



SPA Python Example

Getting Started with Serial Communication

software version: V26006

manual version: March 2026

language: English

Check for latest version of this manual here:

<http://electron.plus/pages/manuals>

1	Introduction	4
2	Requirements	4
3	Connecting the Instrument	4
4	Quick Start	5
5	Command-Line Options	5
5.1	Instrument Selection	6
5.2	Measurement Setup	6
5.3	Bias Source	6
5.4	Testing	6
6	Input Ranges	6
7	Common Examples	7
8	Stopping a Measurement	8
9	Calibration	8
9.1	How It Works	8
9.2	Automatic Calibration Handling	8
9.3	Manual Calibration File	8
10	Using the Script in Your Own Code	9
10.1	SPA Class Reference	9
10.1.1	Constructor	9
10.1.2	Methods	9
10.1.3	read_sample() Return Value	10
11	Troubleshooting	10
11.1	"No SPA instrument found"	10
11.2	"pyserial is not installed"	10
11.3	Overload readings ("** OVR **")	10
11.4	Noisy or unstable readings	10

12 Self-Test (Dry Run)	11
13 Support	11

1 Introduction

This guide accompanies the `SPA_python_example.py` script, which demonstrates how to communicate with an Electron Plus **SPA100** or **SPA120** Source Picoammeter from Python.

The script is a complete, working example. You can run it as-is to read calibrated current measurements, or use it as a starting point for your own measurement software.

Supported instruments:

Model	Channels	Description
SPA100	1	Single-channel picoammeter
SPA120	2	Dual-channel picoammeter

For the latest version of this script, visit: www.electron.plus

2 Requirements

- **Python 3.7** or later (download from python.org if you don't already have it)
- **pyserial** library – the only external dependency

Install pyserial from a terminal or command prompt:

```
pip install pyserial
```

No other libraries are needed. The script uses only Python standard library modules plus pyserial.

3 Connecting the Instrument

1. Connect the SPA to your computer with a USB cable.
2. Wait a few seconds for the CH340 driver to install (Windows usually does this automatically).
3. Note the COM port number that appears in Device Manager, or let the script auto-detect it.

NOTE: On Windows, the port will appear as COMx in Device Manager under "Ports (COM & LPT)". On Linux it will be `/dev/ttyUSBx` and on macOS `/dev/tty.wchusbserialxxxxx`.

4 Quick Start

Open a terminal (Command Prompt, PowerShell, or your preferred shell) and navigate to the folder containing the script.

Read 20 samples on range 5 (20 uA), auto-detect everything:

```
python SPA_python_example.py --range 5 --samples 20
```

The script will:

1. Find the SPA's COM port automatically
2. Load the calibration file (or download it from the instrument if no file exists)
3. Configure the range and sample rate
4. Print calibrated current readings to the console

Example output:

```
[spa] Auto-detected port: COM3
[cal] Loaded calibration from settings/SPA120_cal.JSON (device=SPA120, serial=3H7R1C)
[spa] Connected and streaming.
[spa] Ch1 range set to 20 uA (index 5)
[spa] Ch2 range set to 20 uA (index 5)
[spa] Sample rate set to 10 Hz
```

```
=====
More info: www.electron.plus
=====
```

```
SPA120, 20 samples at 10 Hz, avg 1x
Range: 20 uA
=====
```

#	Ch1 Current	Ch2 Current	Ch1 Avg	Ch2 Avg
1	+12.3456 uA	-0.0023 uA	+12.3456 uA	-0.0023 uA
2	+12.3401 uA	-0.0019 uA	+12.3401 uA	-0.0019 uA
...				

5 Command-Line Options

The full list of options is available with `--help`:

```
python SPA_python_example.py --help
```

Here is a summary of the most useful options:

5.1 Instrument Selection

Option	Description	Default
--device	SPA100 or SPA120 (auto-detected if omitted)	auto-detect
--port	COM port, e.g. COM3 or /dev/ttyUSB0	auto-detect
--cal	Path to calibration JSON file	auto-detect

5.2 Measurement Setup

Option	Description	Default
--range	Ch1 input range, 0-7 (see table below)	5 (20 uA)
--range2	Ch2 input range, SPA120 only	same as --range
--rate	Sample rate: 2, 10, or 100 Hz	10
--avg	Rolling average depth, 1-64	1
--samples	Number of samples (0 = continuous)	20

5.3 Bias Source

Option	Description	Default
--bias	Bias voltage, 0-40 V (0 = off)	0
--polarity	Positive or Negative	Positive

5.4 Testing

Option	Description
--dry-run	Run self-test without hardware connected

6 Input Ranges

The SPA has 8 input ranges, selected by index 0-7:

Index	Full Scale	Best For
0	200 pA	Very small currents (sub-nanoamp)
1	2 nA	Low picoamp-to-nanoamp signals
2	20 nA	Nanoamp-level currents
3	200 nA	Hundreds of nanoamps
4	2 uA	Low microamp signals
5	20 uA	General-purpose (default)
6	200 uA	Hundreds of microamps
7	2 mA	Milliamp-level currents

Lower index numbers give finer resolution but a smaller measurable range. Higher index numbers measure larger currents but with less resolution.

On the SPA120, you can set each channel to a different range using `--range` (for Ch1) and `--range2` (for Ch2).

7 Common Examples

SPA100, single channel, range 4 (2 uA), 50 samples:

```
python SPA_python_example.py --device SPA100 --range 4 --samples 50
```

SPA120, different ranges on each channel:

```
python SPA_python_example.py --range 5 --range2 3 --samples 100
```

Run continuously until you press Ctrl+C:

```
python SPA_python_example.py --range 5 --samples 0
```

Fast sampling at 100 Hz with 8x averaging:

```
python SPA_python_example.py --range 5 --rate 100 --avg 8 --samples 1000
```

Enable the bias source at +10 V:

```
python SPA_python_example.py --range 5 --bias 10.0 --polarity Positive --samples 50
```

Dry-run self-test (no hardware needed):

```
python SPA_python_example.py --dry-run
```

8 Stopping a Measurement

Press **Ctrl+C** at any time to stop the measurement cleanly. The script will:

1. Stop reading samples
2. Disable the instrument's data stream
3. Close the serial port
4. Report how many samples were captured

The instrument is always left in a safe, idle state after disconnecting.

9 Calibration

9.1 How It Works

Each SPA is factory-calibrated with precision reference currents. The calibration data maps raw ADC readings to calibrated current values in Amps.

Each of the 8 input ranges has its own calibration, consisting of two reference points: one at a known positive current and one at a known negative current. The script uses linear interpolation between these two points.

9.2 Automatic Calibration Handling

The script handles calibration automatically:

1. **First choice:** load from a JSON file on disk (e.g. settings/SPA120_cal.JSON)
2. **If no file exists:** download the calibration directly from the instrument's flash memory, verify it, and save it to disk for next time

You do not normally need to do anything – the script takes care of it.

9.3 Manual Calibration File

If you need to use a specific calibration file (for example, if you have multiple instruments), use the `--cal` option:

```
python SPA_python_example.py --cal settings/SPA120_cal.JSON --range 5
```

The calibration file is a standard JSON file. You can inspect it with any text editor.

10 Using the Script in Your Own Code

The SPA class can be imported into your own Python scripts. Here is a minimal example:

```
from SPA_python_example import SPA

# Connect to the instrument
spa = SPA("COM3", cal_file="settings/SPA120_cal.JSON")
spa.connect()

# Configure
spa.set_range(1, 5)           # Ch1 = 20 uA range
spa.set_range(2, 5)           # Ch2 = 20 uA range
spa.set_sample_rate(10)       # 10 Hz
spa.set_averaging(4)          # 4x rolling average

# Read 100 samples
for i in range(100):
    sample = spa.read_sample()
    ch1 = sample["ch1_amps"]    # calibrated current in Amps
    ch2 = sample["ch2_amps"]
    avg1 = sample["ch1_avg"]    # rolling average in Amps
    print(f"Ch1: {ch1:.6e} A    Ch2: {ch2:.6e} A")

# Always disconnect when done
spa.disconnect()
```

10.1 SPA Class Reference

10.1.1 Constructor

```
spa = SPA(port, cal_file=None)
```

- `port` – COM port string, e.g. "COM3" or "/dev/ttyUSB0"
- `cal_file` – path to calibration JSON file (optional; downloads from instrument if not provided or file not found)

10.1.2 Methods

Method	Description
<code>spa.connect()</code>	Open port, load cal, start streaming
<code>spa.disconnect()</code>	Stop streaming, close port
<code>spa.set_range(ch, index)</code>	Set input range (ch=1 or 2, index=0..7)
<code>spa.set_sample_rate(hz)</code>	Set rate: 2, 10, or 100 Hz
<code>spa.set_averaging(n)</code>	Set rolling average depth (1..64)
<code>spa.set_bias(ch, volts, pol, on)</code>	Set bias source voltage and enable/disable

Method	Description
<code>spa.read_sample()</code>	Read one sample, returns a dict (see below)
<code>SPA.format_current(amps)</code>	Format Amps with SI prefix (static method)

10.1.3 read_sample() Return Value

`read_sample()` returns a dictionary:

```
{
  "ch1_raw":    int,    # signed 24-bit ADC count
  "ch2_raw":    int,    # signed 24-bit ADC count
  "ch1_amps":   float,  # calibrated current (Amps)
  "ch2_amps":   float,  # calibrated current (Amps)
  "ch1_avg":    float,  # rolling average (Amps)
  "ch2_avg":    float,  # rolling average (Amps)
  "ch1_overload": bool,  # True if ADC is clipping
  "ch2_overload": bool,  # True if ADC is clipping
}
```

11 Troubleshooting

11.1 "No SPA instrument found"

The script could not auto-detect the COM port. Try:

- Check that the USB cable is connected
- Check Device Manager for the COM port number
- Specify the port manually: `--port COM3`

11.2 "pyserial is not installed"

Run `pip install pyserial` to install the serial library.

11.3 Overload readings ("** OVR **")

The input current exceeds the selected range. Switch to a less sensitive (higher index) range.

11.4 Noisy or unstable readings

- Increase the rolling average: `--avg 8` or `--avg 16`
- Use a slower sample rate: `--rate 2`
- Check your cable shielding and connections

12 Self-Test (Dry Run)

The `--dry-run` option runs a comprehensive self-test that verifies all protocol logic without any hardware connected:

```
python SPA_python_example.py --dry-run
```

This tests packet building, checksums, parsing, sign extension, calibration loading, ADC-to-Amps conversion, SI formatting, and more. All 70 tests should show [PASS]. This is useful for verifying that the script works correctly on your system before connecting an instrument.

13 Support

For technical support, firmware updates, and the latest documentation:

- **Website:** www.electron.plus
- **Email:** support@electron.plus