the serial number of the unit.	
Syntax	[[SYSTem:]COMMunicate:]LAN:HOSTname "<hostname>" [[SYSTem:]COMMunicate:]LAN:HOSTname[?] [{CURRent STATIC}]	
Arguments	"<hostname>"	Quoted string containing the new name, max. 16 characters.
	[{CURRent STATIC}]: Optional argument. If present, it determines if it is the stored name or the actual host name (current) in use which is queried. CURRent is default.	
Query response	Returns the host name as a quoted string.	
Example(s)	>SYST:COMM:LAN:HOST? - Queries the unit's LAN host name "qswitch-1"	
Notes	Changes are automatically saved in non-volatile memory. The hostname is the host portion (left most) of the domain name, which is translated into an IP address. When DHCP is activated, the hostname is registered with the Dynamic Domain Name System (Dynamic DNS) service at power-on or restart.	

SYSTem:COMMunicate:LAN:MAC?

Description	Queries the instrument's Media Access Control (MAC) address	
Syntax	[[SYSTem:]COMMunicate:]LAN:MAC?	
Arguments	None	
Query response	Returns the 12-character hexadecimal MAC address surrounded by quotes.	
Example(s)	>SYST:COMM:LAN:MAC? - Queries the instrument's MAC address. "70B3D5921000"	
Notes	The MAC address is printed in the rear panel of the instrument.	

SYSTem:COMMunicate:LAN:IPADdress

Description	Sets or queries the instrument's IP address.
Syntax	[[SYSTem:]COMMunicate:]LAN:IPADdress "<address>" [[SYSTem:]COMMunicate:]LAN:IPADdress? [{CURRENT STATIC}]
Arguments	"<address>": IP address surrounded by quotes. [{CURRENT STATIC}]: Optional argument. If present, it determines if it is the stored address or the actual address in use which is queried. CURRENT is default.
Query response	Returns either the stored (static) IP address or the actually used IP address (current), which may be different from the stored one if DHCP is ON.
Example(s)	>SYST:COMM:LAN:IPAD "192.168.14.178" -Sets the static IP address >SYST:COMM:LAN:IPAD? STATIC -Queries the static IP address. "192.168.14.178"
Notes	If DHCP is enabled (SYSTem:COMMunicate:LAN:DHCP ON), the (static) IP address should be set to "0.0.0.0", otherwise a dynamic IP address will not be retrieved. Changes are automatically saved in non-volatile memory. To effectuate a new setting the instrument needs to be restarted (system:restart) or power cycled.

SYSTem:COMMunicate:LAN:GATeway

Description	Sets the default gateway for the instrument.
Syntax	[[SYSTem:]COMMunicate:]LAN:GATeway "<address>" [[SYSTem:]COMMunicate:]LAN:GATeway? [{CURRENT STATIC}]
Arguments	"<address>": IP address of the gateway surrounded by quotes. [{CURRENT STATIC}]: Optional argument. If present, it determines if it is the stored gateway address or the current gateway in use which is queried. CURRENT is default.
Query response	Returns either the stored gateway (STATIC) or the actually used gateway (CURRENT) - which may be different from the stored one if DHCP is ON.
Example(s)	>SYST:COMM:LAN:GAT "192.168.1.1" -Sets the static default gateway. >SYST:COMM:LAN:GAT? STAT -Queries the static default gateway. "192.168.1.1"
Notes	Assigns a static default gateway for the instrument. If DHCP is enabled (SYSTem:COMMunicate:LAN:DHCP ON), the specified static gateway is not used. Changes are automatically saved in non-volatile memory. To effectuate a new setting the device needs to be restarted (system:restart) or power cycled.

SYSTem:COMMunicate:LAN:SMASk

Description	Sets or queries the instrument's subnet mask on the local network.	
Syntax	<pre>[[SYSTem:]COMMunicate:]LAN:SMASk <mask> [[SYSTem:]COMMunicate:]LAN:SMASk? [{CURRENT STATIC}]</pre>	
Arguments	<mask>:	Sub-net address in CIRD* format. It specifies the number of bits in a 32 bit mask which are not part of the subnet's IP addresses, e.g. 24 = 255.255.255.0, 25 = 255.255.255.128.
	{CURRENT STATIC}:	Optional argument. If present, it determines if it is the stored mask or the current mask in use which is queried. CURRENT is default.
Query response	Returns either the stored (static) subnet mask or the mask currently in use, which may be different from the stored one if DHCP is ON and active.	
Example(s)	<pre>>SYST:COMM:LAN:SMASK 24 LAN:SMASK? STAT 24</pre>	-Sets the static sub-net mask -Queries the static sub-net mask..
Notes	*CIRD = Classless Inter-Domain Routing If DHCP is enabled and successful (SYSTem:COMMunicate:LAN:DHCP ON), the specified static sub-net mask is not used. Changes are automatically saved in non-volatile memory. To effectuate a new setting the device needs to be restarted (system:restart) or power cycled.	

6.3.2 Error system

SYSTem:ERRor:ALL

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

6.3.3 Other commands

SYSTem:BEEPer:STATe

Description	Sets the state of the built-in error buzzer and LED error signal.	
Syntax	[SYSTem:]BEEPer:STATe {ON OFF 1 0}	
Arguments	{ON OFF 1 0}:	ON 1: The buzzer sounds when an error occurs and the LED starts flashing. OFF 0: The buzzer is silent when errors occur (default), and the front LED will not flash.
Query response	None	
Example(s)	>SYST:BEEP:STAT OFF	- Causes the buzzer not to sound when errors occur.
Notes	The internal buzzer is mainly intended as a help to the user when developing and testing out code. In "production mode" buzzing is rarely desired. Therefore, the buzzer is OFF by default. Another reason for having it OFF by default is to avoid any unnecessary electronic noise generation inside the unit. Therefore, also front LED will refrain from starting blinking when errors occur – unless the BEEPer state is ON.	

SYSTem:BEEPer:IMMediate

Description	Makes the built-in buzzer create a short sound.
Syntax	[SYSTem:]BEEPer[:IMMediate]
Arguments	None
Query response	No query version. Event only.
Example(s)	<code>>BEEP</code> - Makes the instrument produce a buzzing sound.
Notes	

SYSTem:AUTosave

Description	Controls if relay states are automatically saved in non-volatile memory. When autosave is enabled, the relay states are automatically re-established when the QSwitch is REStart'ed or powered on.
Syntax	[SYSTem:]AUTosave {ON OFF 0 1} [SYSTem:]AUTosave?
Arguments	{ON 1 OFF 0}: ON 1: AUTosave is enabled and the current state is saved. OFF 0: AUTosave is switched off.
Query response	Reports if autosave is enabled or off. 0 = OFF, 1 = ON.
Example(s)	<code>>SYST:AUT OFF</code> - Causes autosave of the QSwitch relay-state to be enabled. <code>>AUT?</code> - Queries the autosave state. <code>0</code>
Notes	Note, if AUTosave is enabled the execution time for OPEN and CLOSe command is increased with about 45 ms, which is the time it takes to write the relay states into non-volatile memory, see <i>Timing</i> in section 5.2.

SYSTem:REStart

Description	Restarts the firmware including the LAN interface (effectuating any changes), and resets the device to its power up condition*.
Syntax	[SYSTem:]REStart
Arguments	None
Query response	None
Example(s)	<code>>SYST:REST</code> <code>>rest</code>
Notes	*) If AUTosave is ON, then REStart will effectively not change the states of the relays as the last saved state before REStart will be read from non-volatile memory and all relays will be set accordingly – so no change.

7 Specifications and performance

Main unit

Number of signal lines	24
Signal line input relays	Default <i>off</i>
Number of breakout lines	8
Max number of closed breakout relays	40
Signal line grounding	Through 1 M Ω resistors. Default <i>on</i>
Wiring	Pair-wise twisted (1-2,3-4 ...)
Line to line isolation	≥ 10 G Ω (design)
Line to ground isolation	(to be determined)
Signal line resistance	Typical < 0.4 Ω , including input relay
Signal line pair mutual capacitance	10 / 25 / 42 pF (min. / typ. / max.)
Relay switching time	< 25 ms (< 70 ms when autosave is on)
Maximum signal current	100 mA
Maximum signal voltages	AC: 30 Vrms, 42 Vpp, DC: 60 V
Bandwidth	≥ 10 MHz (-3dB attenuation at 10 MHz)
Power failure behaviour	Signal lines are soft grounded before all other relays open
Power on condition	Signal lines are connected to soft ground and all other relays open. If autosave is enabled, the last switch configuration will be re-established within ~70 ms
Power requirement	DC 9 V, 2.5 A, stable, low noise, use enclosed adapter only.
Operating environment temperature	15 – 35 °C
Operating environment humidity	20 – 80% RH (non-condensing).
Operating altitude	Up to 3000 m
Dimensions (mm)	Width: 485 (19"), height: 88.9 (2U), 94 incl. feet, depth: 314
Weight (excl. cables)	4.7 kg

External power supply

Brand / Model	Mean Well GSM 25B09-P1J
Power requirement	100-240 VAC, 50/60 Hz, 0.7-0.35 A
Output voltage	9 V
Dimensions without cables (mm)	Width: 54, height: 33, depth: 79
Weight (incl. cables)	0.3 kg

EMC test summary

Tests according to EN 61326-1:2013/2021 “Electrical Equipment for measurement, control and laboratory use – EMC requirements – Part 1: General requirements”

Used basic standard	Phenomenon	Tested on
EN 55011:2009 +A1 2010	Radiated emission	Enclosure
EN 55011:2009 +A1 2010	Conducted emission	9V adapter Input AC power port
EN 61000-4-2: 2009	ESD immunity	Enclosure
EN 61000-4-3: 2006 +A1:2008 +A2:2010	Radiated RF immunity	Enclosure
EN 61000-4-4:2012	Conducted fast transients/Burst immunity	9V adapter Input AC power port, Control and Signal ports
EN 61000-4-5: 2006	Conducted surge transient immunity	9V adapter Input AC power port
EN 61000-4-6: 2009	Conducted RF immunity	9V adapter Input AC power port, Control and Signal ports
EN 61000-4-8: 2010	Power frequency magnetic field immunity	Enclosure
EN 61000-4-11: 2004	Voltage dips and interruptions immunity	9V adapter Input AC power port

The tests have been performed with wiring as in a real use case, with 47 activated relays while constantly monitoring that their states are unchanged and constantly communicating to and from the instrument.

Immunity Test for *Industrial Electromagnetic Environment* passed with Performance Criterion A, except AC port voltage dip to 0% for 5 seconds, Performance criterion C.

Radiated Emission tests passed as Group 1 Class B equipment

Conducted Emission test (AC power port) passed as Group 1 Class A equipment

8 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 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 Linux systems the installation executable must have its attribute changed to “executable” by using the “chmod +x” command. On Windows the operating system may need convincing that it is safe to run the executable.

The firmware updater requires a hardware reset to connect to the QSwitch. The hardware reset is performed by powering off the QSwitch (if not already powered off), and then powering it on. Some devices require a few power cycles before updating starts.

The firmware update program will last a few 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 connection on your computer, which is connected to the QSwitch, for example a terminal program or Python code.

Firmware update procedure:

1. Download the relevant updater (Windows, Linux). On Linux systems the installation executable must have its attribute changed to “executable” by using the “chmod +x” command. On Windows the operating system may need convincing that it is safe to run the executable.
2. Connect the QSwitch to the computer with a USB cable. Make sure only one QSwitch is connected to the computer.
3. Power down the QSwitch.
4. Start the updater.
5. Power up the QSwitch
6. If you within 3 seconds do not receive a “*** Connected to Qswitch ***” message, then power off, and then back on. If you are unplugging at the mains side of the power supply instead of unplugging the barrel connector on the QSwitch rear panel, then please wait 4 seconds before reconnecting mains power.
7. When the updating and verification has ended press <enter> – or sometimes the <space> bar.

The updater attempts to connect for up to 30–40 seconds and then gives up. If this happens, just repeat the procedure. If after several attempts updating is not successful, please contact the Quantum Machines Customer Success Team (<https://www.quantum-machines.co/contact-us/>).

Example – executing the firmware installer on Windows

```
> ./qswitch-fw-update.exe  
Updating QSwitch firmware to version 2.0...  
[13:18:31] Attempting to connect...  
[13:18:35] *** Failed to connect, please power the device OFF and then back ON ***  
[13:18:37] *** Failed to connect, please power the device OFF and then back ON ***  
[13:18:38] *** Connected to Qswitch ***  
[13:18:40] Update | ████████████████████████████████████████ | 975/975  
[13:20:17] Verify | ████████████████████████████████████████ | 975/975  
[13:20:41] Resetting device  
Update finished  
(Press Enter to close)  
Press any key to continue . . .>
```

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 as the firmware updater is blocking the connection.

Web address for firmware updates

Updates, when available, can be downloaded from this web site:

<https://docs.quantum-machines.co/latest/docs/Hardware/QSwitch/>

Appendix B Mac and Linux USB/serial driver information

The following is published by Microchip at <https://www.microchip.com/en-us/product/MCP2221>.

Appendix B1. Linux driver instructions

The following is published by Microchip at <https://www.microchip.com/en-us/product/MCP2221>:

MCP2200/MCP2221 - Linux Driver Instructions

In order to be able to use the MCP2200/MCP2221 with Linux, the kernel must have support for USB CDC class drivers.

The cdc_acm driver is used for the CDC interface of the device.

If the cdc_acm driver is compiled as a module, when plugging the MCP2200/MCP2221, this driver will be loaded by the kernel.

To verify that, type:

```
lsmod | grep cdc
```

or you can verify it like this:

```
dmesg | grep ttyACM
```

After that, there will be an entry in /dev

in order to find the entry type:

```
ls /dev/ttyACM*
```

If there are any issues, there will be no entries. But if everything goes well we should see something like:
/dev/ttyACM0

In order to be able to use your favourite terminal program, we will create a link between /dev/ttyACMx and a serial port.

To do that just type:

```
ln -sf /dev/ttyACM0 /dev/ttyS3
```

In some other systems, there might be a different number after "ttyACM".

You have to provide the number that

appears in your system. As for the "ttyS3", I had chosen this one since it was an unused serial device node.

Now, let's make some settings on the serial port. Use "stty" to set serial port parameters. E.g. :

```
stty -F /dev/ttyS3 115200
```

You can start now whatever serial port utility you want (minicom, gtkterm, ...)

Appendix B2. MAC OS driver instructions

The following is published by Microchip at <https://www.microchip.com/en-us/product/MCP2221>:

Important!

=====

This readme file applies only to MAC OS X version starting with 10.7 (incl.) and beyond.

Composite CDC + (any other interface) USB devices (such as the MCP2200 & MCP2221, which are composite CDC+HID devices) will only work on Mac OS X 10.7 (or later).

Mac OS X 10.7 is the first OS X version that supports USB Interface Association

Descriptors (IADs), which are needed when implementing composite USB devices with multiple interfaces, with at least one CDC-ACM function. Prior versions of Mac OS X did not support IADs, and therefore can only support non-composite, single function CDC-ACM devices.

Upon plugging in a USB CDC ACM virtual COM port device into a Mac OS X based machine, the OS should automatically enumerate the USB device successfully, and a new object should show up as:

`/dev/tty.usbmodemXXXX`

(where XXXX is some value, such as "fd1321")

To test the CDC functionality a procedure like follows can be used:

Open TERMINAL. This can be done by clicking SPOTLIGHT and searching for TERMINAL.

Spotlight is the little magnifying glass in the upper right of the screen.

In Terminal, with the USB CDC ACM device NOT plugged in (yet), type:

`ls /dev/tty.*`

This will show all serial devices currently connected to the Mac. In the author's case, the following list appears:

`/dev/tty.Bluetooth-Modem`

`/dev/tty.Bluetooth-Incoming-Port`

`/dev/tty.BluePortXP-115D-SPP`

Now, plug the MCP2200 or MCP2221 device into a USB port of the Mac. Hit the UP cursor, which will bring the search command back (`ls/dev/tty.*`) and hit return. You should get the exact same list as before, but this time, with a new serial device.

In the author's case, it was:

`/dev/tty.usbmodemfd1321`

Once the complete name is known, the received serial port data can be displayed by typing:

`screen /dev/tty.usbmodemfd1321`

(replace "fd1321" in the above line with the value for your machine).

The baud rate can be set by using syntax like follows:

`screen -U /dev/tty.usbmodemfd1321 115200`

where "usbmodemfd1321" should be replaced with the actual value of the device, and "115200" should be replaced with actual desired baud rate (ex: 9600, 19200, 38400, 57600, 115200, etc.).

More details and usage information for screen can be found in the man page of the "screen" utility.

A number of serial port utilities can be found on the App Store. "Serial Tools" is a good example of a free serial terminal application.