Note

You are not reading the most recent version of this documentation. 1.12.0 is the latest version available.

A more complex example

The following example shows how to use SCPI commands with a Keithley 2000 multimeter in order to measure 10 voltages. After having read them, the program calculates the average voltage and prints it on the screen.

I'll explain the program step-by-step. First, we have to initialise the instrument:

```
>>> keithley = rm.open_resource("GPIB::12")
>>> keithley.write("*rst; status:preset; *cls")
```

Here, we create the instrument variable *keithley*, which is used for all further operations on the instrument. Immediately after it, we send the initialisation and reset message to the instrument.

The next step is to write all the measurement parameters, in particular the interval time (500ms) and the number of readings (10) to the instrument. I won't explain it in detail. Have a look at an SCPI and/or Keithley 2000 manual.

```
>>> interval_in_ms = 500
>>> number_of_readings = 10
>>> keithley.write("status:measurement:enable 512; *sre 1")
>>> keithley.write("sample:count %d" % number_of_readings)
>>> keithley.write("trigger:source bus")
>>> keithley.write("trigger:delay %f" % (interval_in_ms / 1000.0))
>>> keithley.write("trace:points %d" % number_of_readings)
>>> keithley.write("trace:feed sense1; feed:control next")
```

Okay, now the instrument is prepared to do the measurement. The next three lines make the instrument waiting for a trigger pulse, trigger it, and wait until it sends a "service request":

```
>>> keithley.write("initiate")
>>> keithley.assert_trigger()
>>> keithley.wait_for_srq()
```

With sending the service request, the instrument tells us that the measurement has finished and that the results are ready for transmission. We could read them with *keithley.query("trace:data?")* however, then we'd get:

```
-000.0004E+0,-000.0005E+0,-000.0004E+0,-000.0007E+0,
-000.0000E+0,-000.0007E+0,-000.0008E+0,-000.0004E+0,
-000.0002E+0,-000.0005E+0
```

which we would have to convert to a Python list of numbers. Fortunately, the *query_ascii_values()* method does this work for us:

```
>>> voltages = keithley.query_ascii_values("trace:data?")
>>> print("Average voltage: ", sum(voltages) / len(voltages))
```

Finally, we should reset the instrument's data buffer and SRQ status register, so that it's ready for a new run. Again, this is explained in detail in the instrument's manual:

```
>>> keithley.query("status:measurement?")
>>> keithley.write("trace:clear; feed:control next")
```

That's it. 18 lines of lucid code. (Well, SCPI is awkward, but that's another story.)