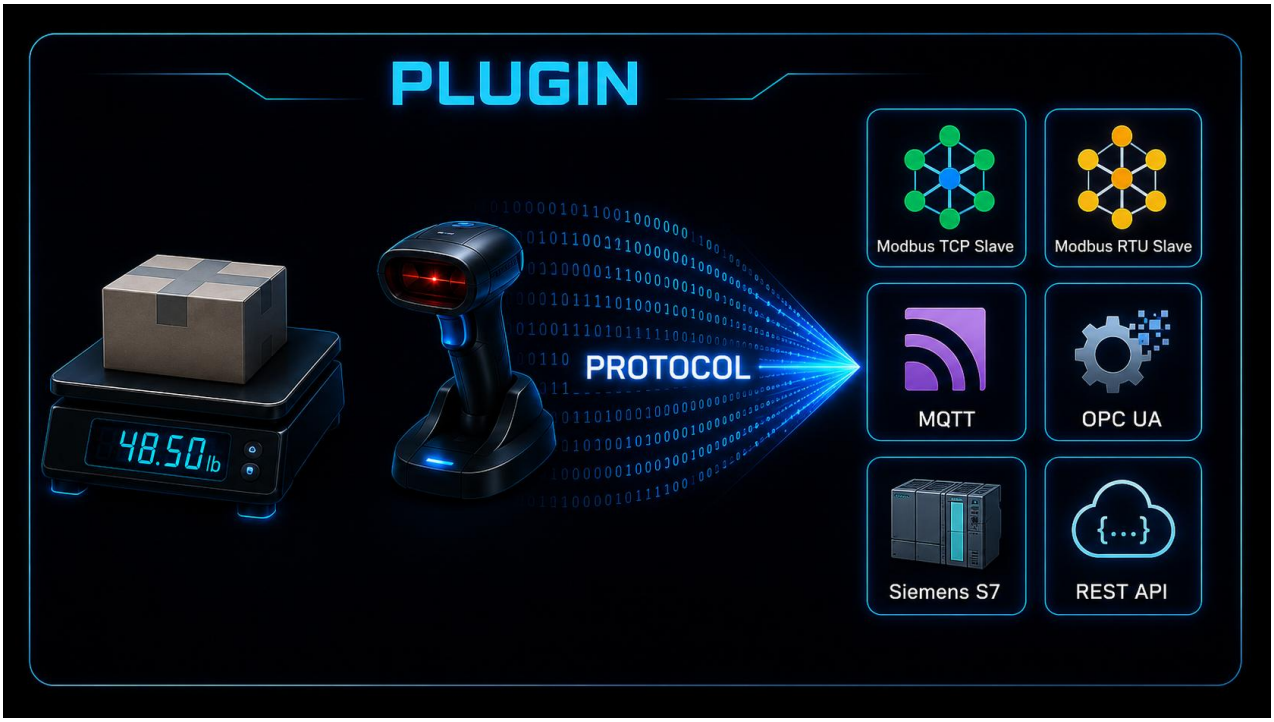



# BillProduction Protocol Plugin

## User Manual

**Version 2026/06/16**

BillProduction Softwares — www.billproduction.com

# 1. Overview

BillProduction Protocol Plugin is a gateway that receives numeric data from a device (scale, barcode scanner, caliper, counter…) through BillRedirect, and re-publishes that value on the industrial or IT protocol of your choice. One input — many possible outputs.

Software outputs available:

- Modbus TCP Slave
- Modbus RTU Slave (serial)
- MQTT
- OPC UA server
- Siemens S7 (write to a PLC data block)
- REST API (HTTP server + webhook)

## Data flow

Device → (RS-232 / USB / TCP) → BillRedirect → (TCP, port 20296) → BillProduction Protocol Plugin → (chosen protocol) → PLC / SCADA / MES / cloud.

# 2. Installation & automatic startup

Run the installer Setup_BillProductionProtocolPlugin.exe and follow the wizard (administrator rights are required). It installs the application and configures it to start automatically every time Windows starts.

How the program loads:

- On startup the plugin loads silently into the system tray (next to the clock). The main window does NOT open automatically — the gateway runs in the background.
- To open the main window, run the program a second time (double-click its shortcut or the .exe) or double-click the tray icon. Only one instance ever runs.
- Right-click the tray icon for Show / Quit.

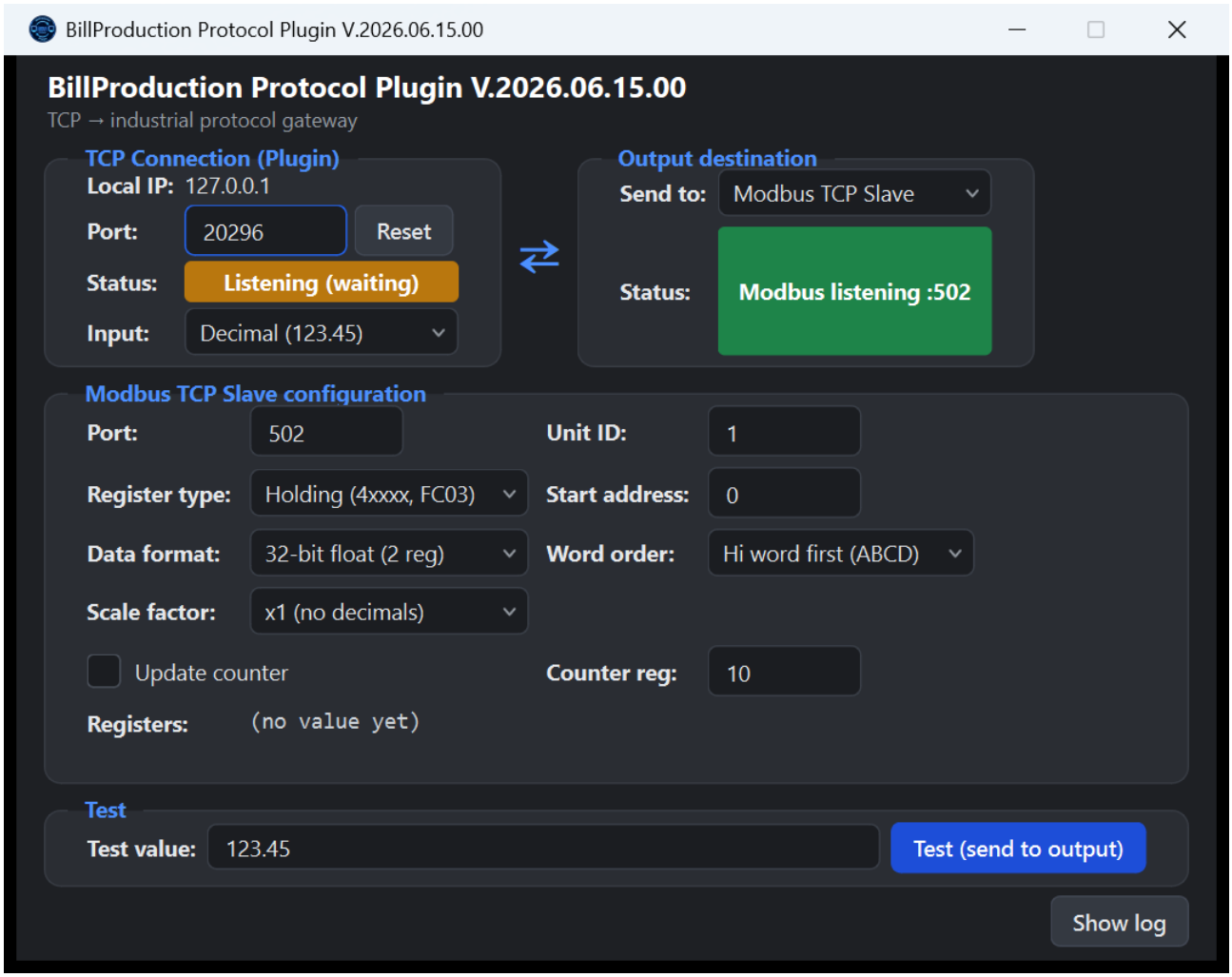
# 3. Configuring BillRedirect

In BillRedirect, configure your device input (for example the Serial Port for an RS-232 device, or the TCP input), then send that data to the plugin: check the Plugin box, set the Plugin section to ON, and click Save Configuration. BillRedirect transmits the data to the plugin over TCP on the port shown in the plugin (default 20296). BillRedirect can also redirect a TCP input to the plugin the same way.


Screenshot: a RS-232 device (Serial Port ON) redirected through Data Filtering to the Plugin (ON).

# 4. The main window

The window is organized in clear sections: the TCP Connection (input from BillRedirect), the Input format selector, the Output destination selector with its configuration panel, a Test box, and a Show log button. Closing the window minimizes the app to the system tray — the gateway keeps running.




## TCP Connection (input)

- Local IP — always 127.0.0.1 (BillRedirect runs on the same PC).
- Port — default 20296. Must match the port configured in BillRedirect.
- Status — Listening (waiting) / Connected / Stopped. Reset restarts listening.

## Input format (Decimal / Hexadecimal)

- Decimal — a number with or without decimals, e.g. 123.45.
- Hexadecimal — a base-16 integer, up to 64-bit, e.g. 1A2B becomes 6699.

The value is interpreted here before it is sent to the selected output.

## Output destination

Use the “Send to:” selector to choose where the value goes; the matching configuration panel appears below. Only one destination is active at a time. The Status badge shows the output state. Settings are applied and saved immediately — there is no Apply button.

# 5. Output protocols

## Modbus TCP Slave

The app acts as a Modbus TCP server; a PLC or SCADA polls it to read the value.

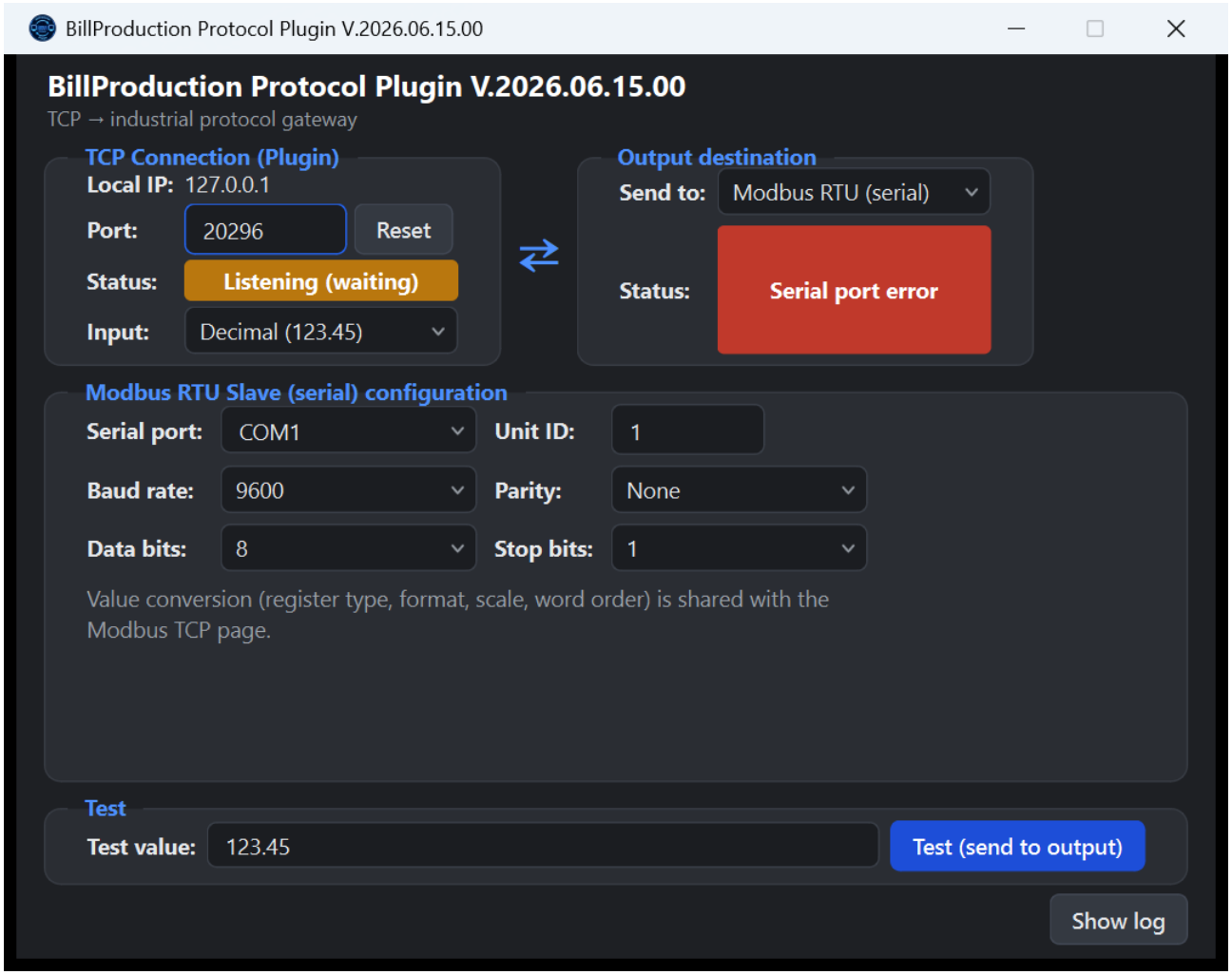
- Port — default 502.
- Unit ID — Modbus slave address.
- Register type — Holding (FC03) or Input (FC04).
- Data format — 16-bit int, 32-bit int, or 32-bit float.
- Start address, Scale factor, Word order.

*▶ A scale sends 48.50 → with 32-bit float at register 4'0000, the PLC reads 48.50.*

## Modbus RTU Slave (serial)

Same as Modbus TCP, but over a serial port (RS-232/485).

- Serial port (COM), Baud rate, Parity, Data bits, Stop bits, Unit ID.
- The value conversion is shared with the Modbus TCP page.

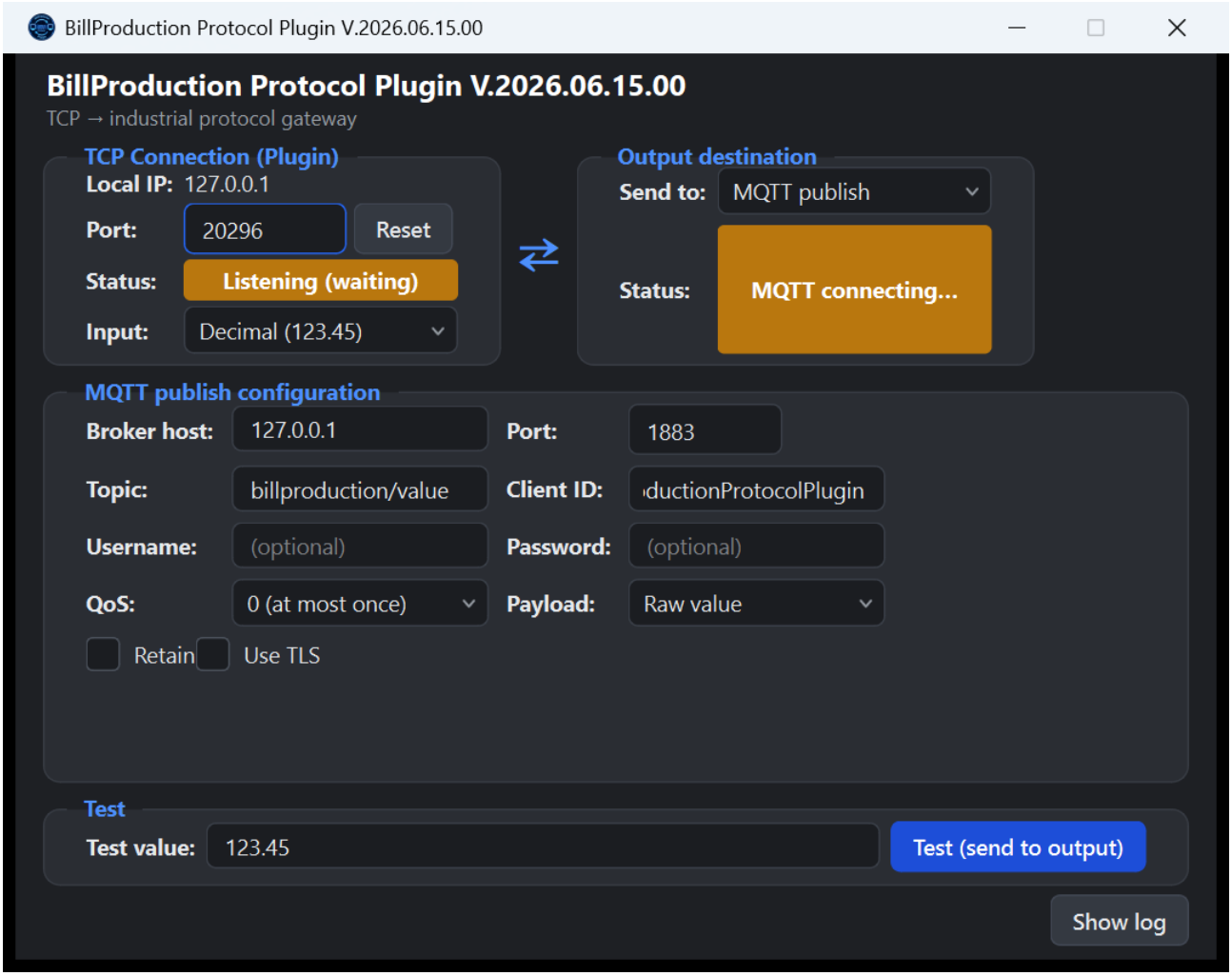


## MQTT

The app publishes the value to an MQTT broker.

- Broker host / Port (default 1883).
- Topic, Client ID, Username / Password.
- QoS (0 or 1), Payload (Raw value or JSON), Retain, Use TLS.

▶ *Topic billproduction/value, JSON payload {"value":48.50,"seq":7,"ts":"..."}.*

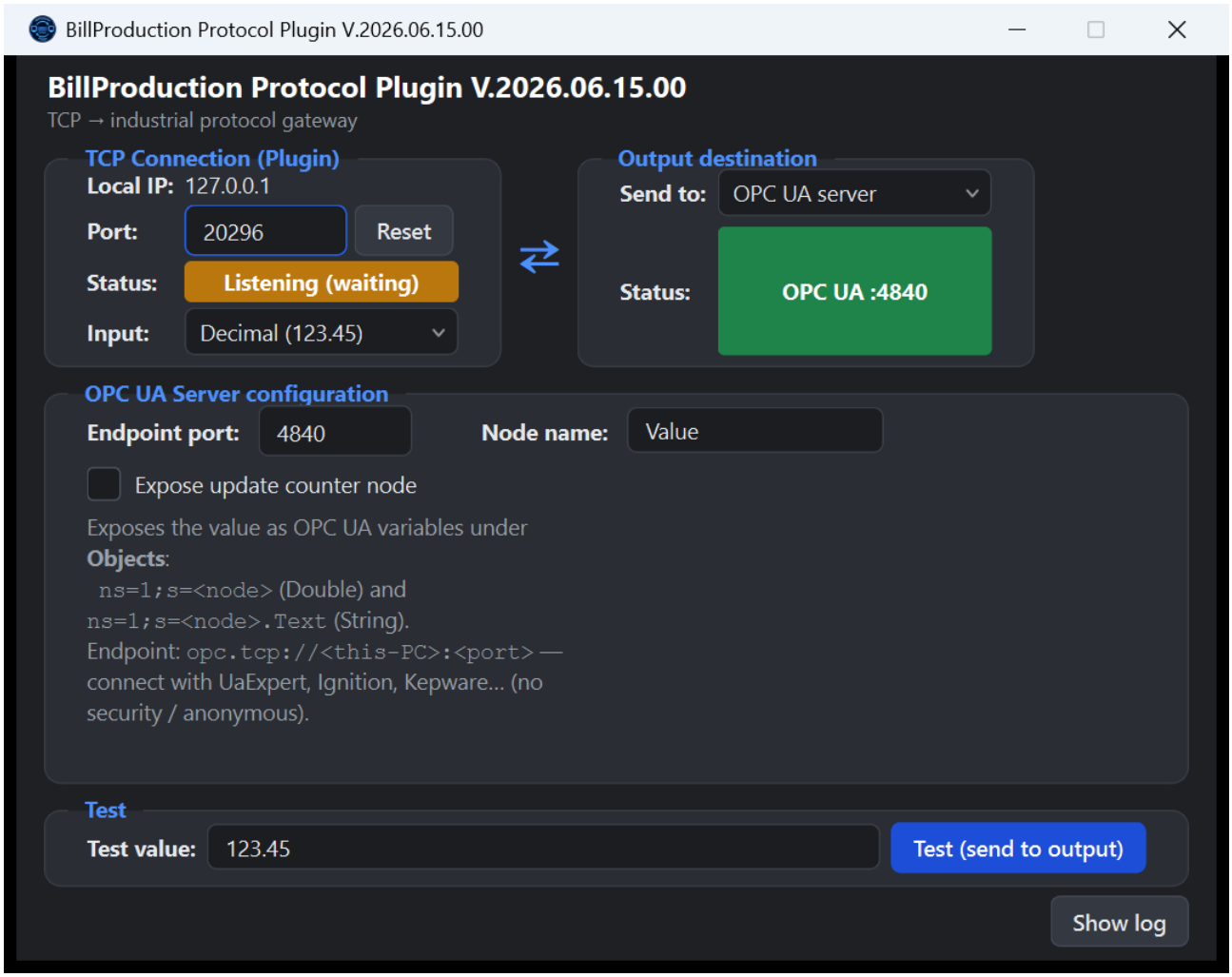



## OPC UA

The app is an OPC UA server; clients (UaExpert, Ignition, Kepware…) read the value.

- Endpoint port — default 4840.
- Node name — exposes ns=1;s=<node> (Double) and <node>.Text (String).

Endpoint: opc.tcp://<this-PC>:4840 — no security / anonymous.

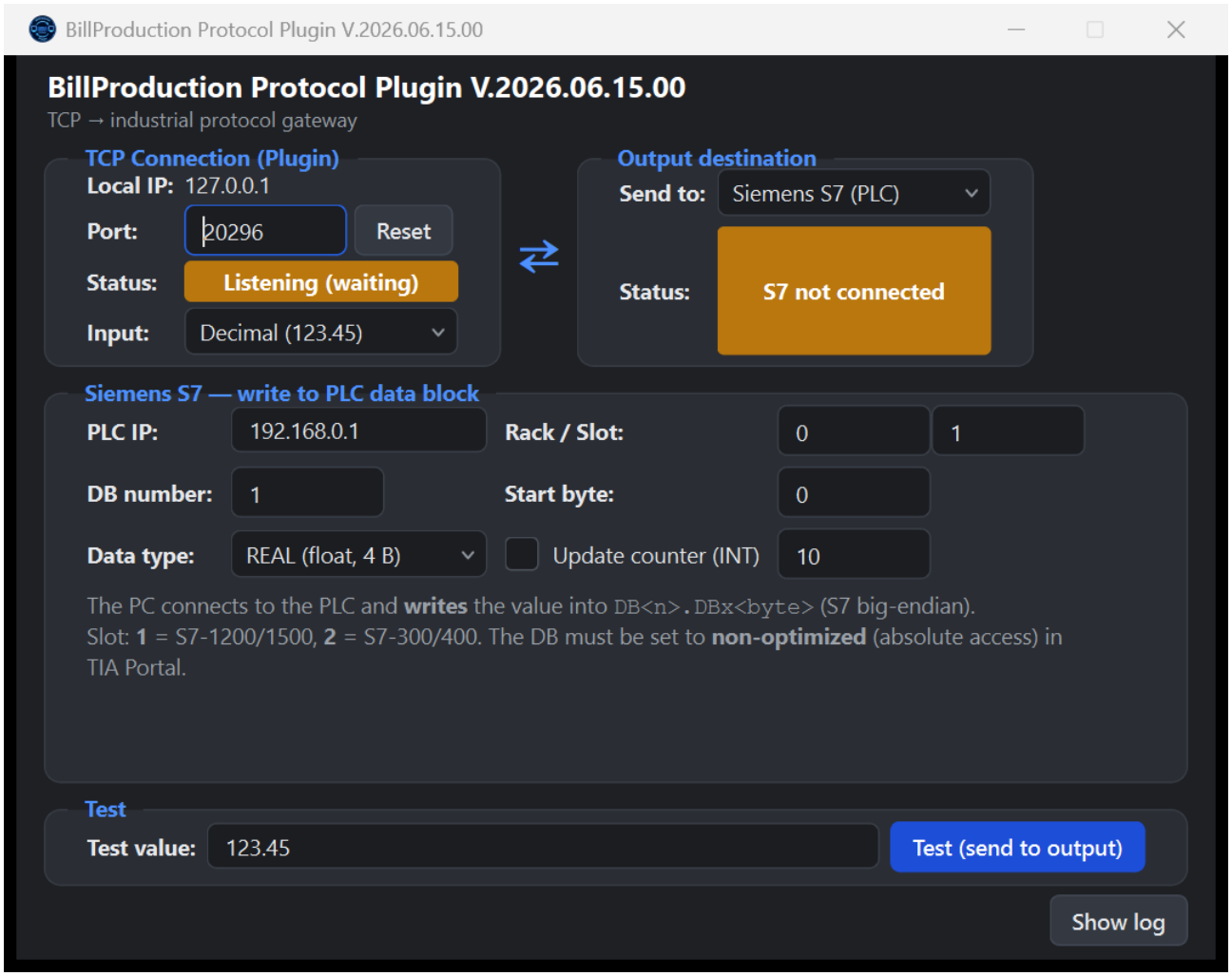



## Siemens S7

The app is an S7 client that connects to a Siemens PLC and writes the value into a data block.

- PLC IP, Rack / Slot — slot 1 for S7-1200/1500, slot 2 for S7-300/400.
- DB number, Start byte, Data type (REAL / DINT / INT).

The DB must be set to non-optimized (absolute access) in TIA Portal.

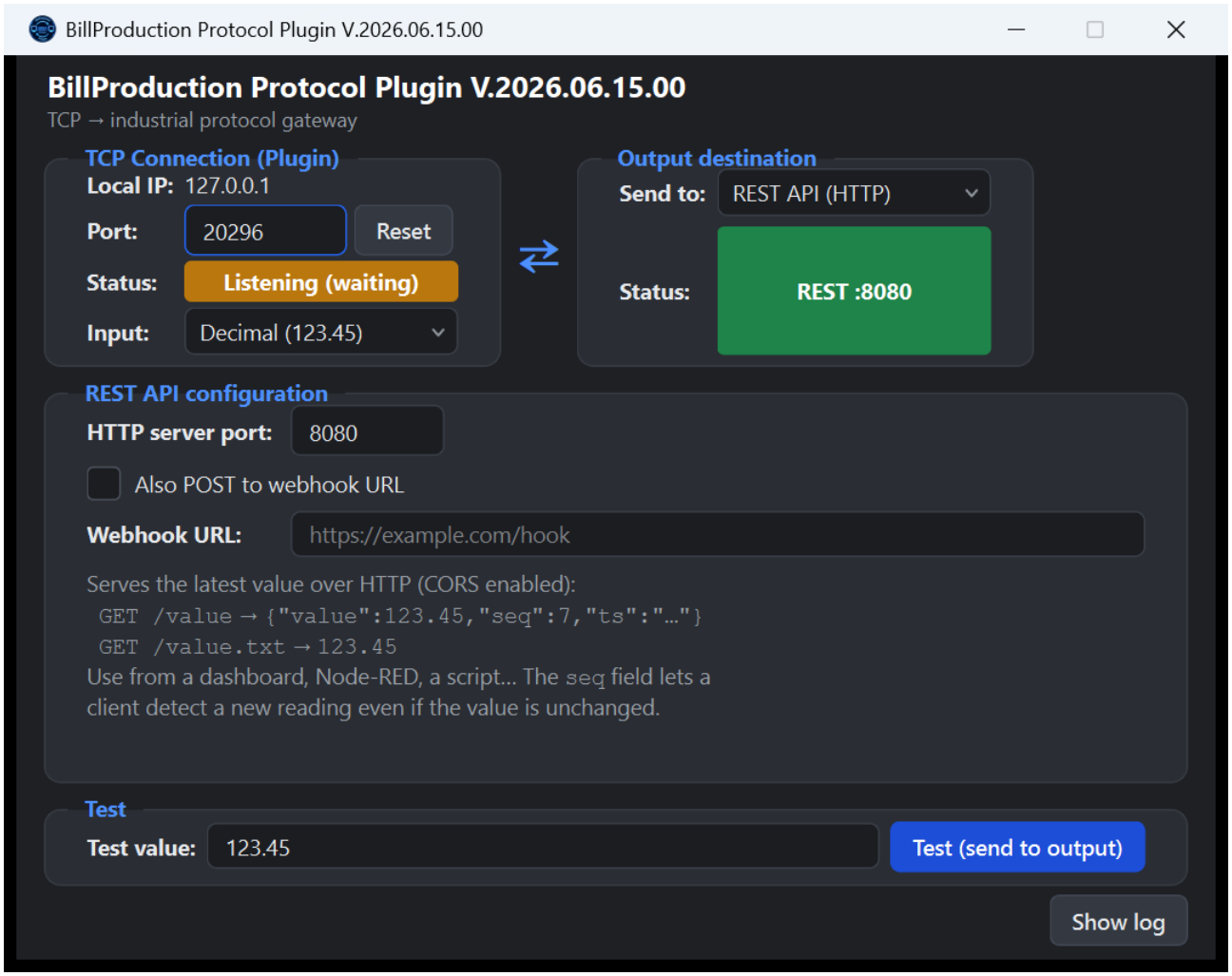



## REST API

The app serves an HTTP endpoint and can also push a webhook.

- GET /value → {"value":48.50,"seq":7,"ts":"..."} (JSON, CORS enabled).
- GET /value.txt → 48.50 (plain number).
- Optional webhook: POST the same JSON to a URL on every value.

▶ *From a dashboard: fetch('http://<PC>:8080/value').then(r => r.json()).*


BillProduction Protocol Plugin V.2026.06.15.00
BillProduction Protocol Plugin V.2026.06.15.00
TCP → industrial protocol gateway
TCP Connection (Plugin)
Local IP: 127.0.0.1
Port: 20296
Reset
Status: Listening (waiting)
Input: Decimal (123.45)
Output destination
Send to: REST API (HTTP)
Status: REST :8080
REST API configuration
HTTP server port: 8080
Also POST to webhook URL
Webhook URL: https://example.com/hook
Serves the latest value over HTTP (CORS enabled):
GET /value → {"value":123.45,"seq":7,"ts":"…"}
GET /value.txt → 123.45
Use from a dashboard, Node-RED, a script… The seq field lets a client detect a new reading even if the value is unchanged.
Test
Test value: 123.45
Test (send to output)
Show log

# 6. Update counter

On polled protocols (Modbus, S7) or subscriptions (OPC UA), an identical value sent twice looks the same to the master. Enable Update counter (optional, per output) to expose a counter that increments on every received value — the master detects a new reading even if the value is unchanged.

- Modbus — a register at a configurable address.
- Siemens S7 — an INT word in the same DB.
- OPC UA — a <node>.Counter node.
- MQTT / REST — the seq field in the JSON.

# 7. Test and Log

The Test box sends a value to the active output without a real device — useful to verify your configuration. The Show log button reveals an activity log (received values, output writes, errors); click it again to hide it.

# 8. System tray

Closing the window does not stop the gateway — it returns to the system tray and keeps running. Double-click the tray icon to show the window again, or right-click it for Show / Quit. To stop the gateway completely, choose Quit.