

ModbusBB

User Manual

Version 2.3.0 · November 2026

Max Energic Apps · modbus.maxenergic.com

Contact / Help

ModbusBB user manual

Version 2.3.0 · Step-by-step instructions with screenshots for the program and the command-line tool.

By Mohammad Ba'ra, Electrical Engineer, SCADA/SAS Specialist · Updated November 15, 2026

---

1. Introduction

ModbusBB is a Modbus tool for Windows. It has three parts:

- **Master (client)**: reads and writes data in Modbus devices such as energy meters, PLCs, inverters and drives.
- **Slave simulator**: acts like a Modbus device, so you can test a SCADA system or a PLC program without real hardware.
- **Command-line tool (CLI)**: the same functions in a terminal, for scripts and automatic tests.

ModbusBB supports five ways (transports) to talk to devices: Modbus TCP, Modbus UDP, Modbus RTU (serial), Modbus ASCII (serial) and RTU over TCP (serial frames through an Ethernet gateway).

What you can do with ModbusBB

- Connect to several devices at the same time. Each connection has its own poll tabs.
- Read coils, discrete inputs, holding registers and input registers, once or every few milliseconds.
- Show values as 16, 32 or 64-bit integers, Float32, Float64, strings, hex or binary, in all four byte orders, with scale, offset and unit.
- Write values to coils and holding registers.
- See every byte that goes over the wire in the traffic monitor.
- Run diagnostics: device identification (FC43), Report Slave ID (FC17), Diagnostics (FC08), Mask Write (FC22), Read/Write Multiple (FC23) and raw requests.

- Find devices and registers with scanners, chart values over time, raise alarms, log data to CSV and save your setup as a workspace.

How to use this manual

Chapters 3 to 9 explain the daily work: install, connect, read and write. The later chapters explain the tools one by one. Numbered red marks in the screenshots match the numbered lists under them. Words in **bold** are names of buttons, menus and fields in the program. Technical words are explained in the glossary.

> **Short on time?**
>
> Read chapter 3 (install), chapter 6 (connect) and chapter 7 (read data). That is enough to read your first device.

2. Requirements

Item	Requirement
Operating system	Windows 10 or Windows 11, 64-bit (x64)
Runtime	Nothing extra. The .NET 8 runtime is included in the program.
Network devices	A network connection to the device (TCP or UDP, default port 502).
Serial devices (RTU / ASCII)	An RS-485 or RS-232 adapter that Windows shows as a COM port (for example COM3).
Internet	Only for activating the license, the license check (at least every 30 days) and the update check. Your Modbus data never leaves your network.

Serial devices behind an Ethernet gateway

If your RS-485 devices are connected to a serial-to-Ethernet gateway (device server), you do not need a COM port. Use the transport **RTU over TCP (gateway)** or, if the gateway converts to Modbus TCP, use **Modbus TCP**.

3. Installation

Install with the installer (recommended)

1. Download `ModbusBB_Setup_{VER}.exe` from the download page.
2. Double-click the file. Windows may ask for administrator rights. Click **Yes**.
3. Follow the wizard. The program is installed in `C:\Program Files\ModbusBB`.
4. Optional: tick **Add CLI to system PATH**. Then you can type `ModbusBB.CLI` in any terminal window.
5. Optional: tick **Create a desktop shortcut**.
6. Click **Finish**. ModbusBB starts.

The installer creates three Start-menu entries: **ModbusBB** (the program), **ModbusBB CLI** (opens a command window with the CLI help) and the uninstaller.

Portable version (no installation)

1. Download `ModbusBB_Portable_{VER}.zip`.
2. Right-click the ZIP file and choose **Extract All**. Choose any folder, for example a USB stick.
3. Start `GUI\ModbusBB.GUI.exe` for the program, or use `CLI\ModbusBB.CLI.exe` in a terminal.

Update to a new version

Run the new installer over the old installation. You do not need to uninstall first. Your license, trial status and settings stay.

Uninstall

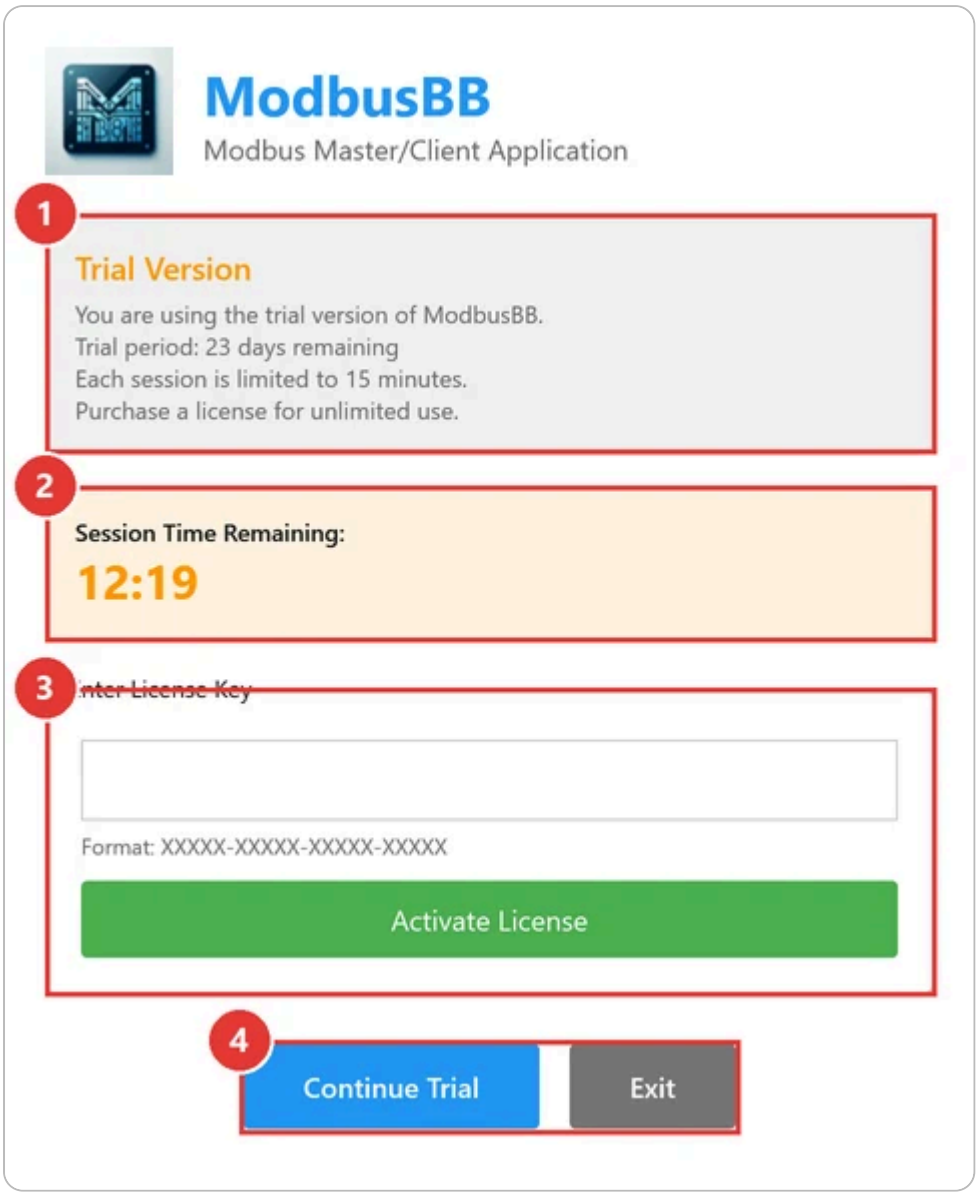
Open Windows **Settings > Apps**, select **ModbusBB** and click **Uninstall**. The folder `%LOCALAPPDATA%\ModbusBB` is kept on purpose. It holds your license activation and the trial record, so a later reinstall keeps working.

Check the download

The download page shows a SHA-256 checksum for each file. In PowerShell, run `Get-FileHash .\ModbusBB_Setup_2.3.0.exe` and compare the result with the checksum on the page.

4. First start and the license window

Every time ModbusBB starts without a license, it shows the license window.



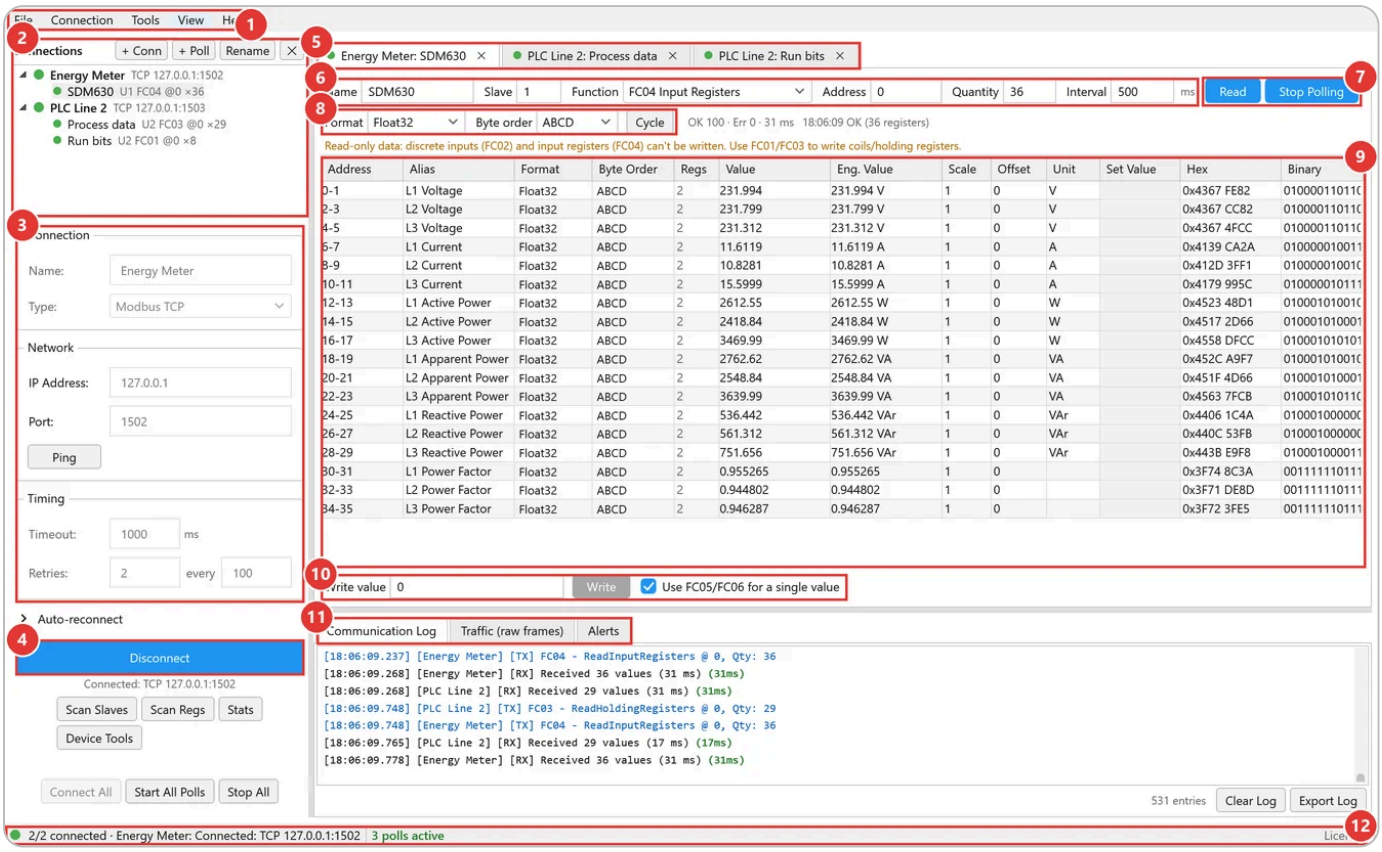
The license window in trial mode.

1. **Trial Version**: how many trial days are left. The trial lasts **30 days** from the first start.
2. **Session Time Remaining**: in the trial, each session lasts **15 minutes**. Then the program closes. Start it again for a new session.
3. **Enter License Key**: type your key in the format xxxxx-xxxxx-xxxxx-xxxxx and click **Activate License**. This needs internet.
4. **Continue Trial** opens the program in trial mode. **Exit** closes it.

With a valid license there is no time limit and the license window does not appear. The status bar shows **Licensed** in the bottom-right corner. More about keys, devices and moving the license: chapter 21.

5. The main window

The main window has a fixed layout. The left side is for connections. The right side is for data.



The main window with two connections and three poll tabs.

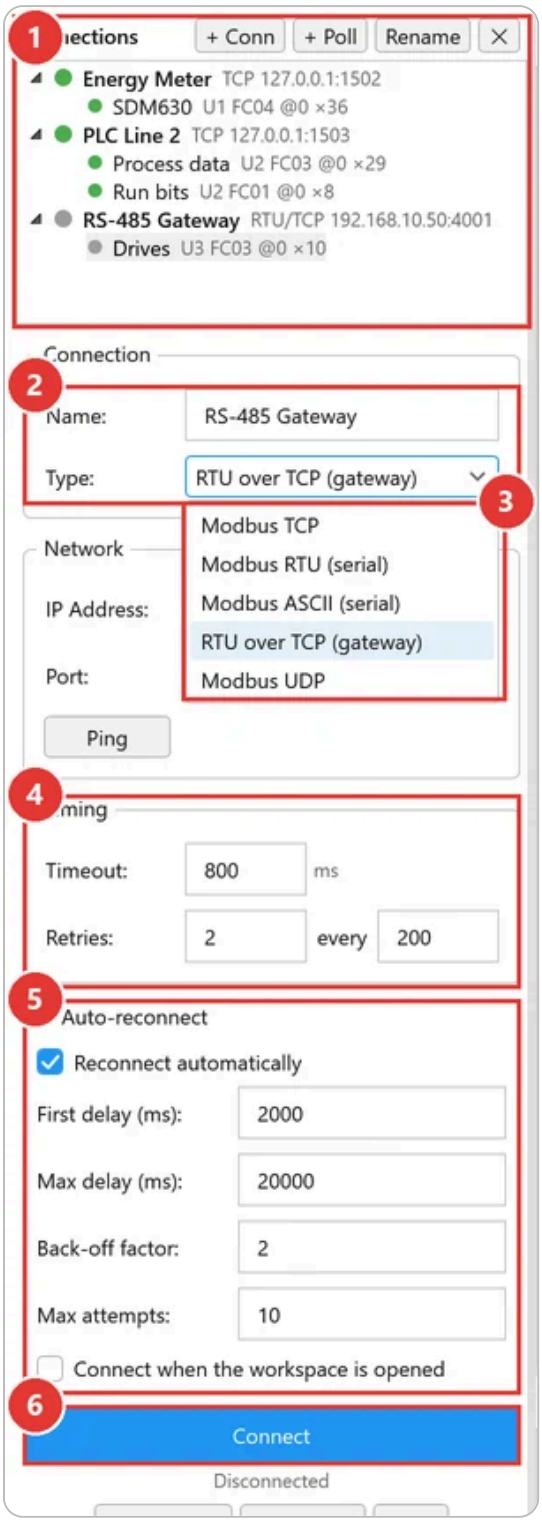
1. **Menu bar**: File, Connection, Tools, View and Help.
2. **Connections tree**: every connection and its polls. The buttons **+ Conn**, **+ Poll**, **Rename** and ✕ add, rename or remove items. A green dot means connected.
3. **Connection settings** of the selected connection: name, type, network or serial settings and timing.
4. **Connect / Disconnect** button. Below it: **Scan Slaves**, **Scan Regs**, **Stats** and **Device Tools**.
5. **Poll tabs**: one tab per poll. The dot shows the state of its connection.
6. **Poll settings**: name, slave ID, function, address, quantity and interval.
7. **Read** (read once, F5) and **Start / Stop Polling** (F6).
8. **Format** and **Byte order** for all rows of this tab, and the **Cycle** button.

9. **Register grid**: one row per value, with address, alias, format, value, engineering value, hex and binary.
10. **Write value** box and **Write** button.
11. Bottom tabs: **Communication Log**, **Traffic (raw frames)** and **Alerts**.
12. **Status bar**: how many connections are connected, active polls, logging and watchdog state, and the license state.

Below the connection buttons you also find **Connect All**, **Start All Polls** and **Stop All**. They act on all connections at once.

6. Connecting to devices

A **connection** is one link to one device or one network/serial line. You can have many connections at the same time. Each connection has its own settings and its own poll tabs.



Connection settings. The Type list shows the five transports.

1. Select the connection in the tree. Use **+ Conn** to add a new one.

2. **Name**: any name, for example "Meter panel A". It appears in the tree, tabs, log and traffic monitor.
3. **Type**: the transport (see the table below).
4. **Timing**: timeout and retries.
5. **Auto-reconnect**: what happens when the link drops.
6. **Connect** / **Disconnect**.

Step by step: connect to a Modbus TCP device

1. Click **+ Conn** (or **Connection > Add Connection**).
2. Type a **Name**.
3. Set **Type** to **Modbus TCP**.
4. Type the device **IP Address** and **Port** (Modbus default: 502).
5. Optional: click **Ping** to check that the IP address answers.
6. Click **Connect**. The dot in the tree turns green and the button changes to **Disconnect**.

Step by step: connect to a serial (RTU) device

1. Connect the RS-485 adapter. Windows gives it a COM port number (see Windows Device Manager).
2. Click **+ Conn** and set **Type** to **Modbus RTU (serial)**.
3. Choose the **COM Port**. Click **Refresh** if the port is not in the list.
4. Set **Baud Rate**, **Data Bits**, **Parity** and **Stop Bits** exactly as in the device. Common: 9600, 8, None, 1 (often written "9600 8N1") or 19200 8E1.

5 Click **Connect**.

The five transports

Type (in the list)	Use it for	Settings
Modbus TCP	Devices with Ethernet that speak Modbus TCP (most meters, PLCs, inverters).	IP address, port
Modbus UDP	Devices that use Modbus over UDP (less common).	IP address, port
RTU over TCP (gateway)	Serial devices behind a transparent serial-to-Ethernet gateway. The RTU frame with CRC is sent through TCP.	IP address, port of the gateway
Modbus RTU (serial)	RS-485 / RS-232 devices with binary RTU frames. Most serial devices use RTU.	COM port, baud rate, data bits, parity, stop bits
Modbus ASCII (serial)	Older serial devices that use ASCII frames (often 7 data bits).	COM port, baud rate, data bits, parity, stop bits

Serial settings

Field	Options	Default
Baud Rate	1200, 2400, 4800, 9600, 19200, 38400, 57600, 115200, 230400	9600
Data Bits	7, 8	8
Parity	None, Odd, Even, Mark, Space	None
Stop Bits	1, 1.5, 2	1

Timing

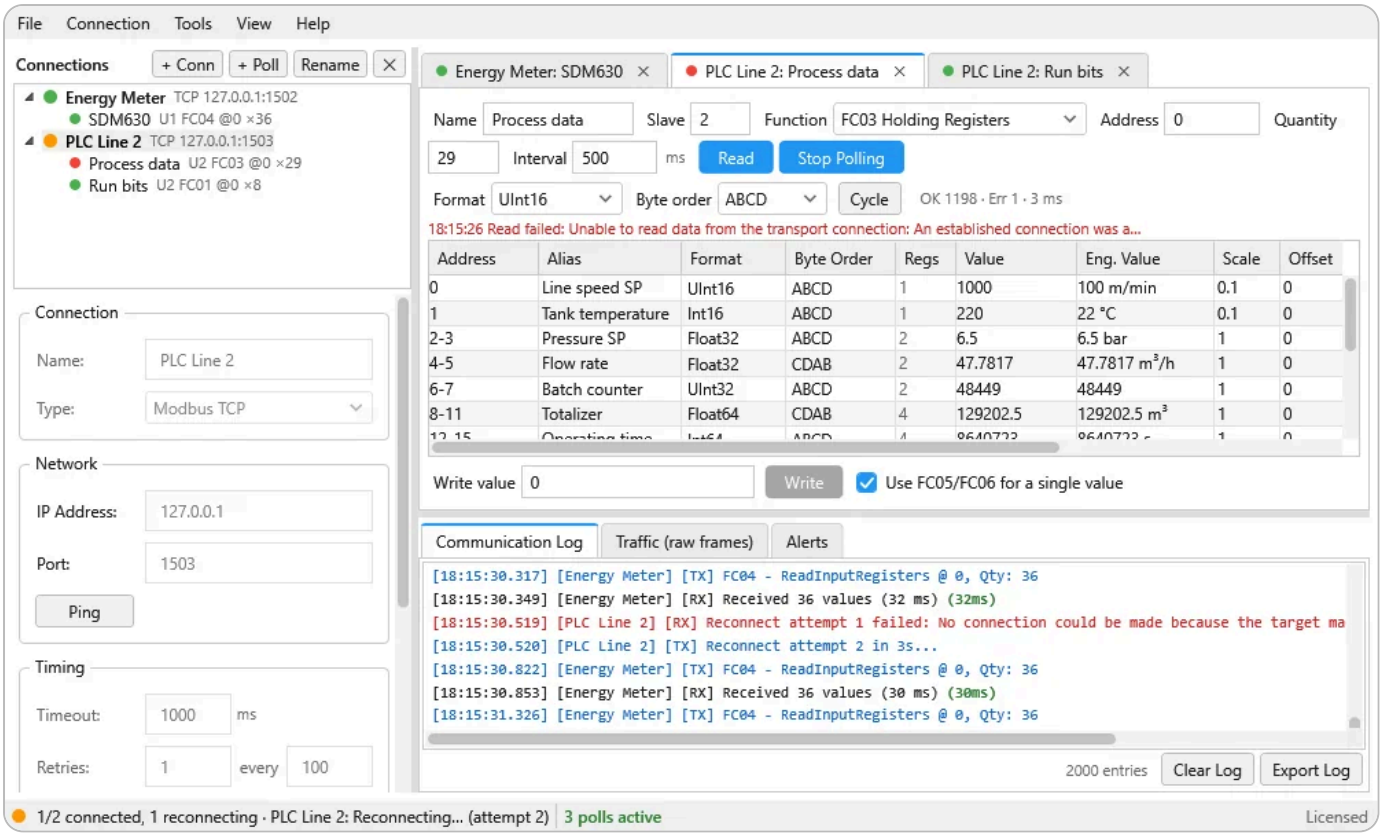
Field	Meaning	Range	Default
Timeout	How long ModbusBB waits for an answer.	50 – 60000 ms	1000 ms
Retries	How many times a failed request is sent again.	0 – 10	3
every	Pause between two retries.	0 – 10000 ms	250 ms

Auto-reconnect

When **Reconnect automatically** is on (default), ModbusBB opens a dropped link again and continues polling. The pause between attempts grows step by step:

Field	Meaning	Default
First delay (ms)	Pause before the first new attempt.	1000
Max delay (ms)	The longest pause between attempts.	30000
Back-off factor	The pause is multiplied by this number after each failed attempt. 1 = always the same pause.	2
Max attempts	Stop after this many attempts. 0 = try forever.	0
Connect when the workspace is opened	When you open a saved workspace, this connection connects and its running polls start again.	off

When **Reconnect automatically** is off, a dropped link stops the polls of this connection.



A dropped link: the tab dot and the status bar show the reconnect attempts.

Several connections at the same time

Add one connection for each device or line. All connections work at the same time. **Connection > Connect All** and **Disconnect All** act on every connection. Several devices on one RS-485 line share *one* connection; use a separate poll tab with its own **Slave** number for each device.

7. Polling and poll tabs

A **poll** is one read request that ModbusBB can repeat. Each poll has its own tab. A connection can have many polls, for example one for measurements and one for status bits.

Step by step: read data

1. Select the connection in the tree and click **+ Poll** (or press `Ctrl+T`). A new tab opens.
2. Type a **Name** for the poll.
3. **Slave**: the device address (unit ID), 0–255. Most TCP devices use 1. Decimal or hex (`0x..`).
4. **Function**: what to read (see the table below).
5. **Address**: the first address, 0-based (see addresses). Decimal (400) or hex (0x190).
6. **Quantity**: how many registers (1–125) or bits (1–2000) to read.
7. **Interval**: time between reads when polling, 100 – 3600000 ms. Default 1000 ms.
8. Click **Read** (`F5`) to read once, or **Start Polling** (`F6`) to read again and again.

The line next to the **Cycle** button shows the counters: **OK** (successful reads), **Err** (failed reads), the last response time and the time of the last read.

Function	Reads	Data	Writable?
FC01 Coils	Coils (on/off outputs)	bits	Yes (FC05 / FC15)
FC02 Discrete Inputs	Discrete inputs (on/off inputs)	bits	No, read-only

Function	Reads	Data	Writable?
FC03 Holding Registers	Holding registers (settings, setpoints, many measured values)	16-bit registers	Yes (FC06 / FC16)
FC04 Input Registers	Input registers (measured values)	16-bit registers	No, read-only

Work with polls

- Rename: **Connection > Rename Poll**, or right-click the poll in the tree.
- Remove: click ✕ on the tab, or **Connection > Remove Poll**.
- Start or stop all polls: **Start All Polls** / **Stop All** (bottom of the left side).
- All polls of all connections can run at the same time.

Poll Groups: all polls in one window

Connection > Poll Groups (all, live)... opens one window that lists every poll of every connection, with live values. Use it when you poll many register groups at once and want to see them all without switching tabs.

Column	What it shows
Group	The poll name
Session	The connection the poll belongs to
Request	Slave, function, address and quantity
Interval	The poll interval in ms
Value	The latest values (the first 8)
Age	How long ago the last answer came
Counters	OK and Err reads and the last response time

- The button in each row starts or stops that one poll. **Start All** and **Stop All** start or stop every poll on a connected connection.

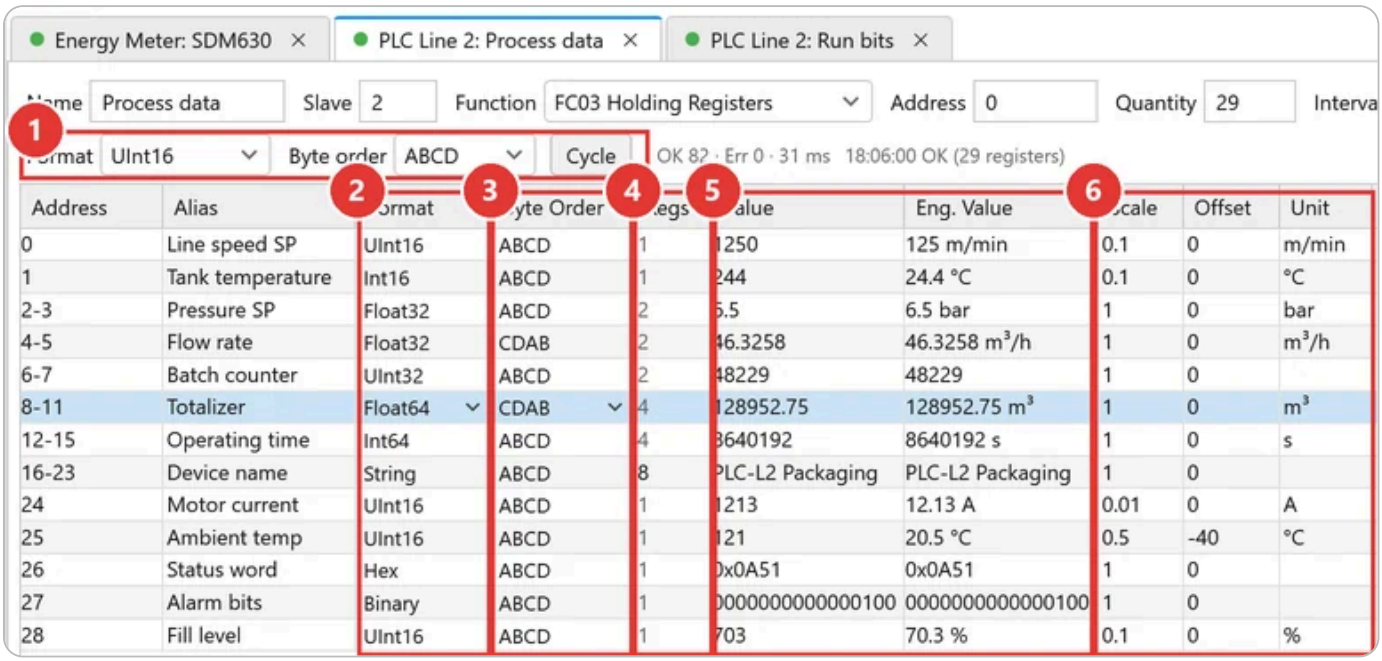
- **Export snapshot...** saves the current values of every group to a CSV file. **Copy** puts the same CSV on the clipboard.
- To add or change a group, use the poll tabs. The window only shows the polls and starts or stops them.
- All polls on one connection share the link. ModbusBB sends their requests one after the other, so polls, trends and manual reads never mix on the wire.

The CLI can do the same with the `pollset` command.

> **Too many errors?**
>
> Increase the **Interval**, reduce the **Quantity**, or increase the **Timeout** of the connection. Slow serial lines need a longer interval. A device that does not answer only costs the timeout: the window does not freeze, and you can stop the poll at any time. See Troubleshooting.

8. Data types, byte order, scaling and addresses

A Modbus register holds 16 bits. Many values need more space: a Float32 or a 32-bit integer uses 2 registers, a Float64 or a 64-bit integer uses 4 registers. ModbusBB combines the registers for you. You choose the **format** and the **byte order**.



Different formats in one poll tab.

1. **Format** and **Byte order** for all rows of the tab. **Cycle** tries the next byte order.
2. **Format** column: change the format of one row.
3. **Byte Order** column: change the byte order of one row.
4. **Regs**: how many registers this value uses.
5. **Value** (raw decoded value) and **Eng. Value** (after scale and offset, with unit).
6. **Scale**, **Offset** and **Unit** of the row.

Formats

Format	Registers	Range / meaning
UInt16	1	0 to 65535
Int16	1	−32768 to 32767

Format	Registers	Range / meaning
Hex	1	Hexadecimal, e.g. 0x0A51
Binary	1	16 bits, e.g. 0000101001010001
UInt32 / Int32	2	32-bit unsigned / signed integer
Float32	2	IEEE 754 single precision (decimal numbers)
UInt64 / Int64	4	64-bit unsigned / signed integer (energy counters, operating hours)
Float64	4	IEEE 754 double precision
String	set by **String regs**	Text, 2 characters per register

Byte order

Devices store multi-register values in different orders. If a value looks wrong (very large, very small or nonsense), the byte order is probably wrong. Click **Cycle** until the value looks right.

Byte order	Name	32-bit bytes	64-bit bytes
ABCD	Big-endian (Modbus standard)	A B C D	A B C D E F G H
CDAB	Word swap (very common in meters and PLCs)	C D A B	G H E F C D A B
BADC	Byte swap inside each register	B A D C	B A D C F E H G
DCBA	Little-endian	D C B A	H G F E D C B A

"A" is the most significant byte. Example: the float 230.5 is `43 66 80 00` in ABCD. In CDAB the device sends `80 00 43 66`.

Scaling and units

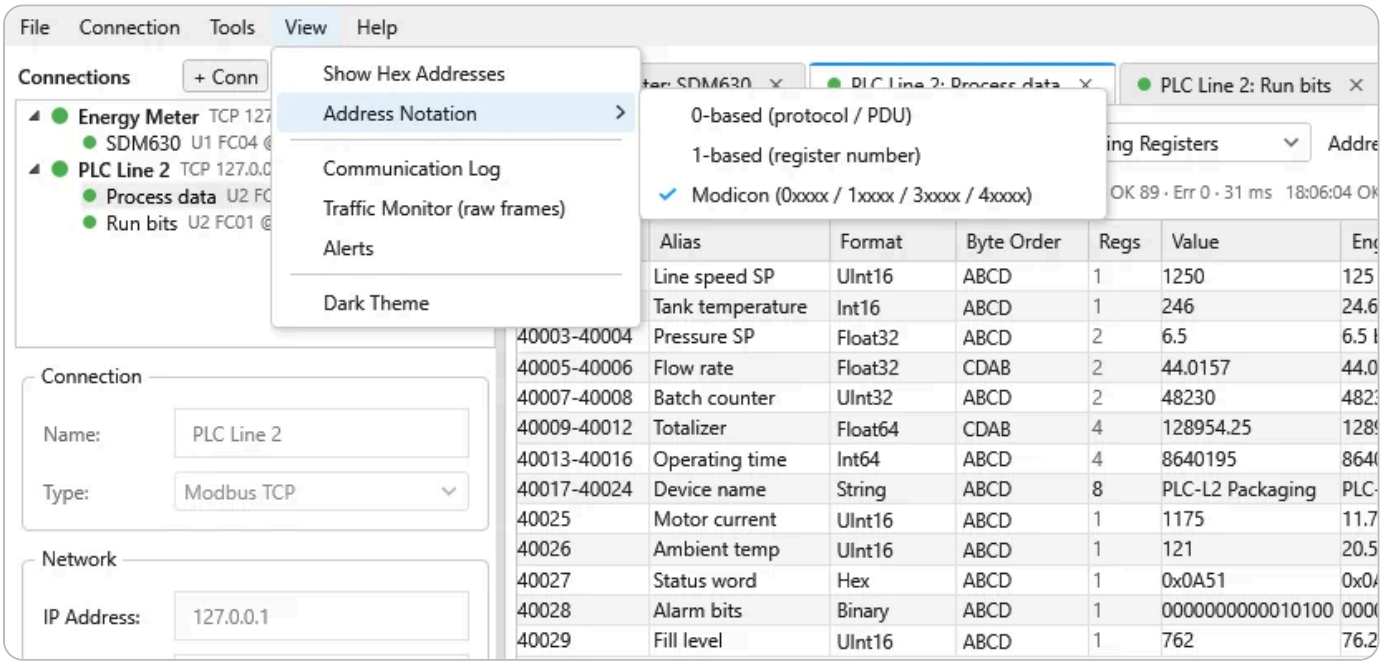
Many devices send integers that you must scale. **Engineering value = raw value × Scale + Offset.** Example: register value 2305 with Scale 0.1 and Unit V gives 230.5 V. Type the scale, offset and unit in the row of the grid.

Addresses: 0-based, 1-based and Modicon

The Modbus message always carries a **0-based** address (0–65535). This is what you type in the **Address** field. Device manuals often use other numbers:

Manual says	Notation	Type in ModbusBB
Register 1 / 40001 (holding)	1-based / Modicon	Address 0, FC03
30011 (input register)	Modicon	Address 10, FC04
10001 (discrete input)	Modicon	Address 0, FC02
00017 (coil)	Modicon	Address 16, FC01

View > Address Notation changes only how the grid shows addresses: **0-based (protocol / PDU)**, **1-based (register number)** or **Modicon (0xxxx / 1xxxx / 3xxxx / 4xxxx)**. Requests always use the 0-based address. **View > Show Hex Addresses** shows addresses in hex.



View > Address Notation.

9. Writing values

You can write to **coils** (FC01 tabs) and **holding registers** (FC03 tabs). Discrete inputs (FC02) and input registers (FC04) are read-only. A read-only tab shows an orange note.

> **Be careful**
>
> A write changes the real device. A wrong value or address can start or stop machinery or change setpoints. Check slave, address, format and value before you write.

Method 1: Set Value column (fastest)

1. Read the tab once (`F5`) so the rows exist.
2. Click the **Set Value** cell of the row.
3. Type the new value in the format of the row (for example `21.5` for Float32, `1` or `0` for a coil).
4. Press `Enter`. ModbusBB writes the value and reads it back on the next poll.

Method 2: Write value box

1. Select a row in the grid.
2. Type the value in the **Write value** box under the grid.
3. Click **Write**.

Comma-separated values (for example `10,20,30`) write consecutive rows in one request.

Which function code is used?

- With **Use FC05/FC06 for a single value** ticked, one coil is written with FC05 and one 16-bit register with FC06.

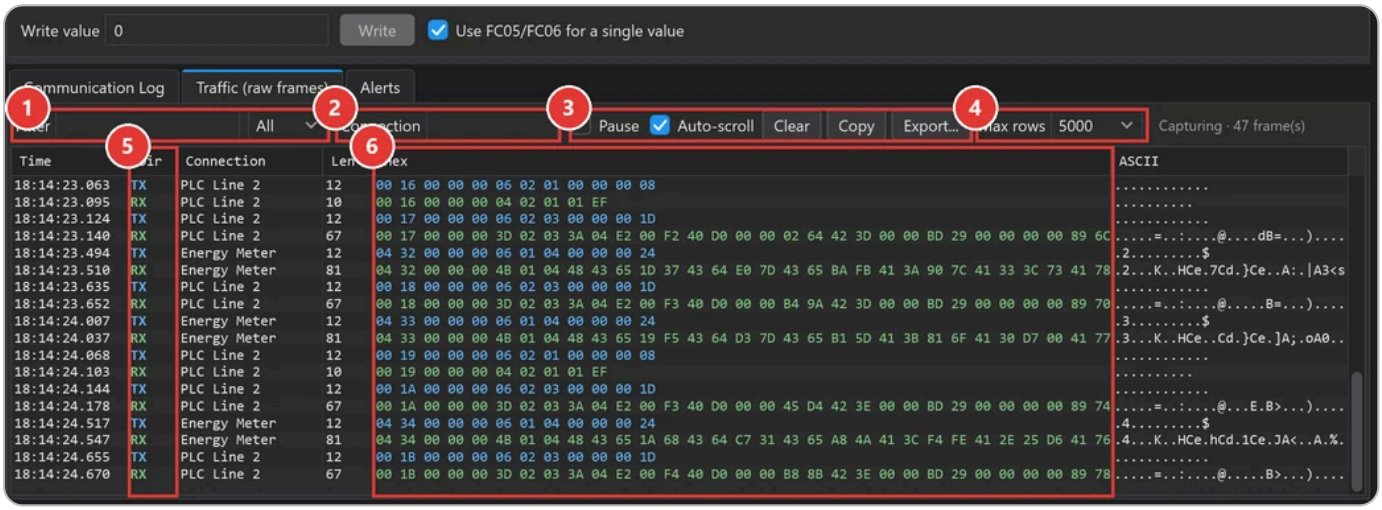
- Without the tick, and for more than one value, ModbusBB uses FC15 (coils) or FC16 (registers).
- 32-bit, 64-bit and String values always use FC16, because they need more than one register.

To write a bit mask or to write and read in one step, use Device Tools (FC22 and FC23).

10. Communication log and traffic monitor

The bottom part of the main window has three tabs.

- **Communication Log**: one line per request and response in plain words, with time and response time. **Clear Log** empties it, **Export Log** saves it to a file.
- **Traffic (raw frames)**: every frame as bytes (hex and ASCII). Use it to see exactly what the device sends.
- **Alerts**: alarms from the watchdog.



The traffic monitor with frames of two connections.

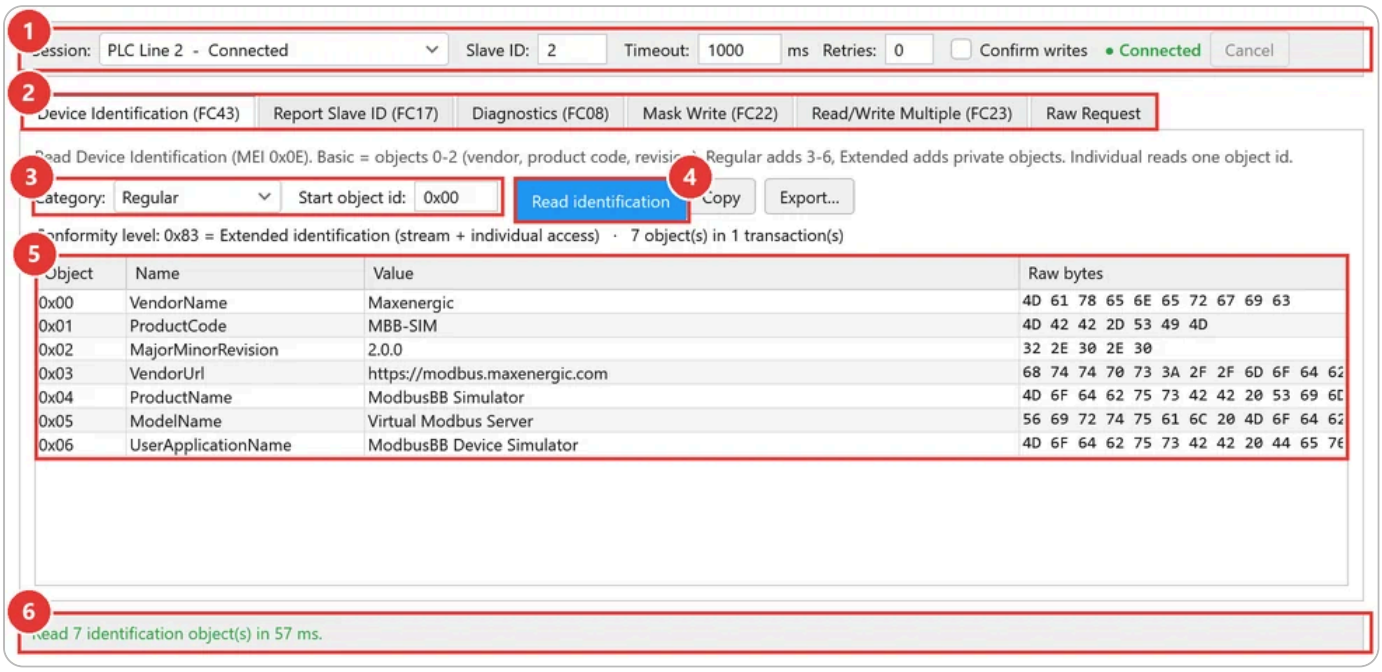
1. **Filter**: show only frames that contain this text in hex, ASCII or connection name (for example `01 03`). Next to it: **All**, **TX** or **RX**.
2. **Connection**: show only frames of connections whose name contains this text.
3. **Pause** stops capturing. **Auto-scroll** follows new frames. **Clear**, **Copy** (`Ctrl+C`) and **Export...** (CSV or TXT).
4. **Max rows**: 1000, 5000 or 20000 frames are kept.
5. **Dir**: TX = sent by ModbusBB, RX = received from the device.
6. **Hex**: the frame bytes. TCP frames start with the MBAP header (transaction ID, protocol ID, length, unit ID).

Tip

When a value looks wrong, compare the RX bytes with the device manual. You then see if the problem is the address, the format or the byte order.

11. Device Tools and diagnostics

Tools > Device Tools (or the **Device Tools** button) opens a window for special Modbus functions. The top bar applies to all tabs.

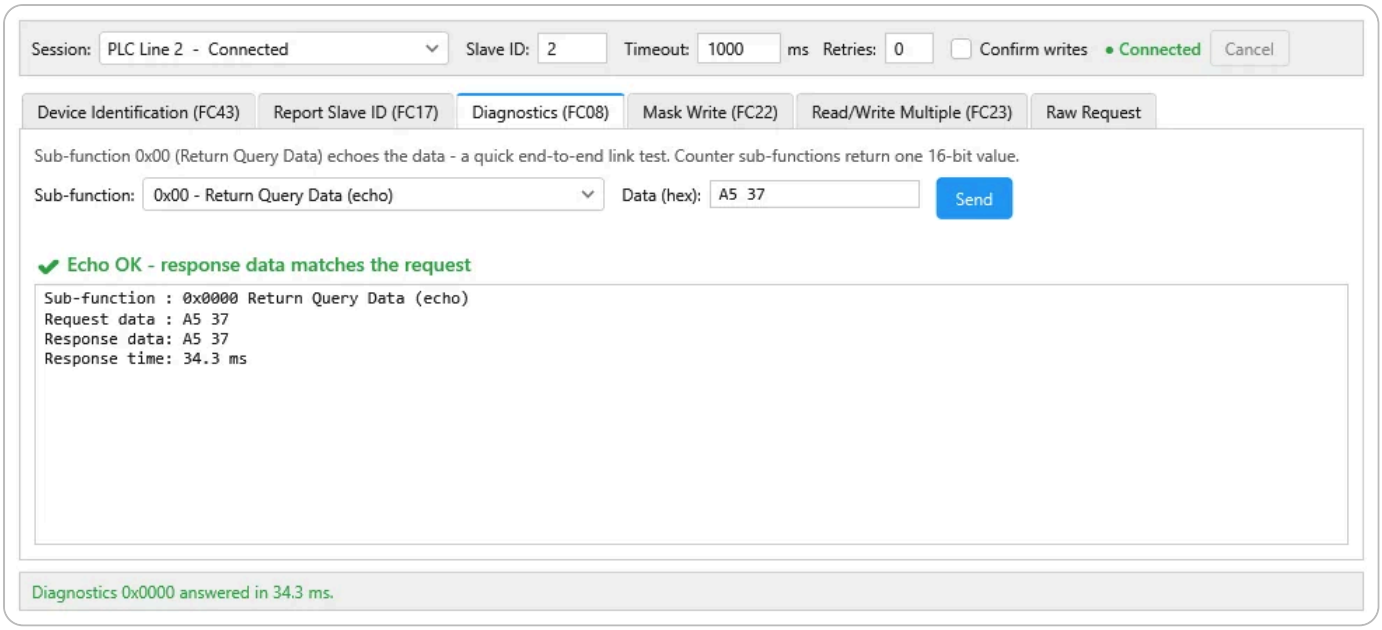


Device Tools: reading the device identification (FC43).

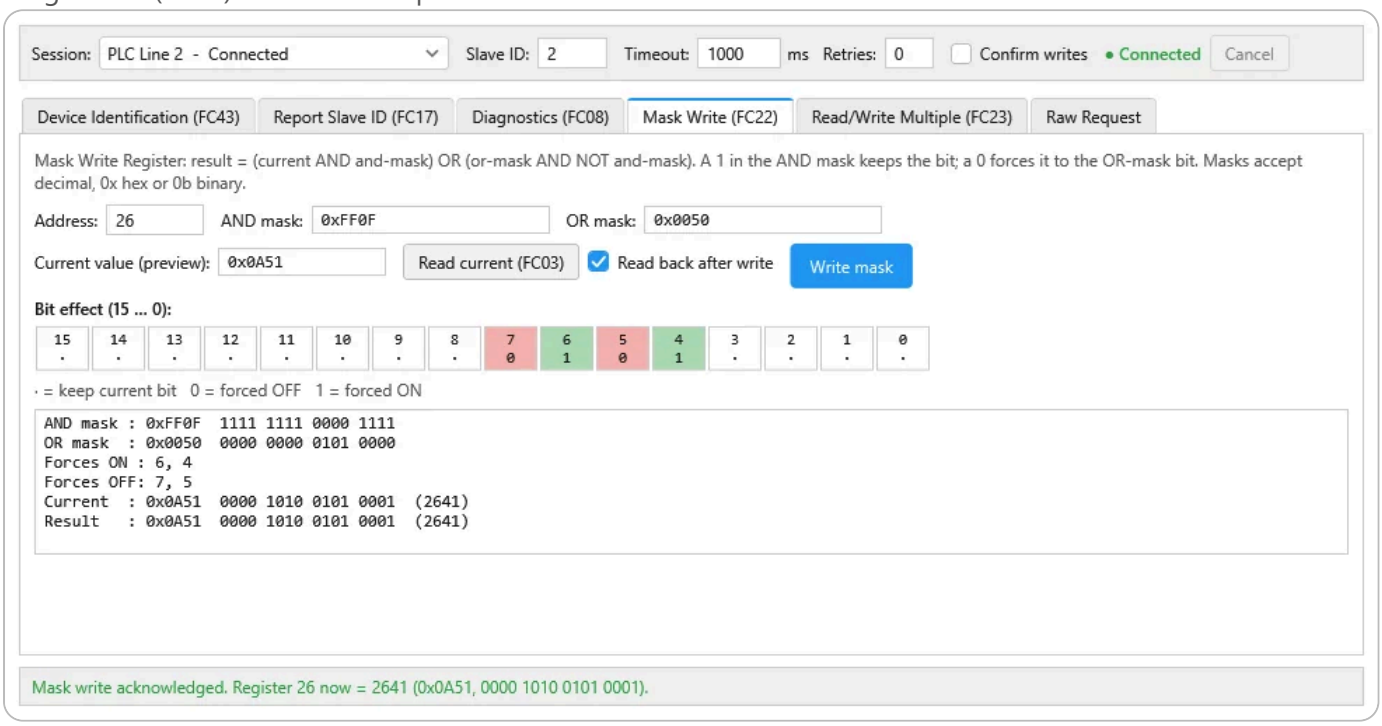
1. **Session** (which connection to use), **Slave ID**, **Timeout** (empty = connection setting), **Retries** and **Confirm writes** (asks before any request that changes data).
2. One tab per function.
3. **Category** and **Start object id**.
4. **Read identification**.
5. The result: object, name, value and raw bytes. **Copy** and **Export...** save it.
6. Status line with the result and response time.

The tabs, explained simply

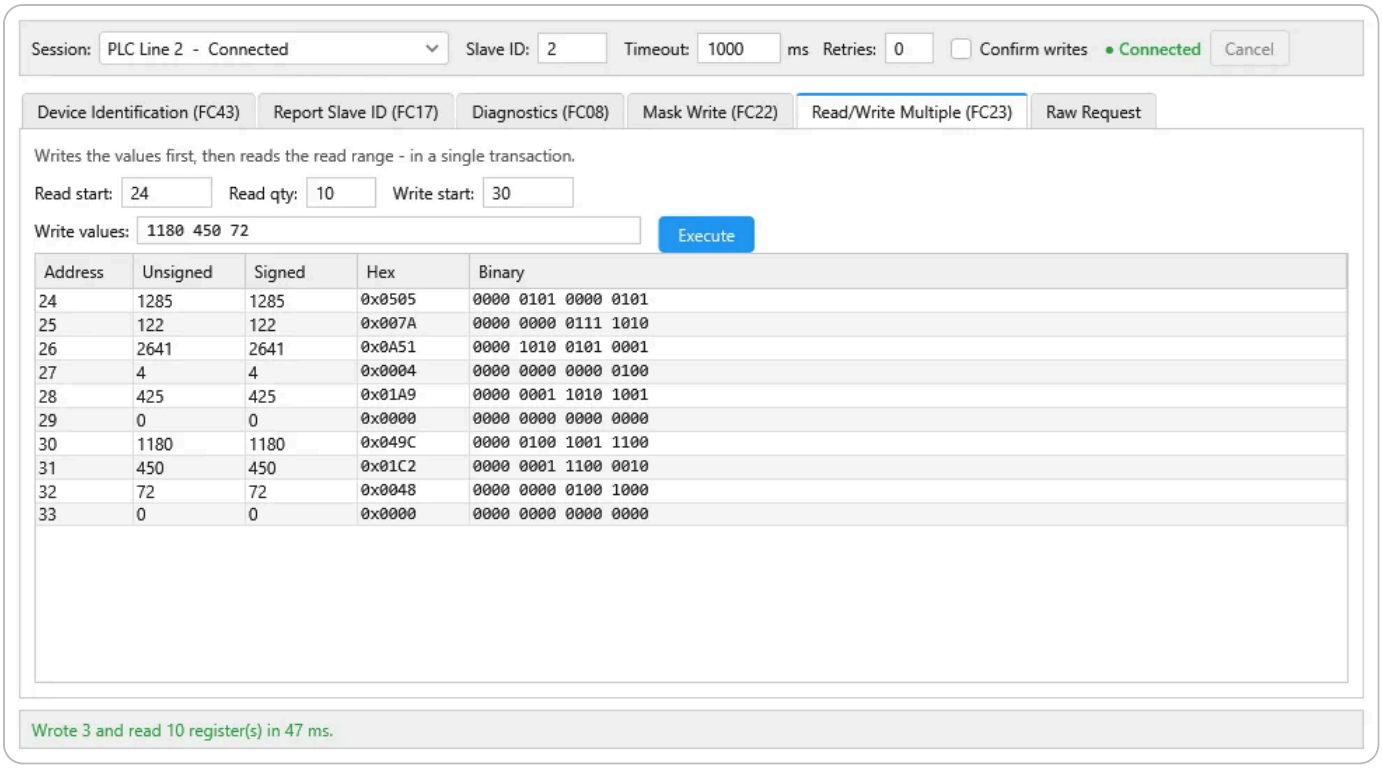
Tab	What it does	Typical use
Device Identification (FC43)	Reads text information from the device: vendor, product code, revision. *Basic* = objects 0–2, *Regular* adds 3–6, *Extended* adds private objects, *Individual* reads one object.	Find out which device and firmware answers.
Report Slave ID (FC17)	Reads a device-specific server ID, the run indicator (0xFF = ON) and extra data.	Older devices that don't support FC43.
Diagnostics (FC08)	Sub-function 0x00 (Return Query Data) sends data that the device must echo. Other sub-functions read or clear counters (bus messages, errors, overruns …).	Test the link end to end. Check error counters on a serial line.
Mask Write (FC22)	Changes single bits in a register. Result = (current AND and-mask) OR (or-mask AND NOT and-mask). The preview shows the effect on each bit.	Switch one bit without touching the others.
Read/Write Multiple (FC23)	Writes values and then reads a range, in one transaction. The write happens first.	Write a command and read the status in one step.
Raw Request	Sends any function code with your data bytes. Choose *Function code + data* or *Full PDU*, or a *Preset*. The history keeps your requests (double-click to reuse).	Test functions that have no own tab.



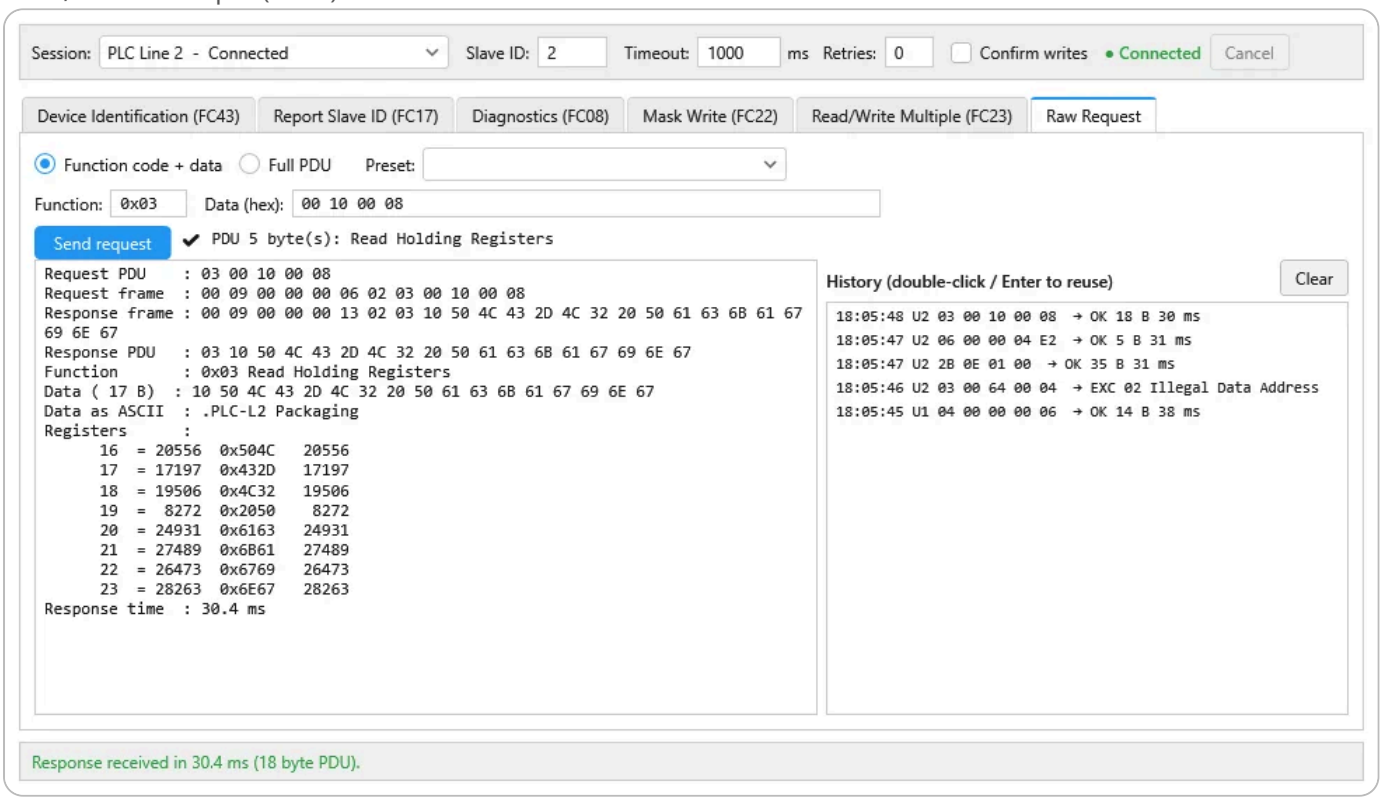
Diagnostics (FC08): the echo test passed.



Mask Write (FC22) with the bit-by-bit preview.



Read/Write Multiple (FC23).



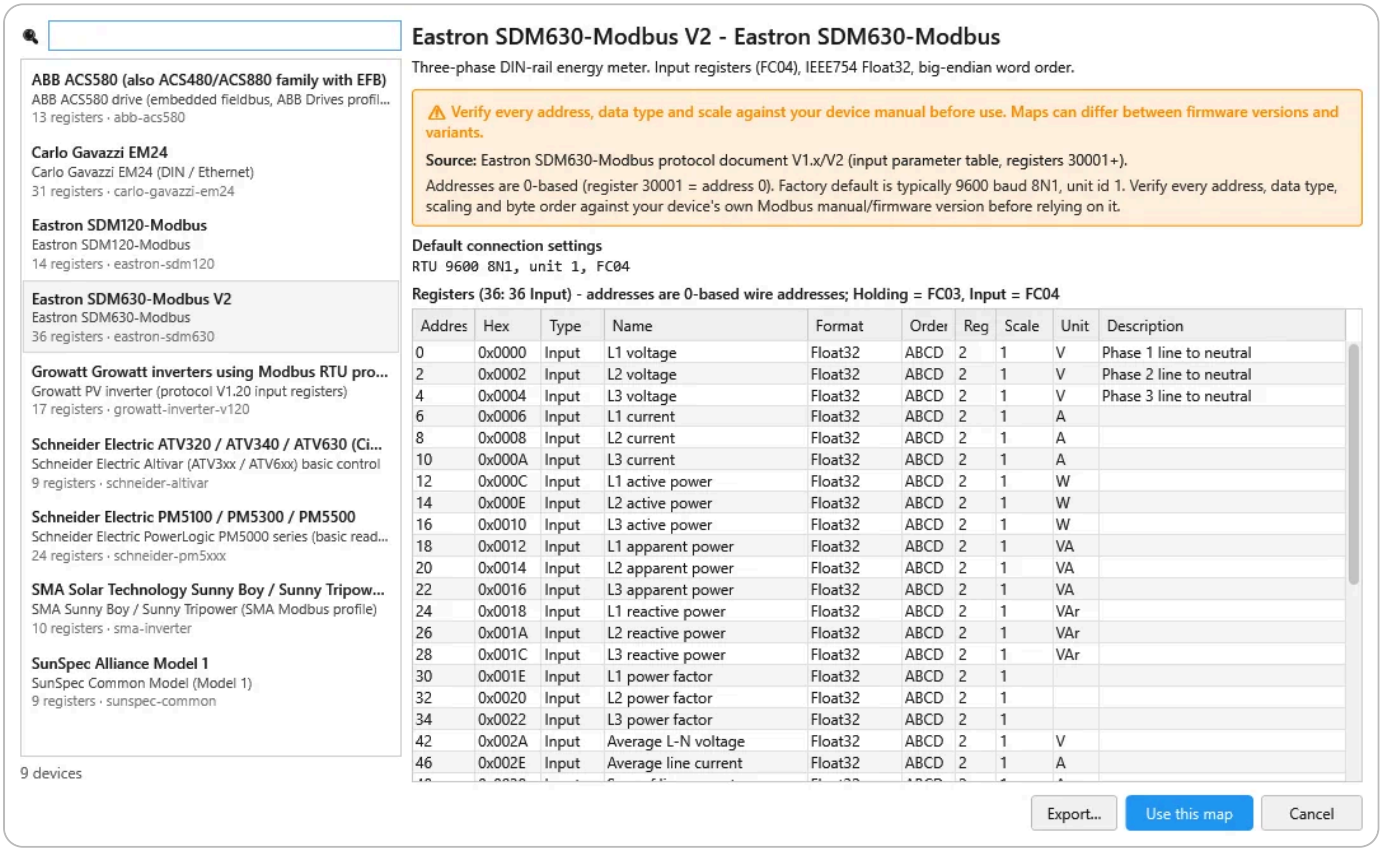
Raw Request: request and response frames decoded.

12. Register maps and the device library

A **register map** gives each address a name (alias), a format, a byte order, a scale and a unit. With a map you see "L1 Voltage 231.9 V" instead of "0x4367".

Use a built-in map from the device library

1. Create a poll tab for the device (for example FC04, address 0).
2. Open **Tools > Device Library**.
3. Type in the search box (manufacturer, model or name) and select the device.
4. Check the **Default connection settings** and the register list on the right.
5. Click **Use this map**. Names, formats and scaling are applied to the active tab.



The device library with the Eastron SDM630 map selected.

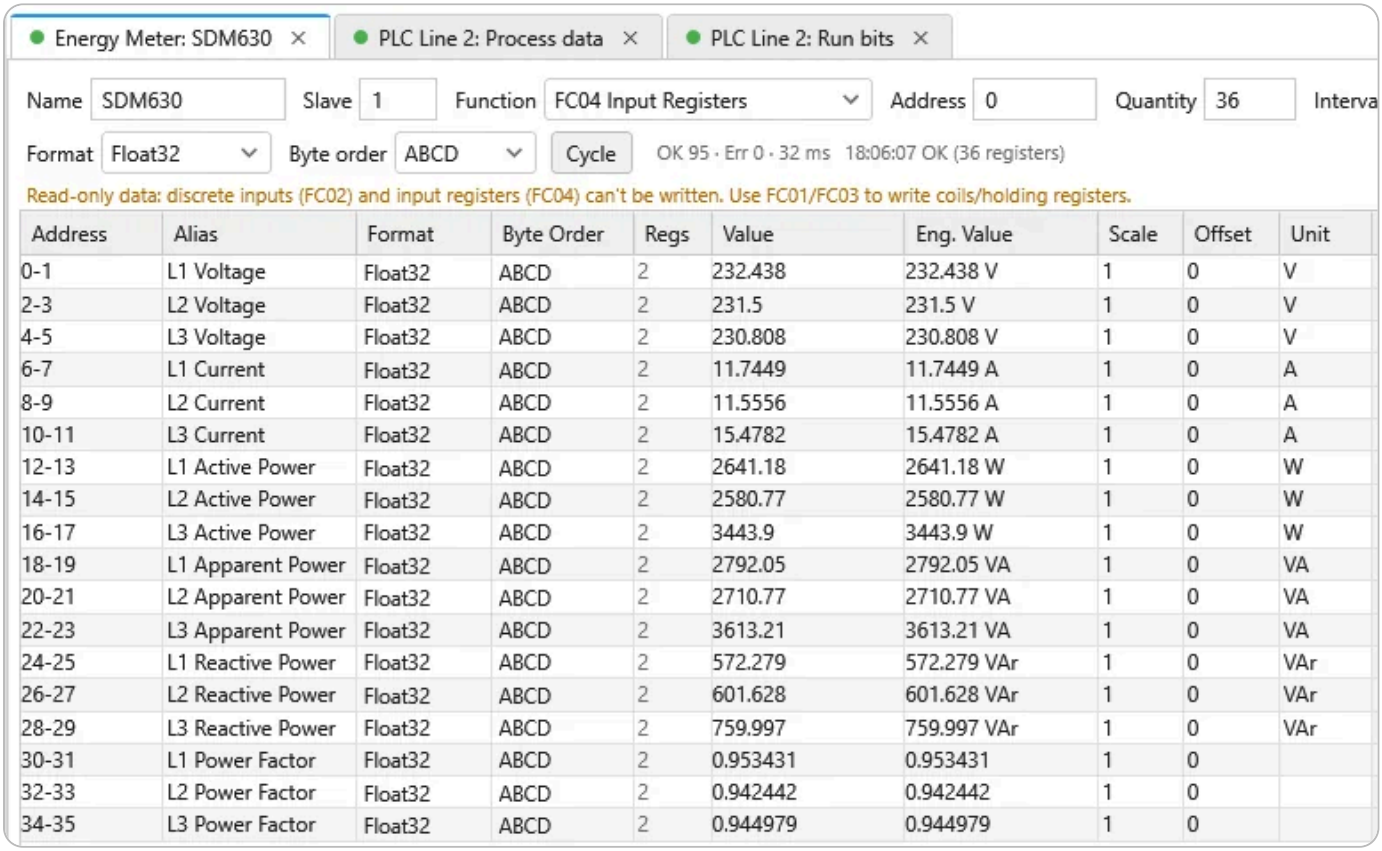
The library contains maps for: ABB ACS580, Carlo Gavazzi EM24, Eastron SDM120, Eastron SDM630, Growatt inverters, Schneider Altivar, Schneider PM5xxx, SMA inverters and SunSpec common model. **Export...** saves a map as JSON or CSV.

> **Check the map**
>
> Maps can differ between firmware versions and device variants. Always compare addresses, data types and scaling with your device manual.

Your own maps (File menu)

- **File > Import Register Map...**: applies a CSV or JSON map to the active tab's connection.
- **File > Export Register Map...**: saves the current names, formats and scaling.
- **File > Create Register Map Template...**: creates an empty CSV file to fill in Excel.

CSV columns: `Address, Name, Description, Format, ByteOrder, Type, RegisterCount, ScaleFactor, Offset, Unit, MinValue, MaxValue, ReadOnly`.



Address	Alias	Format	Byte Order	Regs	Value	Eng. Value	Scale	Offset	Unit
0-1	L1 Voltage	Float32	ABCD	2	232.438	232.438 V	1	0	V
2-3	L2 Voltage	Float32	ABCD	2	231.5	231.5 V	1	0	V
4-5	L3 Voltage	Float32	ABCD	2	230.808	230.808 V	1	0	V
6-7	L1 Current	Float32	ABCD	2	11.7449	11.7449 A	1	0	A
8-9	L2 Current	Float32	ABCD	2	11.5556	11.5556 A	1	0	A
10-11	L3 Current	Float32	ABCD	2	15.4782	15.4782 A	1	0	A
12-13	L1 Active Power	Float32	ABCD	2	2641.18	2641.18 W	1	0	W
14-15	L2 Active Power	Float32	ABCD	2	2580.77	2580.77 W	1	0	W
16-17	L3 Active Power	Float32	ABCD	2	3443.9	3443.9 W	1	0	W
18-19	L1 Apparent Power	Float32	ABCD	2	2792.05	2792.05 VA	1	0	VA
20-21	L2 Apparent Power	Float32	ABCD	2	2710.77	2710.77 VA	1	0	VA
22-23	L3 Apparent Power	Float32	ABCD	2	3613.21	3613.21 VA	1	0	VA
24-25	L1 Reactive Power	Float32	ABCD	2	572.279	572.279 VAr	1	0	VAr
26-27	L2 Reactive Power	Float32	ABCD	2	601.628	601.628 VAr	1	0	VAr
28-29	L3 Reactive Power	Float32	ABCD	2	759.997	759.997 VAr	1	0	VAr
30-31	L1 Power Factor	Float32	ABCD	2	0.953431	0.953431	1	0	
32-33	L2 Power Factor	Float32	ABCD	2	0.942442	0.942442	1	0	
34-35	L3 Power Factor	Float32	ABCD	2	0.944979	0.944979	1	0	

After applying a map: aliases, formats and units in the grid.

13. Scanners and connection statistics

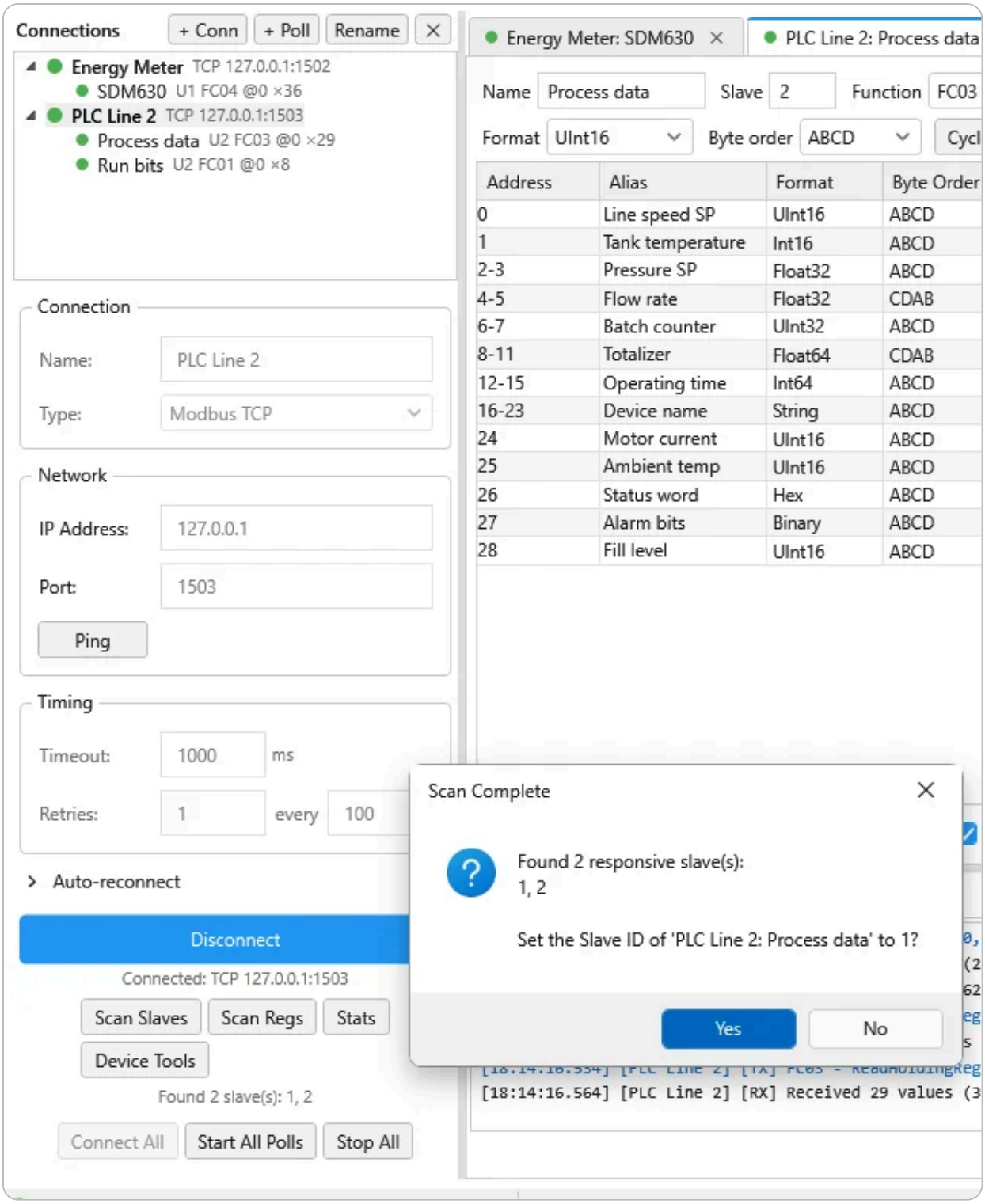
Scan Slaves: find devices on a line

1. Connect the connection.
2. Click **Scan Slaves** (or **Tools > Scan for Slaves**).
3. ModbusBB asks slave IDs 1 to 247. A device that answers, also with an exception, counts as found.
4. At the end ModbusBB lists the IDs and offers to set the found ID in the active tab.

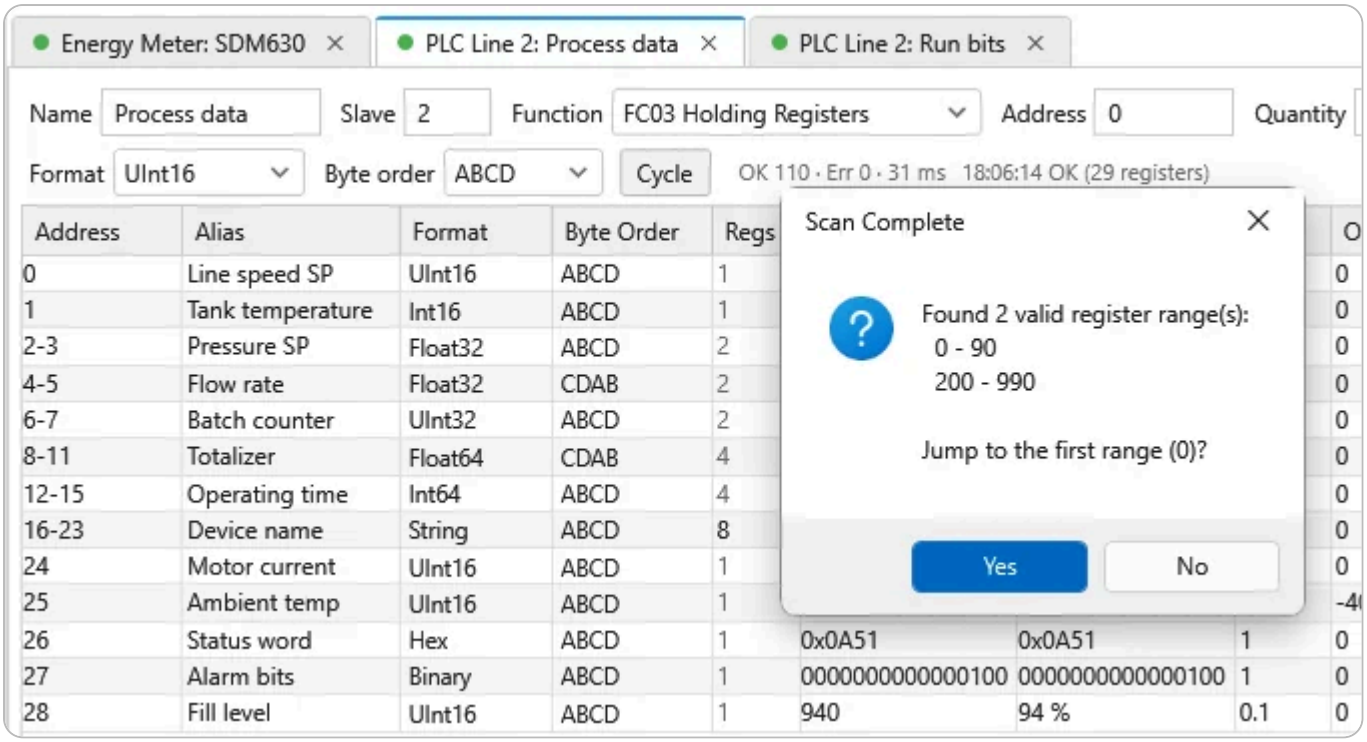
Click **Cancel Scan** to stop. Tip: a short connection timeout (for example 200 ms) makes the scan much faster.

Scan Regs: find valid registers

Scan Regs checks 1000 addresses from the address of the active tab, with the tab's function and slave. At the end it lists the ranges that answered and offers to jump to the first one.



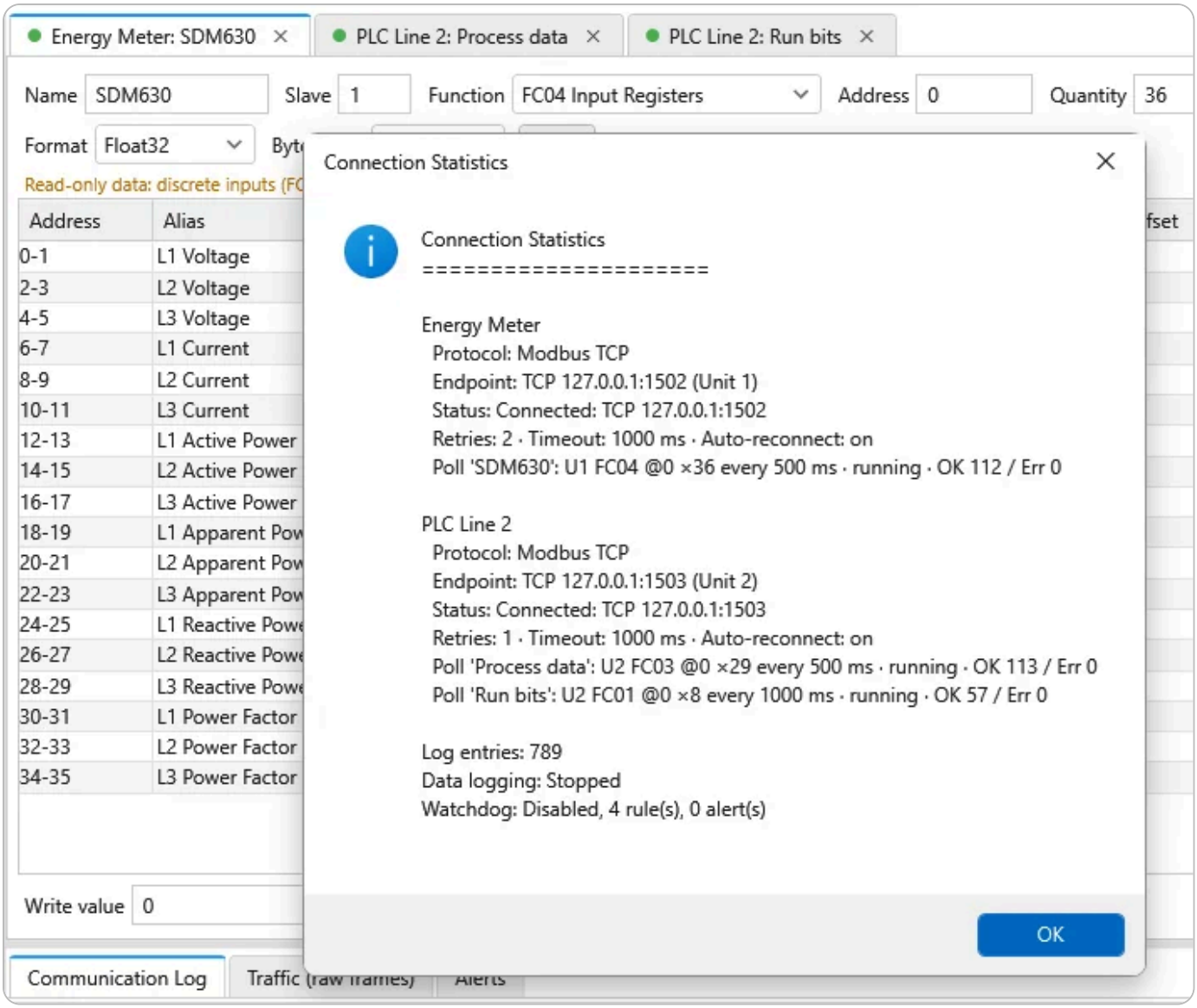
Scan Slaves found two devices.



Scan Regs found two valid ranges.

Connection statistics

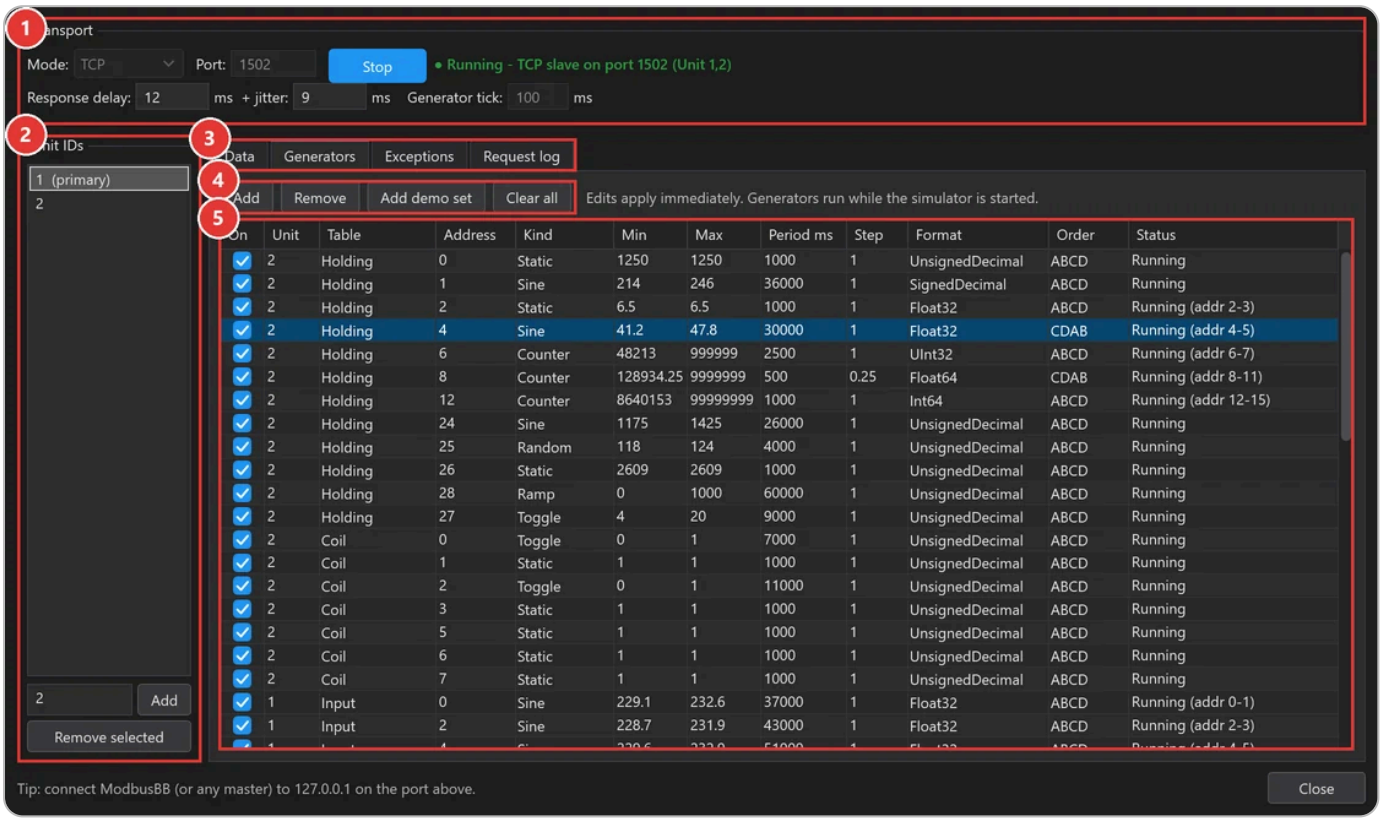
Stats (or **Tools > Connection Statistics**) shows each connection: protocol, endpoint, state, retries, auto-reconnect and, for each poll, the number of good and failed reads.



Connection statistics for two connections.

14. Slave simulator

Tools > Modbus Simulator (slave) turns your PC into a Modbus device. Use it to test a SCADA system, a PLC program or ModbusBB itself, without hardware.



The simulator with value generators on the Generators tab.

1. **Transport**: **Mode** (TCP, UDP, RTU, ASCII), **Port** or COM settings, **Start**/**Stop**. **Response delay** + **jitter** slow down the answers so you can test timeouts.
2. **Unit IDs**: each unit ID has its own data. Type a number and click **Add**.
3. Tabs: **Data**, **Generators**, **Exceptions** and **Request log**.
4. **Add**, **Remove**, **Add demo set** and **Clear all**.
5. The generators: table, address, kind, minimum, maximum, period, step, format and byte order.

Step by step: start a simulator and read it

1. Open **Tools > Modbus Simulator (slave)**.

2. Set **Mode** to **TCP** and **Port** to `5020` (port 502 may need administrator rights or be in use).
3. Open the **Generators** tab and click **Add demo set**.
4. Click **Start**. The status shows "Running".
5. In the main window, add a connection: Modbus TCP, IP `127.0.0.1`, port `5020`. Connect and read holding registers from address 0.

Data tab

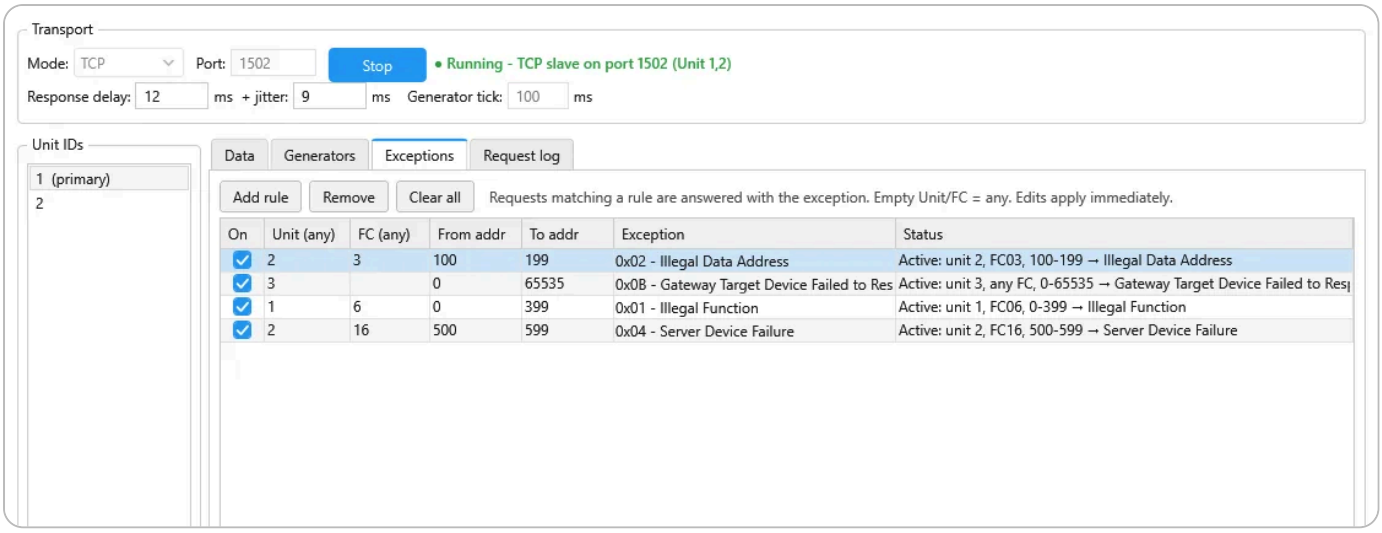
Choose **Unit**, **Table** (holding registers, input registers, coils, discrete inputs), **Start**, **Count** (1–250), **Format** and **Order**. Edit values in the grid and click **Apply edits**. **Auto-refresh** updates the values twice a second.

Generators

Kind	What it does
Static	Writes the Min value once when the simulator starts.
Ramp	Rises from Min to Max in one period, then starts again.
Sine	A sine wave between Min and Max.
Random	A new random value between Min and Max every period.
Counter	Adds Step every period and wraps from Max back to Min.
Toggle	Switches between Min and Max (coils: off/on) every period.

Exceptions

On the **Exceptions** tab, **Add rule** makes the simulator answer certain requests with a Modbus exception (for example 02 Illegal Data Address). A rule can match a unit, a function code and an address range (empty = any).



Exception rules: the simulator answers matching requests with an error code.

Serial simulator

For RTU or ASCII the simulator needs a COM port. To test on one PC, use a virtual COM-port pair (one port for the simulator, one for the master).

15. Live trends

Tools > Live Trends draws the values of the active poll tab over time.



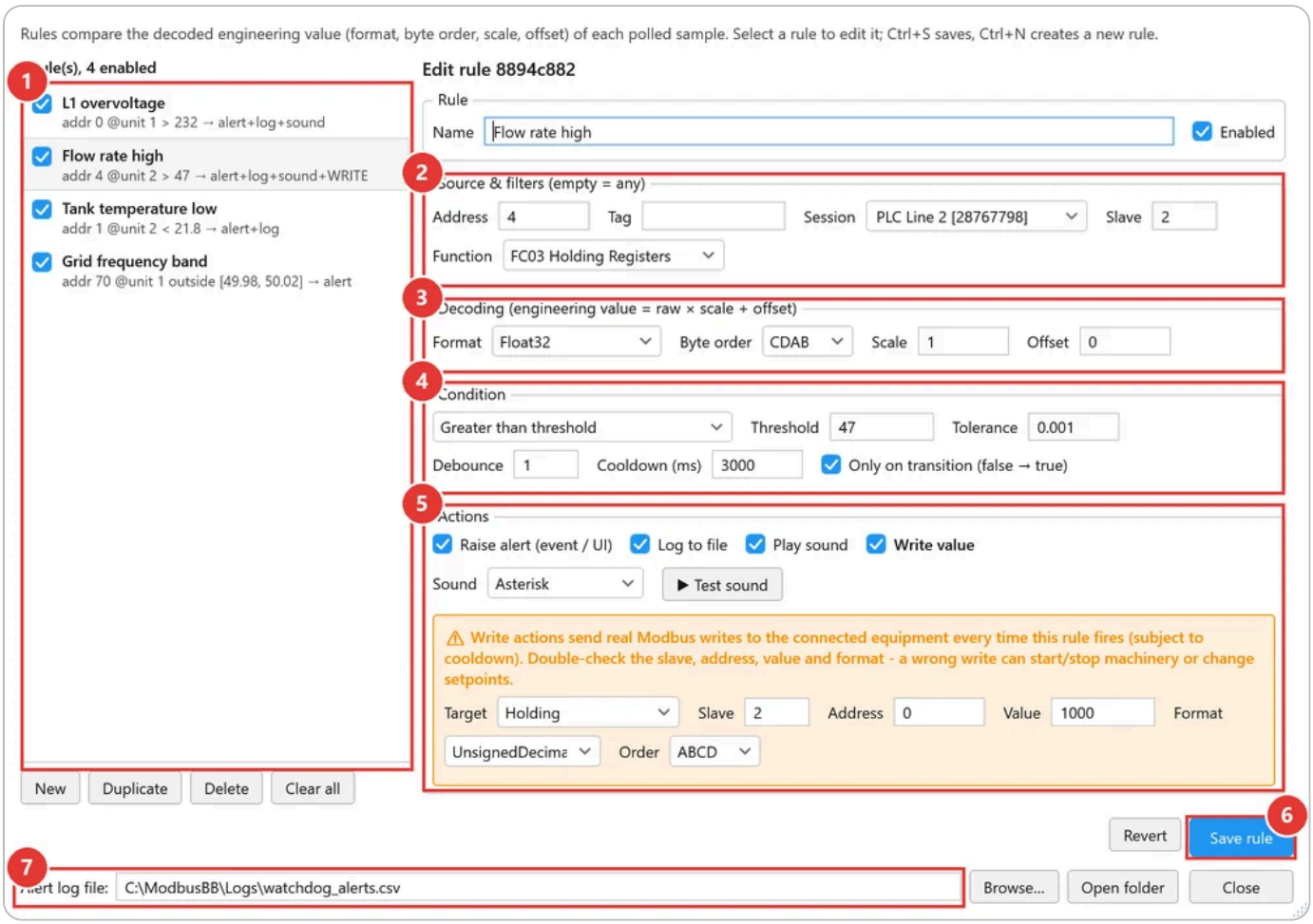
Live trends of four currents.

1. **Pause**, **Clear**, **Follow live**, **Window** (visible time span) and **History** (points kept per series) with an optional age limit.
2. **Auto Y** or a manual **Min**/**Max** with **Set**. **Reset view** returns to live mode.
3. **Export CSV...** (all samples) and **Export PNG...** (picture of the chart).
4. **Series**: tick to show a series. ✕ removes it.
5. The chart. Point at it to see the values at that time.
6. **Sample main grid every**: how often the values of the main grid are recorded.

Mouse: wheel = zoom time, `Ctrl`+wheel = zoom Y, drag = move, `Shift`+drag or right-drag = zoom box, double-click or `Home` = reset.

16. Watchdog alerts

The watchdog checks every value that is read. When a rule is true, it can raise an alert, write a line to a log file, play a sound or write a value to a device.



The watchdog editor with four rules.

1. The list of rules. Tick a rule to enable it. **New** (`Ctrl+N`), **Duplicate**, **Delete**, **Clear all**.
2. **Source & filters**: address, and optionally tag, session (connection), slave and function. Empty = any.
3. **Decoding**: how the raw value is turned into an engineering value (format, byte order, scale, offset).
4. **Condition**, threshold, tolerance, debounce (number of samples the condition must be true), cooldown and **Only on transition**.
5. **Actions**: raise alert, log to file, play sound, write value.
6. **Save rule** (`Ctrl+S`). **Revert** discards changes.

7 **Alert log file**: the CSV file for the "Log to file" action.

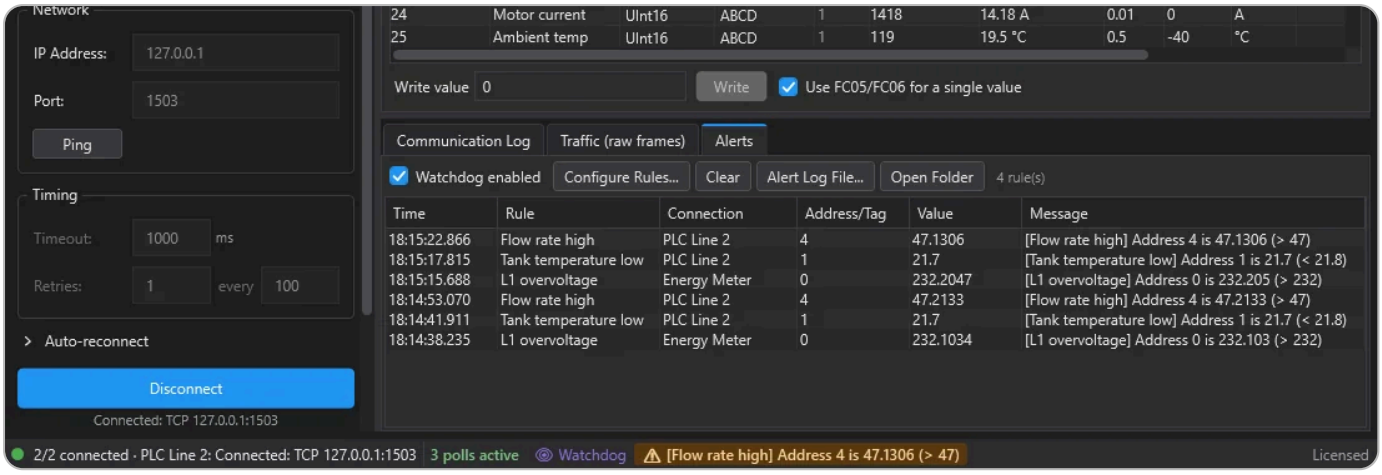
Step by step: alarm when a value is too high

1. Open **Tools > Watchdog Alerts > Configure Rules...**.
2. Click **New** and type a **Name**.
3. Type the **Address** of the value. Choose the **Format** and **Byte order** as in your poll tab.
4. Set **Condition** to **Greater than threshold** and type the **Threshold**.
5. Tick the actions you want, for example **Raise alert** and **Log to file**.
6. Click **Save rule**, then close the window.
7. Make sure **Tools > Watchdog Alerts > Enable Watchdog** is ticked.

Condition	True when
On change	The value changed by more than the tolerance.
Greater than / Less than threshold	The value is above / below the threshold.
Equals threshold (± tolerance)	The value equals the threshold.
Not equal to threshold	The value is different from the threshold.
Outside / Inside range [low, high]	The value is outside / inside the range.

Write actions are real

A **Write value** action sends a real Modbus write every time the rule fires (after the cooldown). Check slave, address, value and format carefully.



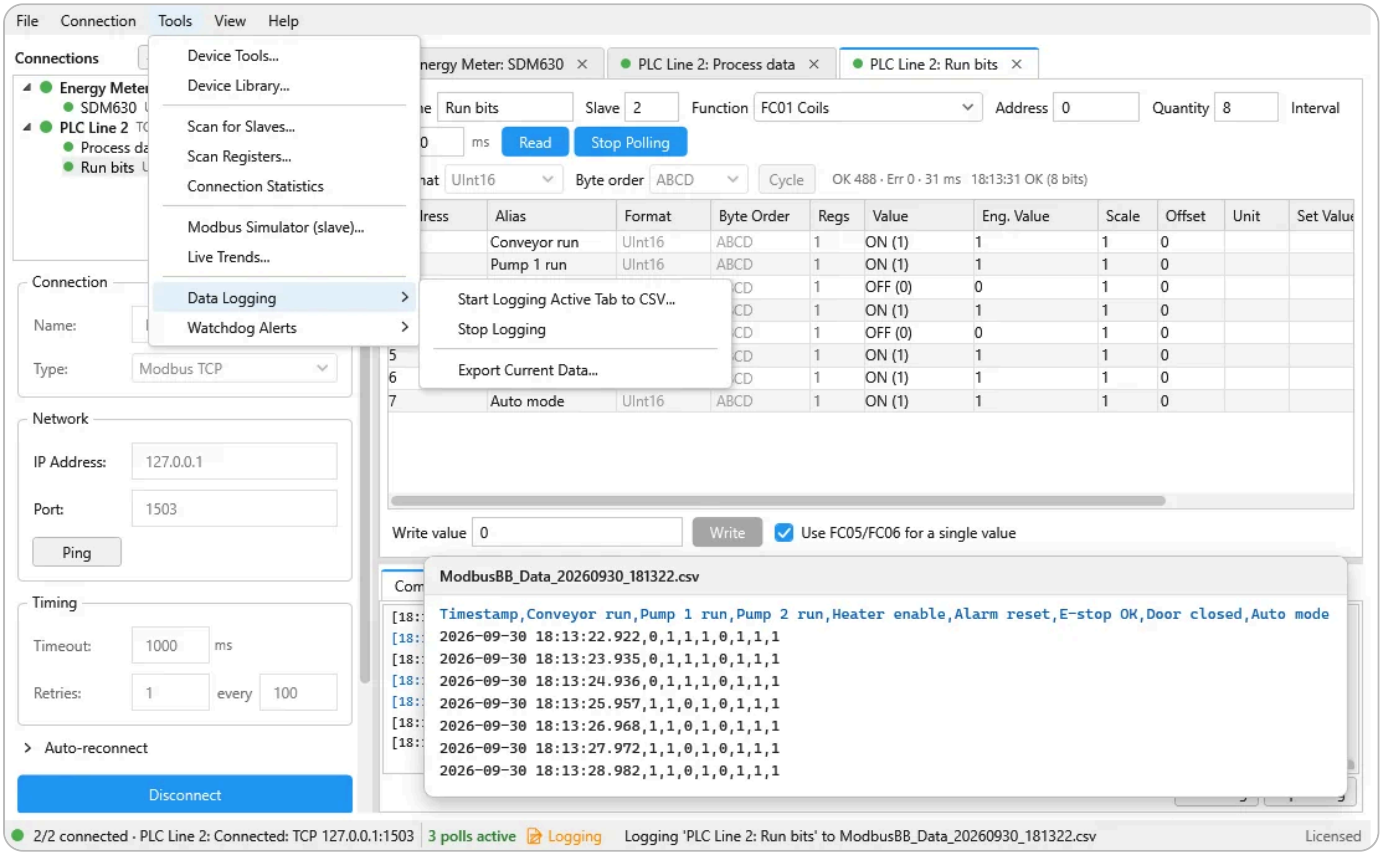
Alerts tab in the main window.

17. Data logging and export

Log a poll to CSV

1. Select the poll tab you want to log.
2. Choose **Tools > Data Logging > Start Logging Active Tab to CSV...** and choose a file name.
3. Every successful read adds one line: `Timestamp` and one column per value.
4. Stop with **Tools > Data Logging > Stop Logging**.

The status bar shows **Logging** while logging runs. The CSV file opens in Excel.



Logging the active tab to a CSV file.

Export one snapshot

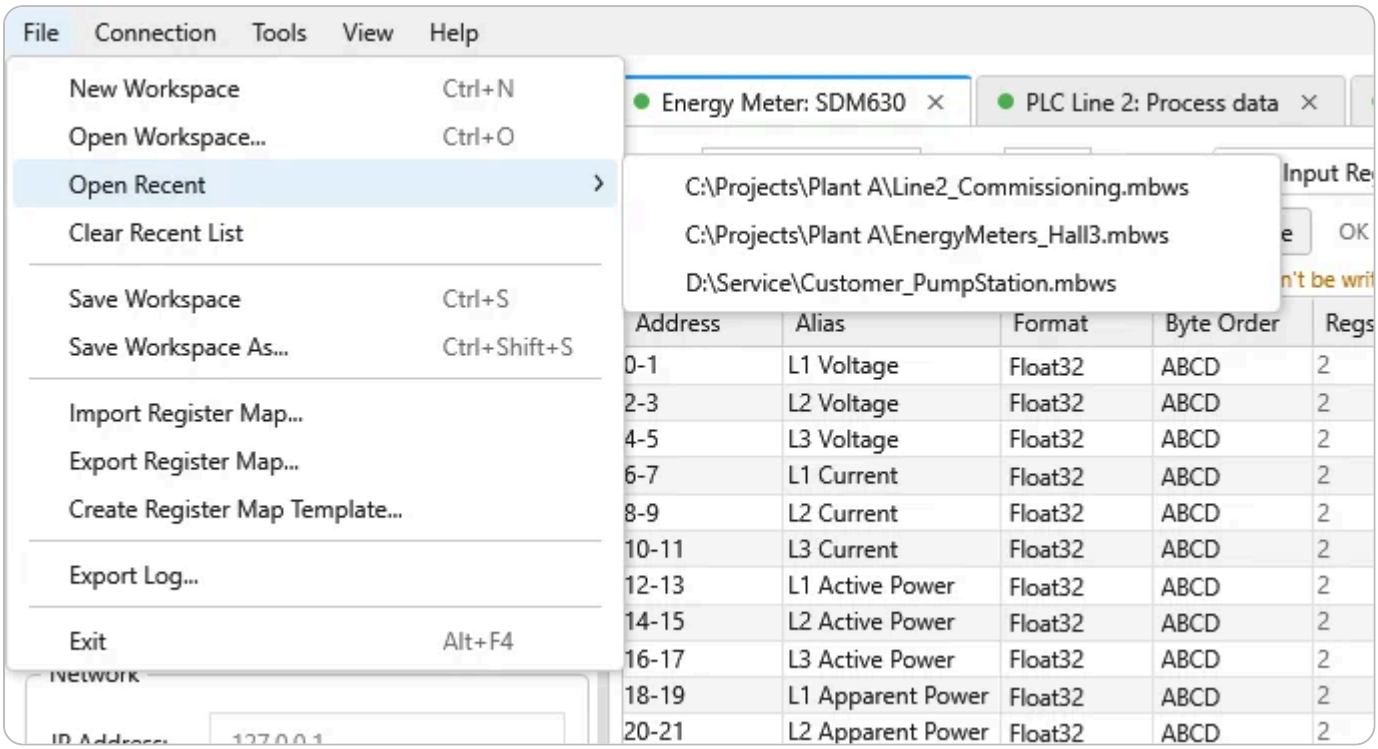
Tools > Data Logging > Export Current Data... saves the values of the last successful read of the active tab to a CSV file.

Export the log

File > Export Log... or **Export Log** under the communication log saves the log lines to a file.

18. Workspaces

A **workspace** (`.mbws` file) saves your whole setup: all connections with their settings, all poll tabs, names, formats, scaling and units.



File menu with the workspace commands.

Menu	Shortcut	What it does
File > New Workspace	`Ctrl+N`	Starts an empty setup.
File > Open Workspace...	`Ctrl+O`	Opens a `.mbws` workspace. Old 1.x profiles (`.json`, `.mbprofile`) and `.mbcfg` files open too.
File > Open Recent		The last workspaces. **Clear Recent List** empties the list.
File > Save Workspace	`Ctrl+S`	Saves to the current file.
File > Save Workspace As...	`Ctrl+Shift+S`	Saves to a new file.

To start polling automatically when a workspace opens, tick **Connect when the workspace is opened** in each connection's **Auto-reconnect** section, then save. The CLI can run a workspace too: `ModbusBB.CLI workspace run plant.mbws`.

19. Command-line tool (CLI)

The CLI (`ModbusBB.CLI.exe`) does the same work as the program, in a terminal. Use it for scripts, scheduled tasks and automatic tests. It is in the `CLI` folder of the installation. If you ticked **Add CLI to system PATH**, you can type `ModbusBB.CLI` anywhere. The full reference is on the CLI page.

The basic pattern

```
ModbusBB.CLI <command> [options] <connection>
```

The connection options are the same for every command:

Option	Meaning
`--tcp host[:port]`	Modbus TCP (default port 502)
`--udp host[:port]`	Modbus UDP
`--rtu-over-tcp host:port`	RTU frames through a TCP serial gateway
`--rtu COM3` / `--ascii COM3`	Serial, with `--baud` (default 9600), `--parity N|E|O|M|S`, `--databits 7|8`, `--stopbits 1|1.5|2`
`-u, --unit <id>`	Slave / unit ID, 1–247 (default 1)
`-t, --timeout <ms>`	Response timeout, 10–60000 ms (default 1000)
`--retries <n>`	Retries per request, 0–10 (default 3; scans and stats use 0)
`--profile file.mbws`	Take the connection from a workspace (`--connection name` selects one)

Example: read Float32 values and see the raw frames

```
Windows PowerShell

PS C:\Plant> ModbusBB.CLI read --tcp 192.168.1.10 --unit 1 --input --address 0 --count 6 --format float32
Address  Value    Raw
--------------------------
0        233.677  4369 AD64
2        232.637  4368 A319
4        231.322  4367 528B

PS C:\Plant> ModbusBB.CLI read --tcp 192.168.1.10 --unit 1 --address 0 --count 2 --trace
[10:45:11.666] TX 00 01 00 00 00 06 01 03 00 00 00 02
[10:45:11.695] RX 00 01 00 00 00 07 01 03 04 03 F5 00 F2
Address  Value  Hex     Binary
--------------------------------------
0        1013   0x03F5  0000001111110101
1        242    0x00F2  0000000011110010
```

`read` with Float32 format, and `read --trace` showing the TX and RX bytes.

Example: log to CSV and scan for devices

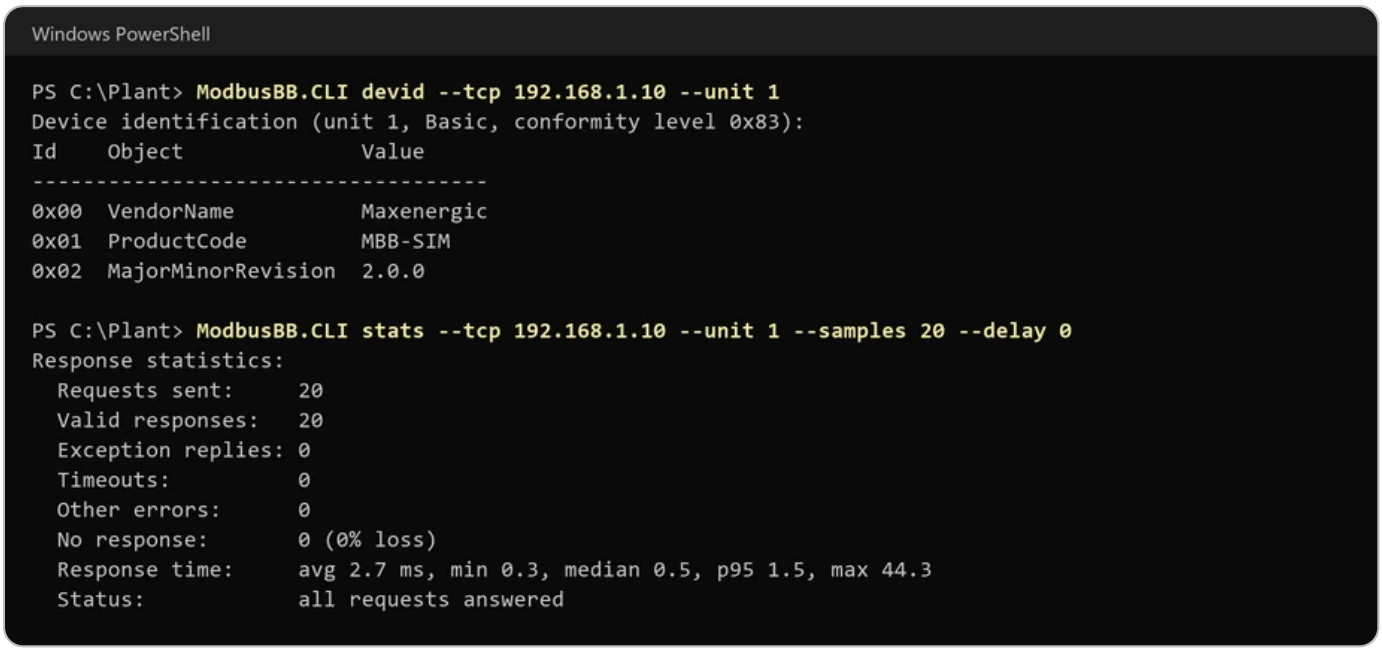
```
Windows PowerShell

PS C:\Plant> ModbusBB.CLI poll --tcp 192.168.1.10 --unit 1 --input --address 0 --count 2 --format float32 --interval 500 --samples 4 --output csv
timestamp,unit,status,response_ms,error,0
2026-10-01T10:45:03.231+03:00,1,ok,43.44,,233.85
2026-10-01T10:45:03.701+03:00,1,ok,1.01,,233.627
2026-10-01T10:45:04.190+03:00,1,ok,1.62,,233.419
2026-10-01T10:45:04.697+03:00,1,ok,1.37,,233.117

PS C:\Plant> ModbusBB.CLI scan --tcp 192.168.1.10 --start 1 --end 5 --timeout 200
Found 2 responding unit(s):
  Unit 1
  Unit 2
```

`poll --output csv` prints one line per sample; `scan` finds unit IDs.

Example: device identification and statistics



`devid` and `stats` against the simulator.

Example: poll several groups at once (pollset)

`pollset` polls several register groups on one connection at the same time. Give one `--group` per group, as a list of `key=value` fields:

Field	Meaning
`address`	Required. 0–65535, decimal or `0x` hex
`type`	`holding` (default), `input`, `coils` or `discrete`. Or use `fc` 1–4
`count`	Registers or bits to read (default: one value of the format, else 10)
`format`, `order`	Data type (`uint16`, `int16`, `hex`, `float32`, `int32`, ..., `string`) and byte order (`ABCD`, `CDAB`, ...). Registers only
`unit`	Slave ID 1–247 (default `--unit` or 1)
`interval`	10–3600000 ms (default `--interval`, which is 1000 if not set)
`name`	Label in the output

```
ModbusBB.CLI pollset --rtu COM3 --unit 1 ^ --group "name=Power,address=12,format=fl
```

- It runs until Ctrl+C, `--samples` (total over all groups) or `--duration` (seconds).
- `--output csv` writes one row per value: `timestamp,group,unit,function,address,value,raw,status,error`. `--output json` writes one JSON object per sample (NDJSON).
- If the link drops, it is opened again automatically.
- `--max-errors n` stops a group after n errors in a row (default 0 = never).
- The exit code is 1 if a group was stopped by `--max-errors`, if a poll stopped because the link could not be restored, or if a group that ran never got one good answer. So a script also sees a partial failure.

All commands

Command	What it does	Example
read	Read coils, discrete inputs, holding or input registers (FC01–04).	`read --tcp 192.168.1.10 --unit 1 --holding --address 0 --count 10`
write	Write holding registers or coils (FC05/06/15/16). `--dry-run` shows the frame without sending.	`write --tcp 192.168.1.10 --address 100 --format float32 --byte-order CDAB --value 21.5`
poll	Read repeatedly and print every sample. Stops with Ctrl+C, `--samples` or `--duration`.	`poll --tcp 192.168.1.10 --input --address 0 --count 2 --format float32 --output csv > log.csv`

Command	What it does	Example
pollset	Poll several register groups at once on one connection, each with its own interval. Output as table, CSV or JSON (see above).	`pollset --tcp 192.168.1.10 --group "name=Power,address=12,format=float32,interval=500" --group "name=Status,type=coils,address=0,count=8" --output csv > log.csv`
scan	Find responding unit IDs (default) or readable register ranges (`--registers`).	`scan --rtu COM3 --baud 19200 --timeout 150 --start 1 --end 32`
stats	Measure response time and loss.	`stats --tcp 192.168.1.10 --samples 100 --delay 0`
devid	Read Device Identification (FC43).	`devid --tcp 192.168.1.10 --category regular`
slaveid	Report Server/Slave ID (FC17).	`slaveid --rtu COM3 --unit 4`
diag	Diagnostics (FC08); sub-function 0 = echo test.	`diag --rtu COM3 --unit 1 --sub 0 --data "12 34"`
maskwrite	Mask Write Register (FC22).	`maskwrite --tcp 192.168.1.10 --address 4 --and 0xFFF0 --or 0x0005`
rw	Read/Write Multiple Registers (FC23).	`rw --tcp 192.168.1.10 --read-address 0 --read-count 4 --write-address 10 --value 1,2,3`
raw	Send any function code + data and print the frames.	`raw --tcp 192.168.1.10 --unit 1 --fc 3 --data "00 00 00 02"`
connect / disconnect / status	Interactive mode: open, close and show a connection. One-shot: test a connection.	`connect --tcp 192.168.1.100:502 --unit 1`

Command	What it does	Example
ports	List serial ports.	`ports`
library	Show or export the built-in register maps.	`library show eastron-sdm630`
workspace	Run or inspect a workspace (.mbws).	`workspace run plant.mbws --duration 3600 --output csv > plant.csv`
simulate	Run the slave simulator until Ctrl+C.	`simulate --tcp 5020 --unit 1,2 --generator "holding:0:sine:0:100:5000"`
gateway	Upload a workspace to ModbusBB SCADA Cloud 24/7 (Windows service). See chapter 25.	`gateway setup --workspace plant.mbws`
help / version	Help for a command or topic (`help connection`, `help scripting`, `help exit-codes`).	`help read`

Output formats

`--output table` (default, for people), `--output csv` (one row per value or sample) or `--output json` (one JSON object; `poll` writes one object per line). Data goes to stdout; messages, errors and `--trace` frames go to stderr. Colors switch off automatically when the output is redirected, or with `--no-color`.

Exit codes

Code	Meaning
0	Success
1	Runtime error: connection failed, timeout, Modbus exception reply, I/O error

Code	Meaning
2	Usage error: unknown option, missing or invalid value
3	License error: trial expired / no valid license
130	Stopped with Ctrl+C (`poll`, `pollset`, `simulate` and `workspace run` exit with 0 when stopped by Ctrl+C)

Scripting tips

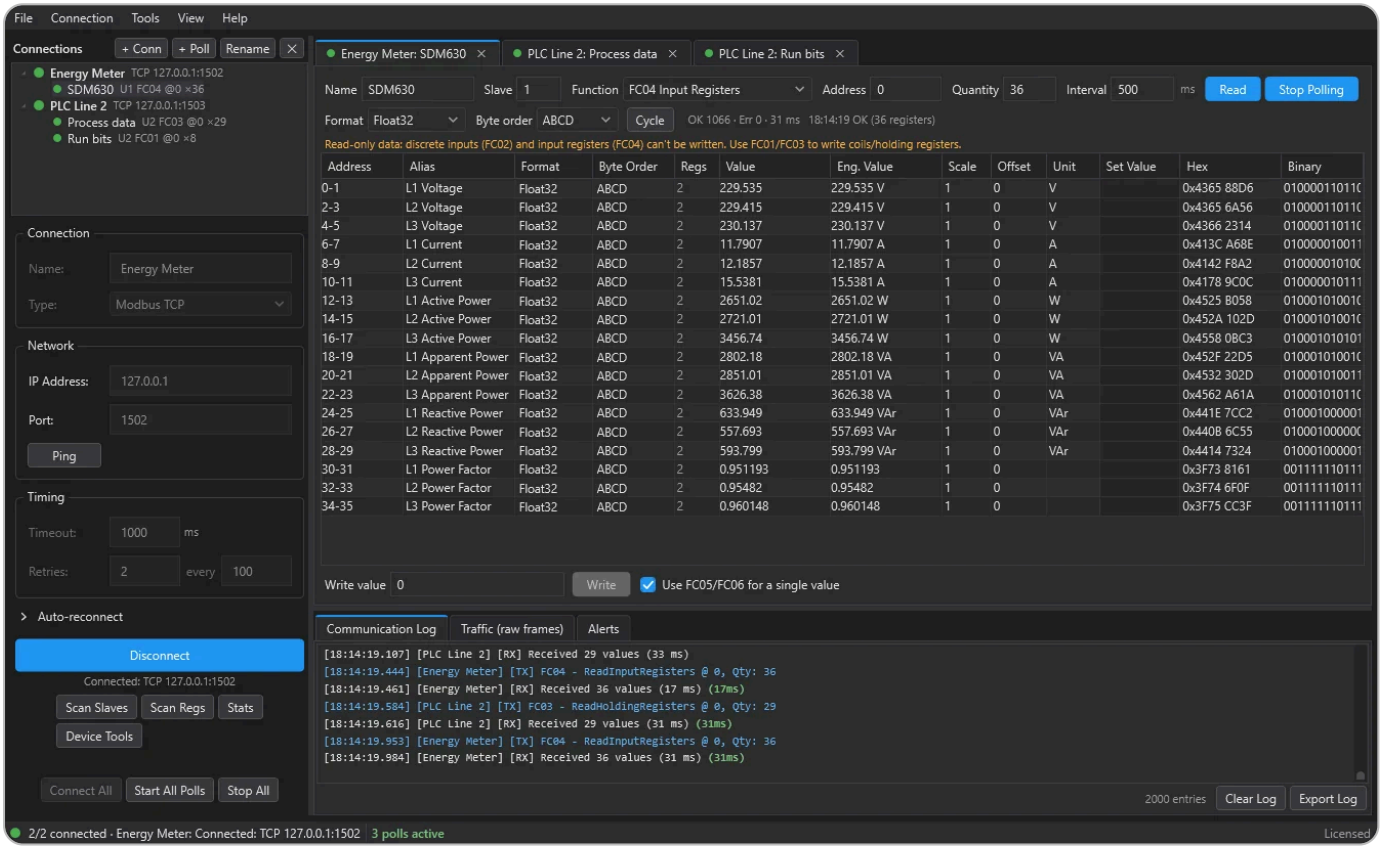
- Set default connection options once: `set MODBUSBB_OPTS=--tcp 192.168.1.10 --unit 1` (PowerShell: `$env:MODBUSBB_OPTS="--tcp 192.168.1.10 --unit 1"`).
- Use `--quiet` to hide the banner and `--no-prompt` (or `MODBUSBB_NO_PROMPT=1`) so the CLI never waits for input.
- Check `%ERRORLEVEL%` (cmd) or `$LASTEXITCODE` (PowerShell) after each command.
- Without a command, `ModbusBB.CLI` starts an interactive mode: `connect` once, then type commands without connection options.

```
foreach ($u in 1..5) { $r = ModbusBB.CLI read --rtu COM3 --unit $u --input --addres
```

The trial limits apply to the CLI as well. With a license, the CLI checks the license online every 7 days.

20. Themes and view options

- **View > Dark Theme**: switches between light and dark. ModbusBB remembers your choice.
- **View > Show Hex Addresses**: addresses in hex.
- **View > Address Notation**: 0-based, 1-based or Modicon (display only).
- **View > Communication Log**, **Traffic Monitor (raw frames)**, **Alerts**: jump to that bottom tab.
- **Help > Keyboard Shortcuts**: the list of shortcuts (see Appendix C).



The dark theme.

21. License

Trial

The trial lasts 30 days from the first start. Each session is limited to 15 minutes. All functions work during the trial.

Buy and activate

1. Buy a license on the buy page. You receive the key by e-mail.
2. Start ModbusBB. In the license window, type the key (xxxxx-xxxxx-xxxxx-xxxxx).
3. Click **Activate License**. The PC must be online.
4. The status bar shows **Licensed**.

How many computers?

One license can be active on **3 computers** at the same time. Changing small hardware parts does not count as a new computer.

Move the license to another computer

1. Open the license portal and sign in with your license key.
2. Deactivate the computer you no longer use. You can deactivate up to 5 computers in 30 days.
3. Activate the key on the new computer.

Working offline

On activation the license server issues a digitally signed license file for this computer, which ModbusBB checks offline. ModbusBB re-checks it online every 7 days and keeps working offline for up to **30 days** after the last successful check. After updating from 2.0.x, it needs internet once to get its license file. If the server rejects the key (for example because it was refunded or is used on too many computers), ModbusBB returns to trial mode. A network error never removes your license.

Lost key?

Open Have your key e-mailed again (also linked from the license portal), or write to support@maxenergic.com.

22. Updates

ModbusBB checks for updates once a day at start. When a new version exists, it tells you. You can also check by hand: **Help > Check for Updates**.

1. Click **Download Update**. The download page opens in your browser.
2. Download the new installer.
3. Close ModbusBB and run the installer over the old version. Your license and settings stay.

23. Troubleshooting

Problem	Likely cause	What to do
Every read times out	Wrong IP/port, wrong slave ID, device off, firewall, or wrong serial settings.	Ping the IP. Check the slave ID (try **Scan Slaves**). For serial, check baud rate, parity and stop bits. Increase the **Timeout**.
Some reads time out	Line too slow or too busy.	Increase **Interval** and **Timeout**. Read fewer registers per poll. On RS-485, check termination and cabling.
Exception 01 Illegal Function	The device does not support this function code.	Try FC03 instead of FC04 (or the opposite). Check the device manual.
Exception 02 Illegal Data Address	The address or the range does not exist.	Check the 0-based address (40001 → 0). Reduce **Quantity**. Use **Scan Regs**.
Exception 03 Illegal Data Value	A value or quantity in the request is not allowed.	Check the written value and the quantity.
Values are huge, tiny or nonsense	Wrong byte order or format.	Click **Cycle**. Try Float32 vs Int32. Compare the hex value with the manual.
Value is 10× or 100× too big	Missing scale factor.	Set **Scale** (for example 0.1) in the row.
Values are off by one register	1-based address used as 0-based.	Subtract 1 from the address in the manual.
COM port can't be opened	Another program uses the port, or the adapter was unplugged.	Close other Modbus programs. Click **Refresh**. Check Windows Device Manager.
Serial: no answer at all	A/B wires swapped, wrong baud rate, no common ground.	Swap A and B. Check all serial settings. Connect the ground (GND/COM).
Serial: random CRC errors	No termination, long stub lines, noise.	Use a 120 Ω terminator at both ends of the bus. Use twisted-pair cable. Lower the baud rate.

Problem	Likely cause	What to do
TCP connects, then drops	Device allows only one or a few connections, or closes idle links.	Close other masters. Keep **Reconnect automatically** on.
Simulator won't start on port 502	Port 502 is in use or needs administrator rights.	Use another port, for example 5020.
Write is not possible	Tab uses FC02 or FC04 (read-only).	Use FC01 for coils or FC03 for holding registers.
Program closes after 15 minutes	Trial session ended.	Start it again, or activate a license.
Activation fails	No internet, key mistyped, or 3 computers already active.	Check the connection and the key. Deactivate an old computer in the license portal.

Look at the bytes

For any unclear problem, open the **Traffic (raw frames)** tab. If there are TX frames but no RX frames, the device does not answer. If RX frames contain an exception (function code + 0x80), the device answers but refuses the request.

RS-485 wiring in short

- Connect all devices in a line (daisy chain), not as a star.
- A to A (often D−/TX−) and B to B (often D+/TX+). Names differ between vendors; if nothing answers, swap A and B.
- Put a 120 Ω terminating resistor at both ends of the line.
- Connect the signal ground of all devices.
- Every device on the line needs its own slave ID and the same serial settings.

More: RS-485 wiring guide, timeout guide, error codes guide.

24. FAQ

Can I connect to several devices at the same time? +

Can ModbusBB act as a Modbus slave? +

Does ModbusBB send my data to the internet? +

Does it run on Linux or macOS? +

Why is the address in my manual 40001 but ModbusBB uses 0? +

Can I use the license on a laptop and a desktop? +

Do I lose my license when I reinstall? +

Can I script ModbusBB? +

25. ModbusBB SCADA Cloud

ModbusBB SCADA Cloud (scada.maxenergic.com) turns any Windows PC with ModbusBB into a 24/7 gateway. A Windows service on the PC polls your devices and uploads the values. In the browser you get dashboards, live values, charts with up to 2 years of history, CSV download and email alerts, and you can invite team members. More on the ModbusBB SCADA Cloud page.

Item	Details
Price	$59 the first year, then $89 per year, paid with PayPal. Cancel anytime.
Included licence	A lifetime ModbusBB desktop licence. You keep it when you cancel.
Limits	10 gateways, 5 team members, 500 MB of data.
Data location	Hosted in the EU (France).
Requires	ModbusBB 2.2.0 or newer on the gateway PC.

Monitoring only

The cloud never writes to your devices. Do not use it for safety-critical functions.

Step by step: connect a PC

1. Register at scada.maxenergic.com and subscribe.
2. On the web, open **Gateways** > **Add gateway**. You get a code. It is valid 15 minutes and works once.
3. In ModbusBB, open **Cloud** > **ModbusBB SCADA Cloud...**.
4. In **1. Connect this PC**, type the **Enrollment code** and a **Gateway name** (how this PC is named on the web), then click **Connect this PC**.
5. In **2. What to upload**, choose the **Workspace** (**Browse...** or **Use open workspace**) and the connections.

6. Set **Store every** (1, 5, 15, 30 or 60 min) and **Live values every** (5–60 s), then click **Save settings**.

7. In **3. Upload 24/7 (Windows service)**, click **Install and start**. Windows asks for administrator rights. Click **Yes**.

8. Open **Cloud > Open Cloud Dashboard** to see the values in the browser.

The workspace must be saved

The service reads the `.mbws` file, not the ModbusBB window. Only polls that were polling when the workspace was saved are uploaded. Changes to the workspace or the settings are picked up when the service restarts.

How the gateway works

- The service is called **ModbusBBGateway**. It runs while Windows runs, also when ModbusBB is closed. ModbusBB itself does not upload.
- The **Start**, **Stop** and **Uninstall** buttons control the service. They also ask for administrator rights.
- When the internet is down, the data is kept on the PC (up to 30 days, about 1 GB) and uploaded when the PC is back online.
- **Disconnect this PC** removes the gateway key from the PC. Stop the service first. The gateway stays listed on the web until you remove it there. To connect again you need a new code.

Files

What	Where
Gateway settings	`%ProgramData%\ModbusBB\cloud\gateway.json`
Gateway key (encrypted)	`%ProgramData%\ModbusBB\cloud\gateway.key`
Data not uploaded yet (outbox)	`%ProgramData%\ModbusBB\cloud\outbox.db`
Last gateway status	`%ProgramData%\ModbusBB\cloud\status.json`

What	Where
Logs (one file per day, kept 7 days)	`%ProgramData%\ModbusBB\logs\gateway-YYYYMMDD.log`

The gateway from the CLI

The CLI can set up and run the gateway without the window. `install`, `uninstall`, `start` and `stop` need an administrator prompt.

Command	What it does	Example
gateway setup	Choose the workspace and settings: `--workspace`, `--interval` (1, 5, 15, 30, 60 min), `--live` (5–60 s), `--connection`, `--proxy`.	`gateway setup --workspace C:\Plants\solar.mbws --interval 5`
gateway enroll	Connect this PC with the code from the web. `--name` sets the gateway name (default: computer name).	`gateway enroll K7QM-2XPA --name "Plant 1"`
gateway key	Save a gateway key (`gw_...`) copied from the web instead of a code.	`gateway key gw_...`
gateway forget	Delete the gateway key from this PC.	`gateway forget`
gateway run	Run the gateway in this window until Ctrl+C (or `--duration` seconds).	`gateway run --duration 600`
gateway status	Show settings, key, service state, last upload and last problem.	`gateway status`
gateway install	Install and start the Windows service (starts with Windows).	`gateway install`
gateway uninstall	Stop and remove the Windows service.	`gateway uninstall`
gateway start / stop	Start or stop the Windows service.	`gateway stop`

The order is: 1. `setup`, 2. `enroll`, 3. `install` (or `run` in this window). `ModbusBB.CLI help gateway` shows all options.

Troubleshooting

Problem	What to do
"Unknown or expired code"	A code is valid 15 minutes and works once. Create a new one on the web (**Gateways > Add gateway**).
"This needs an administrator prompt"	Click **Yes** when Windows asks, or run the CLI from an administrator terminal (right-click Terminal > Run as administrator).
0 values, or "no enabled read polls"	Start the polls you want, save the workspace (`Ctrl+S`), then click **Save settings** and restart the service.
Changes are not uploaded	Save the workspace, then restart the service (**Stop**, **Start**).
The gateway shows "late" on the web	Check the status (**Cloud > ModbusBB SCADA Cloud...** or `ModbusBB.CLI gateway status`), the internet connection and that the service is running.
"Cloud is not active on this account"	Uploads are paused. Check the subscription on the web.
Other problems	Read the log in `%ProgramData%\ModbusBB\logs`.

Appendix A: Modbus function codes

Code	Name	In ModbusBB
01 (0x01)	Read Coils	Poll tab, CLI `read --coils`
02 (0x02)	Read Discrete Inputs	Poll tab, CLI `read --discrete`
03 (0x03)	Read Holding Registers	Poll tab, CLI `read --holding`
04 (0x04)	Read Input Registers	Poll tab, CLI `read --input`
05 (0x05)	Write Single Coil	Write (one coil), CLI `write --coil`
06 (0x06)	Write Single Register	Write (one register), CLI `write`
08 (0x08)	Diagnostics	Device Tools, CLI `diag`
15 (0x0F)	Write Multiple Coils	Write, CLI `write --coil`
16 (0x10)	Write Multiple Registers	Write, CLI `write`
17 (0x11)	Report Server (Slave) ID	Device Tools, CLI `slaveid`
22 (0x16)	Mask Write Register	Device Tools, CLI `maskwrite`
23 (0x17)	Read/Write Multiple Registers	Device Tools, CLI `rw`
43/14 (0x2B/0x0E)	Read Device Identification	Device Tools, CLI `devid`
any	Any function code	Device Tools > Raw Request, CLI `raw`

More: function codes guide.

Appendix B: Modbus exception codes

Code	Name	Meaning
01	Illegal Function	The device does not support this function code.
02	Illegal Data Address	The address (or part of the range) does not exist.
03	Illegal Data Value	A value in the request is not allowed (for example the quantity).
04	Server Device Failure	The device had an internal error.
05	Acknowledge	The device accepted a long command and is still working on it.
06	Server Device Busy	The device is busy. Try again later.
07	Negative Acknowledge	The device cannot do the requested programming function.
08	Memory Parity Error	The device found a memory error (file record functions).
0A	Gateway Path Unavailable	The gateway has no path to the target device.
0B	Gateway Target Device Failed to Respond	The gateway sent the request but the target did not answer.

A device signals an exception by returning the function code + 0x80 (for example 0x83 for FC03) followed by the exception code.

Appendix C: Keyboard shortcuts

Key	Action
`Ctrl+N`	New workspace
`Ctrl+O`	Open workspace
`Ctrl+S`	Save workspace
`Ctrl+Shift+S`	Save workspace as
`Ctrl+T`	Add a poll to the selected connection
`F5`	Read the active tab once
`F6`	Start / stop polling the active tab
`Ctrl+C`	Copy selected frames (traffic monitor)
`Enter`	Write the value typed in a Set Value cell
`Alt+F4`	Exit

In the watchdog editor: `Ctrl+N` new rule, `Ctrl+S` save rule. In the trend window: `Home` resets the view.

Appendix D: File locations

What	Where
Program (installer)	`C:\Program Files\ModbusBB`; the CLI is in its `CLI` folder
License, trial, theme and update-check data	`%LOCALAPPDATA%\ModbusBB` (kept when you uninstall)
Workspaces	Where you save them (`.mbws`)
Register maps	Where you save them (`.csv` or `.json`)
CSV logs, alert log	The files you choose in the program
SCADA Cloud gateway settings	`%ProgramData%\ModbusBB\cloud\gateway.json`
SCADA Cloud gateway key (encrypted)	`%ProgramData%\ModbusBB\cloud\gateway.key`
SCADA Cloud upload buffer (data not uploaded yet)	`%ProgramData%\ModbusBB\cloud\outbox.db`
SCADA Cloud service status	`%ProgramData%\ModbusBB\cloud\status.json`
SCADA Cloud gateway log (one file per day, kept 7 days)	`%ProgramData%\ModbusBB\logs\gateway-YYYYMMDD.log`

Appendix E: Glossary

ASCII (Modbus ASCII)

Serial Modbus variant that sends each byte as two text characters. Older and slower than RTU.

Baud rate

Speed of a serial line in bits per second, for example 9600.

Byte order

The order in which a device stores the bytes of a value that uses more than one register (ABCD, CDAB, BADC, DCBA).

Coil

A single on/off output bit that can be read and written (FC01, FC05, FC15).

CRC

Checksum at the end of an RTU frame. A wrong CRC means the frame was damaged.

Discrete input

A single on/off input bit that can only be read (FC02).

Exception

An error answer from a device: function code + 0x80 and an exception code.

Float32 / Float64

Decimal numbers in IEEE 754 format, using 2 / 4 registers.

Function code (FC)

The number in a Modbus request that says what to do, for example 03 = read holding registers.

Holding register

A 16-bit register that can be read and written (FC03, FC06, FC16).

Input register

A 16-bit register that can only be read (FC04).

Master / client

The device that asks (ModbusBB). The slave / server answers.

MBAP header

The 7-byte header in front of every Modbus TCP frame (transaction ID, protocol ID, length, unit ID).

Modicon notation

Old numbering with a prefix: 0xxxx coils, 1xxxx discrete inputs, 3xxxx input registers, 4xxxx holding registers.

PDU

Protocol Data Unit: the function code and data, without address/CRC or MBAP header.

Poll

A read request that ModbusBB repeats at a fixed interval.

Register

A 16-bit value in a Modbus device.

RS-485

The usual two-wire serial bus for Modbus RTU. Many devices can share one line.

RTU

Remote Terminal Unit: the binary serial Modbus variant with CRC.

RTU over TCP

RTU frames sent unchanged through a TCP connection, usually to a serial gateway.

Slave ID / unit ID

The address of a device on a line (1–247).

Workspace

A file (.mbws) that stores all connections, polls and settings.