# 13.3 Modbus Slave

WCC Lite can act as one (or several) of slave devices in a communication line. This can be used to transmit data to SCADA systems or other RTU devices. It can reply to a messages from Modbus Master with matching device and register addresses.

## Configuring datapoints

To use Modbus Slave 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

> ⚠ If TCP/IP is used as a trasmission medium, only devices with IPs predefined in host column are allowed to connect. All other connections are rejected

### Modbus Slave parameters for Devices tab

| Parameter | Type | Description | Required | | Default Value (when not specified) | Range | |
|---|---|---|---|---|---|---|---|
| | | | TCP | RTU/ASCII | | 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 | unknown | | |
| enable | boolean | Enabling/disabling of a device | No | No | 1 | 0 | 1 |
| protocol | string | Protocol to be used | Yes | Yes | | Modbus serial Slave, Modbus TCP Slave | |
| host | string | Space separated host IP addresses of master device | Yes | - | | | |
| port | integer | TCP port to listen for incoming connections | Yes | - | | | |
| 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 | | |
| keep_alive_timeout | integer | Minimum time a connection can be idle without being closed in milliseconds | No | No | 60 | | |
| 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 | |
| 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 | Communication device’s flow control option. | - | No | none | none |

## Modbus Slave parameters for Signals tab

| Parameter | Type | Description | Required | | Default Value (when not specified) | Range | |
|---|---|---|---|---|---|---|---|
| | | | TCP | RTU/ASCII | | 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 an individual signal | No | No | 1 | 0 | 1 |
| number_type | string | Type of a number (FLOAT, DOUBLE, DIGITAL, etc.). This defines the size that will be read. | Yes | Yes | | | |
| log | integer | Size of this signal’s log in the Event log. | No | No | 0 | | |
| slave_id | integer | Address of a slave device | Yes | Yes | | | |
| function | integer | Function number | Yes | Yes | | | |
| register_address | integer | Register address | Yes | Yes | | | |

## Device status signals

Modbus slave has an additional signal which can be configured to show communication status. It is used to indicate if the master device has disconnected from slave (WCC Lite). To configure such signal for Modbus protocol, job_todo and tag_job_todo fields with string values are required. For Modbus slave required parameters for status signal will be: **signal_name device_alias, signal_alias, number_type, slave_id, function**, **register_address, 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, uknown_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.

## Mapping values to registers

Internally stored values aren’t organised in a register-like order, therefore mapping should be done by the user. This mapping includes setting an address of the device WCC Lite is simulating as well as function number, register number and how much 16-bit registers are used to store a value. These values should be set in `common_address`, `function`, `info_address` and `size` columns respectively in the Excel configuration.

To find out how many register should be used for storing a values, how values can have their values swapped, a user should consult a section `number_type`.

⚠ If a Modbus master device requests a data from a register that is mapped but doesn’t yet have initial value, ILLEGAL DATA ADDRESS error code will be returned. The same error code is returned if a requested size of value is bigger that defined or if register is not configured at all.

## Debugging a Modbus Slave application

If configuration for Modbus Slave 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 Slave command line debugging options

```
modbus-slave
```

```
-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 Slave 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-slave` process and run `modbus-slave` command with respective flags as shown above.

Revision #1
Created 11 January 2024 09:21:50 by Gabriele
Updated 11 January 2024 09:38:19 by Gabriele