Your Global Automation Partner

TURCK

excom I/O System Integration via Modbus TCP

Integration Manual

Contents

Contents

Hans Turck GmbH & Co. KG | T +49 208 4952-0 | F +49 208 4952-264 | more@turck.com | www.turck.com

1 About This Manual

This manual describes the general integration of the GEN-N and GEN-3G excom Ethernet gateways via Modbus TCP. CODESYS is used as the development environment in this example.

Read this manual and the applicable documents carefully before the integration. This will prevent the risk of personal injury and damage to property. Keep this manual safe during the service life of the product. If the product is passed on, hand over this manual as well.

The manual describes the possibilities to integrate the excom system, configure the station, use the I/O data and the associated diagnostics. Other applications of the excom system are described in addition to the general integration:

- Description of the Modbus TCP/IP protocol
- Reading parameter bytes
- Reading diagnostic bytes

1.1 Target groups

These instructions are written for suitably qualified and trained personnel and must be read carefully by anyone entrusted with the mounting, commissioning, operation, maintenance, disassembly or disposal of the device.

When using the device in Ex circuits, the user must also have an additional knowledge of explosion protection (EN 60079-14 etc.).

1.2 Explanation of symbols used

The following symbols are used in these instructions:

	**DANGER** DANGER indicates a dangerous situation with high risk of death or severe injury if not avoided.
	**WARNING** WARNING indicates a dangerous situation with medium risk of death or severe injury if not avoided.
	**CAUTION** CAUTION indicates a dangerous situation of medium risk which may result in minor or moderate injury if not avoided.
	**NOTICE** NOTICE indicates a situation which may lead to property damage if not avoided.
	**NOTE** NOTE indicates tips, recommendations and useful information on specific actions and facts. The notes simplify your work and help you to avoid additional work.
▶	**CALL TO ACTION** This symbol denotes actions that the user must carry out.
⇨	**RESULTS OF ACTION** This symbol denotes relevant results of actions.

1.3 Other documents

Besides this document the following material can be found on the Internet at **www.turck.com**:

- excom manuals
- Data sheets
- GEN… Getting Started
- EU Declaration of Conformity
- Approvals

1.4 Feedback about these instructions

We make every effort to ensure that these instructions are as informative and as clear as possible. If you have any suggestions for improving the design or if some information is missing in the document, please send your suggestions to **techdoc@turck.com**.

2 Notes on the System

2.1 System identification

This manual applies to the following multiprotocol Ethernet gateways for excom:

- GEN-N
- GEN-3G

2.2 Manufacturer and service

Hans Turck GmbH & Co. KG
Witzlebenstraße 7
45472 Mülheim an der Ruhr
Germany

Turck supports you with your projects, from initial analysis to the commissioning of your application. The Turck product database contains software tools for programming, configuration or commissioning, data sheets and CAD files in numerous export formats. You can access the product database at the following address: **www.turck.de/products**

For further inquiries in Germany contact the Sales and Service Team on:

- Sales: +49 208 4952-380
- Technology: +49 208 4952-390

Outside Germany, please contact your local Turck representative.

3 For Your Safety

The product is designed according to state-of-the-art technology. However, residual risks still exist. Observe the following warnings and safety notices to prevent damage to persons and property. Turck accepts no liability for damage caused by failure to observe these warning and safety notices.

3.1 Intended use

These devices are designed solely for use in industrial areas.

The gateway must only be used in the excom I/O system with the appropriate module racks. The gateway forms the interface between the excom I/O system and the higher-level fieldbus system. The gateway supports the industrial Ethernet protocols PROFINET, EtherNet/IP and Modbus TCP.

A ring master enables gateways to be networked in a ring topology.

Any other use is not in accordance with the intended use. Turck accepts no liability for any resulting damage.

3.2 General safety notes

- The device may only be assembled, installed, operated, parameterized and maintained by professionally-trained personnel.
- The device may only be used in accordance with applicable national and international regulations, standards and laws.
- The device only meets the EMC requirements for industrial areas and is not suitable for use in residential areas.

3.3 Notes on Ex protection

- Only use the device in Ex areas when installed in the appropriate protective housing.
- Observe national and international regulations for explosion protection.
- When using the device in explosion-protection circuits, the user must have a working knowledge of explosion protection (EN 60079-14 etc.).
- Use the device only within the permissible operating and ambient conditions (see approval data and Ex approval specifications).
- Fit blank modules (BM1) on unused slots on the module rack.
- Cables and terminals with intrinsically safe circuits must be indicated – use light blue for color-coding. Separate cables and terminals from non-intrinsically safe circuits or isolate accordingly (EN 60079-14).
- Perform "Proof of intrinsic safety".
- Never connect equipment to intrinsically safe circuits if this equipment was previously used once in non-intrinsically safe circuits.

4 Modbus TCP – Basic Principles

NOTE
The following description of the Modbus protocol is laid down in detail in the Modbus Application Protocol Specification V1.1b of the Modbus IDA.

The Modbus protocol is an application protocol based on layer 7 of the OSI reference model. The Modbus protocol enables the establishment of client-server communication between nodes of different bus systems and networks. Modbus is accessed via system port 502 of the TCP/IP stack. Modbus is a request-response protocol and offers different services that are specified by function codes.

Modbus is subdivided into three subprotocols:

- Modbus PLUS
- Asynchronous, serial data transmission
- Modbus TCP/IP

Only Modbus TCP/IP is relevant for the excom system.

4.1 Modbus TCP protocol

The Modbus protocol defines a simple protocol data unit (PDU), that is independent of the communication levels below it. When mapping the Modbus protocol to different bus systems or networks, additional fields are added to the relevant application data unit (ADU).

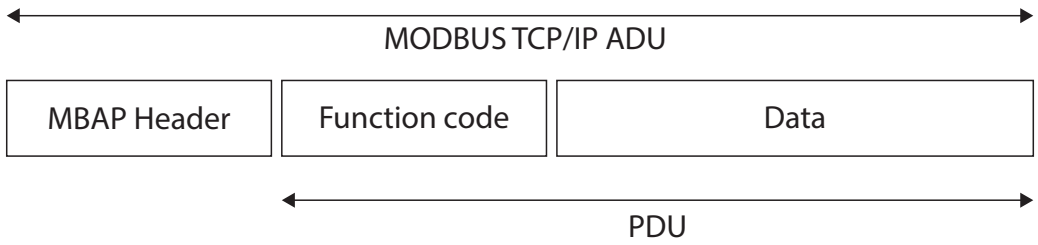

Fig. 1: Modbus TCP – Data units

Modbus ADU is built by the client that initiates the Modbus communication. The MBAP header (Modbus Application Header) is created by the client and enables the receiver (server) to unambiguously interpret the requested data. The MBAP header contains the following elements:

- Transaction number for assigning the request and response telegrams
- Protocol ID for identification of the protocol (Modbus = 0)
- Length specification
- Address byte

The Modbus function code indicates to the server which type of data access is to be executed. The Modbus application protocol (with excom system Modbus TCP/IP) defines the form of the client request. The information data field also contains information which is sent from a client to the server and which the server requires to process the command. This consists for example of bit or register addresses, the number of commands to be processed and the number of actual data bytes in the particular data field. With particular requests, the data field can also be non-existent (=0). In this case, the server does not require any additional information. The function code only defines the command to be executed. The response telegram contains the requested data if the server processes the client request without errors.

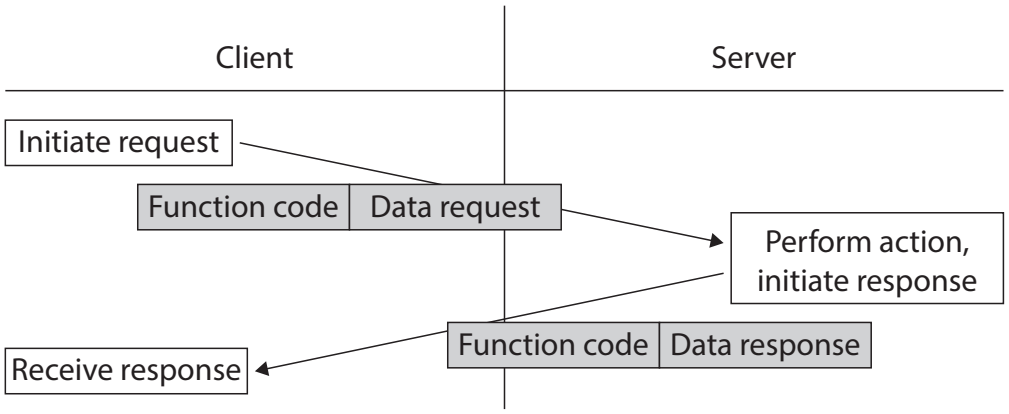

Fig. 2: Client-server communication – Request and response

If an error occurs in the data request, the data field of the response telegram contains an exception code that can be evaluated by the client according to the application.

4.2 Modbus data model and functions

The Modbus data model has four basic data types:

Data type	Object type	Access	Description
Discrete inputs	Bit	Read only	Data is provided by the I/O system.
Coils	Bit	Read/write	Data is changed and written by the application program. Data can also be read back.
Input register	16-bit (word)	Read only	Data is provided by the I/O system.
Holding registers	16-bit (word)	Read/write	Data is changed and written by an application program. Data can also be read back.

All data transferred via Modbus (bits and registers) must be stored in the application memory of the device. This data is accessed via defined access addresses [▶ 11].

The Modbus function code enables access to the four basic data types. The excom system supports the following functions for accessing process data, parameters, diagnostics and other services:

Function code	Function	Description
1	Read coils	Read multiple output bits
2	Read discrete inputs	Read multiple input bits
3	Read holding registers	Read multiple output registers
4	Read input registers	Read multiple input registers
5	Write single coil	Write single output bits
6	Write single registers	Write single output registers
15	Write multiple coils	Write multiple output bits
16	Write multiple registers	Write multiple output registers
23	Read/write multiple registers	Read and write multiple registers

4.3 Modbus register access

Multiple Modbus TCP connections are possible at the same time. Only the Exclusive owner connection can write and read data. The Exclusive owner connection is defined by the active fieldbus protocol after the power supply is switched on or via an appropriate entry in register 0x1140. Each I/O connection has complete read and write access to the configuration settings. Only the Exclusive owner connection has access to the I/O register.

After the Exclusive owner connection has timed out, the I/O data is set to the defined substitute value. Only read access is provided if Modbus is not the active fieldbus protocol. The following table explains the meaning of the registers:

Address	Access	Meaning
0x0000…0x01FF	Read only	Process data of the inputs (slot 0…24 is linked to form a data block)
0x0800…0x09FF	Read/write	Process data of the outputs (slot 0…24 are linked to form a data block)
0x1000…0x1006	Read only	Module ID
0x100C	Read only	Module status
0x1017	Read only	Register mapping revision (must always be 3 otherwise the register mapping is not compatible with this description)
0x1020	Read only	Watchdog, current time in ms
0x1120	Read/write	Watchdog, preset time in ms (default: 500 ms)
0x1130	Read/write	Modbus connection mode register
0x1131	Read/write	Modbus connection timeout in s (default: 300 s)
0x113C…0x113D	Read/write	Modbus parameter restore (reset the parameters to the default settings)
0x113E…0x113F	Read/write	Modbus parameter save (non-volatile saving of parameters)
0x1140	Read/write	Deactivate protocol Explicitly deactivates the selected Ethernet protocol: ■ Bit 0 = Deactivate EtherNet/IP ■ Bit 1 = Deactivate Modbus TCP ■ Bit 2 = Deactivate PROFINET ■ Bit 15 = Deactivate web server
0x1141	Read only	Active protocol ■ Bit 0 = EtherNet/IP active ■ Bit 1 = Modbus TCP active ■ Bit 2 = PROFINET active ■ Bit 15 = Web server active
0x8000…0x8320	Read only	Process data of the inputs (max. 25 modules per station × 32 registers per module)
0x9000…0x9320	Read/write	Process data of the outputs (max. 25 modules per station × 32 registers per module)
0xA000…0xA320	Read only	Diagnostics
0xB000…0xB320	Read/write	Parameter

4.3.1 Data width and assignment of the I/O modules in the Modbus register area

To enable efficient access to the process data of a station, the module data is combined and mapped in a contiguous register area. excom differentiates between digital and smart modules (analog modules and function modules).

The module types are mapped separately in successive register areas. The data bytes are structured in the physical order of the station setup from left to right. Each module is assigned the number of Modbus registers required by its data width, but at least one register. A DM80 module, for example, is assigned a contiguous register (2 byte) in the input and output area.

A Modbus register cannot contain the data of multiple modules. This forces Bit 0 of a digital module to be a word limit.

NOTE
Access to all 512 input and output registers is always possible irrespective of the station setup. Unused registers output "0".

Module	Input bytes	Output bytes	Alignment
Gateways			
GEN…	1	1	Bitwise
Digital inputs			
DM80…	1	1	Bitwise
DM80…S	2	1	Bitwise
DM80…8I	1		Bitwise
DM80…S 8I	2		Bitwise
DI40-N	1		Bitwise
DI401Ex	1		Bitwise
DI80-N	1		Bitwise
Digital outputs			
DO40-N		1	Bitwise
DO401Ex		1	Bitwise
DO60R-N		1	Bitwise
DO80-N		1	Bitwise
Analog inputs			
AI40-N	8		Wordwise
AI401Ex	8		Wordwise
AI41…	8		Wordwise
AI43…	8		Wordwise
AIH40…	8		Wordwise
AIH40… 1H	12		Wordwise
AIH40… 4H	24		Wordwise
AIH40… 8H	40		Wordwise
AIH41…	8		Wordwise
AIH41… 1H	12		Wordwise
AIH41… 4H	24		Wordwise
AIH41… 8H	40		Wordwise
AIH401…	8		Wordwise
AIH401… 1H	12		Wordwise

Module	Input bytes	Output bytes	Alignment
AIH401… 4H	24		Wordwise
AIH401… 8H	40		Wordwise
Analog outputs			
AO40-N		8	Wordwise
AO401Ex		8	Wordwise
AOH40…		8	Wordwise
AOH40… 1H	4	8	Wordwise
AOH40… 4H	16	8	Wordwise
AOH40… 8H	32	8	Wordwise
AOH401…		8	Wordwise
AOH401… 1H	4	8	Wordwise
AOH401… 4H	16	8	Wordwise
AOH401… 8H	32	8	Wordwise
Function modules			
TI40…	8		Wordwise
TI41…	8		Wordwise
DF20 F/P	8	2	Wordwise

4.3.2 Register 0x100C: Module status

This register contains a general gateway station status.

Gateway status word															
Byte 1								Byte 0							
res	FM	Par	MB Wdg	I/O cfg	I/O com	res	res	res	res	res	res	I/O cfg w	FS	res	I/O diag

Meaning of the gateway status bits

Designation	Meaning
res	Reserved
FM	Force mode in the DTM active
Par	I/O parameter error
MB Wdg	Modbus watchdog error
I/O cfg	Configuration error (missing or incorrectly fitted modules)
I/O cfg w	Warning: I/O configuration was changed
I/O com	No communication with the I/O module bus
FS	Warning – Fail-safe mode active
I/O diag	I/O diagnostics active

4.3.3 Register 0x1130: Modbus connection mode

This register affects the behavior of the Modbus connections.

Bit	Description
15…2	Reserved
1	**MB_ImmediateWritePermission**
	0: The write permission for the corresponding Modbus connection is requested with the first write access. In the event of an error, an exception response is generated with exception code 0x01. If successful, the write access is executed and the write permission is retained up to the end of the connection. **1**: The write permission for the corresponding Modbus connection is requested on connection establishment. The first Modbus connection then receives the write permission, all subsequent permissions are ignored (Bit 0 = 1).
0	**MB_OnlyOneWritePermission**
	0: All Modbus connections have write permissions. **1**: Only one Modbus connection can be assigned the write permission. A write permission once assigned is retained until the disconnect. After the connection with write permission is disconnected, the next connection that requested the write permission receives it.

4.3.4 Register 0x1131: Modbus connection timeout

This register defines the period of inactivity after which a Modbus connection is closed by a disconnect.

4.3.5 Registers 0x113C and 0x113D: Modbus parameter restore

Registers 0x113C and 0x113D are used to reset the parameter registers 0x1120 and 0x1130…0x113B to the default settings.

- ▶ Write 0x6C6F to register 0x113C.
- ▶ Write 0x6164 to register 0x113D within 30 s.
- ⇨ The registers are reset.

The FC16 and FC23 functions enable both registers to also be written with a single request. The functions restore the parameters without saving them. The Modbus Save parameter enables the entry to be saved.

4.3.6 Registers 0x113E and 0x113F: Modbus Save parameter

The registers 0x113E and 0x113F are used for the non-volatile saving of parameters in registers 0x1120 and 0x1130…0x113B.

- ▶ Write 0x7361 to register 0x113E.
- ▶ Write 0x7665 to register 0x113F within 30 s.
- ⇨ The parameters are saved.

The FC16 and FC23 functions enable both registers to also be written with a single request.

4.3.7 Registers 0x8000…0x8320: Process data inputs

Registers 0x8000…0x8320 map the process data of the inputs according to slot and are identical to the packed process data of the inputs from registers 0x0000…0x01FF. The gateways (slot 0) and the maximum of 24 I/O modules per station produce the following register mapping:

Register	Module	Slot
0x8000	Gateway	Slot 0
0x8020	I/O module	Slot 1
0x8040	I/O module	Slot 2
0x8…	I/O module	Slot n
0x8320	I/O module	Slot 24

4.3.8 Registers 0x9000…0x9320: Process data outputs

Registers 0x9000…0x9320 map the process data of the outputs according to slot and are identical to the packed process data of the outputs from registers 0x0800…0x09FF. The gateways (slot 0) and the maximum of 24 I/O modules per station produce the following register mapping:

Register	Module	Slot
0x9000	Gateway	Slot 0
0x9020	I/O module	Slot 1
0x9040	I/O module	Slot 2
0x9…	I/O module	Slot n
0x9320	I/O module	Slot 24

4.3.9 Registers 0xA000…0xA320: Diagnostics

Registers 0xA000…0xA320 map the diagnostics messages of the excom station according to slot. The gateways (slot 0) and the maximum of 24 I/O modules per station produce the following register mapping:

Register	Module	Slot
0xA000	Gateway	Slot 0
0xA020	I/O module	Slot 1
0xA040	I/O module	Slot 2
0xA…	I/O module	Slot n
0xA320	I/O module	Slot 24

Gateway diagnostics

The diagnostics data produces the gateway diagnostics in registers 0xA000…0xA002. The diagnostics data contain:

- Gateway status
- Redundancy status
- CAN status

Byte	0…1	2…3	4…5
DIAG DATA	Gateway	Redundancy	CAN

The diagnostics data of the gateway on Byte 0 and 1 (0xA000) corresponds to the reverse byte order of the gateway status word and is formed as follows:

Gateway status word															
Byte 1								Byte 0							
res	res	res	res	I/O cfg w	FS	res	I/O diag	res	FM	Par	MB Wdg	I/O cfg	I/O com	res	res

Meaning of the gateway status bits

Designation	Meaning
res	Reserved
FM	Force mode in the DTM active
Par	I/O parameter error
MB Wdg	Modbus watchdog error
I/O cfg	Configuration error (missing or incorrectly fitted modules)
I/O cfg w	Warning: I/O configuration was changed
I/O com	No communication with the I/O module bus
FS	Warning – Fail-safe mode active
I/O diag	I/O diagnostics active

The redundancy status can be read from Byte 2 and 3 (0xA001):

Function	Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Switchover		Reserved															1
Gateway missing																1	
Not ready															1		
Gateway error														1			
No fieldbus connection													1				
Different configuration												1					
Different firmware											1						

The status of the CAN bus can be read from Byte 4 and 5 (0xA002):

Function	Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CAN redundancy error		Reserved															1
CAN Error on both lines																1	

Hans Turck GmbH & Co. KG | T +49 208 4952-0 | F +49 208 4952-264 | more@turck.com | www.turck.com

I/O module diagnostics

The first two bytes for each I/O module are reserved and are output as zero. The channel-specific diagnostics start from Byte 3 of each I/O module (e.g. 0xA021 for channels 1 and 2 of the first I/O module fitted):

Byte per module n	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**
Diagnostics	res	res	CH1	CH2	CH3	CH4	CH5	CH6	CH7	CH8	res

Bit assignments for the diagnostics messages of module types DM80, DF20, DO…(apart from DO80…) and DI… (apart from DI80…):

Function	**Bit**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Wire break								1	
Short circuit							1		

Bit assignments for the diagnostics messages of module types DI80 and DO80:

Function	**Bit**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Channel faulty									1
Wire break								1	
Short circuit							1		
External power supply missing						1			
Overvoltage					1				
Undervoltage				1					
Short circuit Field device voltage			1						
Bounce error		1							

Bit assignments for the diagnostics messages of module types AO4…, AI4… and TI4…:

Function	**Bit**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Wire break								1	
Short circuit							1		
Overflow						1			
Underflow					1				

Bit assignments for the diagnostics messages of module types AOH4… and AIH4…:

Function	**Bit**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Channel faulty									1
Wire break								1	
Short circuit							1		
Overflow						1			
Underflow					1				
HART status error				1					
HART communication error			1						
Channel error		1							

5 Integrating excom in Modbus TCP: Application example

The following example shows the integration of excom with CODESYS, the handling of the I/O data and the diagnostics.

Hardware used

This example uses the following hardware components:

- MT08-3G
- PSM24-3G
- DM80Ex
- DO401Ex
- AIH401Ex
- AOH401Ex
- Ethernet cable with RJ45 connector
- PC

Software used

This example uses the following software:

- Gateway firmware V1.4.0.0
- CODESYS version 3.5.12 (download free of charge from **www.turck.com**)

Requirements

- The programming software has been opened.
- A new project has been created.
- CODESYS PLC has been started via CODESYS Control Win SysTray.
- The excom station has been configured via the DTM (see GEN… Getting Started).

5.1 Connecting the device with the controller

To connect the device to the controller, the following components must be added in CODESYS first of all:

- Ethernet adapter
- Modbus TCP master
- Modbus TCP slave

Scanning the PLC

- ▶ Double-click **Device**.
- ▶ Click **Scan network…**
- ▶ Select the interface and confirm with **OK**.

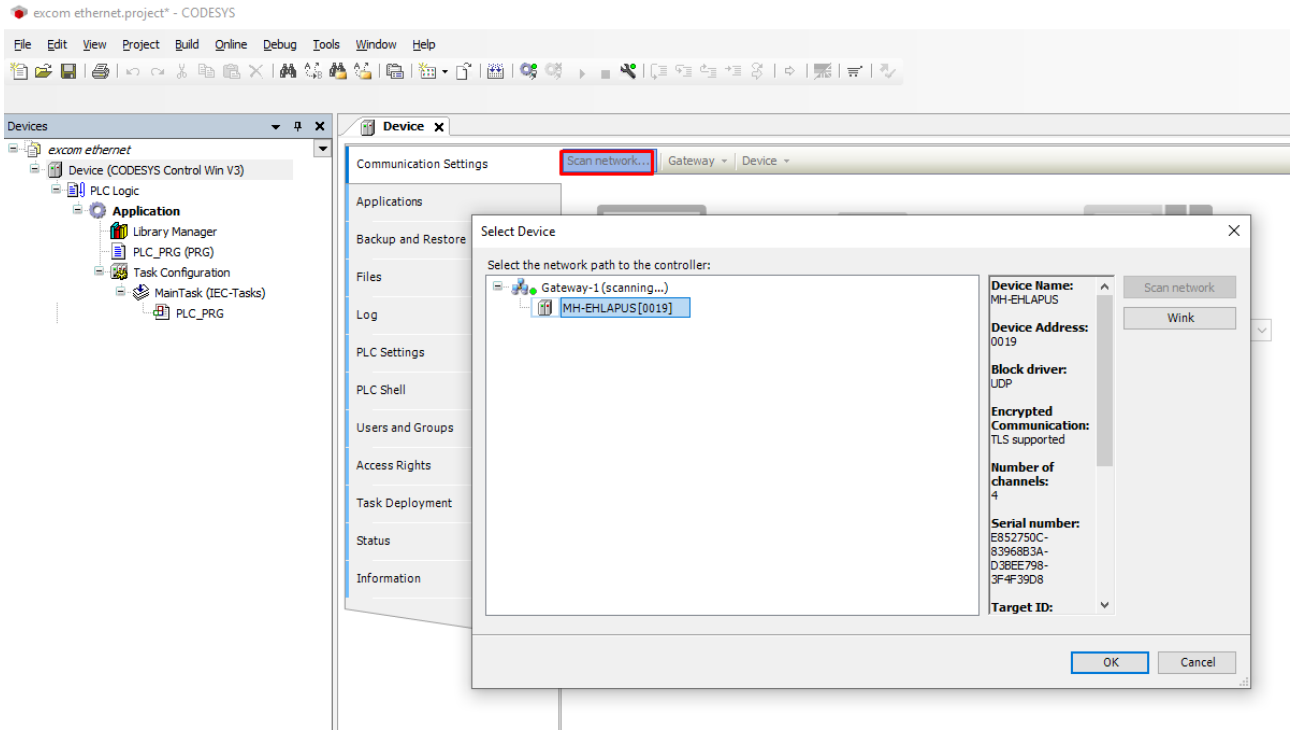

Fig. 3: Scanning the network

Adding an Ethernet port

- ▶ Right-click **Device** in the project tree.
- ▶ Select **Add device**
- ▶ Select the Ethernet port.
- ▶ Click **Add device**.
- ⇨ The Ethernet port appears as **Ethernet** in the project tree.

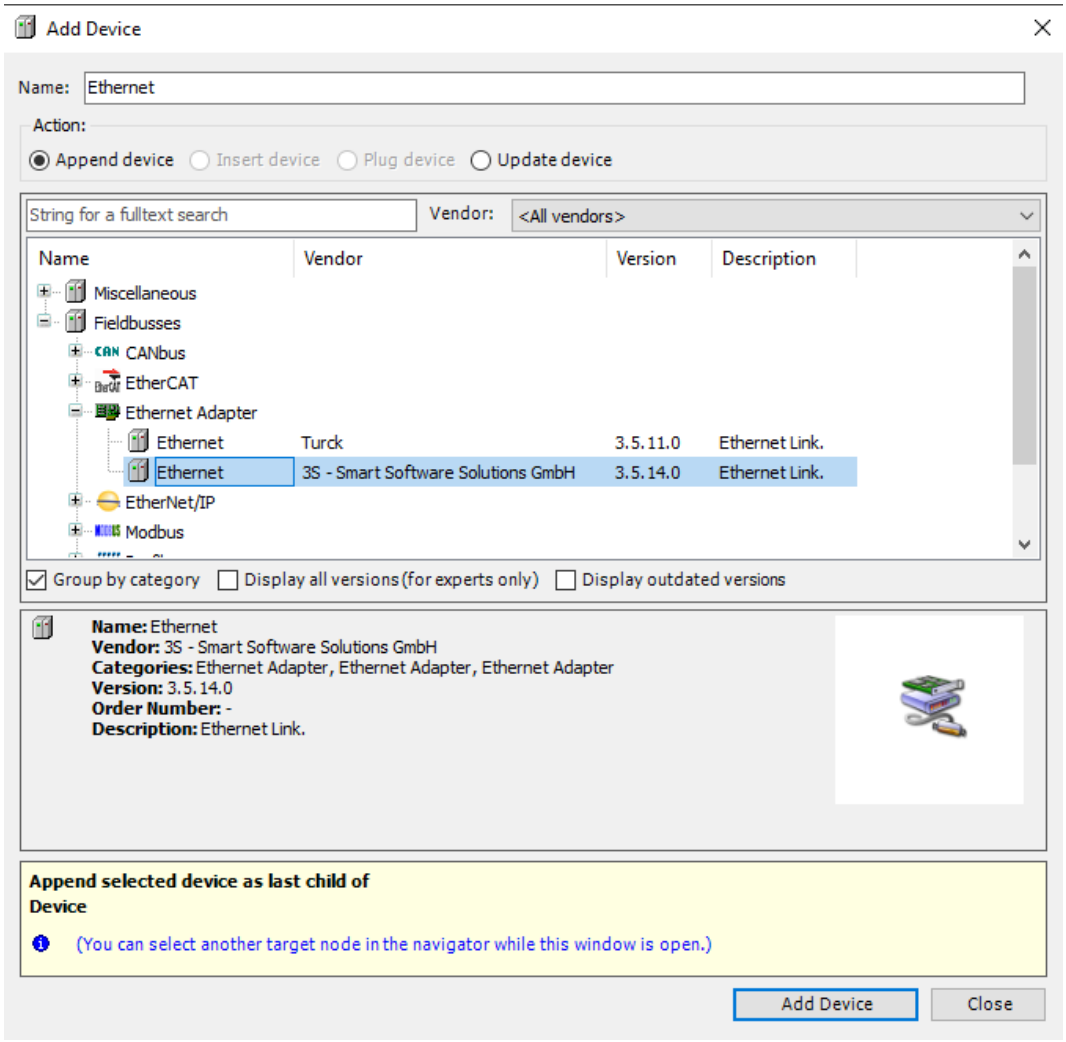

Fig. 4: Adding an Ethernet port

Setting an IP address

- ▶ Double-click Ethernet adapter (here: **Ethernet (Ethernet)**).
- ▶ Set the IP address (here: 192.168.1.1).

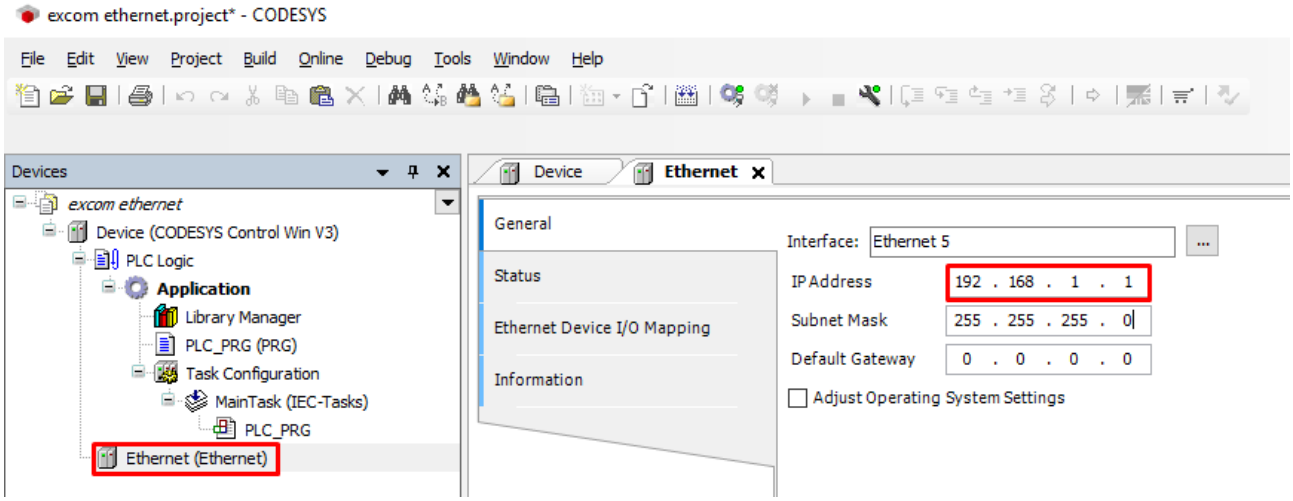

Fig. 5: Setting an IP address

Adding a Modbus master

- ▶ Right-click **Ethernet** in the project tree.
- ▶ Select **Add device**
- ▶ Double-click **Modbus TCP Master**.
- ⇨ The Modbus TCP master appears as **Modbus_TCP_Master** in the project tree.

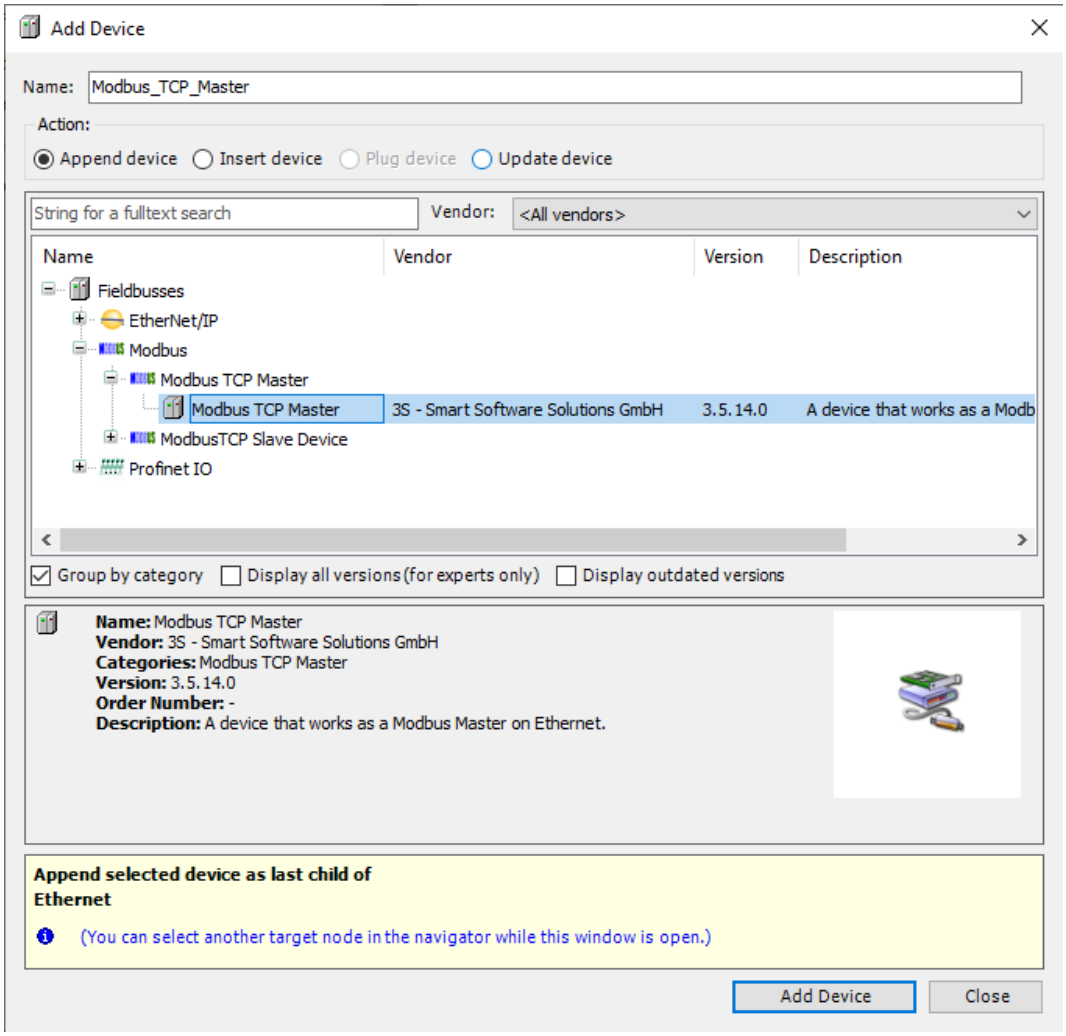

Fig. 6: Adding a Modbus master

Adding a Modbus slave

- ▶ Right-click **Modbus TCP Master** in the project tree.
- ▶ Select **Add device**
- ▶ Double-click **Modbus TCP Slave**.
- ⇨ The Modbus slave appears as **Modbus_TCP_Slave** in the project tree.

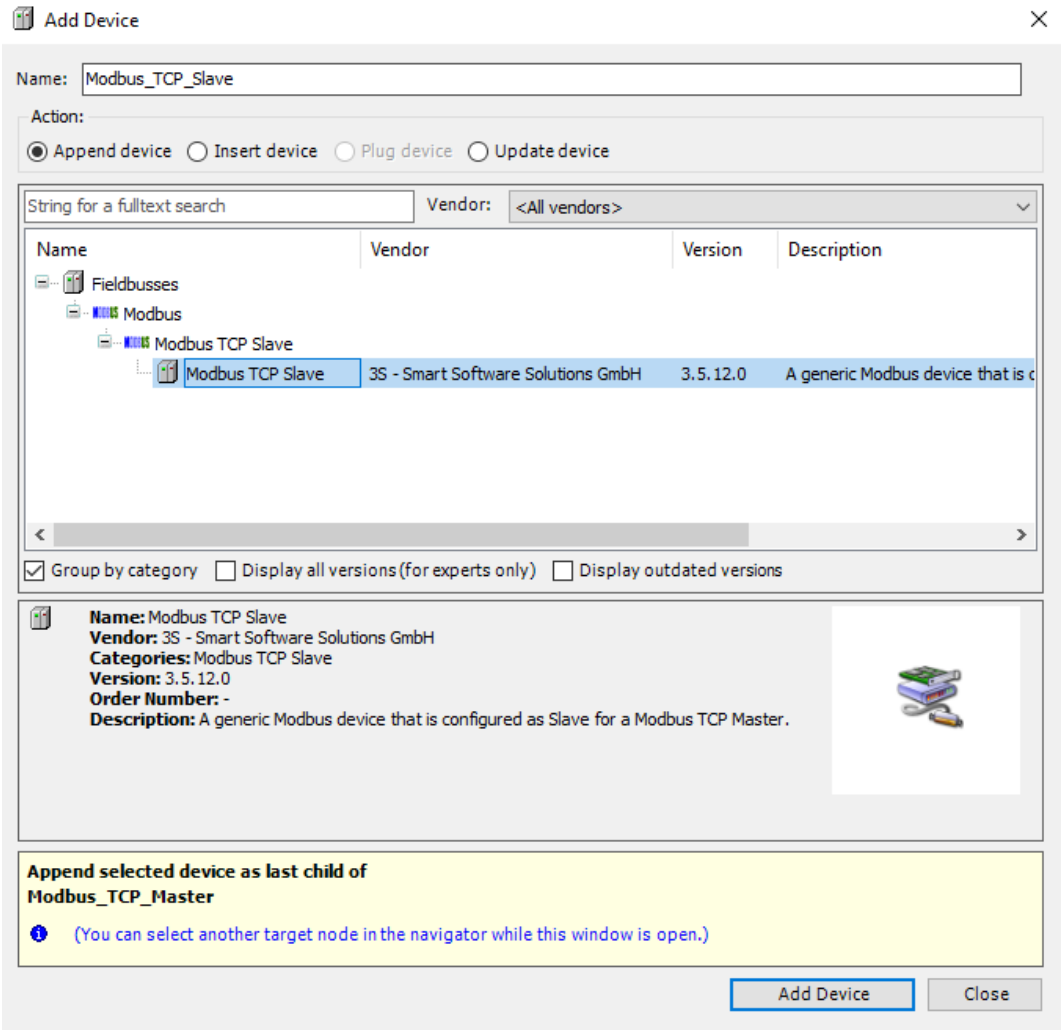

Fig. 7: Adding a Modbus slave

Renaming a Modbus slave

- Click Modbus slave in the project tree.
- Press [F2].
- Adjust the name of the slave in the project tree of the application.

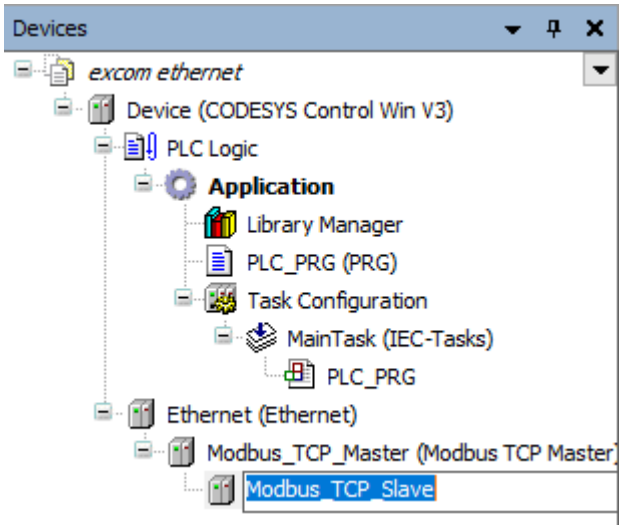

Fig. 8: Renaming a Modbus slave

5.2 Setting up an excom station in Modbus TCP

Setting an IP address

- Double-click the Modbus slave.
- Set the IP address.

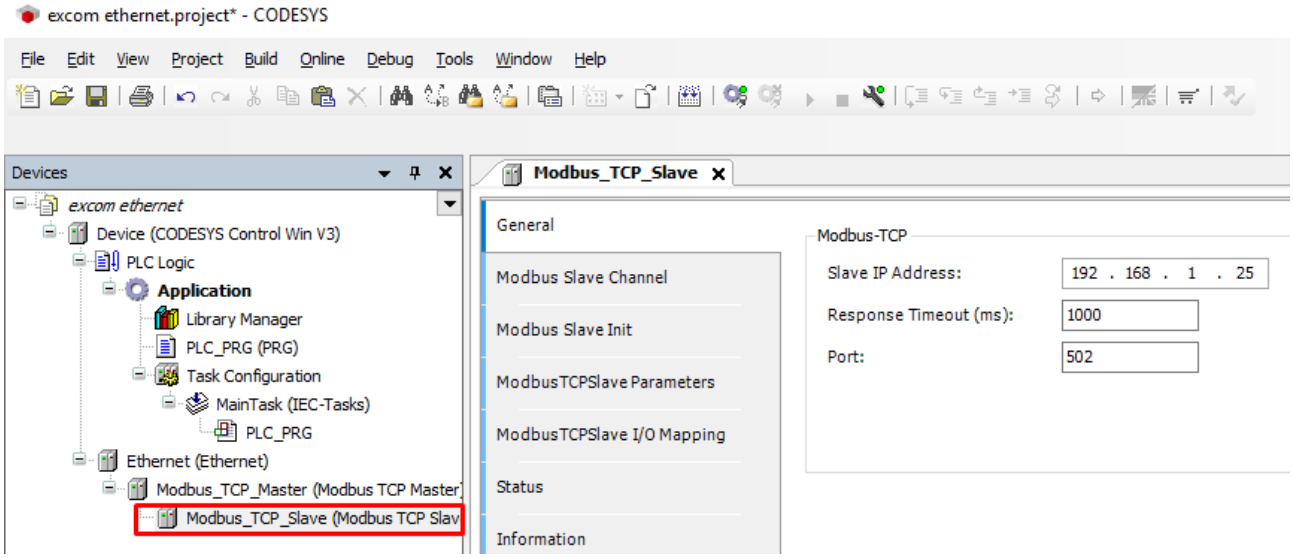

Fig. 9: Setting the Modbus slave IP address

The communication between Modbus TCP master and Modbus slaves is performed using defined function calls (Modbus communication channels). These channels are set up with the relevant Modbus slaves in the **Modbus Slave Channel** tab via the **Add channel** button.

The Modbus communication channels are defined by:

Access type	Modbus function code that defines the type and method of function calls (bitwise, word-wise, read or write)
READ registers and WRITE registers Offset	Start address of the registers of the Modbus slave registers to be read or written. Refer to the Modbus mapping of the excom system for the relevant information (see web server or manual).

The mapping of the data depends on the structure and parameters of the excom station. The data mapping can be derived from the Modbus registers [▶ 11] or the data width of the configured modules [▶ 12] or viewed on the web server under **Modbus TCP Memory Map**.

The example structure used produces the following data mapping:

NOTE
The AIH401 and AOH401 I/O modules were configured in the 4H variant.

Slot 0 - Station (Input Data Mapping)			
Description	**Register**	**Bit Offset**	**Bit Length**
Gateway active	0x0000 (0)	0	1
Gateway slot left	0x0000 (0)	1	1
Gateway redundancy available	0x0000 (0)	2	1
Right supply module available	0x0000 (0)	3	1
Left supply module available	0x0000 (0)	4	1
Slot 1 - DM80 S (Input Data Mapping)			
Description	**Register**	**Bit Offset**	**Bit Length**
Channel 1 - Value	0x0001 (1)	0	1
Channel 2 - Value	0x0001 (1)	1	1
Channel 3 - Value	0x0001 (1)	2	1
Channel 4 - Value	0x0001 (1)	3	1
Channel 5 - Value	0x0001 (1)	4	1
Channel 6 - Value	0x0001 (1)	5	1
Channel 7 - Value	0x0001 (1)	6	1
Channel 8 - Value	0x0001 (1)	7	1
Channel 1 - Status	0x0001 (1)	8	1
Channel 2 - Status	0x0001 (1)	9	1
Channel 3 - Status	0x0001 (1)	10	1
Channel 4 - Status	0x0001 (1)	11	1
Channel 5 - Status	0x0001 (1)	12	1
Channel 6 - Status	0x0001 (1)	13	1
Channel 7 - Status	0x0001 (1)	14	1
Channel 8 - Status	0x0001 (1)	15	1

Fig. 10: Input data mapping – Slot 0 and slot 1

Slot 3 - AIH40. 4H (Input Data Mapping)			
Description	**Register**	**Bit Offset**	**Bit Length**
Channel 1 - Value	0x0002 (2)	0	16
Channel 2 - Value	0x0003 (3)	0	16
Channel 3 - Value	0x0004 (4)	0	16
Channel 4 - Value	0x0005 (5)	0	16
HART variable 1	0x0006 (6)	0	32
HART variable 2	0x0008 (8)	0	32
HART variable 3	0x000a (10)	0	32
HART variable 4	0x000c (12)	0	32
Slot 4 - AOH40. 4H (Input Data Mapping)			
Description	**Register**	**Bit Offset**	**Bit Length**
HART variable 1	0x000e (14)	0	32
HART variable 2	0x0010 (16)	0	32
HART variable 3	0x0012 (18)	0	32
HART variable 4	0x0014 (20)	0	32

Fig. 11: Input data mapping – Slot 3 and slot 4

Gateway Status Word (Input Data Mapping)			
Description	**Register**	**Bit Offset**	**Bit Length**
Module Diagnostics Available	0x0016 (22)	0	1
Modulebus Failsafe Mode Enabled	0x0016 (22)	2	1
Station Configuration Changed	0x0016 (22)	3	1
Overcurrent Isys	0x0016 (22)	5	1
Overvoltage Field Supply UI	0x0016 (22)	6	1
Undervoltage Field Supply UI	0x0016 (22)	7	1
Overvoltage Field Supply Usys	0x0016 (22)	8	1
Undervoltage Field Supply Usys	0x0016 (22)	9	1
Modulebus Communication Lost	0x0016 (22)	10	1
Modulebus Configuration Error	0x0016 (22)	11	1
Modulebus Status Error	0x0016 (22)	12	1
Modulebus Parameter Error	0x0016 (22)	13	1
Force Mode Enabled	0x0016 (22)	14	1

Fig. 12: Input data mapping – Gateway status word

Hans Turck GmbH & Co. KG | T +49 208 4952-0 | F +49 208 4952-264 | more@turck.com | www.turck.com

Slot 0 - Station (Output Data Mapping)			
Description	**Register**	**Bit Offset**	**Bit Length**
Red switching	0x0800 (2048)	0	2
Slot 1 - DM80 (Output Data Mapping)			
Description	**Register**	**Bit Offset**	**Bit Length**
Channel 1 - Value	0x0801 (2049)	0	1
Channel 2 - Value	0x0801 (2049)	1	1
Channel 3 - Value	0x0801 (2049)	2	1
Channel 4 - Value	0x0801 (2049)	3	1
Channel 5 - Value	0x0801 (2049)	4	1
Channel 6 - Value	0x0801 (2049)	5	1
Channel 7 - Value	0x0801 (2049)	6	1
Channel 8 - Value	0x0801 (2049)	7	1
Slot 2 - DO40. (Output Data Mapping)			
Description	**Register**	**Bit Offset**	**Bit Length**
Channel 1 - Value	0x0802 (2050)	0	1
Channel 2 - Value	0x0802 (2050)	1	1
Channel 3 - Value	0x0802 (2050)	2	1
Channel 4 - Value	0x0802 (2050)	3	1
Slot 4 - AOH40. 4H (Output Data Mapping)			
Description	**Register**	**Bit Offset**	**Bit Length**
Channel 1 - Value	0x0803 (2051)	0	16
Channel 2 - Value	0x0804 (2052)	0	16
Channel 3 - Value	0x0805 (2053)	0	16
Channel 4 - Value	0x0806 (2054)	0	16

Fig. 13: Output data mapping – Slot 0...4

Creating a new Modbus channel in CODESYS

- At **Modbus Slave Channel** click **Add channel…**

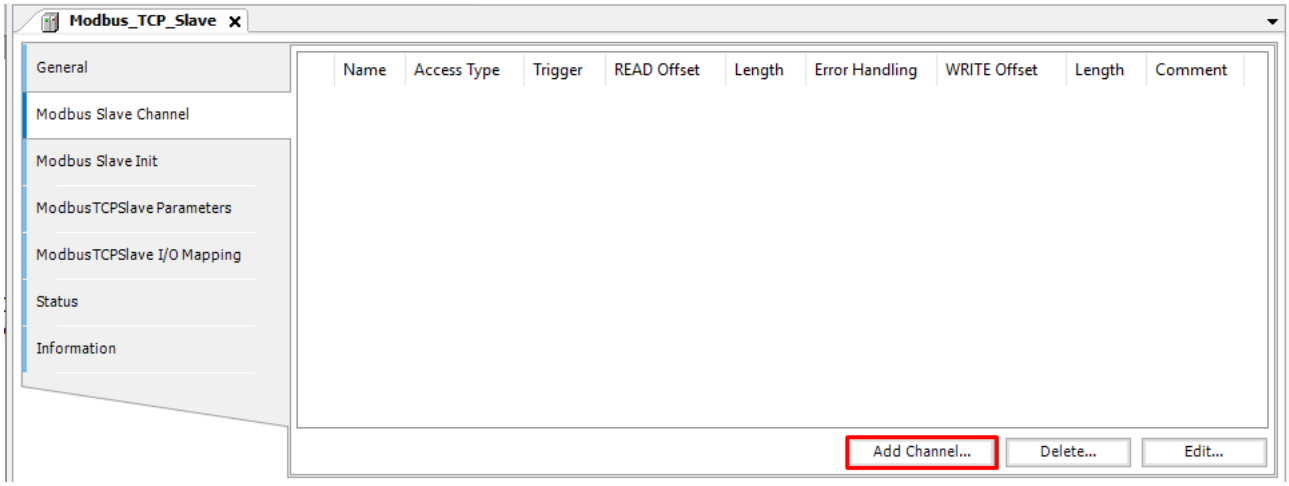

Fig. 14: Adding a Modbus channel

Creating gateway process data bits

- Create a Modbus channel with the access type **Read Holding Registers**.
- Set **READ register 0x0000** with a length of **1**.
- Confirm with **OK**.

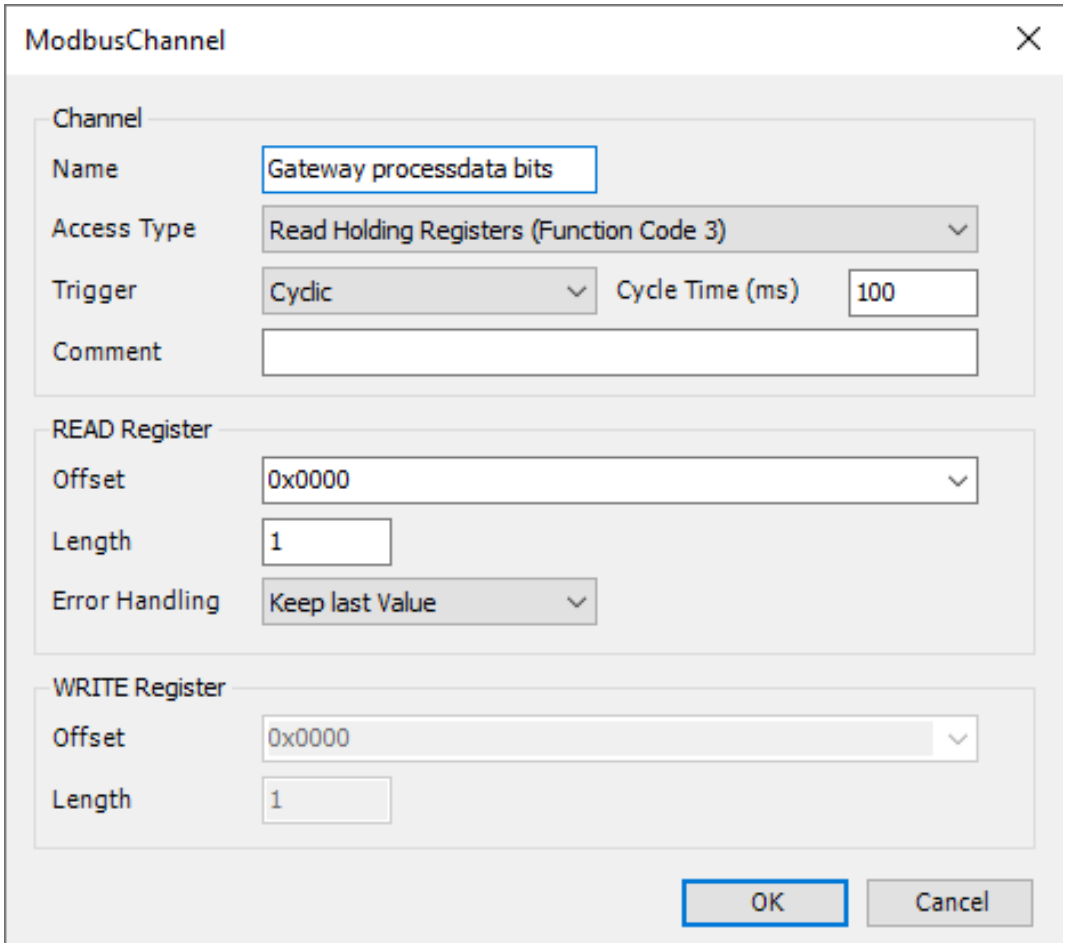

Fig. 15: Modbus channel – Creating a gateway process data bit

▶ Read the register at **ModbusTCPSlave\I/O image**.

In the figure the left gateway is fitted and there is no gateway redundancy function. A redundant power supply is provided.

Variable	Mapping	Channel	Address	Type	Current Value	Prepared Value	Unit	Description
		Gateway-Prozessdatenbits	%IW0	ARRAY [0..0] OF WORD				Read Holding Registers
		Gateway-Prozessdatenbits[0]	%IW0	WORD	27			0x0000
		Bit0	%IX0.0	BOOL	TRUE			
		Bit1	%IX0.1	BOOL	TRUE			
		Bit2	%IX0.2	BOOL	FALSE			
		Bit3	%IX0.3	BOOL	TRUE			
		Bit4	%IX0.4	BOOL	TRUE			
		Bit5	%IX0.5	BOOL	FALSE			
		Bit6	%IX0.6	BOOL	FALSE			
		Bit7	%IX0.7	BOOL	FALSE			
		Bit8	%IX1.0	BOOL	FALSE			
		Bit9	%IX1.1	BOOL	FALSE			
		Bit10	%IX1.2	BOOL	FALSE			
		Bit11	%IX1.3	BOOL	FALSE			
		Bit12	%IX1.4	BOOL	FALSE			
		Bit13	%IX1.5	BOOL	FALSE			
		Bit14	%IX1.6	BOOL	FALSE			
		Bit15	%IX1.7	BOOL	FALSE			

Fig. 16: Modbus channel – Reading a gateway process data bit

DM80 – Reading channels

The process data of the I/O module can be read via the register **0x0001**.

- Create a Modbus channel with the access type **Read Holding Registers**.
- Set **READ register 0x0001** with a length of **1**.
- Confirm with **OK**.

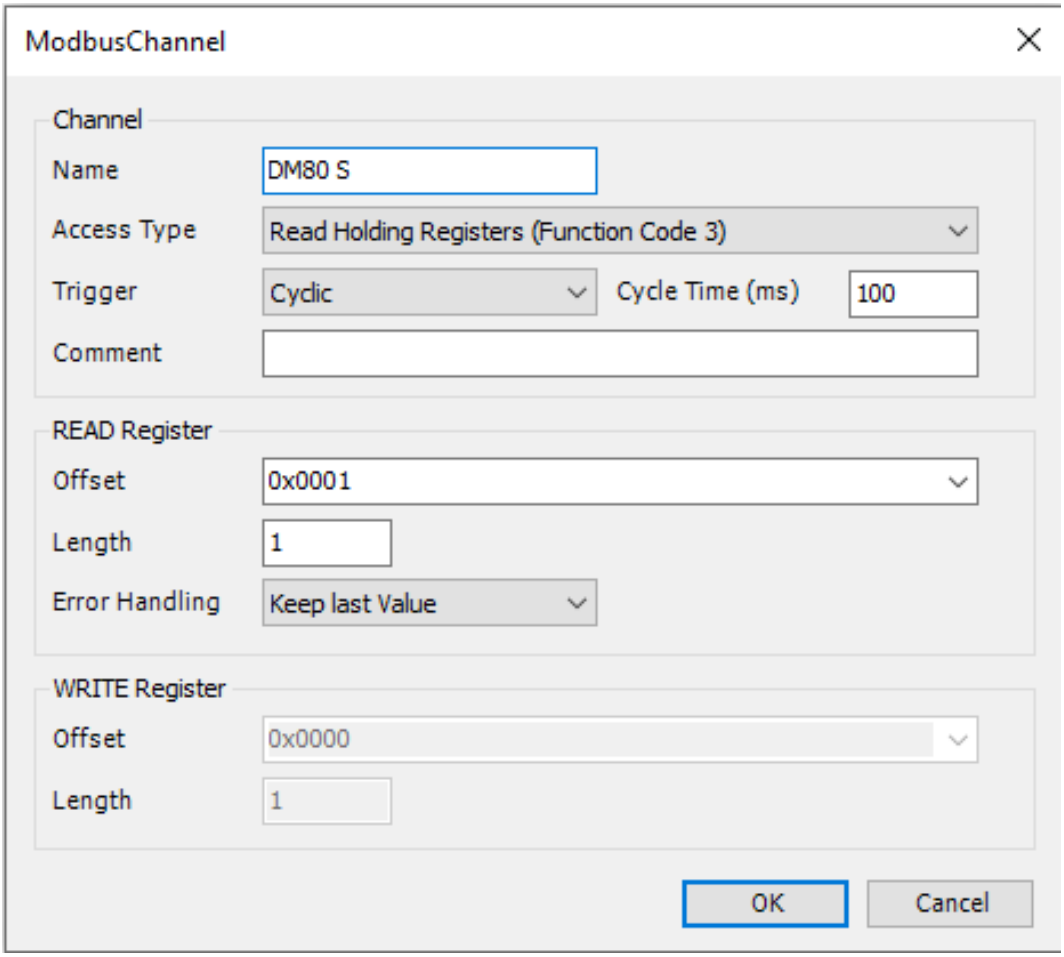

Fig. 17: DM80 – Creating a Modbus channel

- Read the register at **ModbusTCPSlaveI/O image**.

Eight sensors are connected to the DM80. On channels 5…8 the sensors detect an object and a status message is present on channels 1…4.

	DM80 S	%IW1	ARRAY [0..0] OF WORD				Read Holding Registers
	DM80 S[0]	%IW1	WORD	4080			0x0001
	Bit0	%IX2.0	BOOL	FALSE			
	Bit1	%IX2.1	BOOL	FALSE			
	Bit2	%IX2.2	BOOL	FALSE			
	Bit3	%IX2.3	BOOL	FALSE			
	Bit4	%IX2.4	BOOL	TRUE			
	Bit5	%IX2.5	BOOL	TRUE			
	Bit6	%IX2.6	BOOL	TRUE			
	Bit7	%IX2.7	BOOL	TRUE			
	Bit8	%IX3.0	BOOL	TRUE			
	Bit9	%IX3.1	BOOL	TRUE			
	Bit10	%IX3.2	BOOL	TRUE			
	Bit11	%IX3.3	BOOL	TRUE			
	Bit12	%IX3.4	BOOL	FALSE			
	Bit13	%IX3.5	BOOL	FALSE			
	Bit14	%IX3.6	BOOL	FALSE			
	Bit15	%IX3.7	BOOL	FALSE			

Fig. 18: DM80 – Creating a Modbus channel

AIH401 – Reading channels

The process data of the I/O module can be read via the register **0x0002**.

- Create a Modbus channel with the access type **Read Holding Registers**.
- Set **READ register 0x0002** with a length of **4**.
- Confirm with **OK**.

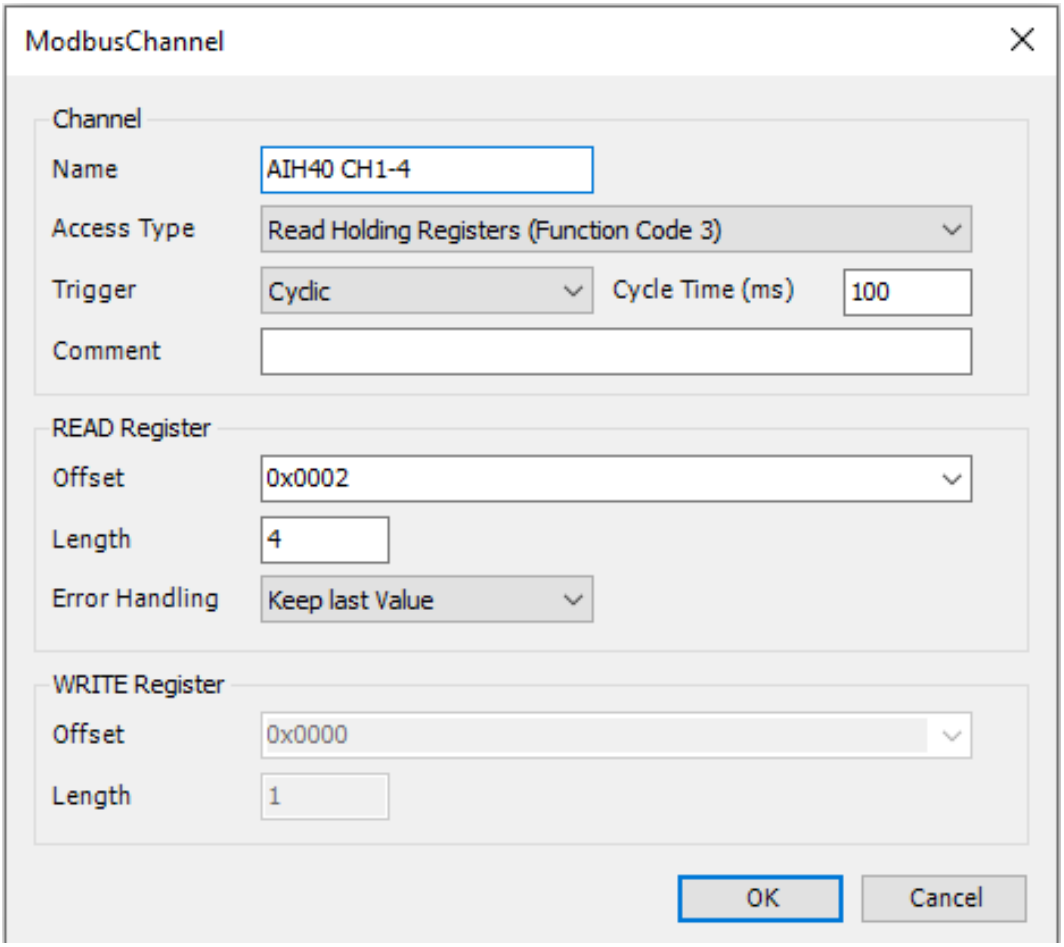

Fig. 19: AIH401 – Creating a Modbus channel

- Read the register at **ModbusTCPSlave\I/O image**.

		AIH401 CH1-4	%IW2	ARRAY [0..3] OF WORD		Read Holding Registers
		AIH401 CH1-4[0]	%IW2	WORD	7608	0x0002
		AIH401 CH1-4[1]	%IW3	WORD	36368	0x0003
		AIH401 CH1-4[2]	%IW4	WORD	36368	0x0004
		AIH401 CH1-4[3]	%IW5	WORD	36368	0x0005

Fig. 20: AIH401 – Creating a Modbus channel

Reading HART values

Each HART signal consists of 32 bytes. **READ registers** with a length of **2** must be read.

- Create a Modbus channel with the access type **Read Holding Registers**.
- Set **READ register 0x0006** with a length of **2**.
- Confirm with **OK**.

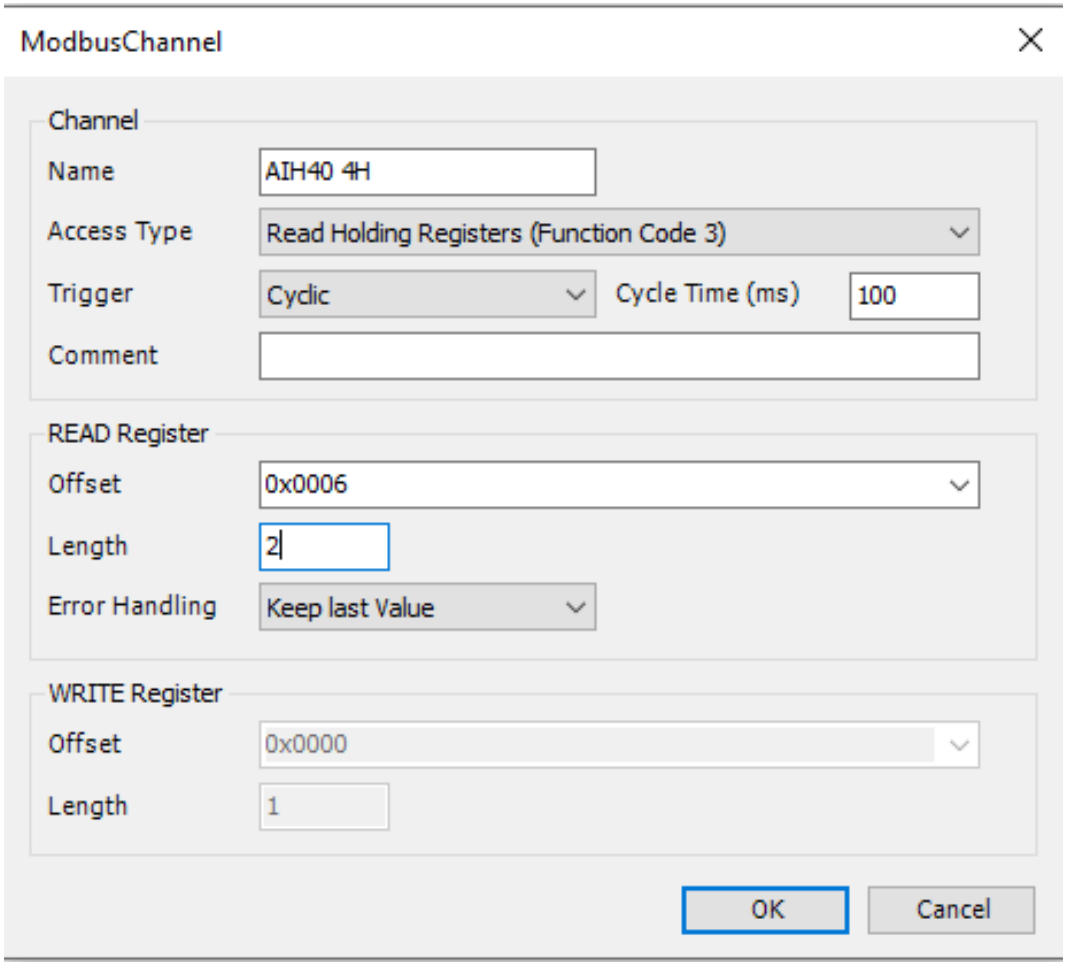

Fig. 21: AIH401 – Creating HART values

- Read the register at **ModbusTCPSlave\I/O image**.

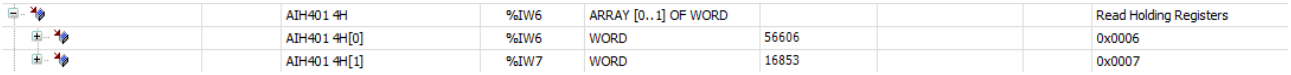

Fig. 22: AIH401 – Reading HART values

Reading out process data by slot-based addressing

I/O process data can also be read by slot-based addressing in addition to accessing packed I/O process data via registers 0x000…0x01FF and 0x0800…0x09FF. The slot-based addressing of the AIH401 I/O module in slot 3 consists of register **0x8060** with a length of **20**.

- Create a Modbus channel with the access type **Read Holding Registers**.
- Set **READ register 0x8060** with a length of **20**.
- Confirm with **OK**.

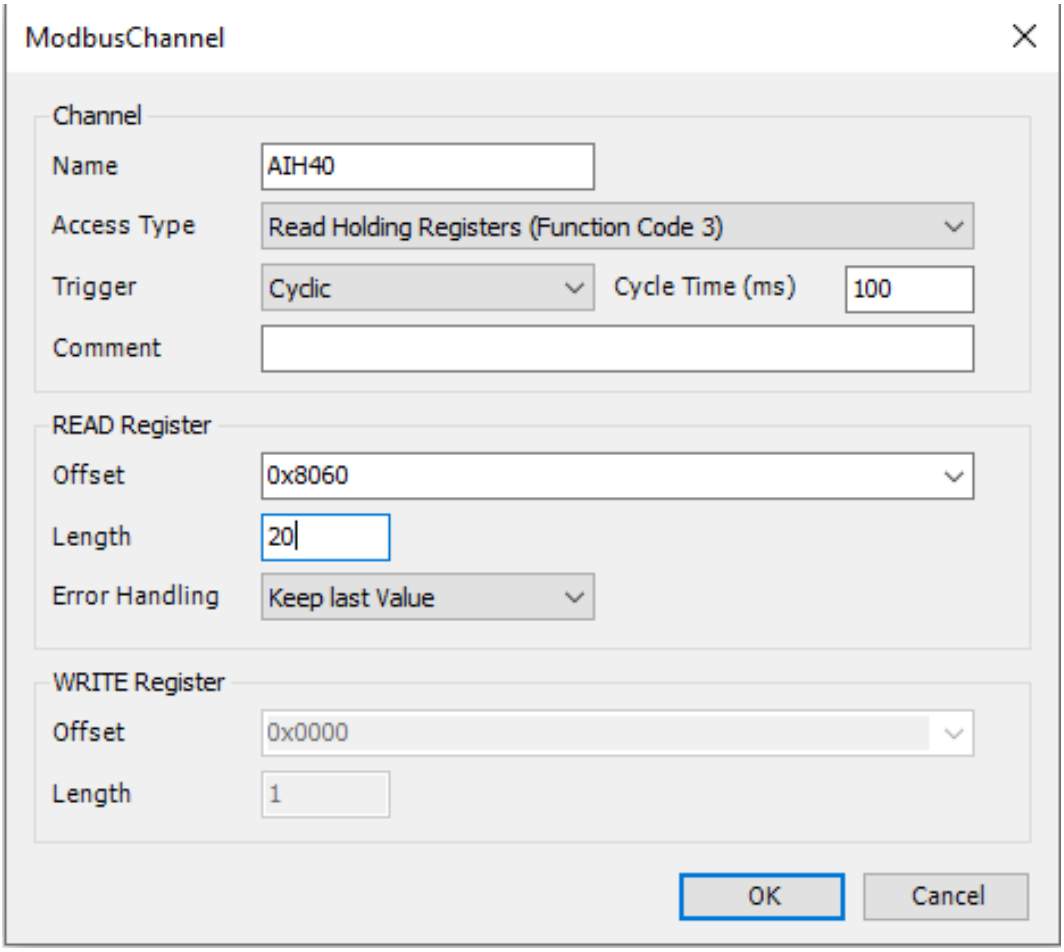

Fig. 23: AIH401 – Creating a Modbus channel by slot

- Read the register at **ModbusTCPSlave\I/O image**.

The packed and slot-based process data is identical:

	AIH40 CH1-4	%IW2	ARRAY [0..3] OF WORD					Read Holding Registers
	AIH40 CH1-4[0]	%IW2	WORD	8100				0x0002
	AIH40 CH1-4[1]	%IW3	WORD	36368				0x0003
	AIH40 CH1-4[2]	%IW4	WORD	36368				0x0004
	AIH40 CH1-4[3]	%IW5	WORD	36368				0x0005
	AIH40 4H	%IW6	ARRAY [0..1] OF WORD					Read Holding Registers
	AIH40 4H[0]	%IW6	WORD	5040				0x0006
	AIH40 4H[1]	%IW7	WORD	16845				0x0007
	AIH40	%IW8	ARRAY [0..19] OF WORD					Read Holding Registers
	AIH40 [0]	%IW8	WORD	8101				0x8060
	AIH40 [1]	%IW9	WORD	36368				0x8061
	AIH40 [2]	%IW10	WORD	36368				0x8062
	AIH40 [3]	%IW11	WORD	36368				0x8063
	AIH40 [4]	%IW12	WORD	5040				0x8064
	AIH40 [5]	%IW13	WORD	16845				0x8065

Fig. 24: AIH401 – Reading Modbus channels

5.2.1 Writing outputs

- Create a Modbus channel with the access type **Write Single Register**.
- Address the **WRITE register** with **0x0802**
- Confirm with **OK**.

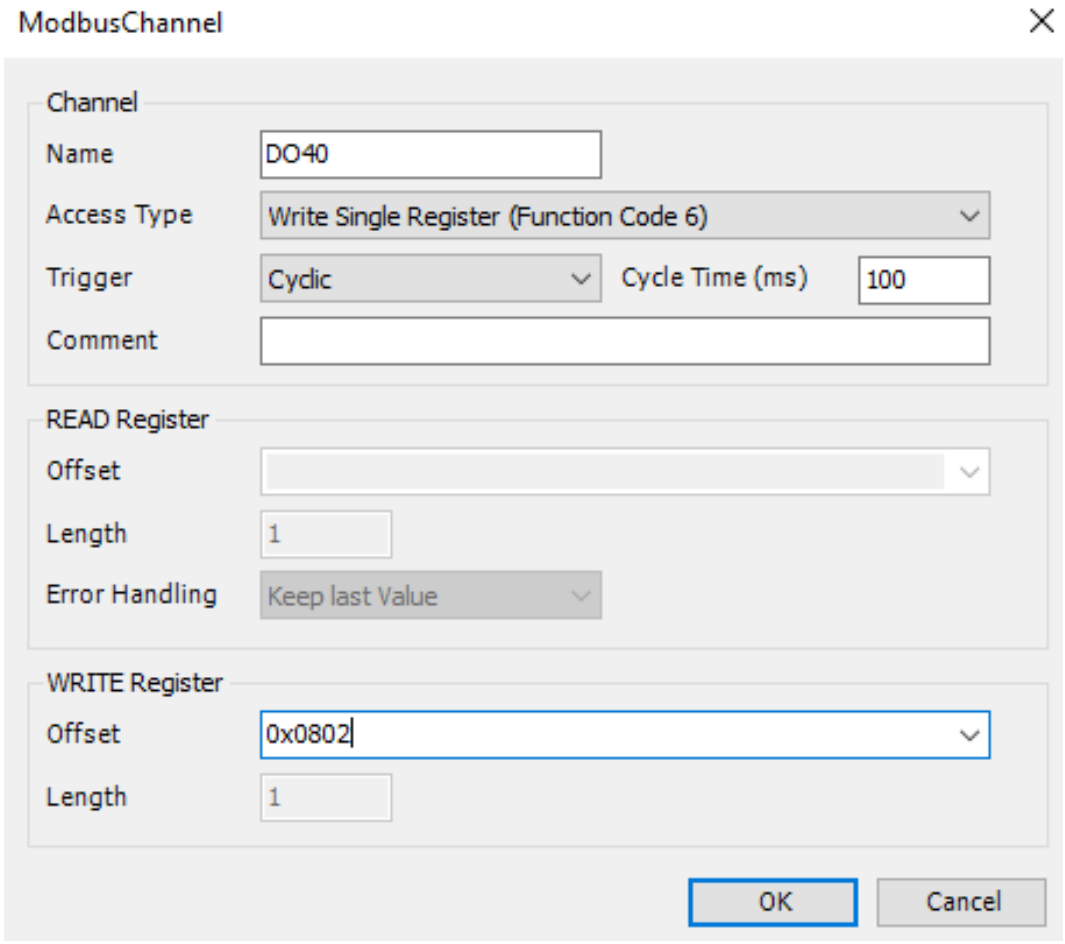

Fig. 25: DO40 – Creating a Modbus channel

Mapping output registers

The output register of the station is mapped in the **ModbusTCPSlavel/O image**:

- ▶ Double-click the **Variable** line in the corresponding field.
- ⇨ This opens the **Input Assistant** window.
- ▶ Find the variable to be linked.
- ▶ **ACTIVATE** is located under **GVL** as it was defined beforehand.

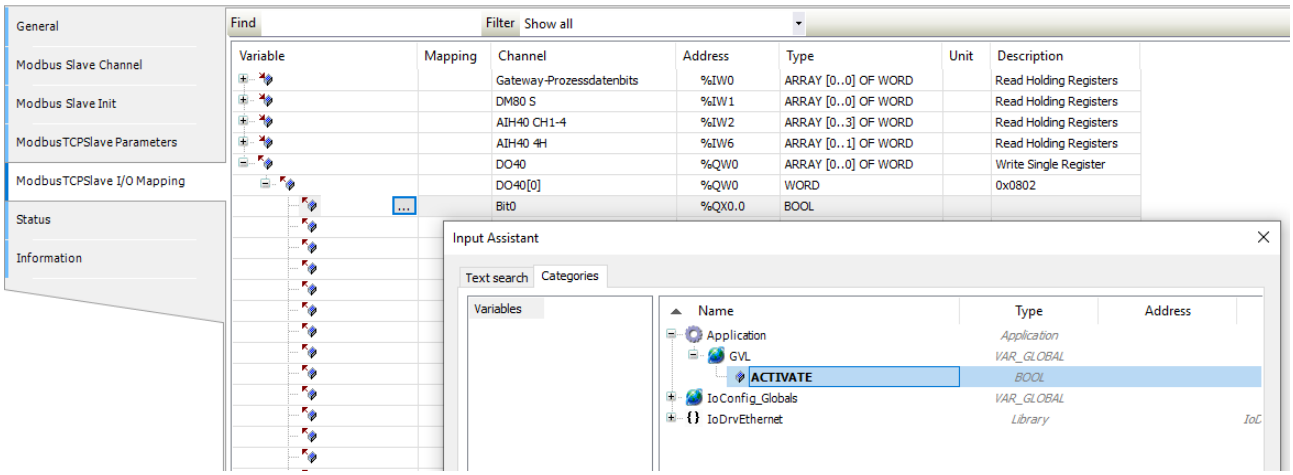

Fig. 26: DO40 – Writing output registers

The **ACTIVATE** variable makes it possible to activate or deactivate different functions. Example: Link LED with the first channel of the DO40 and activate:

Variable	Mapping	Channel	Address	Type	Current Value	Prepared Value	Unit	Description
		Gateway-Prozessdatenbits	%IW0	ARRAY [0..0] OF WORD				Read Holding Registers
		DM80 S	%IW1	ARRAY [0..0] OF WORD				Read Holding Registers
		AIH40 CH1-4	%IW2	ARRAY [0..3] OF WORD				Read Holding Registers
		AIH40 4H	%IW6	ARRAY [0..1] OF WORD				Read Holding Registers
		DO40	%QW0	ARRAY [0..0] OF WORD				Write Single Register
		DO40[0]	%QW0	WORD				0x0802
Application.GVL...		Bit0	~~%QX0.0~~	BOOL	TRUE			
		Bit1	%QX0.1	BOOL	FALSE			
		Bit2	%QX0.2	BOOL	FALSE			
		Bit3	%QX0.3	BOOL	FALSE			

Fig. 27: DO40 – Activating LEDs

5.2.2 Reading diagnostics

In order to read diagnostics in the excom station, Modbus channels must be created by slot. Registers **0xA000…0xA320** can be used to read diagnostics for each slot in the excom station. The following shows the reading of the diagnostics for the gateway:

- Create a Modbus channel with the access type **Read Holding Registers**.
- Set **READ register 0xA000** with a length of **3**.
- Confirm with **OK**.

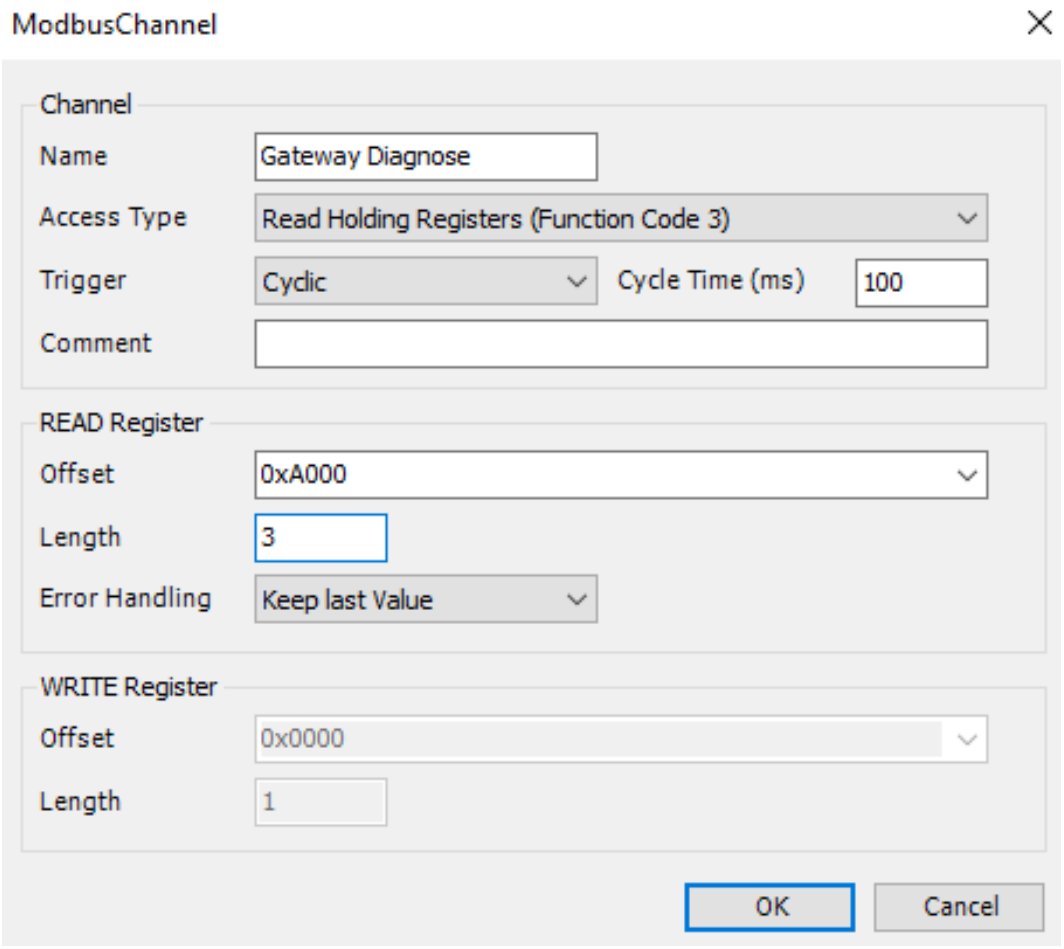

Fig. 28: Gateway diagnostics – Creating a Modbus channel

▶ Read the register at **ModbusTCPSlaveI/O image**.

From the following bit assignments [▷ 15] it is possible read that I/O diagnostics are present and that the I/O configuration in the module rack was changed. The registers **0xA001** and **0xA002** are zero, indicating that no redundancy or CAN diagnostics are present.

	Gateway Diagnose	%IW8	ARRAY [0..2] OF WORD				Read Holding Registers
	Gateway Diagnose[0]	%IW8	WORD	2304			0xA000
	Bit0	%IX16.0	BOOL	FALSE			
	Bit1	%IX16.1	BOOL	FALSE			
	Bit2	%IX16.2	BOOL	FALSE			
	Bit3	%IX16.3	BOOL	FALSE			
	Bit4	%IX16.4	BOOL	FALSE			
	Bit5	%IX16.5	BOOL	FALSE			
	Bit6	%IX16.6	BOOL	FALSE			
	Bit7	%IX16.7	BOOL	FALSE			
	Bit8	%IX17.0	BOOL	TRUE			
	Bit9	%IX17.1	BOOL	FALSE			
	Bit10	%IX17.2	BOOL	FALSE			
	Bit11	%IX17.3	BOOL	TRUE			
	Bit12	%IX17.4	BOOL	FALSE			
	Bit13	%IX17.5	BOOL	FALSE			
	Bit14	%IX17.6	BOOL	FALSE			
	Bit15	%IX17.7	BOOL	FALSE			
	Gateway Diagnose[1]	%IW9	WORD	0			0xA001
	Gateway Diagnose[2]	%IW10	WORD	0			0xA002

Fig. 29: Gateway diagnostics – Reading a Modbus channel

AIH401 – Reading diagnostics

AIH401 on slot 3 is selected as an example of reading diagnostic:

- Create a Modbus channel with the access type **Read Holding Registers**.
- Set **READ register 0xA060** with a length of **3**.
- Confirm with **OK**.

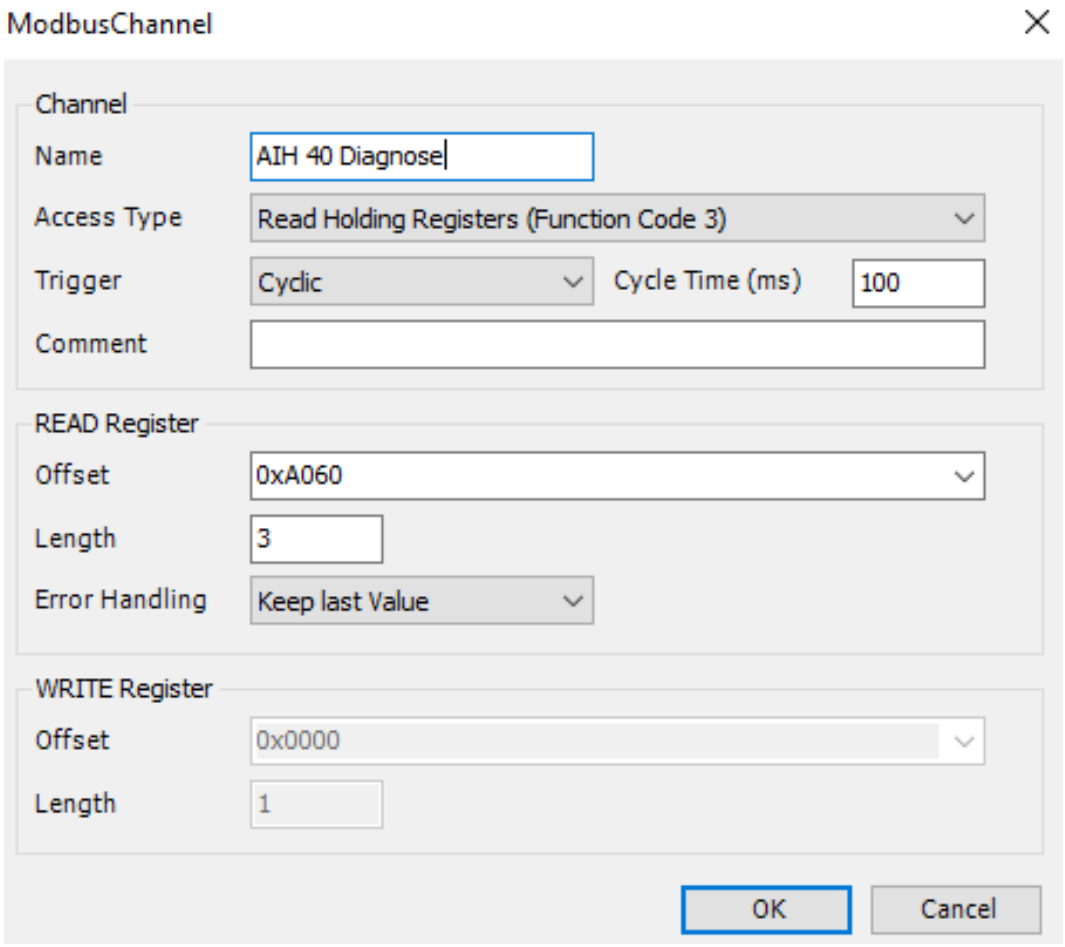

Fig. 30: AIH401 – Creating diagnostics

▶ Read the register at **ModbusTCPSlaveI/O image**.

The first register **0xA060** is reserved and zero. The channel-specific diagnostics can be read from register **0xA061**.

Register **0xA061** indicates at **Byte 2**, **Bit 2** (**%IX65.1**) a wire break on the second channel. **Bit 7** and **Bit 8** (**%IX65.6** and **%IX65.7**) indicate a HART communication error and diagnostics messages for this channel.

	AIH 40 Diagnose	%IW31	ARRAY [0..2] OF WORD				Read Holding Registers
	AIH 40 Diagnose[0]	%IW31	WORD	0			0xA060
	AIH 40 Diagnose[1]	%IW32	WORD	49664			0xA061
	Bit0	%IX64.0	BOOL	FALSE			
	Bit1	%IX64.1	BOOL	FALSE			
	Bit2	%IX64.2	BOOL	FALSE			
	Bit3	%IX64.3	BOOL	FALSE			
	Bit4	%IX64.4	BOOL	FALSE			
	Bit5	%IX64.5	BOOL	FALSE			
	Bit6	%IX64.6	BOOL	FALSE			
	Bit7	%IX64.7	BOOL	FALSE			
	Bit8	%IX65.0	BOOL	FALSE			
	Bit9	%IX65.1	BOOL	TRUE			
	Bit10	%IX65.2	BOOL	FALSE			
	Bit11	%IX65.3	BOOL	FALSE			
	Bit12	%IX65.4	BOOL	FALSE			
	Bit13	%IX65.5	BOOL	FALSE			
	Bit14	%IX65.6	BOOL	TRUE			
	Bit15	%IX65.7	BOOL	TRUE			

Fig. 31: AIH401 – Reading diagnostics for channel 1 and 2

	AIH 40 Diagnose[2]	%IW33	WORD	49858			0xA062
	Bit0	%IX66.0	BOOL	FALSE			
	Bit1	%IX66.1	BOOL	TRUE			
	Bit2	%IX66.2	BOOL	FALSE			
	Bit3	%IX66.3	BOOL	FALSE			
	Bit4	%IX66.4	BOOL	FALSE			
	Bit5	%IX66.5	BOOL	FALSE			
	Bit6	%IX66.6	BOOL	TRUE			
	Bit7	%IX66.7	BOOL	TRUE			
	Bit8	%IX67.0	BOOL	FALSE			
	Bit9	%IX67.1	BOOL	TRUE			
	Bit10	%IX67.2	BOOL	FALSE			
	Bit11	%IX67.3	BOOL	FALSE			
	Bit12	%IX67.4	BOOL	FALSE			
	Bit13	%IX67.5	BOOL	FALSE			
	Bit14	%IX67.6	BOOL	TRUE			
	Bit15	%IX67.7	BOOL	TRUE			

Fig. 32: AIH401 – Reading diagnostics for channel 3 and 4

The diagnostics can also be viewed in the web server at **Gateway Diagnostics** irrespective of the Modbus channels created:

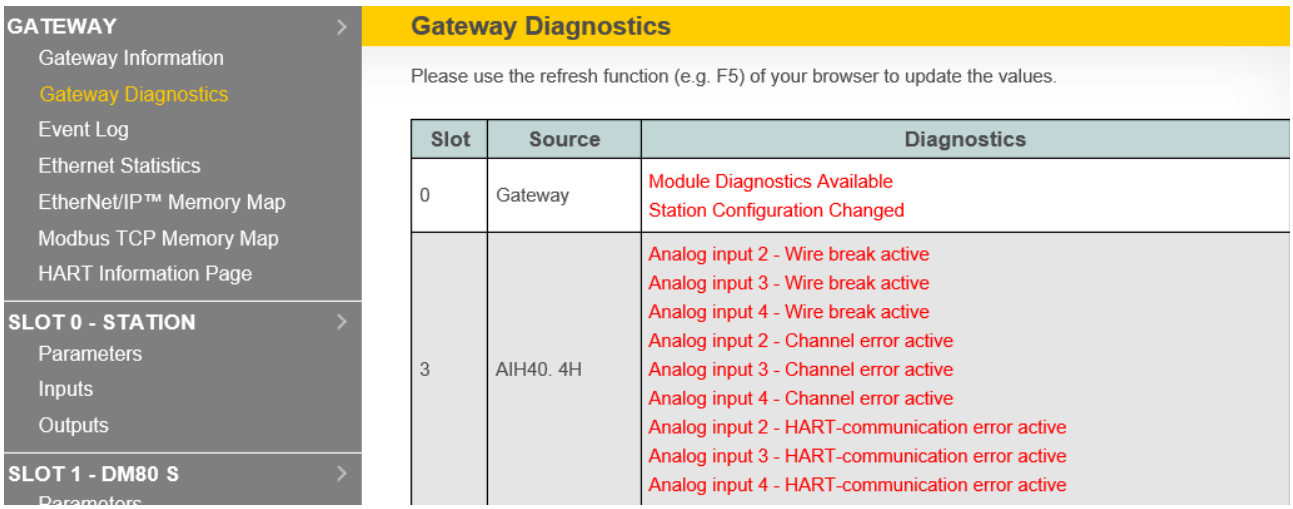

Fig. 33: Diagnostics in the web server

6 Redundancy Strategies

6.1 Topology

The general topology of the Turck-specific system redundancy with the Ethernet protocols EtherNet/IP, Modbus TCP and PROFINET has the following structure:

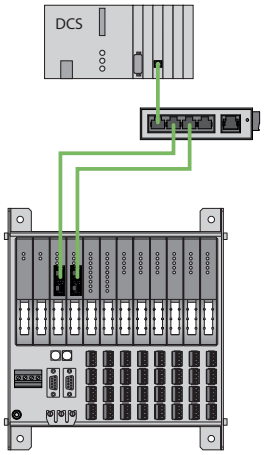

Fig. 34: System redundancy with one master and two gateways

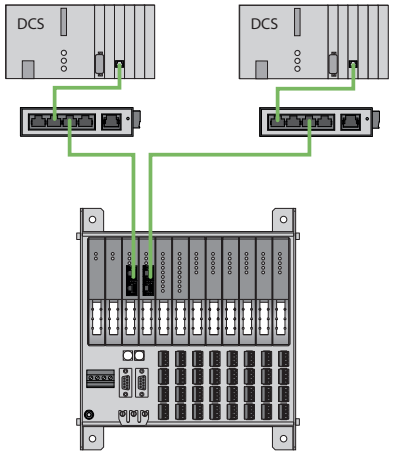

Fig. 35: System redundancy with two masters and two gateways

The system redundancy with one master and two gateways is a Turck-specific, parameterizable redundancy function of the excom system. The two gateways are provided here with separate IP addresses. The separate IP addresses are used to set up independent communication. The gateways transmit the input data and receive the output data via the IP addresses. One gateway is the primary gateway while the second gateway acts as a backup. If the primary gateway fails, a bumpless switchover to the backup gateway is carried out automatically. The redundancy function makes it possible to implement interruption-free communication. The output word of the gateway enables the forcing of a redundancy switchover.

When system redundancy is implemented with two masters and two gateways, two independent Ethernet masters communicate with the associated gateway. Both masters can be controlled via one or two process control system controllers. The process data is processed via two separate and independent Ethernet connections to the excom system.

6.2 Redundancy setup

NOTE
Both gateways must have the same configuration, parameterization and firmware.

The **Redundancy mode** gateway parameter must be set for **system redundancy**.

6.3 System redundancy

NOTE
Both gateways must have the same configuration, parameterization and firmware.

If the **Redundancy mode** parameter is set to **system redundancy** in the DTM, web server or control system, the excom station operates in system redundancy mode. Both gateways communicate with their respective master. The PRIO LED is lit on the active gateway. The active gateway takes over the output data transferred by the master and sends this to the output modules.

The gateway communicating with the secondary master ignores the received output data as the secondary module does not have write access to the output modules.

If the gateway is configured in the controller as “GEN... C”, the gateway is provided with an input word as well as an output word for monitoring redundancy. The input word describes the current state of the gateway.

The output word is used for the manual redundancy switchover in the master. It is possible to switch in the process control system from the primary gateway to the secondary gateway. A switchover is carried out in response to the following events:

- The primary gateway was removed.
- Communication to the primary gateway was interrupted.
  The outputs are set to 0 until switch-over to the other gateway. After elapse of the timer for interrupted connections, the system switches to the other gateway.

After a switchover, an automatic switchover to the former primary gateway is no longer carried out.

When the excom system is started, the gateway on the left starts to operate as the primary gateway. If communication with the left gateway fails, the gateway on the right tries to establish primary communication.

Assignment of gateway process data bits

The input word of the gateway process data is used to view the gateway and system redundancy of the excom station:

	Bit							
Byte	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
0	Not used			Left power supply unit	Right power supply unit	Gateway redundancy	Slot	Active/ passive
1	Not used							

Meaning of the gateway process data bits

Designation	Meaning
Left power supply unit	0: Left power supply unit not present
	1: Left power supply unit fitted
Right power supply unit	0: Right power supply unit not present
	1: Right power supply unit fitted
Gateway redundancy	0: Redundant gateway or redundant communication not available
	1: Redundancy available
Slot	0: Gateway is located on the right slot (GW2)
	1: Gateway is located on the left slot (GW1)
Active/ passive	0: Gateway is passive
	1: Gateway is active

Assignment of the command bits

The output word of the gateway enables the forcing of a redundancy switchover in the “Red switching” web server:

	Bit							
Byte	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
0	Not used					Control bit	Redund-ancy switchover is initiated	Activation of the right or left gate-way
						Control bits for edge change		
1	Not used							

Meaning of the command bits

Designation	Meaning
Bit 2 = 0 Redundancy switchover is initi-ated	11 → 01: Receiver is the passive gateway. The passive gateway requests control from the active gateway and becomes active.
	11 → 10: Receiver is the active gateway. The active gateway gives control to the passive gateway and becomes passive.
Bit 2 = 1 Activation of the right or left gateway	11 → 01: Receiver is the left gateway. The left gate-way requests control from the right gateway and becomes active.
	11 → 10: Receiver is the right gateway. The right gateway requests control from the left gateway and becomes active.

TURCK

Over 30 subsidiaries and over
60 representations worldwide!

100022828 | 2020/07

www.turck.com