# Modbus driver for the REX Control System (the `MbDrv` module)

## User guide

REX Controls s.r.o.

Version 2.50.5
Plzeň (Pilsen), Czech Republic
2017-09-06

# Contents

# Chapter 1

# The MbDrv driver and the REX Control System

## 1.1 Introduction

This manual describes the MbDrv driver for data exchange between the REX Control System and various devices supporting the Modbus protocol [1]. The driver supports both the RTU version for the RS-232 or RS-485 serial lines and the TCP/IP version.

The driver supports both Master and Slave modes. Thus in fact the MbDrv driver contains 4 drivers – serial line Master, serial line Slave, TCP/IP Master and TCP/IP Slave. All versions have much in common and must be installed all at once.

## 1.2 System requirements

The MbDrv driver can be used on Windows and Linux target devices. The RTU version requires a serial port while the TCP/IP version requires the TCP/IP stack (Ethernet card, USB WiFi dongle etc.).

In order to use the driver, the host computer (development) and the target computer (runtime) must have the following software installed:

| | |
|---|---|
| **Host computer** | |
| Operating system | one of the following: Windows 7/8/10 |
| REX Control System | version for Windows operating system |
| | |
| **Target device** | |
| REX Control System | runtime core for the corresponding operating system |
| IO driver | version for the corresponding operating system |

## 1.3 Installation of the driver on the host computer

The MbDrv driver is included in the installation package of the Development tools of the REX Control System. It is necessary to select the corresponding package in the installer. The REX Control System typically installs to the
C:\Program Files (x86)\REX Controls\REX <version> folder.

The following files are copied to the installation folder:

Bin\MbDrv_H.dll – Configuration part of the MbDrv driver.

Bin\MbDrv_T.dll – Target part of the MbDrv driver which is called by the RexCore runtime module.

Doc\PDF\ENGLISH\MbDrv_ENG.pdf – This user manual.

## 1.4 Installation of the driver on the target device

### 1.4.1 Windows machines

The target part of the driver, which is used for running REX Modbus Master or Slave on Windows 7/8/10 is included in the Development tools of the REX Control System as mentioned above.

### 1.4.2 Linux machines

If there is no RexCore runtime module installed on your target device, install it first using the Getting started guide of the REX Control System [2].

In order to enable Modbus communication in the REX Control System the driver must be installed. This is done from command line using the command

**Debian:**

```
sudo apt-get install rex-mbdrvt
```

**OpenWrt:**

```
opkg install rex-mbdrvt
```

# Chapter 2

# Including the driver in the project

The driver is included in the project as soon as the driver is added to the project main file and the inputs and outputs are connected in the control algorithms.

## 2.1 Adding the MbDrv driver

The project main file with the MbDrv driver included is shown in Figure 2.1. The Modbus Master of the TCP/IP version is shown.
There are 2 blocks which must be added to the project to include the driver. First the MODULE block is attached the the Modules output of the EXEC function block. It must be renamed to MbDrv.

The other block of type IODRV is named MTM and it is connected to the Drivers output of the main EXEC block. The three most important parameters are:

module – name of the module linked to the driver, in this case MbDrv – the name is CASE SENSITIVE!

classname – class of the driver, which defines the role of the target device and the Modbus version to use:

- MbmDrv – for Modbus RTU Master
- MbsDrv – for Modbus RTUSlave
- MtmDrv – for Modbus TCP/IP Master
- MtsDrv – for Modbus TCP/IP Slave

The name is CASE SENSITIVE!

cfgname – name of the driver configuration file (*.rio, REX Input/Output file), which is discussed in chapter 3

The name of this block (MTM, see Fig. 2.1), is the prefix of all input and output signals provided by this driver for Modbus TCP/IP Master. Similarly, the IODRV block can be named MTS, MBM and MBS for TCP/IP Slave, RTU Master and RTU Slave.

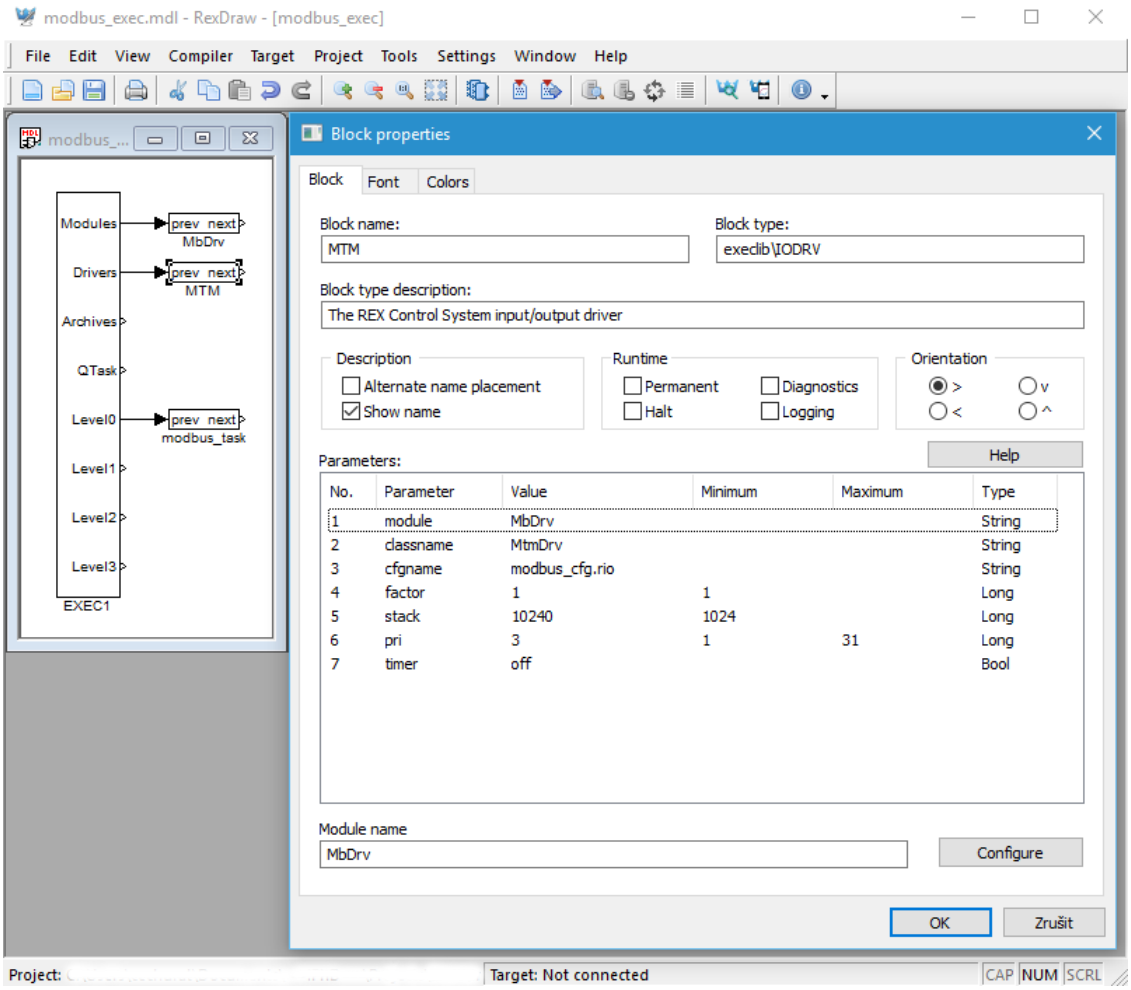


Figure 2.1: An example of project main file with the `MbDrv` driver included

The above mentioned parameters of the `IODRV` function block are configured in the RexDraw program. The configuration dialog is shown also in Fig. 2.1.

The `Configure` button opens the configuration dialog of the `MbDrv` driver, which is described in chapter 3.

## 2.2 Connecting the inputs and outputs in the control algorithm

The inputs and outputs of the driver must be interconnected with the individual tasks (`.mdl` files). The individual tasks (`QTASK` or `TASK` blocks) are connected to the `QTask`, `Level0`,..., `Level3` outputs of the main `EXEC` block. Use the blocks depicted in Fig. 2.2

to interchange data between the control algorithm and the `MbDrv` driver. Figure 2.3

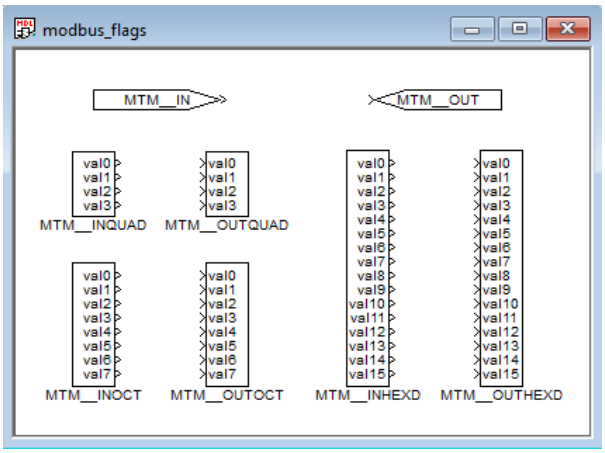


Figure 2.2: Example of input and output flags of the `MbDrv` driver

shows an example of a PID control loop with one input (temperature) and one output (power) signal provided by the `MbDrv` driver.

The `From` block allowing the user to read one input signal has the `Goto tag` set to `MTM__temperature`. The `Goto` block allowing the user to set one output signal has the `Goto tag` set to `MTM__power`. The blocks always have the `MTM` prefix right at the beginning of the tag followed by two `__` underscore. The blocks with multiple inputs/outputs have this prefix directly in their name.

The use of multi-input/output blocks is recommended if data exchange rate (sampling frequency) is the priority. See the function block reference manual [3] for details about `INOCT`, `OUTOCT`, `INHEXD`, `OUTHEXD` blocks.

Example project with input and output flags for the `MbDrv` driver are prepared by default in the folder
`C:\Program Files (x86)\REX Controls\REX <version>\Examples\Modbus_examples\00_IO_Flags`.

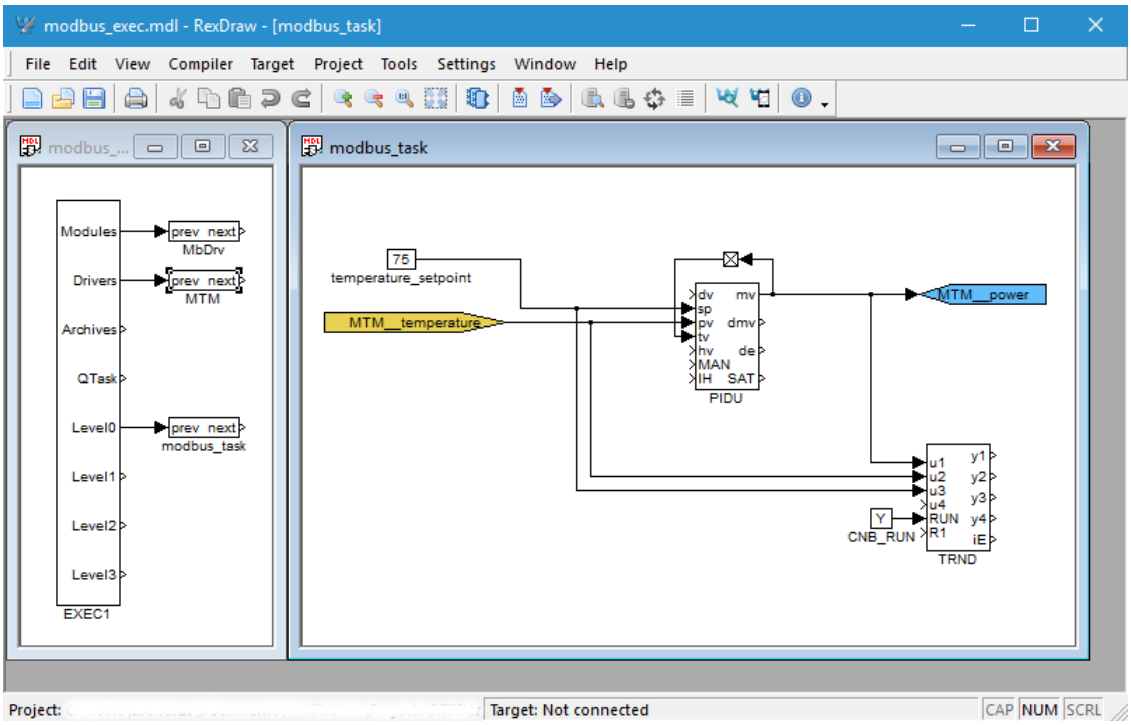


Figure 2.3: A simple PID control loop

# Chapter 3

# I/O driver configuration

This chapter describes the configuration of individual input and output signals and their symbolic naming. The signals are mapped to addresses of the Modbus protocol in individual stations.

The configuration dialog is part of the `MbDrv_H.dll` file. It can be activated from RexDraw by pressing the `Configure` button in the parameters dialog of the `IODRV` block (see chapter 2).

## 3.1 Modbus Master – configuration dialog

### 3.1.1 Modbus TCP/IP Master configuration

The configuration dialog is shown in Figure 3.1.

The upper left part of the dialog contains `Connection` parameters. Namely:

`Response timeout` – Maximal time (in seconds) to wait for the response from Slave station. The station is considered malfunctioning if no valid response is received.

`Retry time` – Time interval (in seconds) for testing the malfunctioning stations.

If the REX Control System acts as `TCP/IP` Master, it is necessary to define the Slave stations. The center part of dialog depicted in Fig. 3.1 displays the defined Slave stations. New Slave station can be added by pressing `Add slave` in the right column. Existing slaves can be edited directly in the Slave list or by pressing `Edit slave` button.

Following parameters define slave device:

`Name` – A unique station name.

`Address` – IP address of the Slave station.

`Port` – Port (TCP address) of the Slave station. The default port for Modbus is 502.

`Max requests` – Maximum number of Modbus telegrams in the queue. Especially the embedded devices with limited memory usually have a small TCP/IP stack buffer.

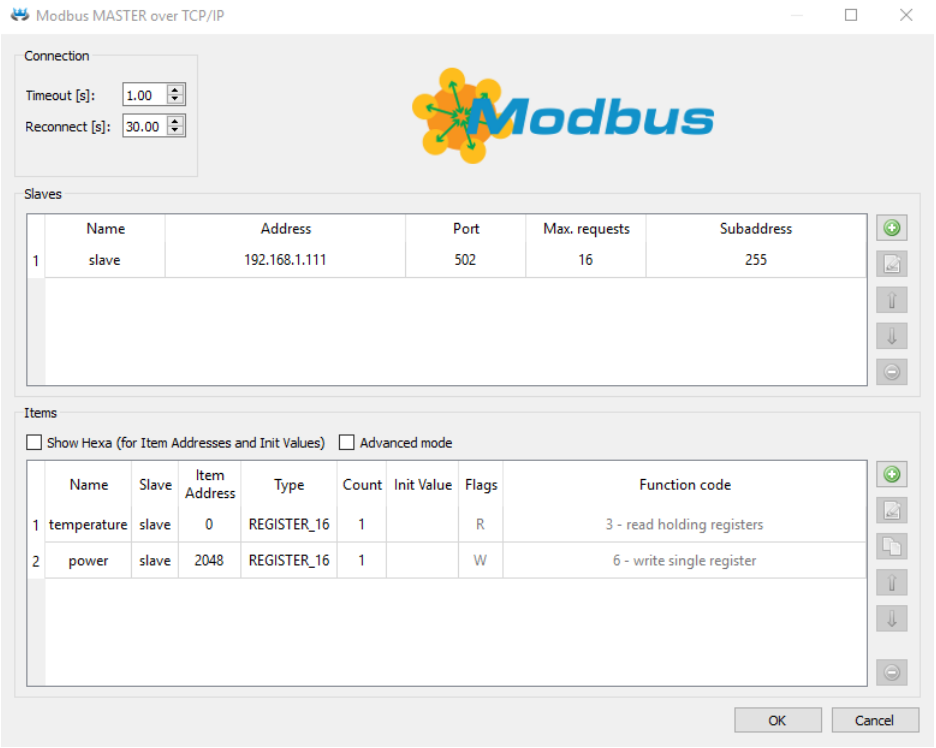


Figure 3.1: Modbus TCP/IP Master configuration dialog

The Master station tracks the requests and responses of each Slave station and postpones the requests if the Slave station fails to respond in a timely manner.

`Subaddress` – Additional address field, values 1 to 255. The value of 255 is reserved for broadcast. Do not use 255 unless you have a special reason.

### 3.1.2 Modbus RTU Master configuration

The configuration dialog is shown in Figure 3.2.

The upper left part of the dialog contains `Connection` parameters. Namely:

`Port` – The serial line used for communication. Usually `COM*` for Windows target devices or `/dev/ttyS*` for Linux target devices. Replace "`*`" symbol according to the chosen serial port!

`Baud rate` – All stations on one bus must use the same baud rate. In bits per second.

`Parity` – The error checking mechanism. All stations on one bus must use the same parity.

`Stopbit` – Stop bits sent at the end of every character. All stations on one bus must use the same parity.

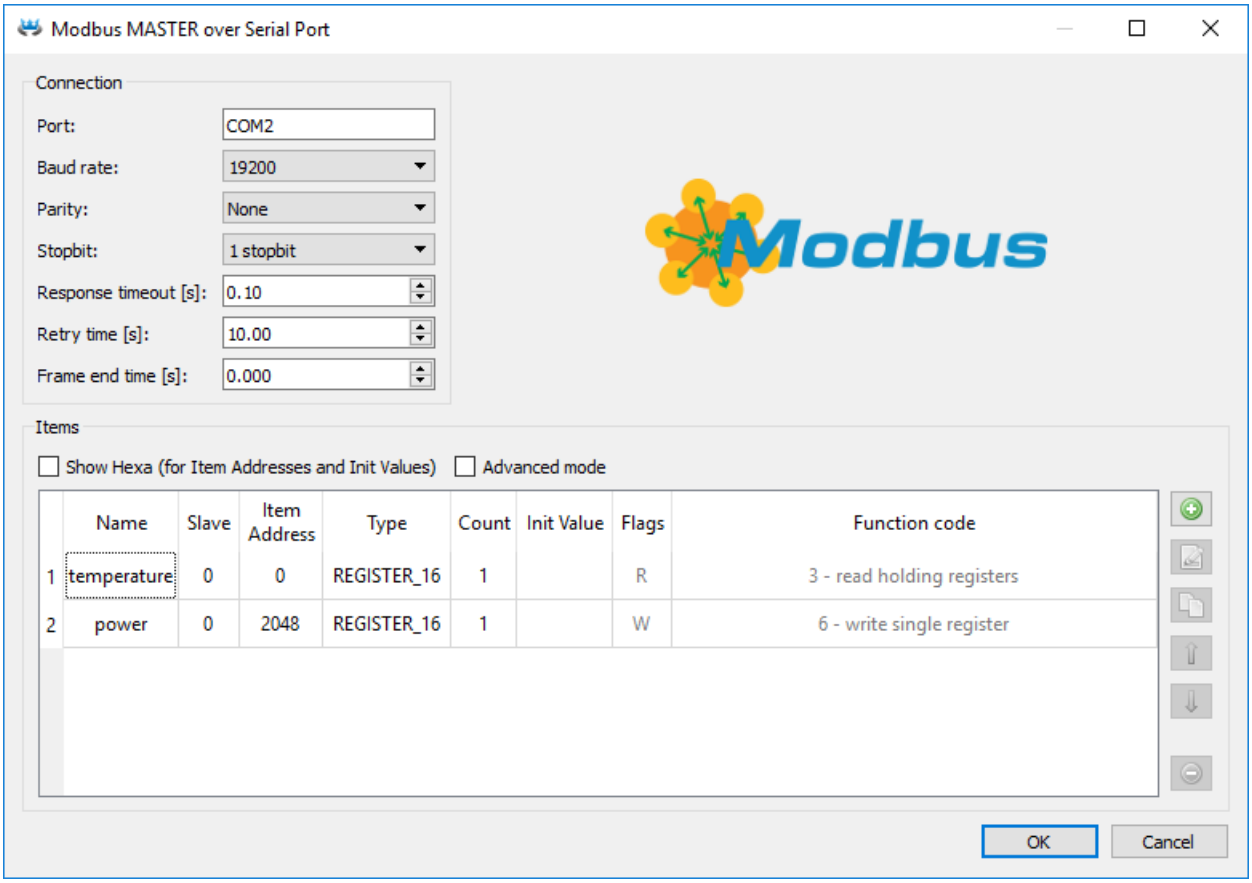


Figure 3.2: Modbus RTU Master configuration dialog

`Response timeout` – Maximal time (in seconds, default 0.1 s) to wait for the 1st byte of response frame from Slave station after the request was sent out completely. The station is considered malfunctioning if no valid response is received 3 times in a row. Do not use long timeouts, especially when there are multiple Slave stations on the bus. In case of a malfunctioning station it is not possible to communicate with other stations for the whole timeout period.

`Retry time` – Time interval (in seconds, default 10 s) to wait before retrying communication with malfunctioning stations.

`Frame end time` – How long silence time on the line is considered as the frame end. Value 0 means automatic configuration based on baud rate according to Modbus specification (3.5 characters) with some safety margin for processing in the operating system. Please try to increase this value up to the value of `Response timeout` parameter if you are facing `Read serial device timed out` errors in system log (driver behaviour changed since version 2.50.5).

## 3.2 Modbus Slave – configuration dialog

### 3.2.1 Modbus TCP/IP Slave configuration

The configuration dialog is shown in Figure 3.3.

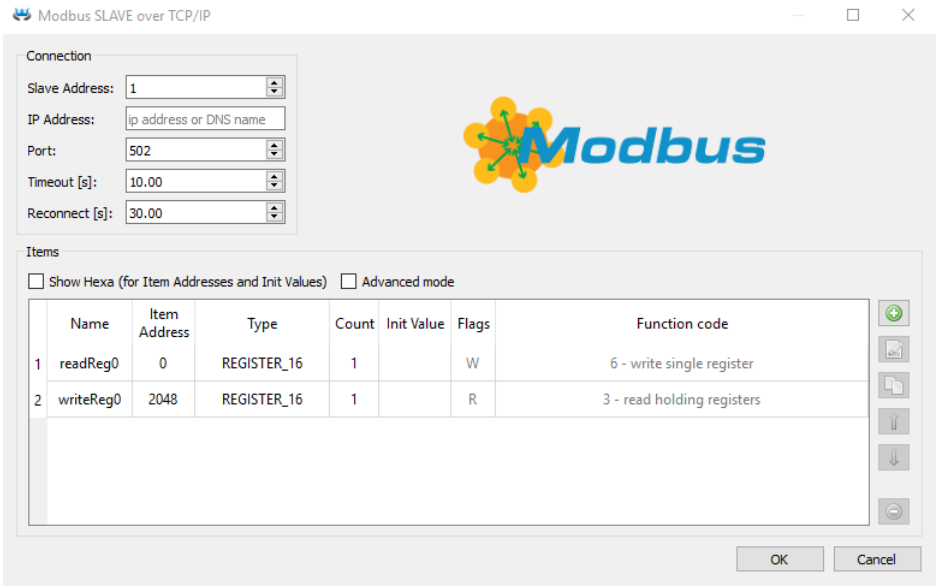


Figure 3.3: Modbus TCP/IP Slave configuration dialog

The upper left part of the dialog contains `Connection` parameters. Namely:

`Slave address` – Slave device address. There can be up to 255 Slave stations on same IP address. The address 255 is reserved for broadcast.

`IP Address` – Defines the network adapter where the Slave is listening. Leave it blank for all network adapters or specify IP Address of a chosen one.

`Port` – Port (TCP address) of the Slave station. The default port for Modbus is 502.

### 3.2.2 Modbus RTU Slave configuration

The configuration dialog is shown in Figure 3.4.

The upper left part of the dialog contains `Connection` parameters. Namely:

`Slave address` – Slave device address. There can be up to 32 Slave stations. The address 255 is reserved for broadcast.

`Port` – The serial line used for communication. Usually `COM*` for Windows target devices or `/dev/ttyS*` for Linux target devices. Replace "`*`" symbol according to the chosen serial port!

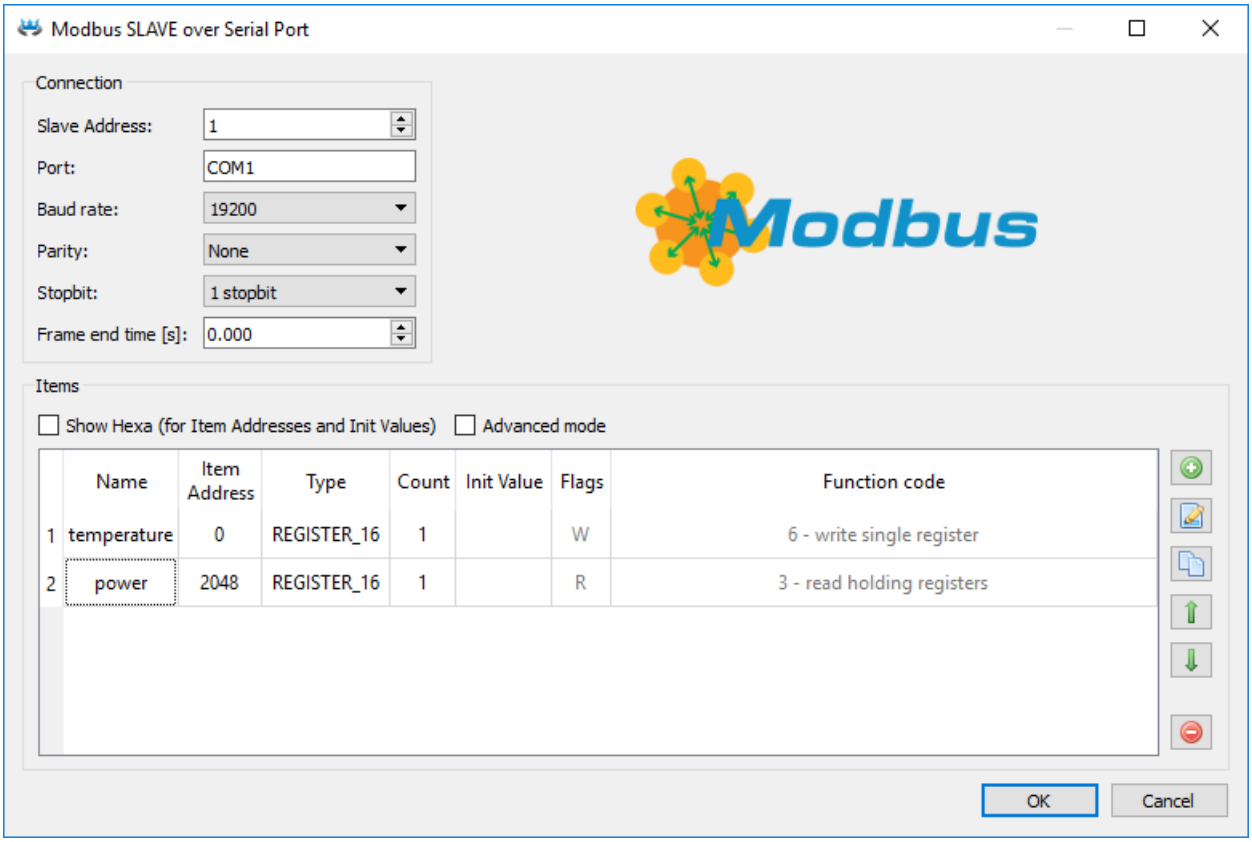


Figure 3.4: Modbus RTU Slave configuration dialog

`Baud rate` – All stations on one bus must use the same baud rate. In bits per second.

`Parity` – The error checking mechanism. All stations on one bus must use the same parity.

`Stopbit` – Stop bits sent at the end of every character. All stations on one bus must use the same parity.

`Frame end time` – How long silence time on the line is considered as the frame end. Value 0 means automatic configuration based on baud rate according to Modbus specification (3.5 characters) with some safety margin for processing in the operating system. Please try to increase this value up to the value of `Response timeout` parameter if you are facing `Read serial device timed out` errors in system log (driver behaviour changed since version 2.50.5).

## 3.3 Modbus items configuration

The configuration dialog is shown in Figure 3.5.

The lower part of the configuration dialog displays the configured signals. Each line has a symbolic name and corresponds to one signal or a group of signals. New item can be added by pressing the `Add item` button in the right column of the configuration dialog. The item can be similarly edited by pressing the `Edit item` button. Some of the parameters (`Name`, `Item Address`, `Type` and `Count`) can be edited directly in the Item list. If you choose `Advanced mode` it unlocks all other parameters for direct edit in the Item list.

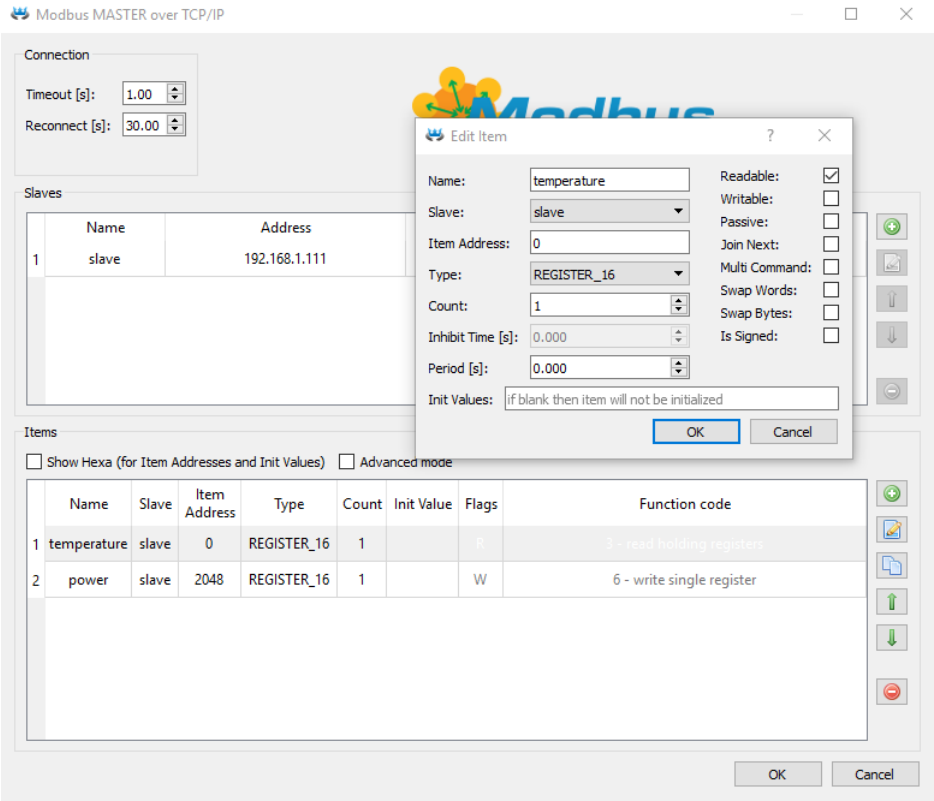


Figure 3.5: Modbus item configuration dialog

The individual columns have the following meaning:

`Name` – A unique signal name.

`Slave` – Select slave device. Available only in Modbus TCP/IP Master item configuration.

`Slave Address` – Slave device address. Available only for Modbus RTU Master item configuration.

`Item Address` – Address of the object in the device. All objects (values) within one station have an identification number (address) from the range 0 to 65535.

`Type` – Signal type. The names respect the Modbus specification:

| | |
|---|---|
| `Input` | Digital input |
| `Coil` | Digital output |
| `Register (16bit)` | integer number 0...65535, |
| `Register (32bit)` | integer number 0...4294967296, |
| `Register (float)` | 4-byte floating point number |
| `Register (double)` | 8-byte floating point number |

`Count` – Number of values. One item can represent a group of values which are read at once using the multi-input/output block. This number defines how many values to read. The number does not necessarily correlate with the number of inputs/outputs of the block. In such a case, the unused pins are not updated.

`Inhibit Time` – Valid only for `Writable (W)` items. Defines the minimum length of time that must be allowed to elapse between the transmissions of the item.

`Period` – Refresh period in seconds. Nonzero values define the period, zero respects the period given by the corresponding `IODRV` block.

`Initial value` – The initial value to set the signal to when initializing the driver. Use square brackets for groups of values. Separate the values by spaces.

`Readable (R)` – Tick this checkbox to allow reading of the value in the REX algorithm. In Slave configuration, such items will be writable for the Master.

`Writable (W)` – Tick this checkbox to allow writing of the value from the REX algorithm. In Slave configuration, such items will be readable for the Master.

`Passive (P)` – In some cases it is more efficient to transmit a large group of signals at once (maximum is 125 values). But we still want to work with individual signals or less populated groups in the algorithm. In that case we define one big array and a number of small groups overlaying the data registers. Only the big array is active, the other signals are set to passive. This field is available only in the Master configuration.

`Join next (J)` – Tick this checkbox to use the read-write command of the Modbus specification. The readable signal is joined with its successor (the writable item one line below) and the data is exchanged using a single command. This field is available only in the Master configuration.

`Multi command (m)` – Tick this checkbox to force multi-read or multi-write command even if only one value is transmitted. Useful for Modbus stations supporting only a subset of Modbus commands. This field is available only in the Master configuration.

`Swap word (a)` – 32-bit numbers are represented by 2 successive 16-bit numbers in Modbus. This flag defines the order of lower and upper word. The same holds for real numbers as they are represented by 2 or 4 successive 16-bit registers.

`Swap byte (b)` – The upper byte of the 16-bit number is transmitted first in Modbus (the so-called big-endian format). However, some devices use little-endian implementation. This flag becomes handy in such a case.

`Is Signed (S)` – Tick this checkbox to treat the integer number as a signed number.

## 3.4 Special signals

Additional diagnostic signals are available, namely:

| | |
|---|---|
| `ErrorFrame` | number of invalid frames received |
| `ErrorTimeout` | number of timeout occurences |
| `ErrorData` | number of valid frames with unexpected data or error code |
| `ErrorReset` | flag for resetting the above mentioned counters |
| `Browse` | non-zero value switch driver into browsing address space mode. Output is in system log. |

Every signal also has its attributes, namely:

| | |
|---|---|
| `_Value` | alias for signal value |
| `_ReadEnable` | configuration parameter `Readable` |
| `_WriteEnable` | configuration parameter `Writable` |
| `_Address` | configuration parameter `Item address` |
| `_Slave` | configuration parameter `Slave address` |
| `_Fresh` | time elapsed since the last data update (in seconds) |
| `_Period` | configuration parameter `Refresh rate` |
| `_Send` | flag indicating that the value is waiting to be stored in the Slave device |

The `Fresh` attribute is read-only, the other attributes are both readable and writable. Beware the `Fresh` attribute is updated even if the read/write operation fails. In such a case the signal quality is set to `BAD` or `UNCERTAIN`.

Thus if we want to know the freshness of the `MBM__IN` signal, we use the `From` block and set the `Goto Tag` parameter to `MBM__IN_Fresh`.

There is also one more signal representing the state of the Slave device. E.g. the `MTM__SlaveStation` signal tells us if the `SlaveStation` device is operational. To determine whether the Master is trying to establish connection with the slave device, use the `MTM__SlaveStation_Connecting` signal (it remains `on` even after the Slave station starts responding). Once the connection is established the `MTM__SlaveStation_Connected` signal is `on`.

# Chapter 4

# Implementation details

Additional information about Modbus implementation in the REX Control System is gathered in this chapter.

- Do not change the `Sync. time` parameter unless necessary (default setting: 0). Only minor devices with slow CPU detect the packets incorrectly and need longer time between them.
- The `Timeout` period is measured from the end of the request to the end of the response. The time necessary to send the Modbus message is added to the timeout. Modbus message contains 10 to 16 bytes plus the values to transmit. Maximal length is 256 bytes.
- The `Reconnect` parameter is important only if there are multiple stations on one bus.
- The Modbus protocol uses only 16-bit registers or groups of successive 16-bit registers. The `Register (32bit)` and `Register (float)` thus occupy two 16-bit registers. If we use the address 100 for `Register (float)`, we cannot use the address 101 as it is already occupied by the float register.
- It is recommended not to combine read and write operations on one signal. Use two strictly read-only and write-only signals pointing to the same register if you need both reading and writing the data.
- The `Readable` and `Writable` flags are always relative to the REX control algorithm running on the REX target device. It is intuitive for Modbus Master but it might be confusing for the Slave. The Modbus Master writes to the registers, which are configured as readable in the Slave.
- Only the address of the first item should be used when defining groups of signals (`Count` $> 1$).
- The `Refresh rate` cannot be faster than reading and writing all signals. Use this parameter to avoid repetitive transmission of slowly changing signals and save the

bandwidth for other signals. The real refresh rate can be displayed in the RexView program.

- The values are read/written in the same order as they are shown in the configuration dialog.
- Some Modbus configuration tools use register addresses starting from 40001 (or from 400001 in the case of 984-series devices) for holding registers. The REX Control System always uses the physical addresses starting from 0.
- The Modbus communication is asynchronous to the REX control algorithm. The driver contains a cache for all signals. The Modbus Master cycles through all signals. If the signal is blocked by the `Refresh rate` parameter or the corresponding station is not responding, the signal is skipped. Otherwise the read query or write command is issued. If there is no response within the timeout period, the Master marks the corresponding station as *Disconnected*. The Modbus Slave only waits for requests of the Master and returns the cached values or updates the cache. All possible situations are described in the following table:

| | Master | Slave |
|---|---|---|
| read | reads the cached value (obtained in the previous cycle of the Modbus line) | reads the cached value (updated by the last write command on the Modbus line) |
| write | writes the value to cache (if it differs from the previous value, it is transmitted during the nearest Modbus cycle) | writes the value to the cache (Modbus Master receives the value as soon as it issues the read command) |
| read and write | cached value is read, standard write operation follows | cached value is read, standard write operation follows |

# Chapter 5

# Troubleshooting

In the case that the diagnostic tools of the REX Control System (e.g. RexView) report unexpected or incorrect values of inputs or outputs, it is desirable to test the functionality outside the REX Control System (command line tools, Modbus simulators, etc.). Also double check the configuration – the most common problems include:

Hardware problem – incorrect wiring (cross-cable vs. direct-cable, connector pinout), TTL vs. RS232 signal levels, signal polarity, etc.

The configured serial channel is occupied by another program

The device uses non-standard Modbus implementation (byte-order or word-order)

The device is very slow and needs longer `Sync. time`

Incorrect Slave address or register number.

Mismatch in readable and writable flags.

The signal is defined in inappropriate configuration file (when using multiple Modbus lines).

Changing the parameters from algorithm. Check the values, especially during startup phase.

ADAM 5000 TCP does not work correctly if TCP stack combines two or more Modbus messages into one Ethernet frame. Messages indicating that timeout expired appear in the system log. A workaround is to set `Max. requests = 1` for that particular slave device, however it reduces communication throughput and response times.

Setting `Max. requests = 1` is recommended whenever an unexpected behavior of the Modbus communication is observed.

In the case that the given input or output works with other software tools and does not work in the REX Control System, report the problem to us, please. E-mail is preferred, reach us at support@rexcontrols.com. Please include the following information in your description to help us process your request as soon as possible:

- Identification of the REX Control System you are using. Simply export it to a file using the RexView program (Target → Licence → Export).
- Short and accurate description of your problem.
- The configuration files of the REX Control System (`.mdl` and `.rio` files) reduced to the simplest case which still demonstrates the problematic behavior.

# Bibliography

[1] *Modbus Application Protocol Specification V1.1b*. http://www.Modbus-IDA.org, 2006.

[2] REX Controls s.r.o.. *Getting started with REX on Raspberry Pi*, 2017.

[3] REX Controls s.r.o.. *Function blocks of the REX Control System – reference manual*, 2017.