

# 13.2 Modbus Master

Modbus communication contains a single Master and may include more than 1, but not more than 247 devices. To gather data from peripheral devices, master device request a cluster of slave devices for data. If any device understand that this message is addressed for it – it replies with data. As no timestamp is sent along with data, having recent data requires frequent polling. WCC Lite can be configured to acquire data periodically in custom-defined intervals.

## Configuring datapoints

To use Modbus Master in WCC Lite, it has to be configured via an Excel configuration. This configuration contains two Excel sheets where parameters have to be filled in - Devices and Signals

### Modbus Master parameters for Devices tab

| Parameter    | Type    | Description   | Required |               | Default Value<br>(when not specified) | Range   |       |
|--------------|---------|---|----------|---------------|---------------------------------------|---|-------|
|              |         |   | TCP      | RTU/A<br>SCII |                                       | Min   | Max   |
| name         | string  | User-friendly name for a device   | Yes      | Yes           |                                       |   |       |
| description  | string  | Description of a device   | No       | No            |                                       |   |       |
| device_alias | string  | Alphanumeric string to identify a device  | Yes      | Yes           |                                       |   |       |
| enable       | boolean | Enabling/disabling of a device  | No       | No            | 1                                     | 0   | 1     |
| protocol     | string  | Protocol to be used   | Yes      | Yes           |                                       | Modbus RTU, Modbus TCP  |       |
| ip           | string  | IP address of TCP slave device  | Yes      | -             |                                       |   |       |
| port         | integer | TCP communication port  | Yes      | -             | 502                                   |   |       |
| bind_address | string  | IP address of network adapter used to connect to slave device (Default: "0.0.0.0")                                    | No       | No            | 0.0.0.0                               |   |       |
| id           | integer | Modbus Slave ID   | Yes      | Yes           |                                       |   |       |
| mode         | string  | Choosing between RTU ("rtu"), ASCII ("ascii") and TCP("tcp") modes. ASCII is the same as RTU, but with ASCII symbols. | No       | No            | TCP (for TCP)<br>RTU (for Serial)     | rtu, ascii, tcp   |       |
| timeout_ms   | integer | Response timeout in milliseconds  | Yes      | Yes           | 10000                                 |   |       |
| device       | string  | Communication port ("PORT1"/"PORT2")  | -        | Yes           |                                       | PORT1   | PORT2 |
| baudrate     | integer | Communication speed, baud/s   | -        | Yes           | 9600                                  | 300, 600, 1200, 2400, 4800, 9600, 19200, 38400, 57600, 115200 |       |

|                    |         |   |     |     |      |                 |   |
|--------------------|---------|---|-----|-----|------|-----------------|---|
| databits           | integer | Data bit count for communication  | -   | Yes | 8    | 6               | 9 |
| stopbits           | integer | Stop bit count for communication  | -   | Yes | 1    | 1               | 2 |
| parity             | string  | Communication parity option   | -   | Yes | none | none, even, odd |   |
| flowcontrol        | string  | Number of requests, before link is considered lost (device status signals are changed) and reconnect attempt will be issued   | -   | Yes | none | none            |   |
| scan_rate_ms       | integer | If provided and positive - all jobs will have similar scan rate - all reads and writes will be executed within this timeframe (parameter scan_rate_ms in Signals tab will be ignored) | Yes | Yes | 300  |                 |   |
| retry_count        | integer | Number of requests, before link is considered lost (device status signals are changed) and reconnect attempt will be issued   | No  | No  | 3    |                 |   |
| serial_delay       | integer | RS485 delay between read and write operations in milliseconds   | -   | Yes | 50   |                 |   |
| keep_alive_timeout | integer | Time interval for sending a keep alive packet (in milliseconds)   | No  | No  | 60   |                 |   |
| modbus_multi_write | boolean | Use 15/16 functions to write 1 register/coil (Default: 0)   | No  | No  | 0    | 0               | 1 |
| comm_restart_delay | integer | Time delay between disconnecting from slave device and restarting connection (in milliseconds) (Default: 500)   | No  | -   | 500  |                 |   |
| update             | boolean | Enable to keep updating the tags even if they have the same value.  | No  | No  | 0    | 0               | 1 |

### Modbus Master parameters for Signals tab

| Parameter    | Type   | Description   | Required |               | Default Value<br>(when not specified) | Range |     |
|--------------|--------|---|----------|---------------|---------------------------------------|-------|-----|
|              |        |   | TCP      | RTU/A<br>SCII |                                       | Min   | Max |
| signal_name  | string | User-friendly signal name                             | Yes      | Yes           |                                       |       |     |
| device_alias | string | Alphanumeric string to identify a device              | Yes      | Yes           |                                       |       |     |
| signal_alias | string | Unique alphanumeric name of the signal to be Yes used | Yes      | Yes           |                                       |       |     |

|                     |         |   |     |     |   |   |   |
|---------------------|---------|---|-----|-----|---|---|---|
| enable              | boolean | Enabling/disabling of an individual signal  | No  | No  | 1 | 0 | 1 |
| job_todo            | string  | Request to send according to Modbus specification without device address and checksum. This field can be identical on several tags to fetch them in single request              | Yes | Yes |   |   |   |
| tag_job_todo        | string  | Similar format to job_todo field. Address and length must be a subset of job field. Defines the individual tag's register(s) or coil(s). Can be described in HEX or DEC formats | Yes | Yes |   |   |   |
| number_type         | string  | Type of a number (FLOAT, DOUBLE, DIGITAL, etc.)   | Yes | Yes |   |   |   |
| log                 | integer | 1 enables logging to Events log, 0 disables   | No  | No  | 0 |   |   |
| pulse_short_time_ms | integer | Time interval for short output pulse to stay active   | No  | No  | 0 |   |   |
| pulse_long_time_ms  | integer | Time interval for long output pulse to stay active  | No  | No  | 0 |   |   |

## Device status signals

Modbus Master has an additional signal which can be configured to show communication status. It is used to indicate if the slave device has disconnected from master (WCC Lite). To configure such signal for Modbus protocol, `job_todo` and `tag_job_todo` fields with string values are required. For Modbus master required parameters for status signal will be: **signal\_name** **device\_alias**, **signal\_alias**, **number\_type**, **job\_todo** and **tag\_job\_todo**. `job_todo` value must be `device_status` and for `tag_job_todo` there are 4 variations: `communication_status`, `device_running`, `device_error`, `unknown_error`. Each signal has 4 possible values and are based on the same logic. If signal returns value of 0, it means unknown error has appeared, 1 - device or protocol connection is on and working properly, 2 - device is off or protocol is disconnected, 3 - error or service is down.

Different device vendors can have different implementations of a Modbus protocol stack. A register table can be one of the primary differences. WCC Lite Modbus Master transmits the most significant word (byte) first, however, devices from some vendors might require transmitting the least significant word (byte) first. If that is the case, make sure to switch bytes as needed. To find out more about setting a correct number format, one should consult a section `number_type`.

Modbus job or tag (as a task to be completed) can be built in two different formats - user can select a more convenient way for him:

- hexadecimal format with every single byte separated by | symbol. Device address, bytes containing output information and CRC (LRC) bytes should be excluded from the message;
- decimal format containing function number, first address and address count, separated by ; symbol. All other information should be excluded from the message;

`job_todo` can group several `tag_job_todo`'s. That way one Modbus message can be used to extract several tags. Grouping is accomplished dynamically meaning that if several identical jobs are found, their tags are grouped automatically.

## Debugging a Modbus Master application

If configuration for Modbus Master is set up, handler for protocol will start automatically. If configuration is missing or contains errors, protocol will not start. It is done intentionally to decrease unnecessary memory usage.

Modbus Master command line debugging options

`modbus-master`

```
-h [ -help ] Display help information
-V [ -version ] Show version
-d<debug level> Set debugging level
-c [ -config ] Config path
-r [ -raw ] Show raw telegram data
-f [ -frame ] Show frame data
-s [ -serial ] Show serial port data
-tcp Show tcp packets
-ascii Show ASCII messages
```

```
-rtu Show RTU messages  
-e [ -redis ] Show redis debug information  
-R [ -readyfile ] Ready notification file
```

If Modbus Master does not work properly (e.g. no communication between devices, data is corrupted, etc.), a user can launch a debug session from command line interface and find out why link is not functioning properly. To launch a debugging session, a user should stop modbus-master process and run modbus-master command with respective flags as shown above.

---

🔄Revision #2

★Created 11 January 2024 09:21:50 by Gabriele

✎Updated 13 February 2024 07:30:19 by Lukas Taroza