HOW-TO
Integration via Modbus

Important Note

Services like REST API, MQTT and Modbus require a license.

Please visit www.whatwatt.ch/pricing for more information.

Version 1.0

Date 26/03/2025

whatwatt.ch
info@whatwatt.ch

whatwatt

Version	Date	Author	Changes
1.0	26.03.2025	SJ	Initial Version

1. Introduction

1.1. Modbus

Starting from firmware version 1.2.20, the device supports Modbus TCP in slave mode. Enabling or disabling the Modbus service (and selecting the port it runs on) is done via the REST API at api/v1/settings. In the .services.modbus object, the field enable (boolean) toggles the Modbus service, and the field port specifies the port number (default is 502). Once the Modbus service is enabled, it is advertised via mDNS.

The device has a default Modbus register layout (refer to the Default Modbus Configuration section) but also allows for defining a custom register layout.

1.2. Summary

This document provides comprehensive technical guidance regarding the support and configuration of the Modbus TCP protocol implemented in the device. It covers the detailed operation of the device in slave mode, including the methods to enable, disable, and configure the Modbus service using the REST API, along with automatic network discovery through mDNS.

Specifically, the document includes:

Configuration Methods

- Instructions for enabling or disabling the Modbus service and modifying the default network port via REST API.
- A description of REST API methods (GET, POST, DELETE) used for managing Modbus register configurations.

Modbus Register Configuration

- Detailed JSON-based structure for defining custom register configurations, including data types, scaling factors, and measurement identifiers such as energy, power, voltage, and current, accompanied by illustrative examples.

Default Register Configuration

- A predefined register layout compatible with industry standards (e.g., Siemens PAC2200), facilitating immediate operational readiness without additional configuration.

Technical Specifications

- Supported Modbus functions, specifically the Read Input Registers function.
- Compliance with Modbus TCP data transmission standards, including Big-Endian byte and word order.
- Comprehensive reference of OBIS identifiers with corresponding data types, units, and measurement details.

Practical Usage Examples

- Examples demonstrating register reading and data interpretation using the mbpoll tool.
- Sample REST API commands employing the curl tool to manage Modbus service settings and register configurations.

2. Modbus Register Configuration

The device offers a flexible configuration of register mapping, allowing the selection of data type and scaling for each register. This configuration is performed through the following REST API endpoint:

Endpoint	api/v1/modbus
Method	GET, POST, DELETE
Content Type	application/json

- **GET** – Retrieves the current configuration (returns HTTP 200 with JSON content, or 404 if no configuration is set or it has been deleted).
- **POST** – Saves a new configuration (returns HTTP 200 with the submitted configuration if valid, or 400 if the configuration is invalid).
- **DELETE** – Deletes the current configuration (returns HTTP 204 if the deletion was successful).

2.1. JSON Configuration Format

The configuration is defined as a JSON array of register set objects with the following structure for each element:

Field	Type	Required	Range	Remarks
[]	array	yes	0..	Array containing objects that describe register sets
[].start_address	uint	yes	0..9998	Starting address of the register set (0-based addressing)
[].registers[]	array	yes	0..	Array containing the definitions of registers in the set
[].registers[].id	string	yes	see list of possible identifiers	Register identifier (measurement ID)
[].registers[].type	string	no	double float long int short	Data type for representing the register value (applies only to numeric registers) the default is double.
[].registers[].scaler	int	no	-6..6	Scaling factor applied to the reported value to derive the register value (power of 10; default 0). Applies only to numeric registers.

Example Configuration

```
[
  {
    "start_address": 0,
    "registers": [
      {"id": "1.7.0", "type": "float", "scaler": 3},
      {"id": "21.7.0", "type": "float", "scaler": 3},
      {"id": "41.7.0", "type": "float", "scaler": 3},
      {"id": "61.7.0", "type": "float", "scaler": 3},
      {"id": "31.7.0", "type": "float", "scaler": 0},
      {"id": "51.7.0", "type": "float", "scaler": 0},
      {"id": "71.7.0", "type": "float", "scaler": 0},
      {"id": "32.7.0", "type": "float", "scaler": 0},
      {"id": "52.7.0", "type": "float", "scaler": 0},
      {"id": "72.7.0", "type": "float", "scaler": 0},
      {"id": "report.timestamp", "type": "int", "scaler": 0}
    ]
  },
  {
    "start_address": 1000,
```

```
    "registers": [
      {"id": "1.8.0", "type": "int", "scaler": 3},
      {"id": "1.8.1", "type": "int", "scaler": 3},
      {"id": "1.8.2", "type": "int", "scaler": 3}
    ]
  },
  {
    "start_address": 2000,
    "registers": [
      {"id": "sys.id"},
      {"id": "sys.firmware"},
      {"id": "sys.date_time"}
    ]
  }
]
```

Explanation

The above configuration defines three groups of registers at different base addresses. Registers starting at:

- **0** represent instantaneous measurement values,
- **1000** represent energy counters,
- **2000** provide system information.

2.2. Possible Register Identifiers and Data Types

Identifier (OBIS)	Data Type	Unit	Description
0	number		Zero-filled padding register
1	number		One-filled padding register
1.8.0	number	kWh	Active energy (positive), total consumed
1.8.1	number	kWh	Active energy (positive), consumed in tariff T1
1.8.2	number	kWh	Active energy (positive), consumed in tariff T2
2.8.0	number	kWh	Active energy (negative), total (exported/delivered)
2.8.1	number	kWh	Active energy (negative), exported in tariff T1
2.8.2	number	kWh	Active energy (negative), exported in tariff T2
3.8.0	number	kvarh	Reactive energy (positive), total
3.8.1	number	kvarh	Reactive energy (positive), tariff T1
3.8.2	number	kvarh	Reactive energy (positive), tariff T2
4.8.0	number	kvarh	Reactive energy (negative), total
4.8.1	number	kvarh	Reactive energy (negative), tariff T1
4.8.2	number	kvarh	Reactive energy (negative), tariff T2
5.8.0	number	kvarh	Reactive energy imported (inductive), total
5.8.1	number	kvarh	Reactive energy imported (inductive), tariff T1
5.8.2	number	kvarh	Reactive energy imported (inductive), tariff T2
6.8.0	number	kvarh	Reactive energy imported (capacitive), total
6.8.1	number	kvarh	Reactive energy imported (capacitive), tariff T1
6.8.2	number	kvarh	Reactive energy imported (capacitive), tariff T2
7.8.0	number	kvarh	Reactive energy exported (inductive), total
7.8.1	number	kvarh	Reactive energy exported (inductive), tariff T1
7.8.2	number	kvarh	Reactive energy exported (inductive), tariff T2
8.8.0	number	kvarh	Reactive energy exported (capacitive), total

Identifier (OBIS)	Data Type	Unit	Description
8.8.1	number	kvarh	Reactive energy exported (capacitive), tariff T1
8.8.2	number	kvarh	Reactive energy exported (capacitive), tariff T2
1.6.0	number	kW	Max demand for positive active energy, total
1.6.1	number	kW	Max demand for positive active energy, tariff T1
1.6.2	number	kW	Max demand for positive active energy, tariff T2
2.6.0	number	kW	Max demand for negative active energy, total
2.6.1	number	kW	Max demand for negative active energy, tariff T1
2.6.2	number	kW	Max demand for negative active energy, tariff T2
1.7.0	number	W	Instantaneous positive active power, total
21.7.0	number	W	Instantaneous positive active power, line 1
41.7.0	number	W	Instantaneous positive active power, line 2
61.7.0	number	W	Instantaneous positive active power, line 3
2.7.0	number	W	Instantaneous negative active power, total
22.7.0	number	W	Instantaneous negative active power, line 1
42.7.0	number	W	Instantaneous negative active power, line 2
62.7.0	number	W	Instantaneous negative active power, line 3
power.active	number	W	Signed instantaneous total active power
power.active.l1	number	W	Signed instantaneous active power, line 1
power.active.l2	number	W	Signed instantaneous active power, line 2
power.active.l3	number	W	Signed instantaneous active power, line 3
3.7.0	number	var	Instantaneous positive reactive power, total
23.7.0	number	var	Instantaneous positive reactive power, line 1
43.7.0	number	var	Instantaneous positive reactive power, line 2
63.7.0	number	var	Instantaneous positive reactive power, line 3
4.7.0	number	var	Instantaneous negative reactive power, total
24.7.0	number	var	Instantaneous negative reactive power, line 1
44.7.0	number	var	Instantaneous negative reactive power, line 2
64.7.0	number	var	Instantaneous negative reactive power, line 3
power.reactive	number	var	Signed instantaneous total reactive power
power.reactive.l1	number	var	Signed instantaneous reactive power, line 1
power.reactive.l2	number	var	Signed instantaneous reactive power, line 2
power.reactive.l3	number	var	Signed instantaneous reactive power, line 3
9.7.0	number	VA	Instantaneous apparent power, total
31.7.0	number	A	Current on line 1 (unsigned)
51.7.0	number	A	Current on line 2 (unsigned)
71.7.0	number	A	Current on line 3 (unsigned)
current.l1	number	A	Signed current on line 1
current.l2	number	A	Signed current on line 2
current.l3	number	A	Signed current on line 3
32.7.0	number	V	Voltage on line 1
52.7.0	number	V	Voltage on line 2
72.7.0	number	V	Voltage on line 3

Identifier (OBIS)	Data Type	Unit	Description
13.7.0	number	– (ratio)	Instantaneous power factor
96.14.0	number	–	Current tariff
conv_factor	number	–	User-defined conversion factor
report.date_time	hex		Report generation time in date & time format
report.timestamp	number	s	Report generation timestamp in Unix timestamp
96.1.0	hex		Meter identifier, size 24B
96.1.1	hex		Meter model, size 24B
42.0.0	hex		Meter type, size 24B
sys.id	hex		System ID, size 6B
sys.firmware	hex		System firmware version, size 6B
sys.timestamp	number	s	System time expressed as UNIX timestamp
sys.date_time	hex		System time expressed in date & time format

Note – In the configuration, any register representing a timestamp should always use the int type.

2.3. Numeric Types and their Lengths

Type	Length (bytes)	Length (words)
short	2	1
int	4	2
long	8	4
float	4	2
double	8	4

2.4. Slave Device Address

The configured Modbus slave address is not significant for the device; it will respond to any address (for example, it can be set to 1).

3. Supported Functions

Currently, the only supported function is **Read Input Registers.**

- **Address Range** – 30001 – 39999 (logical addressing) or 0 – 9998 (0-based addressing)
- **Access** – Read-only
- **Modbus Function Code** – 04 – Read Input Registers

3.1. Endianness

- **Modbus TCP uses Big-Endian** for data transmission.
- This means that the most significant byte (MSB) is transmitted first.
- The byte order within a 16-bit register is: **MSB → LSB.**

3.2. Word Order for 32-bit Values

For 32-bit values (e.g. int32 or 32-bit IEEE 754 float), there are two main interpretations of word order:

Big-Endian (Modbus standard, MSW → LSW)

- The first register contains the more significant word (MSW – Most Significant Word).
- The second register contains the less significant word (LSW – Least Significant Word).
- Byte order within each 16-bit register: MSB → LSB.

Example of a 32-bit value (0x12345678)

```
Register 1 (40001) → 0x1234 (MSW)
Register 2 (40002) → 0x5678 (LSW)
```

Similarly, the same word order applies to 64-bit values (e.g. double or long).

3.3. Byte Order within a Word

Big-Endian (MSB → LSB, Modbus standard)

Currently, it is not possible to configure a different endianness.

3.4. Byte Order in a Byte Array

For values represented as a byte array, bytes 0 and 1 are transmitted in the first word (word index 0). In that word, byte 0 is the MSB and byte 1 is the LSB. This pattern continues with bytes 2 and 3 in the next 16-bit word, and so on.

3.5. Date and Time Type

Field	Description	Index in array
year high	Higher 8 bits of the year	0
year low	Lower 8 bits of the year	1
month	Month (starting from 1)	2
day	Day (starting from 1)	3
hour	Hour (0–23)	4
minute	Minute (0–59)	5
second	Second (0–59)	6
is DST	Daylight Saving Time flag (1 if active)	7

Example

```
[2007]:         0x07E9
[2008]:         0x0311
[2009]:         0x1105
[2010]:         0x3A00
```

Interpretation

- Register 2007 contains the year (0x07E9 = 2025).
- Register 2008 contains the month and day: 0x03 = March, 0x11 = 17th day.
- Register 2009 contains the hour and minute: 0x11 = 17 hours, 0x05 = 5 minutes.
- Register 2010 contains the second and DST flag: 0x3A = 58 seconds, 0x00 = DST not active.

4. Examples of Reading with the mbpoll Tool

4.1. Reading Instantaneous Values

```
mbpoll -B -t 3:float -c10 -0 -r0 <whatwattGoIP>
```

- **`-B`** – indicates that 32-bit values (e.g. float) are interpreted in Big-Endian format.
- **`-t`**
  - specifies the register table and data type for interpretation.
    - **`3:float`** – means the Input Registers table (denoted by 3, corresponding to Modbus function 04) and interprets the data as 32-bit floating-point (IEEE 754).
    - Each float value occupies **two consecutive 16-bit registers**.
- **`-c10`**
  - the program will read 10 registers (16-bit each).
    - Since each 32-bit float value spans 2 registers:
      - 10 registers = 5 float values.
      - The program will return 5 floating-point numbers.
- **`-0`** – forces 0-based addressing. Enabling -0 means the specified register address is interpreted as zero-based (0 = first register) instead of 1-based.
- **`-r0`** – sets the starting address to 0 (which here means the first Input Register, since is used).

Note – If the device used 1-based addressing, register 0 in the mbpoll command would correspond to 40001 in Modbus notation.

Result and Data Interpretation

```
[0]:    24          total instantaneous active power: 24 W
[2]:    24          instantaneous active power on phase 1: 24 W
[4]:    0           instantaneous active power on phase 2: 0 W
[6]:    0           instantaneous active power on phase 3: 0 W
[8]:    0.29        current on phase 1: 0.29 A
[10]:   0           current on phase 2: 0 A
[12]:   0           current on phase 3: 0 A
[14]:   233.96      voltage on phase 1: 233.96 V
[16]:   32.3        voltage on phase 2: 32.3 V
[18]:   32.49       voltage on phase 3: 32.49 V
```

4.2. Reading Report Generation Time

```
mbpoll -B -t 3:int -c1 -0 -r40 <whatwattGoIP>
```

- **`-r40`** – starts reading from address 40 (zero-based).
- **`-c1`** – reads 1 register (in this context, one 32-bit value of type int).
  (The other options `-B, -t 3:int`, and `-0` are as described above.)

Result and Data Interpretation

```
[40]:   1742228790      UNIX timestamp: Mon Mar 17 2025 18:26:30 GMT+0000
```

Note – When integrating with a Modbus TCP master, all values can be read in one request. Reading all relevant registers in a single poll ensures that measurement values and the timestamp are synchronized.

4.3. Reading Energy Counters (Range 2)

```
mbpoll -B -t 3:int -c3 -0 -r1000 <whatwattGoIP>
```

- **`-r1000`** – starts reading from address 1000 (zero-based).
- **`-c3`** – reads 3 registers (16-bit each) as 32-bit integers (`-t 3:int`).

Result and Data Interpretation

```
[1000]:         94255   total consumed energy
[1002]:         61344   energy consumed in tariff T1
[1004]:         32911   energy consumed in tariff T2
```

4.4. Reading System Information (Range 3)

```
mbpoll -B -m tcp -t 3:hex -c10 -0 -r2000 <whatwattGoIP>
```

- **`-r2000`** – starts reading from address 2000 (zero-based)
- **`-c10`** – reads 10 registers (16-bit each).
- **`-t 3:hex`** – interprets the values as raw hex. (The `-m tcp` option specifies TCP mode for mbpoll.)

Result and Data Interpretation

```
[2000]:         0xA842          System ID (first two bytes)
[2001]:         0xE39F          System ID (next two bytes)
[2002]:         0x8124          System ID (last two bytes)
[2003]:         0x0102          Firmware version - MSB (major): 1, LSB (minor): 2
[2004]:         0x1200          Firmware version - MSB (release): 18, LSB: not used
[2005]:         0x0000          Firmware version: not used
[2006]:         0x07E9          System time: year
[2007]:         0x0311          System time: month, day
[2008]:         0x1122          System time: hour, minute
[2009]:         0x1400          System time: second, DST flag
```

5. Example of the curl command to control the Modbus service

`<whatwattGoIP>` is the IP address (in local network) of the whatwatt Go device, for example (replace <whatwattGoIP> with) 192.168.1.100

The below command enables the Modbus service; to disable it, replace true with false

```
curl -i -X PUT -d '{"services":{"modbus":{"enable":true}}}' http://<whatwattGoIP>/
api/v1/settings
```

It is also possible to change the default port from 502 to another one, as shown below

```
curl -i -X PUT -d '{"services":{"modbus":{"enable":true,"port":5020}}}' http://
<whatwattGoIP>/api/v1/settings
```

An example of using the curl command to set the configuration. First, the JSON register definition should be saved in the modbus.json file

```
curl -i -d @modbus.json http://<whatwattGoIP>/api/v1/modbus
```

The above command writes the register definition. After sending, the returned JSON is supplemented with offset addresses and sizes, as shown below


```
[
  {
    "start_address": 0,
    "end_address": 22,
    "size": 22,
    "registers": [
      {"id":"1.7.0","type":"float","scaler":3,"address":0,"size":2},
      {"id":"21.7.0","type":"float","scaler":3,"address":2,"size":2},
      {"id":"41.7.0","type":"float","scaler":3,"address":4,"size":2},
      {"id":"61.7.0","type":"float","scaler":3,"address":6,"size":2},
      {"id":"31.7.0","type":"float","scaler":0,"address":8,"size":2},
      {"id":"51.7.0","type":"float","scaler":0,"address":10,"size":2},
      {"id":"71.7.0","type":"float","scaler":0,"address":12,"size":2},
      {"id":"32.7.0","type":"float","scaler":0,"address":14,"size":2},
      {"id":"52.7.0","type":"float","scaler":0,"address":16,"size":2},
      {"id":"72.7.0","type":"float","scaler":0,"address":18,"size":2},
      {"id":"report.timestamp","type":"int","scaler":0,"address":20,"size":2}
    ]
  },
  {
    "start_address": 1000,
    "end_address": 1006,
    "size": 6,
    "registers": [
      {"id":"1.8.0","type":"int","scaler":3,"address":1000,"size":2},
      {"id":"1.8.1","type":"int","scaler":3,"address":1002,"size":2},
      {"id":"1.8.2","type":"int","scaler":3,"address":1004,"size":2}
    ]
  },
  {
    "start_address": 2000,
    "end_address": 2010,
    "size": 10,
    "registers": [
      {"id":"sys.id","address":2000,"size":3},
      {"id":"sys.firmware","address":2003,"size":3},
      {"id":"sys.date_time","type":"double","scaler":0,"address":2006,"size":4}
    ]
  }
]
```

Example of reading the current register configuration using curl with formatting as above

```
curl -s http://<whatwattGoIP>/api/v1/modbus | jq '.[].registers |= map(@json)' |
sed -e 's/"{/{/g' -e 's/}"/}/g' -e ’s/\\"/"/g'
```

Example of restoring the configuration to default using curl:

```
curl -i -X DELETE http://<whatwattGoIP>/api/v1/modbus
```

A 204 code should be returned.

6. Default configuration

The default Modbus configuration (register layout and data types) is similar to that of the Siemens PAC2200 power meter. Registers marked as padding always return a value of 0; they are included solely to maintain continuity within the register address space, which is useful when reading multiple registers at once. All registers are read-only and can only be accessed using the Read Input Registers function; writing is not supported.

6.1. Default configuration table

Offset	Registers Count	Name	Format	Unit	ID
1	2	Voltage L1	float	V	32.7.0
3	2	Voltage L2	float	V	52.7.0
5	2	Voltage L3	float	V	72.7.0
7	2	padding (Voltage L1-L2)	float	V	0
9	2	padding (Voltage L2-L3)	float	V	0
11	2	padding (Voltage L3-L1)	float	V	0
13	2	Current L1	float	A	current.l1
15	2	Current L2	float	A	current.l2
17	2	Current L3	float	A	current.l3
19	2	padding (Apparent Power L1)	float	VA	0
21	2	padding (Apparent Power L2)	float	VA	0
23	2	padding (Apparent Power L3)	float	VA	0
25	2	Active Power L1	float	W	power.active.l1
27	2	Active Power L2	float	W	power.active.l2
29	2	Active Power L3	float	W	power.active.l3
31	2	Reactive Power L1	float	var	power.reactive.l1
33	2	Reactive Power L2	float	var	power.reactive.l2
35	2	Reactive Power L3	float	var	power.reactive.l3
37	2	padding (Power Factor L1)	float		0
39	2	padding (Power Factor L2)	float		0
41	2	padding (Power Factor L3)	float		0
55	2	padding (Frequency)	float	Hz	0
57	2	padding (Voltage Avg L-N)	float	V	0
59	2	padding (Voltage Avg L-L)	float	V	0
61	2	padding (Average Current)	float	A	0
63	2	Apparent Power Total	float	VA	9.7.0
65	2	Active Power	float	W	power.active
67	2	Reactive Power	float	var	power.reactive
69	2	Power Factor Total	float		13.7.0
205	2	padding (diagnostic, status)	int		0
207	2	padding (digital outputs status)	int		0
209	2	padding (digital inputs status)	int		0
211	2	Tariff	int		96.14.0
213	2	padding (working time counter)	int	s	0

Offset	Registers Count	Name	Format	Unit	ID
215	2	padding (universal counter)	int		0
217	2	padding (parameters change counter)	int		0
219	2	padding (total parameter changes counter)	int		0
501	2	Active Power Imported	float	W	1.7.0
503	2	Reactive Power Imported	float	var	3.7.0
505	2	Active Power Exported	float	W	2.7.0
507	2	Reactive Power Exported	float	var	4.7.0
509	2	padding (max active power)	float	W	0
511	2	padding (min active power)	float	W	0
513	2	padding (max reactive power)	float	var	0
515	2	padding (min reactive power)	float	var	0
517	2	padding (current measurement period)	int	s	0
519	2	padding (time since demand period)	int	s	0
549	2	Active Energy Imported	float	Wh	1.8.0
551	2	Reactive Energy Imported	float	varh	3.8.0
553	2	Active Energy Exported	float	Wh	2.8.0
555	2	Reactive Energy Exported	float	varh	4.8.0
801	4	Active Energy Imported T1	double	Wh	1.8.1
805	4	Active Energy Imported T2	double	Wh	1.8.2
809	4	Active Energy Exported T1	double	Wh	2.8.1
813	4	Active Energy Exported T2	double	Wh	2.8.2
817	4	Reactive Energy Imported T1	double	varh	3.8.1
821	4	Reactive Energy Imported T2	double	varh	3.8.2
825	4	Reactive Energy Exported T1	double	varh	4.8.1
829	4	Reactive Energy Exported T2	double	varh	4.8.2
833	4	padding (Apparent Energy T1)	double	VAh	0
837	4	padding (Apparent Energy T2)	double	VAh	0
841	4	padding (L1 Active Energy Imported T1)	double	Wh	0
845	4	padding (L1 Active Energy Imported T2)	double	Wh	0
849	4	padding (L1 Active Energy Exported T1)	double	Wh	0
853	4	padding (L1 Active Energy Exported T2)	double	Wh	0
857	4	padding (L1 Reactive Energy Imported T1)	double	varh	0
861	4	padding (L1 Reactive Energy Imported T2)	double	varh	0
865	4	padding (L1 Reactive Energy Exported T1)	double	varh	0
869	4	padding (L1 Reactive Energy Exported T2)	double	varh	0
873	4	padding (L1 Apparent Energy T1)	double	VAh	0
877	4	padding (L1 Apparent Energy T2)	double	VAh	0
881	4	padding (L2 Active Energy Imported T1)	double	Wh	0
885	4	padding (L2 Active Energy Imported T2)	double	Wh	0
889	4	padding (L2 Active Energy Exported T1)	double	Wh	0
893	4	padding (L2 Active Energy Exported T2)	double	Wh	0
897	4	padding (L2 Reactive Energy Imported T1)	double	varh	0

Offset	Registers Count	Name	Format	Unit	ID
901	4	padding (L2 Reactive Energy Imported T2)	double	varh	0
905	4	padding (L2 Reactive Energy Exported T1)	double	varh	0
909	4	padding (L2 Reactive Energy Exported T2)	double	varh	0
913	4	padding (L2 Apparent Energy T1)	double	VAh	0
917	4	padding (L2 Apparent Energy T2)	double	VAh	0
921	4	padding (L3 Active Energy Imported T1)	double	Wh	0
925	4	padding (L3 Active Energy Imported T2)	double	Wh	0
929	4	padding (L3 Active Energy Exported T1)	double	Wh	0
933	4	padding (L3 Active Energy Exported T2)	double	Wh	0
937	4	padding (L3 Reactive Energy Imported T1)	double	varh	0
941	4	padding (L3 Reactive Energy Imported T2)	double	varh	0
945	4	padding (L3 Reactive Energy Exported T1)	double	varh	0
949	4	padding (L3 Reactive Energy Exported T2)	double	varh	0
953	4	padding (L3 Apparent Energy T1)	double	VAh	0
957	4	padding (L3 Apparent Energy T2)	double	VAh	0

6.2. Default configuration in JSON

```
[
  {
    "start_address": 1,
    "registers": [
      {"type": "float", "scaler": 0, "id": "32.7.0"},
      {"type": "float", "scaler": 0, "id": "52.7.0"},
      {"type": "float", "scaler": 0, "id": "72.7.0"},

      {"type": "float", "scaler": 0, "id": "0"},
      {"type": "float", "scaler": 0, "id": "0"},
      {"type": "float", "scaler": 0, "id": "0"},

      {"type": "float", "scaler": 0, "id": "current.l1"},
      {"type": "float", "scaler": 0, "id": "current.l2"},
      {"type": "float", "scaler": 0, "id": "current.l3"},

      {"type": "float", "scaler": 0, "id": "0"},
      {"type": "float", "scaler": 0, "id": "0"},
      {"type": "float", "scaler": 0, "id": "0"},

      {"type": "float", "scaler": 3, "id": "power.active.l1"},
      {"type": "float", "scaler": 3, "id": "power.active.l2"},
      {"type": "float", "scaler": 3, "id": "power.active.l3"},

      {"type": "float", "scaler": 3, "id": "power.reactive.l1"},
      {"type": "float", "scaler": 3, "id": "power.reactive.l2"},
      {"type": "float", "scaler": 3, "id": "power.reactive.l3"},

      {"type": "float", "scaler": 0, "id": "0"},
      {"type": "float", "scaler": 0, "id": "0"},
      {"type": "float", "scaler": 0, "id": "0"}
    ]
  },
  {
    "start_address": 55,
    "registers": [
```

```
      {"type": "float", "scaler": 0, "id": "0"},
      {"type": "float", "scaler": 0, "id": "0"},
      {"type": "float", "scaler": 0, "id": "0"},
      {"type": "float", "scaler": 0, "id": "0"},

      {"type": "float", "scaler": 3, "id": "9.7.0"},
      {"type": "float", "scaler": 3, "id": "power.active"},
      {"type": "float", "scaler": 3, "id": "power.reactive"},
      {"type": "float", "scaler": 3, "id": "13.7.0"}
    ]
  },
  {
    "start_address": 205,
    "registers": [
      {"type": "int", "scaler": 0, "id": "0"},
      {"type": "int", "scaler": 0, "id": "0"},
      {"type": "int", "scaler": 0, "id": "0"},
      {"type": "int", "scaler": 0, "id": "96.14.0"},
      {"type": "int", "scaler": 0, "id": "0"},
      {"type": "int", "scaler": 0, "id": "0"},
      {"type": "int", "scaler": 0, "id": "0"},
      {"type": "int", "scaler": 0, "id": "0"}
    ]
  },
  {
    "start_address": 501,
    "registers": [
      {"type": "float", "scaler": 3, "id": "1.7.0"},
      {"type": "float", "scaler": 3, "id": "3.7.0"},
      {"type": "float", "scaler": 3, "id": "2.7.0"},
      {"type": "float", "scaler": 3, "id": "4.7.0"},
      {"type": "float", "scaler": 0, "id": "0"},
      {"type": "float", "scaler": 0, "id": "0"},
      {"type": "float", "scaler": 0, "id": "0"},
      {"type": "float", "scaler": 0, "id": "0"},
      {"type": "int",   "scaler": 0, "id": "0"},
      {"type": "int",   "scaler": 0, "id": "0"}
    ]
  },
  {
    "start_address": 549,
    "registers": [
      {"type": "float", "scaler": 3, "id": "1.8.0"},
      {"type": "float", "scaler": 3, "id": "3.8.0"},
      {"type": "float", "scaler": 3, "id": "2.8.0"},
      {"type": "float", "scaler": 3, "id": "4.8.0"}
    ]
  },
  {
    "start_address": 801,
    "registers": [
      {"type": "double", "scaler": 3, "id": "1.8.1"},
      {"type": "double", "scaler": 3, "id": "1.8.2"},
      {"type": "double", "scaler": 3, "id": "2.8.1"},
      {"type": "double", "scaler": 3, "id": "2.8.2"},
      {"type": "double", "scaler": 3, "id": "3.8.1"},
      {"type": "double", "scaler": 3, "id": "3.8.2"},
      {"type": "double", "scaler": 3, "id": "4.8.1"},
      {"type": "double", "scaler": 3, "id": "4.8.2"},
      {"type": "double", "scaler": 0, "id": "0"},
      {"type": "double", "scaler": 0, "id": "0"},
      {"type": "double", "scaler": 0, "id": "0"},
      {"type": "double", "scaler": 0, "id": "0"},
      {"type": "double", "scaler": 0, "id": "0"},
      {"type": "double", "scaler": 0, "id": "0"},
      {"type": "double", "scaler": 0, "id": "0"},
      {"type": "double", "scaler": 0, "id": "0"},
      {"type": "double", "scaler": 0, "id": "0"},
      {"type": "double", "scaler": 0, "id": "0"},
      {"type": "double", "scaler": 0, "id": "0"},
```

```
      {"type": "double", "scaler": 0, "id": "0"},
      {"type": "double", "scaler": 0, "id": "0"},
      {"type": "double", "scaler": 0, "id": "0"},
      {"type": "double", "scaler": 0, "id": "0"},
      {"type": "double", "scaler": 0, "id": "0"},
      {"type": "double", "scaler": 0, "id": "0"},
      {"type": "double", "scaler": 0, "id": "0"},
      {"type": "double", "scaler": 0, "id": "0"},
      {"type": "double", "scaler": 0, "id": "0"},
      {"type": "double", "scaler": 0, "id": "0"},
      {"type": "double", "scaler": 0, "id": "0"},
      {"type": "double", "scaler": 0, "id": "0"},
      {"type": "double", "scaler": 0, "id": "0"},
      {"type": "double", "scaler": 0, "id": "0"},
      {"type": "double", "scaler": 0, "id": "0"},
      {"type": "double", "scaler": 0, "id": "0"},
      {"type": "double", "scaler": 0, "id": "0"},
      {"type": "double", "scaler": 0, "id": "0"},
      {"type": "double", "scaler": 0, "id": "0"},
      {"type": "double", "scaler": 0, "id": "0"}
    ]
  }
]
```


Reading returns


```
[
  {
    "start_address": 1,
    "end_address": 43,
    "size": 42,
    "registers": [
      {"id":"32.7.0","type":"float","scaler":0,"address":1,"size":2},
      {"id":"52.7.0","type":"float","scaler":0,"address":3,"size":2},
      {"id":"72.7.0","type":"float","scaler":0,"address":5,"size":2},
      {"id":"0","type":"float","scaler":0,"address":7,"size":2},
      {"id":"0","type":"float","scaler":0,"address":9,"size":2},
      {"id":"0","type":"float","scaler":0,"address":11,"size":2},
      {"id":"current.l1","type":"float","scaler":0,"address":13,"size":2},
      {"id":"current.l2","type":"float","scaler":0,"address":15,"size":2},
      {"id":"current.l3","type":"float","scaler":0,"address":17,"size":2},
      {"id":"0","type":"float","scaler":0,"address":19,"size":2},
      {"id":"0","type":"float","scaler":0,"address":21,"size":2},
      {"id":"0","type":"float","scaler":0,"address":23,"size":2},
      {"id":"power.active.l1","type":"float","scaler":3,"address":25,"size":2},
      {"id":"power.active.l2","type":"float","scaler":3,"address":27,"size":2},
      {"id":"power.active.l3","type":"float","scaler":3,"address":29,"size":2},
      {"id":"power.reactive.l1","type":"float","scaler":3,"address":31,"size":2},
      {"id":"power.reactive.l2","type":"float","scaler":3,"address":33,"size":2},
      {"id":"power.reactive.l3","type":"float","scaler":3,"address":35,"size":2},
      {"id":"0","type":"float","scaler":0,"address":37,"size":2},
      {"id":"0","type":"float","scaler":0,"address":39,"size":2},
      {"id":"0","type":"float","scaler":0,"address":41,"size":2}
    ]
  },
  {
    "start_address": 55,
    "end_address": 71,
    "size": 16,
    "registers": [
      {"id":"0","type":"float","scaler":0,"address":55,"size":2},
      {"id":"0","type":"float","scaler":0,"address":57,"size":2},
      {"id":"0","type":"float","scaler":0,"address":59,"size":2},
      {"id":"0","type":"float","scaler":0,"address":61,"size":2},
      {"id":"9.7.0","type":"float","scaler":3,"address":63,"size":2},
      {"id":"power.active","type":"float","scaler":3,"address":65,"size":2},
      {"id":"power.reactive","type":"float","scaler":3,"address":67,"size":2},
      {"id":"13.7.0","type":"float","scaler":3,"address":69,"size":2}
```

```
    ]
  },
  {
    "start_address": 205,
    "end_address": 221,
    "size": 16,
    "registers": [
      {"id":"0","type":"int","scaler":0,"address":205,"size":2},
      {"id":"0","type":"int","scaler":0,"address":207,"size":2},
      {"id":"0","type":"int","scaler":0,"address":209,"size":2},
      {"id":"96.14.0","type":"int","scaler":0,"address":211,"size":2},
      {"id":"0","type":"int","scaler":0,"address":213,"size":2},
      {"id":"0","type":"int","scaler":0,"address":215,"size":2},
      {"id":"0","type":"int","scaler":0,"address":217,"size":2},
      {"id":"0","type":"int","scaler":0,"address":219,"size":2}
    ]
  },
  {
    "start_address": 501,
    "end_address": 521,
    "size": 20,
    "registers": [
      {"id":"1.7.0","type":"float","scaler":3,"address":501,"size":2},
      {"id":"3.7.0","type":"float","scaler":3,"address":503,"size":2},
      {"id":"2.7.0","type":"float","scaler":3,"address":505,"size":2},
      {"id":"4.7.0","type":"float","scaler":3,"address":507,"size":2},
      {"id":"0","type":"float","scaler":0,"address":509,"size":2},
      {"id":"0","type":"float","scaler":0,"address":511,"size":2},
      {"id":"0","type":"float","scaler":0,"address":513,"size":2},
      {"id":"0","type":"float","scaler":0,"address":515,"size":2},
      {"id":"0","type":"int","scaler":0,"address":517,"size":2},
      {"id":"0","type":"int","scaler":0,"address":519,"size":2}
    ]
  },
  {
    "start_address": 549,
    "end_address": 557,
    "size": 8,
    "registers": [
      {"id":"1.8.0","type":"float","scaler":3,"address":549,"size":2},
      {"id":"3.8.0","type":"float","scaler":3,"address":551,"size":2},
      {"id":"2.8.0","type":"float","scaler":3,"address":553,"size":2},
      {"id":"4.8.0","type":"float","scaler":3,"address":555,"size":2}
    ]
  },
  {
    "start_address": 801,
    "end_address": 961,
    "size": 160,
    "registers": [
      {"id":"1.8.1","type":"double","scaler":3,"address":801,"size":4},
      {"id":"1.8.2","type":"double","scaler":3,"address":805,"size":4},
      {"id":"2.8.1","type":"double","scaler":3,"address":809,"size":4},
      {"id":"2.8.2","type":"double","scaler":3,"address":813,"size":4},
      {"id":"3.8.1","type":"double","scaler":3,"address":817,"size":4},
      {"id":"3.8.2","type":"double","scaler":3,"address":821,"size":4},
      {"id":"4.8.1","type":"double","scaler":3,"address":825,"size":4},
      {"id":"4.8.2","type":"double","scaler":3,"address":829,"size":4},
      {"id":"0","type":"double","scaler":0,"address":833,"size":4},
      {"id":"0","type":"double","scaler":0,"address":837,"size":4},
      {"id":"0","type":"double","scaler":0,"address":841,"size":4},
      {"id":"0","type":"double","scaler":0,"address":845,"size":4},
      {"id":"0","type":"double","scaler":0,"address":849,"size":4},
      {"id":"0","type":"double","scaler":0,"address":853,"size":4},
      {"id":"0","type":"double","scaler":0,"address":857,"size":4},
      {"id":"0","type":"double","scaler":0,"address":861,"size":4},
      {"id":"0","type":"double","scaler":0,"address":865,"size":4},
      {"id":"0","type":"double","scaler":0,"address":869,"size":4},
      {"id":"0","type":"double","scaler":0,"address":873,"size":4},
      {"id":"0","type":"double","scaler":0,"address":877,"size":4},
```

```
      {"id":"0","type":"double","scaler":0,"address":881,"size":4},
      {"id":"0","type":"double","scaler":0,"address":885,"size":4},
      {"id":"0","type":"double","scaler":0,"address":889,"size":4},
      {"id":"0","type":"double","scaler":0,"address":893,"size":4},
      {"id":"0","type":"double","scaler":0,"address":897,"size":4},
      {"id":"0","type":"double","scaler":0,"address":901,"size":4},
      {"id":"0","type":"double","scaler":0,"address":905,"size":4},
      {"id":"0","type":"double","scaler":0,"address":909,"size":4},
      {"id":"0","type":"double","scaler":0,"address":913,"size":4},
      {"id":"0","type":"double","scaler":0,"address":917,"size":4},
      {"id":"0","type":"double","scaler":0,"address":921,"size":4},
      {"id":"0","type":"double","scaler":0,"address":925,"size":4},
      {"id":"0","type":"double","scaler":0,"address":929,"size":4},
      {"id":"0","type":"double","scaler":0,"address":933,"size":4},
      {"id":"0","type":"double","scaler":0,"address":937,"size":4},
      {"id":"0","type":"double","scaler":0,"address":941,"size":4},
      {"id":"0","type":"double","scaler":0,"address":945,"size":4},
      {"id":"0","type":"double","scaler":0,"address":949,"size":4},
      {"id":"0","type":"double","scaler":0,"address":953,"size":4},
      {"id":"0","type":"double","scaler":0,"address":957,"size":4}
    ]
  }
]
```


6.3. Example of reading the first 21 registers using the mbpoll command

```
mbpoll -B -t 3:float -c21 -0 -r1 <whatwattGoIP>
```