

# MODBUS Server

## MODBUS Server communication protocol

[Supported device types and versions](#)  
[Communication line configuration](#)  
[Communication station configuration](#)  
[Line protocol parameters](#)  
[I/O tag configuration](#)  
[Literature](#)  
[Changes and modifications](#)  
[Document revisions](#)

### Supported device types and versions

The protocol executes a slaver communication with arbitrary devices that support standard MODBUS RTU in versions of serial communication and MODBUS over TCP/IP.

### Communication line configuration

- Line category: [Serial](#) (serial communication), [SerialOverUDP Device Redundant](#) (serial communication).
- Line category [TCP/IP-TCP](#) (MODBUS over TCP/IP). Use a symbolic address *ALL*, KOM process listens on selected TCP port on all existing network interfaces. TCP port 502 is commonly used, but any of the ports can be used.  
Line number - set the value for example 1. KOM process works as multitask TCP server and that is why it is able to operate more clients at the same time.

### Line protocol parameters

[Configuration dialog box](#) - table **Parameters**.

They influence some optional parameters of protocol. Following line protocol parameters can be set:

**Table 1**

Parameter	Meaning	Unit	Default value
Silent Interval	Delay which is entered between transmission begin of each data packet.	ms	50
No Request Timeout	If timeout passes and no valid call comes, all station on the line will be in status of communication error. Values of output I/O tags will never become valid (server protocol).	mi:ss	1:00
Single Server	If the value of parameter is set on YES, KOM process replies by error SLAVE_DEVICE_FAILURE upon each call which is send to station and this station is not configured. If the value is NO, KOM process ignores this call.	YES /NO	YES
Moxa Timeout	Switching time of Moxa redundant devices in case of communication error or some problems. As this is a server protocol which wait on demands from other external devices, a failure of the communication demand for longer time than the value of this parameter is considered to be an error. It is used only for <a href="#">SerialOverUDP Device Redundant</a> line.	sec	10 sec

### Communication station configuration

- Communication protocol "**Modbus Server**".
- Station address is decimal figure in the range of 1 up to 247. Address 0 is reserved for broadcast.

There is possible to configure more stations with different addresses on one line, KOM process will reply to proper station. See also the protocol parameters of line [Single Server](#).

### Station protocol parameters

[Configuration dialog box](#) - tab **Parameter**.

They influence some optional parameters of protocol. Following station protocol parameters can be set:

**Table 2**

Parameter	Meaning	Unit	Default value
Addressing model	Sets an address model of MODBUS protocol: <b>MODBUS PDU</b> data are addressed from 0 up to 65535. <b>MODBUS data Model</b> data are addressed from 1 up to 65536.  <b>Note:</b> <i>MODBUS PDU</i> is a default value. If <i>MODBUS data Model</i> is set, the object with the address X is addressed as X-1 in <i>MODBUS PDU</i> . After you change this parameter, restart a proper communication process.	MODBUS PDU MODBUS data Model	MODBUS PDU

## I/O tag configuration

---

Possible value types: **Ai, Ao, Di, Do, Ci, Co.**

### I/O tag address:

In the protocol MODBUS there is the main address space divided into registers of Coils type (reading/writing), Discrete Inputs (reading), Holding Registers (reading/writing) and Input Registers (reading).

An independent address space of 2 byte, i.e. addresses from 0 up to 65535, is in each address space of given type of register.

#### Address format of I/O tag:

Address format is `[I|U|f|F|C|D]Fn.Address[.BitNr]` in which:

- The first optional character defines the type of I/O tag:
  - **I** - Integer 16 bit
  - **U** - Unsigned 16 bit (default)
  - **f** - Float 32 bit (two registers) in *big endian* format (bytes B4, B3, B2, B1 will be sent, B4 is highest byte and B1 is lowest byte of float)
  - **F** - Float 32 bit (two registers) in *little endian* format (bytes B2, B1, B4, B3 will be sent, B4 is highest byte and B1 is lowest byte of float)
  - **C** - Request counter up (16 bit unsigned, which is incremented by every read request). Works only for Fn=3 or Fn=4
  - **D** - Request counter down (16 bit unsigned, which is decremented by every read request). Works only for Fn=3 or Fn=4
- **Note:** Request counter up/down can be used to configure a "watchdog" I/O tag to monitor status and speed of Modbus communication.
- Parameter *Fn* is a function of Modbus protocol for data reading, which inserts I/O tag into proper address space:
  - **1** - Coils: binary statuses
  - **2** - Discrete Inputs: binary inputs
  - **3** - Holding Registers: status registers
  - **4** - Input Registers: input registers
- Parameter *Address* is 2-bytes address of register in the range of 0 up to 65535.
- Parameter *BitNr* optionally specifies a bit of register in the range of 0 upto 15.

Implementation of protocol supports the following functions:

- **1** - Read Coils: reading of binary status - KOM process sends values of I/O tags of Do type.
- **2** - Read Discrete Inputs: reading of binary inputs - KOM process sends values of I/O tags of Do type.
- **3** - Read Holding Registers: reading of status registers - KOM process sends values of I/O tags of Co, Ao type (signed/unsigned).
- **4** - Read Input Registers: reading of input registers - KOM process sends values of I/O tags of Co, Ao type (signed/unsigned).
- **5** - Write Single Coil: writing of binary statuses - KOM process writes a received binary value in I/O tag of Di, Do type into system.
- **15** - Write Multiple Coils - KOM process writes all received binary values of I/O tag of Di, Do type into system.
- **6** - Write Single Register: writing of status registers - KOM process writes the received value in I/O tag of Ai, Ao, Ci, Co type into system.
- **16** - Write Multiple registers: writing of multiple registers - KOM process writes all received values in I/O tags of Ai, Ao, Ci, Co type into system.

**Note:** This is a server type of protocol which is primary intended for sending the values out of D2000 system. Therefore the I/O tags should be configured as output (Ao, Co, Do) because of manipulation of their values directly or by control objects. If the I/O tag is configured as input (Ai, Ci, Di), KOM process does not make to send a valid value at reading by functions 1 - 4 until the value is written by function 5, 15, 6 or 16 from outside.

If KOM process does not have the valid value of I/O tag or request to read a nonexistent I/O tag is received, an implicit value False or 0 is sent at call to read (MODBUS protocol does not support the transfer of value quality).

## Literature

---

- MODBUS APPLICATION PROTOCOL SPECIFICATION V1.1b, December 28, 2006. <http://www.modbus.org>

## Changes and modifications

---

-

## Document revisions

---

- Ver. 1.0 - April 24th, 2009 - document creating
- Ver. 1.1 - November 21st, 2010 - document update.
- Ver. 1.2 - November 11th, 2011 - document update.



#### Related pages:

[Communication protocols](#)