Heinzmann GmbH & Co. KG
Engine & Turbine Management

Am Haselbach 1
D-79677 Schönau
Germany

Phone +49 7673 8208-0
Fax +49 7673 8208-188
e-mail info@heinzmann.com
www.heinzmann.com

V.A.T. No.: DE145551926

HEINZMANN®

Engine & Turbine Management

Digital Speed Governors

Modbus/TCP

Instruction manual

10508 Publication DG 24 001-e / 07-24

	**The appropriate manuals must be thoroughly studied before in-stallation, initial start-up or maintenance** All instructions pertaining to the system and safety must be followed in full. Non-observance of the instructions may lead to injury to per-sons and/or material damage. HEINZMANN shall not be held liable for any damage caused through non-observance of instructions. Independent tests and inspections are of particular importance for all applications in which a malfunction could result in injury to persons or material damage.
	All examples and data, as well as all other information in this manual are there solely for the purpose of instruction and they may not be used for special application without the operator running independent tests and inspections beforehand. HEINZMANN does not guarantee, neither expressly nor tacitly, that the examples, data or other information in this manual is free from er-ror, complies with industrial standards or fulfils the requirements of any special application.
	**To avoid any injury to persons and damage to systems, following monitoring and protective systems must be provided:** - Thermal overload protection - Overspeed protection independent of the rpm controller HEINZMANN shall not be held liable for any damage caused through missing or insufficiently rated overspeed protection. **For ship propulsion additionally following must be provided:** - Second power supply for the engine control system If the governor is used for propulsion, a second power supply has to be provided to the engine control system if the governor can't keep its position in case of power failure. **For alternator systems additionally following must be provided:** - Overcurrent protection - Protection against faulty synchronisation for excessively-large frequency, voltage or phase difference - Directional contactor Reasons for overspeed may be: - Failure of actuator, control unit or its auxiliary devices - Jamming or sluggishness of linkage

	**To avoid material damage with electronically controlled injection (MVC), the following must be observed additionally:** For common rail systems each injector line must be equipped with a separate mechanical flow-rate limiter For unit pump (PLD) and pump-injector unit (PDE) systems, the fuel enable is first made possible by the solenoid valve's control plunger motion. This means that in the event of the control plunger sticking, the fuel supply to the injection valve is stopped.
	**Before installation the following must be observed:** Always disconnect electrical mains supply before any interventions to the system. Only use cable screening and mains supply connections that correspond with *European Union EMC Directive.* Check proper function of all installed protection and monitoring systems.
	As soon as the actuator is supplied with power, its output shaft may move automatically at any time. The working area of the output shaft or the linkage must therefore be secured against unauthorised access.
	HEINZMANN expressly rejects any implied guarantee pertaining to any marketability or suitability for a special purpose, including in the event that HEINZMANN was notified of such a special purpose or the manual contains a reference to such a special purpose.
	HEINZMANN shall not be held liable for any indirect and direct damage nor for any incidental and consequential damage that results from application of any of the examples, data or miscellaneous information as given in this manual.
	HEINZMANN shall not provide any guarantee for the design and planning of the overall technical system. This is a matter of the operator its planners and its specialist engineers. They are also responsible for checking whether the performances of our devices match the intended purpose. The operator is also responsible for a correct initial start-up of the overall system.

Revisions index

Revision No.	Date of modification	Name	Description of modification or remarks
01	may 2024	EnJ	First edition document “Modbus/TCP“ DG 24 001-e / 05-24
02	july 2024	EnJ	Corrections and amendments chp. 4 & 5

Table of contents

Seite

1 Safety instructions and related symbols 6
1.1 Basic safety measures for normal operation 7
1.2 Basic safety measures for servicing and maintenance 7
1.3 Before commissioning after maintenance or repair work 7

2 Introduction 8
2.1 Data transfer message format 8
2.2 Example of data transmission 9

3 Module functions 11
3.1 Unit address 11
3.2 Supported function codes 11
3.3 Supported exception codes 12

4 Parameterization and commissioning 13
4.1 Reading out data 14
4.2 Writing data 16
4.2.1 Timeout monitoring 17
4.2.2 Assigning inputs to the sensors 17
4.2.3 Assignment of the binary values to the switch functions 18
4.2.4 Assignment of Modbus functions 19
4.3 Diagnostic counter 20

5 Parameter description 21
5.1 List 1: Parameters 21
5.2 List 2: Measured values 21
5.3 List 3: Functions 22
5.4 List 4: Characteristics and maps 22

6 List of figures 23

7 List of tables 23

8 Index 23

1 Safety instructions and related symbols

This publication offers wherever necessary practical safety instructions to indicate inevitable residual risks when operating the engine. These residual risks imply dangers to

- Personnel
- Product and machine
- Environment

The most important purpose of the safety instructions is to prevent personal injury!

The signal words used in this document are specifically designed to direct the attention to extent of possible damage!

DANGER indicates a hazardous situation which, if not avoided, will result in death or serious injury.

WARNING indicates a hazardous situation which, if not avoided, could result in death or serious injury.

CAUTION indicates a hazardous situation which, if not avoided, may result in minor or moderate injury.

NOTICE indicates a property damage message.

Safety instructions are also marked by triangular warning symbols in addition to the signal word. In some cases, round command symbols supplement the warnings. The symbols are intended to illustrate the danger as well as the measure for protection.

In any case, the symbols cannot replace the text of the safety notice. The text must therefore always be read in full!

This symbol does not refer to any safety instructions but provides important information for a better understanding of the functions. These should be observed and complied with.

1.1 Basic safety measures for normal operation

The system may only be operated by trained and authorized persons who are familiar with the operating instructions and can work in accordance with them!

Before each start of a machine or plant:

- Check it for visible damage and ensure that it is in perfect condition!
  Report any defects found immediately to the supervisor!
- Remove all materials or objects that are not required from the working area of the machine or plant!
- Check all safety devices and ensure their proper function!
- Ensure that only authorized persons are in the working area of the machine or plant and that nobody can be injured by the start-up of the machine or plant!!

1.2 Basic safety measures for servicing and maintenance

Before maintenance or repair work:

- Block access to the working area of the machine or plant for unauthorized persons! Attach a sign, which draws attention to the maintenance work!
- Switch off the main switch for the power supply and secure it against unauthorized switching on! Always keep the associated switch cabinets closed!
- Make sure that all parts of the machine or plant, which could be touched, are cooled down to room temperature and do not carry any electric voltage!
- Replace damaged lines or cables immediately!
- Never spray switch cabinets and other housings of electrical equipment with water for cleaning!!

1.3 Before commissioning after maintenance or repair work

- Reattach all loosened screw connections and check for tight fit!
- Ensure that the control linkage is reattached properly!
- Make sure that all cable connections are reconnected!
- Make sure that all safety devices of the system are activated and work properly!

2 Introduction

The HEINZMANN control units support Modbus transfer types Modbus RTU/ASCII and Modbus/TCP. The transfer types Modbus RTU and ASCII are described in more detail in the HEINZMANN document *DG_05_002_e_08_14_Modbus_6878*.

The sections below provide a general overview of the functionality, structure and structure of the Modbus TCP protocol. Experienced Modbus users can continue with Chapter *↑3 Module* functions.

2.1 Data transfer message format

The Modbus TCP protocol uses the client/server model, in which a client (master at RTU) requests or sends data from a server (slave at RTU). In the OSI model, the Modbus TCP protocol builds up on the TCP/IP protocol. A Modbus RTU message is mapped into the payload of a TCP/IP frame with a so-called MBAP header. The figure below shows the relationship between the transmission types and the classification of Modbus TCP in the so-called OSI model.

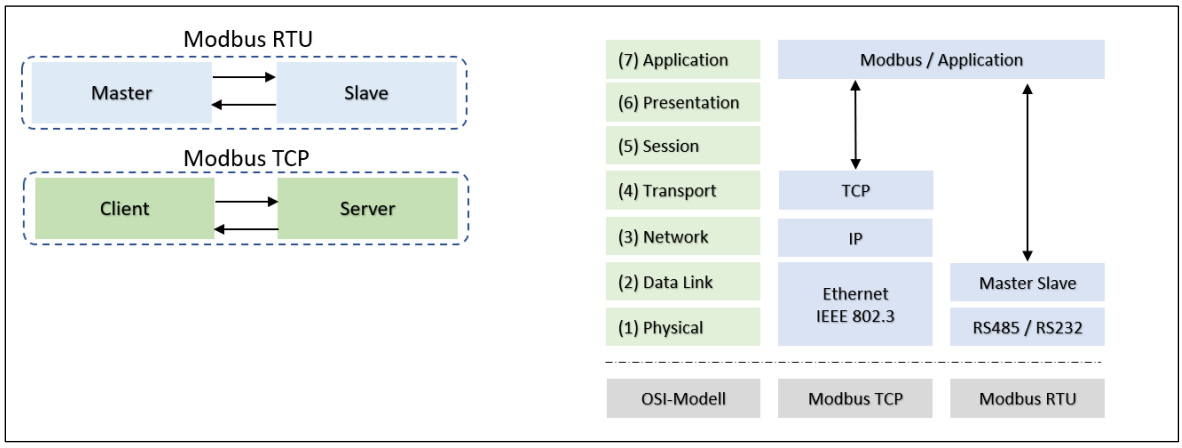

Fig. 1: Modbus transmission types

The Modbus TCP message consists of the Modbus Application Protocol header (MBAP header) and the Protocol Data Unit (PDU). In contrast to Modbus RTU, no checksum is attached at the end of the message as the functionality is adopted by the lower transmission layers. The following figure shows the detailed structure of a Modbus TCP message.

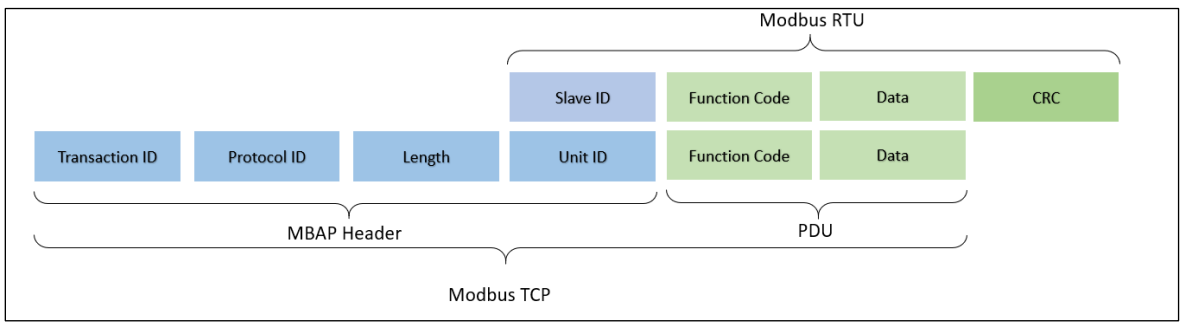

Fig. 2: Modbus TCP message

Transaction ID (2 bytes): Identification ID (non-sequential) from the client.
These bytes are copied by the server in the response.

Protocol ID (2 bytes): The protocol identification is always 0x0000, which corresponds to the Modbus protocol.

Length: Length (2 bytes), number of bytes that are sent.
This includes the unit ID + PDU

UnitID: (1byte) Corresponds to the Slave ID at Modbus RTU

2.2 Example of data transmission

The function code 0x03 *"Read Holding Registers"* explains the sequence of client request and server response in the following tables. The form structure of the sent character strings is shown. In the example, only the pure Modbus TCP message is addressed; additional bytes of other protocol layers are not taken into account. In the last column, a numerical example is entered in hexadecimal representation.

Transaction ID	Transaction ID	High byte	00
		Low byte	01
Protocol ID	Protocol identification	High byte	00
		Low byte	00
Length	Length of PDU data + unit ID	Length (high byte)	00
		Low byte	06
Unit ID	0 to 247 (decimal)	Slave address (in example 247)	F7
Function	1 to 255 (decimal)	Function to be performed	03
Data	Start address	Start address of the data register (high byte)	00
		Start address of the data register (low byte)	00
	No. of registers	Number of data (high byte)	00
		Number of data (low byte)	01

Table 1: Client enquiry

Transaction ID	Transaction ID	High byte	00
		Low byte	01
Protocol ID	Protocol identification	High byte	00
		Low byte	00
Length	Length of PDU data + unit ID	Length (high byte)	00
		Low byte	05
Unit ID	0 to 247 (decimal)	Slave address	01
Function	1 to 255 (decimal)	Function to be performed	03
Byte count	2 to 255	Number of data bytes	00
Data	Registers	Data value (high byte)	00
		Data value (low byte)	00

Table 2: Server response

3 Module functions

The HEINZMANN control unit always functions as a server and can be initiated by a client in the network via a request for a data transfer determined by the function code. The control unit's response to this request can be read-out of data, confirmation of receipt of data, an exception response or a no-response in accordance with the function code.

3.1 Unit address

The unit addresses of the members (ID) of the network must be set by the user on the member unit. Each ID can only be assigned once in a bus segment.

When assigning unit addresses, it is essential to ensure that several units are not assigned the same address. Otherwise abnormal behaviour of the serial bus may occur and the master is unable to communicate with all slaves on the bus.

3.2 Supported function codes

The functions supported by HEINZMANN from the Modbus function list are listed in the following Table 3 and provided with a brief description. If an unknown function code is requested from the master, the corresponding exception response is returned.

Function code	Name	Brief description
0x03	*Read holding register*	Read access to one or more data words for transmitting parameter or measured values of the control unit
0x06	*Write single register*	Write access to a data word
0x08	*Diagnostics*	Diagnostic function, e.g. reading out the diagnostic counter
0x10	*Write multiple register*	Write access to one or more data words

Table 3: Function code

The diagnostic function is limited to the subfunction codes shown in Table 4, all other subfunctions are not supported and rejected with an exception. The data field or word of the request must be assigned with the value 0x0000 except for the subfunction code 0x0000, which supports any length and selectable values of the data words.

This function is used to diagnose the communication that occurred and enables counter readings that are incremented after success, errors, etc. to be read out individually or together and reset.

Sub-function code	Name	Brief description
0x0000	*Return query data*	Query is returned as echo
0x0002	*Return diagnostics register*	Data of the subfunction codes 0x0B to 0x12 are sent back in this sequence
0x000A	*Clear Counters and Diagnostic Registers*	Counter readings of subfunction codes 0x0B to 0x12 are reset to zero
0x000B	*Return Bus Message Count*	Number of valid received messages is returned
0x000C	*Return Bus Communiction Error Count*	Number of incorrectly received messages is returned
0x000D	*Return Bus Exception Error Count*	Number of sent exception messages is returned
0x000E	*Return Slave Message Count*	Number of sent messages is returned
0x0012	*Return Bus Character Overrun Count*	Number of messages that were not received correctly due to character overrun

Table 4: Sub-functions of the diagnostic function

3.3 Supported exception codes

The supported function codes result in the possible exceptional codes, which may be included in an exception response. An exception response is sent to a request from the master if the information on the data to be transferred exceeds the unit's internal address and data ranges or if a function code is not supported. The Table 5 shows the exceptional codes of a HEINZMANN control unit.

Exception code	Name	Brief description
0x00	*No Exception*	No error code
0x01	*Illegal Function*	Function code is not supported
0x02	*Illegal Data Address*	Data address is outside the unit's internal ecessary data area
0x03	*Illegal Data Value*	Number of data exceeds the specified data area
0x04	*Slave Device Fail*	Unit cannot process the data

Table 5: Exception code

4 Parameterization and commissioning

The module is parameterised exclusively using the HEINZMANN basic unit software. DcDesk 6 is required as a Windows® programme to visualise and configure the interface.

Configuration of the interface requires the parameterisation of the IP address and the associated subnet and TCP port. In addition, the entire Modbus function scope must be switched on in the HEINZMANN control unit. The user has the parameters or function parameters for this purpose.

21500-21503	IPAddress(0)-(3)	Configuration of the IP address in the network
21504-21507	SubnetMask(0)-(3)	Configuration of the subnet mask
21508-21511	StandardGateway(0)-(3)	Configuration of the standard gateway in the network
31800	MB_TCP:Port	TCP port (standard is 502)
21801	Modb:SlaveID	Own unit address as slave,
35800	MB_TCP:On	Modbus functions on/off switch, to be set

Parametrization example:

The Modbus function scope must be activated. The HEINZMANN control unit has the IP address 192.168.0.10 in the network. The gateway should have the IP address 192.168.0.1. It must be ensured that the settings require a control unit reset to activate the range of functions.

Number	Parameters	Value	Unit
21500	*IPAddress(0)*	*192*	
21501	*IPAddress(1)*	*168*	
21502	*IPAddress(2)*	*0*	
21503	*IPAddress(3)*	*10*	
21504	*SubnetMask(0)*	*255*	
21505	*SubnetMask(1)*	*255*	
21506	*SubnetMask(2)*	*255*	
21507	*SubnetMask(3)*	*0*	
21508	*StandardGateway(0)*	*192*	
21509	*StandardGateway(1)*	*168*	
21510	*StandardGateway(2)*	*0*	
21511	*StandardGateway(3)*	*1*	
31800	*MB_TCP:Port*	*502*	

The standard settings apply to the unit's as-delivered condition.

They can be adapted on a customer- or project-specific basis by agreement.

Number	Parameters	Value	Unit
21500	*IPAddress(0)*	*10*	
21501	*IPAddress(1)*	*1*	
21502	*IPAddress(2)*	*1*	
21503	*IPAddress(3)*	*10*	
21504	*SubnetMask(0)*	*255*	
21505	*SubnetMask(1)*	*255*	
21506	*SubnetMask(2)*	*255*	
21507	*SubnetMask(3)*	*0*	
21508	*StandardGateway(0)*	*0*	
21509	*StandardGateway(1)*	*0*	
21510	*StandardGateway(2)*	*0*	
21511	*StandardGateway(3)*	*0*	
31800	*MB_TCP:Port*	*502*	

4.1 Reading out data

Read access to one or more data words is used to transmit parameter or measured values of the control unit to the master unit on the Modbus. The function code 0x03 *"Read Holding Register"* is used for this purpose. See also section ↑*3.2 Supported function* codes.

In the curve parameters section, the data field 39200 … 39299 *MB_TCP:ParSet(x)* is available to the user. This data field is assigned the parameter numbers whose measured or parameter values are to be transferred.

It must be ensured that the numbers are entered consecutively and without a gap, starting with the index zero. From the first invalid parameter number, all following entries are ignored.

An invalid parameter number is defined as:

- Parameter number zero,
- a parameter number that is unknown to the control unit or
- a parameter number to which a parameter belongs, whose level is greater than the access level for the Modbus, which is limited to level four.

The permanently set access level of four for the Modbus enables all measured and display values as well as the application-specific parameters to be transferred.

The function code 0x03 can be used to address the data field and to read out the parameter or measured values. The start address, given in the output corresponds to the data field index, for which the data field is limited to 100 entries. The number of data transferred from the index results from the number of registers specified in the request message.

The maximum number of data words that can be read is displayed to the user in the measured value

23810 Modb:NoOfTxParams Number of valid parameter numbers in the data field

Parametrization example:

The speed, oil pressure and oil temperature should be read out via Modbus.

The following are used:

Number	Parameters	Value	Unit
39200	*MB_TCP:ParSet(0)*	*2000*	
39201	*MB_TCP:ParSet(1)*	*2905*	
39202	*MB_TCP:ParSet(2)*	*2909*	
39203	*MB_TCP:ParSet(3)*	*0*	
Display			
23810	*Modb:NoOfTxParams*	*3*	
2000	*Speed*	*1500.2*	*rpm*
2905	*OilPressure*	*3.15*	*bar*
2909	*OilTemp*	*-10.2*	*°C*
External value range			
2000	*Speed*	*0.0 ... 4000.0*	*rpm*
2905	*OilPressure*	*0.00… 10.00*	*bar*
2909	*OilTemp*	*-100.0 ... 1000.0*	*°C*

The data is transferred in the external value range. The value range of each individual parameter can be taken from the documentation on the HEINZMANN control units or read directly from the control unit using a HEINZMANN diagnostic tool. To correctly interpret the transferred data words, only the number of decimal places and possibly the negative range of values must be taken into account.

In the above example, for the speed the data word with the value 15002, which is calculated from $1500{,}2 \cdot 10$ for a decimal place, is transferred for the oil pressure with 315, which is calculated from $3{,}15 \cdot 100$ for two decimal places and for the oil temperature -102, calculated from $-10{,}2 \cdot 10$.

As the speed and the oil pressure can only accept positive values, only the number of decimal places corresponding to the tenth potential must be taken into account. If the oil temperature is already interpreted as a sign, only the tens of ten need to be evaluated for the decimal places.

If the oil temperature is not yet indicated, the value 65434 results in a data word with the value 65434 in the above example. If the external value range allows negative values, all values ≥ 32768 must be interpreted as negative values. The correct signed value is obtained as a difference in the data word compared to 65536. In the second step, the decimal places must then be recalculated.

If, in the example above, the enquiry exceeds a start address ≥ 2 or the number of data to be read in two data words, an exception response is sent.

4.2 Writing data

The function codes 0x06 *"Write Single Register"* (individual-write mode) and 0x10 *"Write Multiple Register"* (multiple-write mode) can be used to describe a specified data field. See also section ↑*3.2 Supported function* codes. The write access to one or more data words enables further processing of received data in the control unit as external input variables.

Measured values (Modbus functions)

23815	*Modb:ResetDevice*	Force reset of control unit
23816	*Modb:SaveParameter*	Save parameter values in the control unit

Measured values (Modbus binary values)

23820	*Modb:RxBinary*	Binary values 16 bit wide

Measured values (Modbus sensor values)

23821	Modb:RxSensor(0) up to	
23824	*Modb:RxSensor(3)*	Sensor values

The sensor values and binary values are freely configurable via the data field 39200 *MB_TCP:Paret(x)*. In addition to sensor and switching functions, parameters with an access level ≤ 4 can also be specified. The data field that is used to read out data (see ↑*4.1 Reading out* data) should be populated with the corresponding parameter numbers, the parameter values of which should also be specified.

The data field can be activated using the function code 0x06 or 0x10, and the parameter and measured values specific to Modbus can be set. The start address given in the output corresponds to the data field index, for which the data field is limited to 100 entries. The number of data entries that will be transferred from the index is determined by the number of registers that should be populated, which is given in the output. If an array is to be populated, all of the registered parameters in the given array must possess write access. If this requirement is not fulfilled, a corresponding exception code (0x04 Slave Device Failure) will be returned.

Parametrization example:

Once again, the speed, oil pressure and oil temperature should be read out via Modbus. It must be possible to specify the PID parameters for the speed governor 100 Gain, 101 Stability and 102 Derivative using Modbus. A control unit from the XIOS series is used.

Number	Parameters	Value	Unit
39200	*MB_TCP:ParSet (0)*	*2000*	
39201	*MB_TCP:ParSet (1)*	*2905*	
39202	*MB_TCP:ParSet (2)*	*2909*	
39203	*MB_TCP:ParSet (3)*	*100*	
39204	*MB_TCP:ParSet (4)*	*101*	
39205	*MB_TCP:ParSet (5)*	*102*	
39206	*MB_TCP:ParSet (6)*	*0*	

Activation			
25801	*MB_TCP:ExtendedOn*	*1*	
Display			
23810	*Modb:NoOfParams*	*6*	
2000	*Speed*	*1500.2*	*rpm*
2905	*OilPressure*	*3.15*	*bar*
2909	*OilTemp*	*-10.2*	*°C*

4.2.1 Timeout monitoring

If binary and/or sensor values are to be transferred to the HEINZMANN control unit in write mode, note that these values must always be sent again and again in cyclic sequence in order to detect failures or damage to the data transfer devices. For this purpose, the first write access starts timeout monitoring, the time limit of which can be set with the parameter 21820 *Modb:RxTimeOut*. Each valid receipt of a request with the function code 0x06 or 0x10 re-initialises the timeout monitoring. If the write access for the set receiving time limit remains off, the error flag 3074 *ErrModbusComm* is set.

In the event of an error in Modbus communication, the standard reactions from the HEINZMANN control units are applied to a sensor error for the sensor value. The binary values are assumed to be zero and the assigned switch functions are thus reset.

If it is only parameters (e.g. at the start of communication) that are to be specified in write mode, timeout monitoring is not strictly necessary. In this case, timeout monitoring can be deactivated by setting the time limit parameter to zero.

For further parameterization of the sensors and switch functions, refer to the chapters *Sensors* and *Inputs and Outputs* for the relevant basic information. Only the respective extensions of the special function via Modbus are discussed here.

4.2.2 Assigning inputs to the sensors

Sensor values are assigned to the sensors as inputs via Modbus by entering the required channel numbers in the associated parameters 900 *Assign…* and the following. This is the respective index plus one here. The selection of the sensor as the Modbus sensor value is achieved by specifying the channel type in parameters 4900 *ChanTyp…* and the following. For Modbus, the channel type must always be specified with 6.

49xx *ChanTyp…* = 6 Sensor value is received via Modbus

The transmission of a sensor value from the master to the HEINZMANN control unit as slave always takes place in the internal value range, which is defined as 0 to 65535.

The scaling of the sensor value from 0 to 100 % corresponds to a value of 0 … 65535 transferred via the Modbus. This results in 32767 for 50 % for the above example.

Parametrization example:

The power setpoint is received as a second sensor value via Modbus. The sensor value should be updated cyclically every 2 s. A control unit from the THESEUS series is used.

Number	Parameters	Value	Unit
900	*Assign_PowerSetpoint*	*2*	
980	*PowerSetpointLow*	*0*	*%*
981	*PowerSetpointHigh*	*100*	*%*
4900	*ChanType_PowerSetp*	*6*	
31820	*MB_TCP:RxTimeOut*	*2.5*	*s*
39200	*MB_TCP:ParSet(0)*	*23822*	
Activation			
25804	*Modb:ExtendedOn*	*1*	
Display			
2900	*PowerSetpoint*	*50*	*%*
23016	*ErrModbusTCP*	*0*	
33821	*MB_TCP:RxSensor(0)*	*0*	*%*
33822	*MB_TCP:RxSensor(1)*	*50*	*%*
33823	*MB_TCP:RxSensor(2)*	*0*	*%*
23824	*MB_TCP:RxSensor(3)*	*0*	*%*

4.2.3 Assignment of the binary values to the switch functions

The assignment parameters from parameter number 810 *Funct…* can only be used for the switch functions, whose status is to be changed using digital hardware inputs. The assignment parameters from parameter number 20810 *Comm…* are doubled for the extended function of the switch functions. These parameters can be used for switching functions that are to be via a communication type.

A binary value via Modbus can simply be assigned to a switching function by entering the bit number in the corresponding assignment parameter. For switching functions, the channel type for Modbus must also be specified with 6 for the sensor assignment.

248xx *ChanTyp…* = 6 Binary value is received via Modbus

If a switching function is received both via the hardware and via the selected communication, the two states are internally linked with OR. The prerequisite for this is that this switching function is configured correctly.

Parametrization example:

Set the binary input three via Modbus to run a fixed speed 1. The sensor value should be updated cyclically every 10 s. A control unit from the HELENOS series is used.

Number	Parameters	Value	Unit
20815	*CommSpeedFix1*	*3*	
21820	*Modb:RxTimeOut*	*11.0*	*s*
24815	*ChanType_SpeedFix1*	*6*	
39200	*MB_TCP :ParamSet(0)*	*23820*	
Activation			
25804	*Modb:ExtendedOn*	*1*	
Display			
2815	*SwitchSpeedFix1*	*1*	
23016	*ErrModbusTCP*	*0*	
33820	*MB_TCP:RxBinary(0)*	*04*	

4.2.4 Assignment of Modbus functions

It is possible to perform certain actions in the control unit. The parameter number must be entered into the data field to enable the relevant function to be carried out. The following Modbus functions are possible:

23815	*Modb:ResetDevice*	Force reset of control unit
23816	*Modb:SaveParameter*	Save parameter values in the control unit

Parametrization example:

The parameters must be stored in the control unit using Modbus.

Number	Parameters	Value	Unit
39200	*MB_TCP:ParamSet(0)*	*23816*	
Activation			
25804	*Modb:ExtendedOn*	*1*	
Display			
3851	*LastIdentifier*	*94*	
23816	*Modb:SaveParameter*	*1*	

4.3 Diagnostic counter

For the HEINZMANN control units, the following diagnostic counters are implemented according to the Modbus specification, which can also be read out using the function code 0x08 *"Diagnostics"*, see also section ↑*3.2 Supported function* codes:

33800	*MB_TCP:RxMsgCnt*	Number of valid received messages,
33801	*MB_TCP:TxMsgCnt*	Number of sent messages,
33802	*MB_TCP:ExceptCnt*	Number of sent exception messages,
33803	*MB_TCP:CommErrCnt*	Number of incorrectly received messages,

5 Parameter description

5.1 List 1: Parameters

31800	**MB_TCP:Port**		
	Level:	4	TCP port on the incoming connections are accepted (standard=502).
	Range:	1..65535	
	Page(s):	13	

5.2 List 2: Measured values

23016	**ErrModbusTCP**		
	Level:	1	Error indication for Modbus TCP communication
	Range:	0..0xFFFF Hex	
23815	**Modb:ResetDevice**		
	Level:	4	Modbus function: Force reset of the control unit
	Range:	0..1	
	Page(s):	16, 19	
23816	**Modb:SaveParameter**		
	Level:	4	Modbus function: Save parameter values in control unit
	Range:	0..1	
	Page(s):	16, 19	
23820	**Modb:RxBinary**		
	Level:	4	Binary values for write access
	Range:	0000..FFFF Hex	
	Page(s):	16	
23821 to 23824	**Modb:RxSensor(x)**		
	Level:	4	Sensor values for write access
	Range:	0..65535	
	Page(s):	16	
33800	**MB_TCP:RxMsgCnt**		
	Level:	4	Number of valid received messages
	Range:	0..65535	
	Page(s)	20	
33801	**MB_TCP:TxMsgCnt**		
	Level:	4	Number of sent messages
	Range:	0..65535	
	Page(s):	20	
33802	**MB_TCP:ExceptCnt**		
	Level:	4	Number of exceptions detected.
	Range:	0..65535	
	Page(s):	20	
33803	**MB_TCP:CommErrCnt**		
	Level:	4	Number of communication errors (e.g. MBAP headers)
	Range:	0..65535	
	Page(s):	20	
33810	**MB_TCP:NoOfParams**		
	Level:	4	Number of valid parameter numbers entered in the data field for read access and write access
	Range:	0..100	

5.3 List 3: Functions

35800	**MB_TCP:On**		
	Level:	4	Functions on/off switch Modbus TCP, change only effective after reset
	Range:	0..1	
	Page(s):	13	

5.4 List 4: Characteristics and maps

39200	**MB_TCP:ParSet(x)**		
to	Level:	4	Data field for read access and write access. The parameter and measured value numbers whose values are to be transferred or received on request must be entered
39299	Range:	0..29999	
	Page(s):	14	

6 List of figures

Fig. 1: Modbus transmission types 8
Fig. 2: Modbus TCP message 8

7 List of tables

Table 1: Client enquiry 9
Table 2: Server response 10
Table 3: Function code 11
Table 4: Sub-functions of the diagnostic function 12
Table 5: Exception code 12

8 Index

Access level 14, 16
Baud rate 13, 21
Binary value 16, 17, 18, 21
Channel type 17, 18
DcDesk 2000 13
Diagnostic function 11
Diagnostic tool 15
Exception code 12
Function code 11
  0x03 ..... 9, 14
  0x06 16
  0x08 20
  0x10 16
Interface 13
Master 11, 12, 17
Parameterisation 13
Read access 11, 14, 21, 22
Receiving time limit 17
Reset control unit 16, 19, 21
Save parameter values 16, 19, 21
Sensor value 16, 17, 21
Slave 17
Sub-function code 11
Switch function 18
Unit address 11
Write access 11, 16, 17, 21, 22