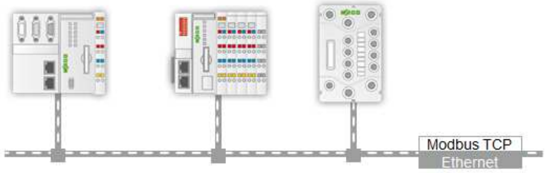
Application Note



MODBUS

with WAGO Ethernet Couplers and Controllers

A300003, English

Version 2.1.0

Imprint

WAGO Kontakttechnik GmbH & Co. KG
Hansastraße 27
D-32423 Minden

Phone: +49 (0) 571/8 87 – 0
Fax: +49 (0) 571/8 87 – 1 69

Email: info@wago.com

Web: http://www.wago.com

Technical Support
Phone: +49 (0) 571/8 87 – 5 55
Fax: +49 (0) 571/8 87 – 85 55

Email: support@wago.com

Every conceivable measure has been taken to ensure the accuracy and completeness of this documentation. However, as errors can never be fully excluded, we always appreciate any information or suggestions for improving the documentation.

We wish to point out that the software and hardware terms, as well as the trademarks of companies used and/or mentioned in the present manual, are generally protected by trademark or patent.

TABLE OF CONTENTS

1 Important Notes

To ensure fast installation and start-up of the units, we strongly recommend that the following information and explanations are carefully read and adhered to.

1.1 Legal Principles

1.1.1 Copyright

This document, including all figures and illustrations contained therein, is subject to copyright. Any use of this document that infringes upon the copyright provisions stipulated herein is prohibited. Reproduction, translation, electronic and phototechnical filing/archiving (e.g., photocopying), as well as any amendments require the written consent of WAGO Kontakttechnik GmbH & Co. KG, Minden, Germany. Non-observance will entail the right of claims for damages.

WAGO Kontakttechnik GmbH & Co. KG reserves the right to make any alterations or modifications that serve to increase the efficiency of technical progress. WAGO Kontakttechnik GmbH & Co. KG owns all rights arising from granting patents or from the legal protection of utility patents. Third-party products are always mentioned without any reference to patent rights. Thus, the existence of such rights cannot be excluded.

1.1.2 Personnel Qualification

The use of the product described in this document is exclusively geared to specialists having qualifications in PLC programming, electrical specialists or persons instructed by electrical specialists who are also familiar with the appropriate current standards. WAGO Kontakttechnik GmbH & Co. KG assumes no liability resulting from improper action and damage to WAGO products and third-party products due to non-observance of the information contained in this document.

1.1.3 Intended Use

For each individual application, the components are supplied from the factory with a dedicated hardware and software configuration. Modifications are only admitted within the framework of the possibilities documented in this document. All other changes to the hardware and/or software and the non-conforming use of the components entail the exclusion of liability on part of WAGO Kontakttechnik GmbH & Co. KG.

Please send your requests for modified and new hardware or software configurations directly to WAGO Kontakttechnik GmbH & Co. KG.

1.2 Scope of Validity

This application note is based on the stated hardware and software from the specific manufacturer, as well as the associated documentation. This application note is therefore only valid for the described installation.
New hardware and software versions may need to be handled differently.

Please note the detailed description in the specific manuals.

1.3 Symbols

DANGER ⚠	**Warning against personal injury!** Indicates a high-risk, imminently hazardous situation which, if not avoided, will result in death or serious injury.
DANGER ⚡	**Do not work on components while energized!** Indicates a high-risk, imminently hazardous situation which, if not avoided, will result in death or serious injury.
WARNING ⚠	**Warning against personal injury!** Indicates a moderate-risk, potentially hazardous situation which, if not avoided, could result in death or serious injury.
CAUTION ⚠	**Warning against personal injury!** Indicates a low-risk, potentially hazardous situation which, if not avoided, may result in minor or moderate injury.
NOTICE ⚠	**Warning: Damage to property!** Indicates a potentially hazardous situation which, if not avoided, may result in damage to property.
Note ➔	**Important note!** Indicates a potential malfunction which will not result in damage to property, however, if not avoided.
Information ℹ	**Additional Information** Refers to additional information which is not an integral part of this documentation (e.g., the Internet).

1.4 Number Notation

Table 1: Number Notation

Number code	**Example**	**Remark**
Decimal	100	Normal notation
Hexadecimal	0x64	C notation
Binary	'100' '0110.0100'	In quotation marks, nibble separated by a period

1.5 Font Conventions

Table 2: Font Conventions

Font type	**Explanation**
italic	Names of paths and files are displayed in italics, e.g.: *C:\Programs\WAGO-I/O-CHECK*
Menu	Menu options are displayed in bold, e.g., : **Save**
>	A “greater than” symbol between two names denotes the selection of a menu option, e.g.: **File** > **New**
Input	Designation of input or optional fields are displayed in bold, e.g.: **Start of measurement range**
“Value”	Input or selection values are displayed in quotation marks, e.g.: Enter the value “4mA” under **Start of measurement range**.
[Button]	Button labels within the dialogs are bold and enclosed in square brackets, e.g.: **[Input]**
[Key]	Keys on the keyboard are bold and enclosed in square brackets, e.g.: **[F5]**

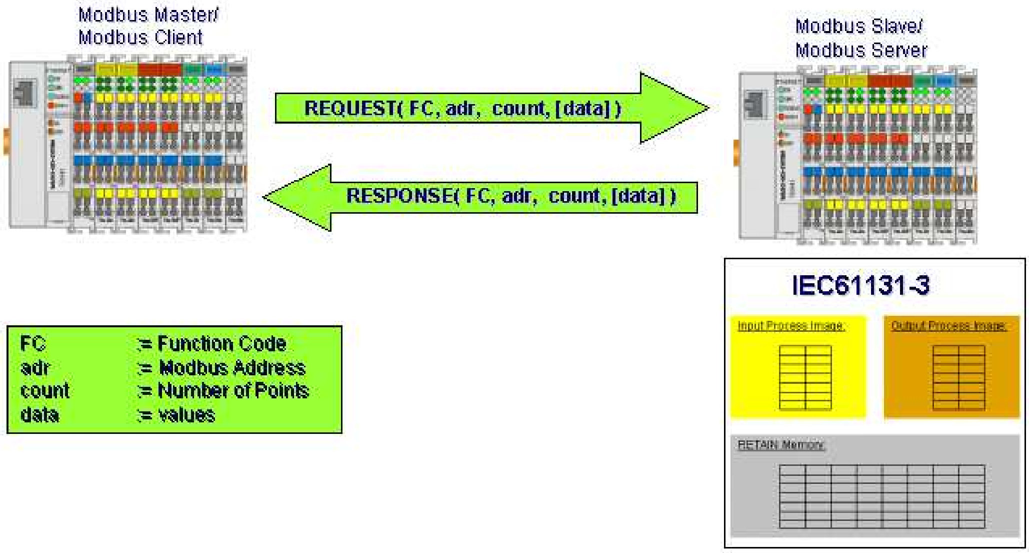
2 Introduction

This application note explains how to use the MODBUS protocol in conjunction with the WAGO-I/O-SYSTEM.

The modular design of the WAGO-I/O-SYSTEM makes it possible to attach a number of different I/O modules to the fieldbus coupler in almost any order. The variability of the node configuration, however, prevents the static assignment of data points and MODBUS addresses.

This application note shows the relationships between node configuration, process images, IEC-61131 addresses and MODBUS addresses.

Metaphorically, MODBUS communication is a “game of questions and answers”. It always involves a MODBUS master and one or more MODBUS slaves.



The MODBUS master makes a REQUEST to the MODBUS slave. The request includes the function code (FC), MODBUS address (adr) and the number (count) of data objects ([data]) to which read or write access is required. The slave processes the request and returns a corresponding RESPONSE.

MODBUS knows only two data types (coils and register).
A “Coil” stands for the state of a digital value (1-bit).
A “Register” is a 16-bit (WORD) analog value.

With fieldbus couplers and programmable fieldbus controllers, WAGO offers two types of head stations.

- Fieldbus couplers (remote IO) allow direct access to the data of connected I/O modules via a fieldbus interface.
  (e.g., 750-352)

- Fieldbus controllers are programmable fieldbus couplers. CODESYS 2.3 is used to program the fieldbus couplers. (e.g., 750-88x, 750-820x). Additional memory ranges (PFC variables and flags) are available for data exchange between PLC program and fieldbus.

The MODBUS protocol can be used to read or modify memory location in the process image. However, the node configuration or PLC program determines which I/O module or what data is located at a specific memory location.

After the power is turned on, a WAGO fieldbus coupler or programmable fieldbus controller determines the current node configuration and creates process images for inputs and outputs.

Complex and digital I/O modules are differentiated:

- Complex I/O modules have a data width of more than one byte, including analog modules, counters, steppers, serial interfaces, etc. During the first step of process image creation, the complex I/O modules are arranged in the process image based on their physical order after the fieldbus coupler.

- The data of the digital I/O modules is packaged to full bytes in a second step based on their position behind the fieldbus coupler and arranged directly behind those of the “Complex” I/O modules.

All WAGO fieldbus couplers and programmable fieldbus controllers have exactly one process image for physical inputs and one for physical outputs. The data of the complex I/O modules is saved in the respective process image, followed by the data of the digital I/O modules.

The process image configuration is described by IEC 61131-3 language elements. This is necessary since the MODBUS protocol only defines services on the basic data types and does not control their meaning or specific addresses.

Address assignment for the PFC200 family looks a bit different. Due to the system property that I/O modules cannot be accessed directly via the MODBUS, they also do not appear in the mapping table.

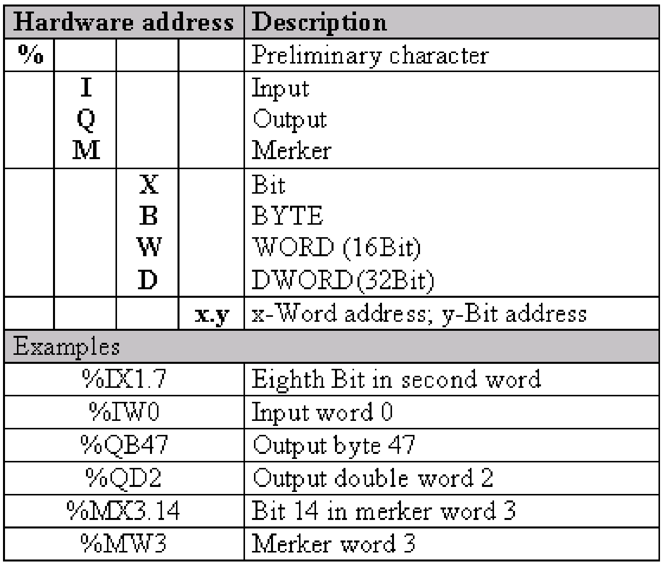
Please also note the following:

Variable monitoring has been optimized in CODESYS 2.3.
The result is that declared variables not used in the program code are not monitored and are therefore always displayed in the debugger with the value “0”.

2.1 The IEC 61131-3 Addresses

In the PLC program, an IEC 61131-3 address is used to access the process image and specific data ranges (PFC variables and flags).

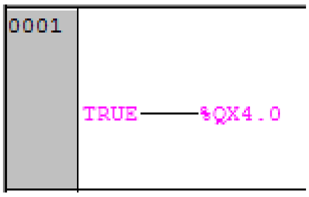
The following table shows the structure of a hardware address in the IEC 61131 syntax.



Hardware address				Description
%				Preliminary character
	I Q M			Input Output Merker
		X B W D		Bit BYTE WORD (16Bit) DWORD(32Bit)
			x.y	x-Word address; y-Bit address
Examples				
%IX1.7				Eighth Bit in second word
%IW0				Input word 0
%QB47				Output byte 47
%QD2				Output double word 2
%MX3.14				Bit 14 in merker word 3
%MW3				Merker word 3

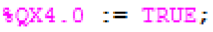
If the first digital output of the sample node from Section 2.2 is to be set, the respective assignment appears as follows:

Assignment in FUP:



```
0001
TRUE———%QX4.0
```

Assignment in ST:



```
%QX4.0 := TRUE;
```

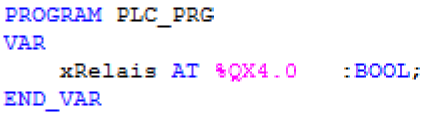
Direct access via hardware addresses in the user program is possible, but not recommended.

The designation “%QX4.0” is not very descriptive and complicates the readability of the program.

To increase readability, symbolic addressing is recommended. Variables are explicitly declared to a hardware address.
The advantage is that the output can be give an descriptive name (e.g., function name or resource ID).

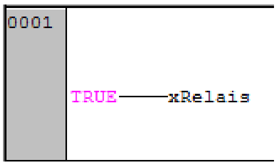
In addition, if the address shifts because a node is added, only the address in the variable declaration has to be adjusted:

Variable declaration:

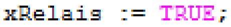


```
PROGRAM PLC_PRG
VAR
    xRelais AT %QX4.0   :BOOL;
END_VAR
```

Assignment in FUP:



Assignment in ST:



```
xRelais := TRUE;
```

In this way, any typed variables can be positioned in the memory.

```
TYPE TMyType :
 STRUCT
   wState      : WORD;     (* actual state *)
   dwJobAct    : DWORD;    (* Actual job *)
   dwJobLast   : DWORD;    (* Last job *)
   dwJobNext   : DWORD;    (* Next job *)
   xFlagDoIt   : BOOL;     (* something should happen *)
   xFlagDone   : BOOL;     (* something have been done *)
 END_STRUCT
END_TYPE

VAR
   xMyOutput  AT %QX0.2 : BOOL;   (* A digital output *)
   wMyInput   AT %IW1   : WORD;   (* A analog input *)
   oInterface AT %MW0   : TMyType;(* A userdefined type  *)
VAR_END
```

This approach can be advantageous when implementing software interfaces between the control system and MODBUS field devices. However, caution is advised because empty bytes are added due to the DWORD alignment and thus the data is inaccessible via the expected MODBUS address.

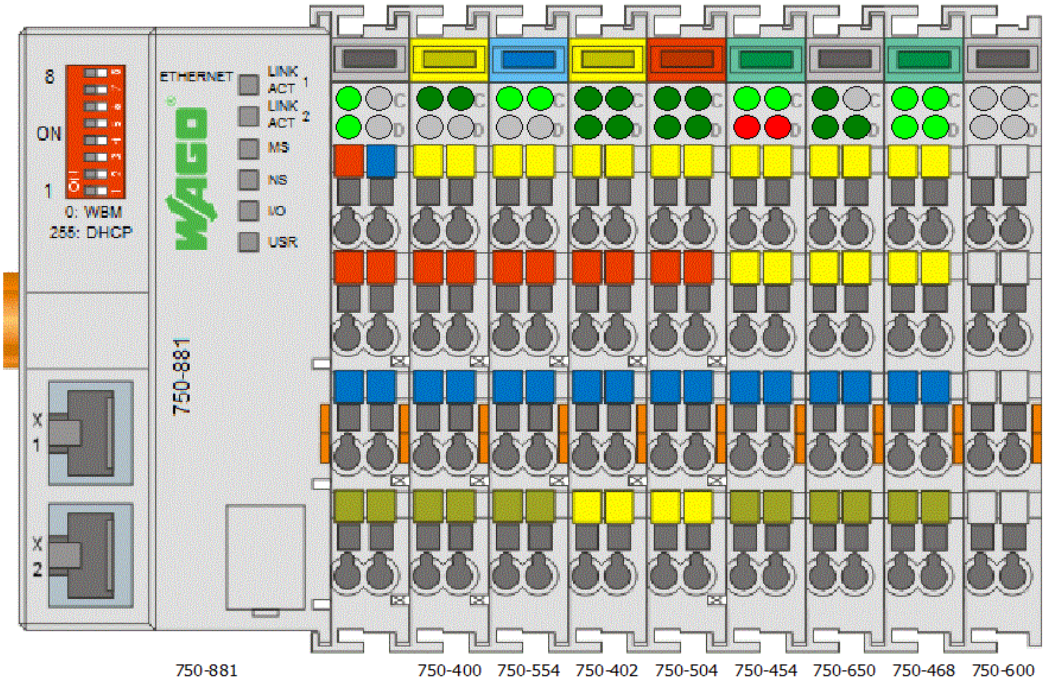
2.2 Structure of the Process Images

A process image is a part of the memory with a fixed size in which the process values of the I/O modules are stored.

One input process image and one output process image are generated.
The process values of the individual I/O modules are stored in the respective process image depending on the type and position behind the fieldbus coupler.

Digital and complex I/O modules are differentiated.

Complex I/O modules (often referred to as “analog” modules) represent all I/O modules having a data width greater than one byte.
Examples are analog inputs and outputs, counter modules, I/O modules for angle and distance measurement, communication modules such as RS-232 C, etc. or in other words, “All non-digital I/O modules”.



For both the input and the output process image, the data of the I/O modules is stored in the respective process image according to the order of their position after the fieldbus coupler.

First, the complex module data is stored in the process image, followed by the digital module data.

The bits of the digital I/O module are combined into bytes. If the amount of digital I/O information exceeds eight bits, the controller automatically starts a new byte.

The data width of an I/O module is between 0 and 48 bytes. Please find more details in the fieldbus coupler manual as well as in the manual of the applicable I/O module.

The following table illustrates this relation using specific example.

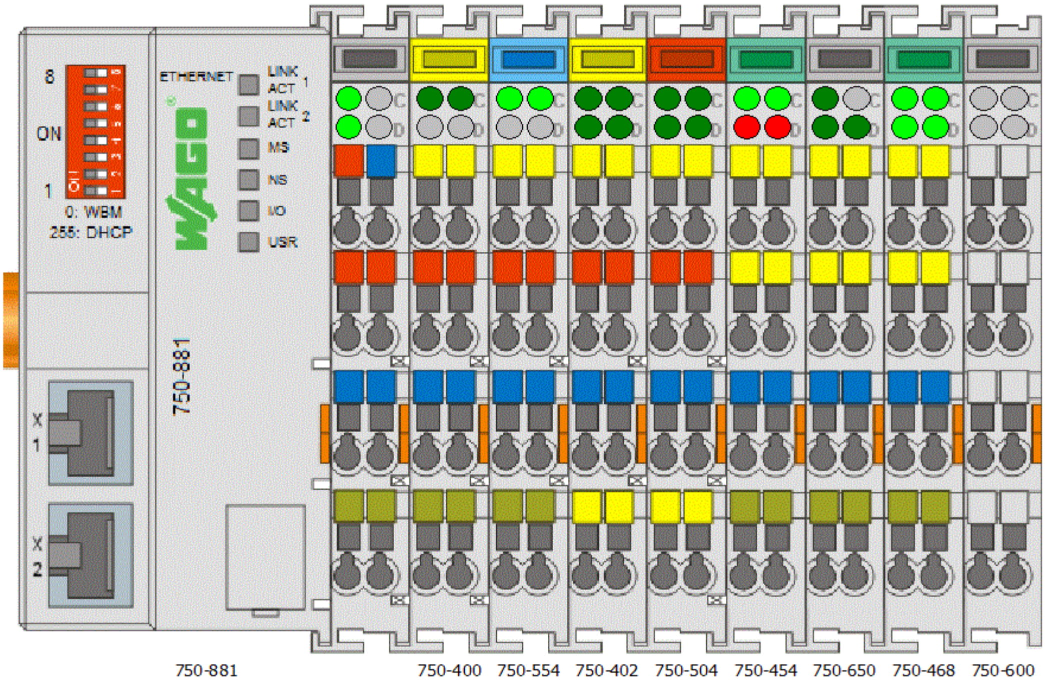
I/O Modules		**Input image**		**Output image**		**Description**
Type	**C**	**run1**	**run2**	**run3**	**run4**	
750-400	1		%IX8.0			**2 DI 24VDC 3ms**: First digital input module with a data width of two bits. Since the complex input modules from “run1” already occupy the first 8 words, the digital inputs are put to lower order bits in word 8.
	2		%IX8.1			
750-554	1			%QW0		**2 AO 4-20mA:** First analog output module with a data width of two words. This I/O module occupies the first two words in the output process image.
	2			%QW1		
750-402	1		%IX8.2			**4 DI 24VDC**: The four digital inputs of this I/O module are put behind the two inputs of the 750-400 and are in the eighth word of the input process image.
	2		%IX8.3			
	3		%IX8.4			
	4		%IX8.5			
750-504	1				%QX4.0	**4 DO 24VDC**: First I/O module of the digital output. The I/O modules of the analog outputs already occupy the first four words in the output process image.
	2				%QX4.1	
	3				%QX4.2	
	4				%QX4.3	
750-454	1	%IW0				**2 AI 4-20mA:** First I/O module of the analog inputs. This I/O module occupies the first two words of the input table.
	2	%IW1				
750-650	1	%IW2				**RS232 C 9600/8/N/1:** The serial interface module 750-650 is a complex I/O module represented both in the input and output process images with 4 bytes each.
		%IW3				
				%QW2		
				%QW3		
750-468	1	%IW4				**4 AI 0-10V S.E:** The I/O module 750-468 follows the 2 input words of the 750-454 and the 2 input words of the 750-650. The I/O module 750-467 occupies 4 input words (4 channels 0-10V).
	2	%IW5				
	3	%IW6				
	4	%IW7				
750-600						**End module** The I/O module 750-600 is a passive module.

C* : Channelcount

“run1” to “run4” in the above table describe the chronological order during compilation of the input and output process images when a WAGO coupler or controller is booted.

3 WAGO Controller 750-88x as MODBUS Slave

For the node configuration from the previous section, the status of the third channel, digital output module 750-504, is to be changed via the MODBUS protocol.



In general, each of the following MODBUS services can be used to change the status of the third channel of the digital output module 750-504.

FC	Name	Description
FC5	Write coil	Write a single digital output
FC6	Write single register	Write a single analog output
FC15	Force multiple coils	Write several digital outputs
FC16	Write multiple registers	Writing several analog outputs
FC22	Mask write	Masked writing of a register
FC23	Read/write multiple registers	Read/write operation to analog inputs/outputs

First choice would certainly be the FC5 (Write coil), but use of FC15, FC22 and FC16 is displayed for this task.

It is necessary to determine the MODBUS address corresponding to the IEC address for the selected MODBUS service.

This information can be used to configure the MODBUS master were it not for the matter of the write permission for programmable fieldbus controllers of the 750-8xx series.

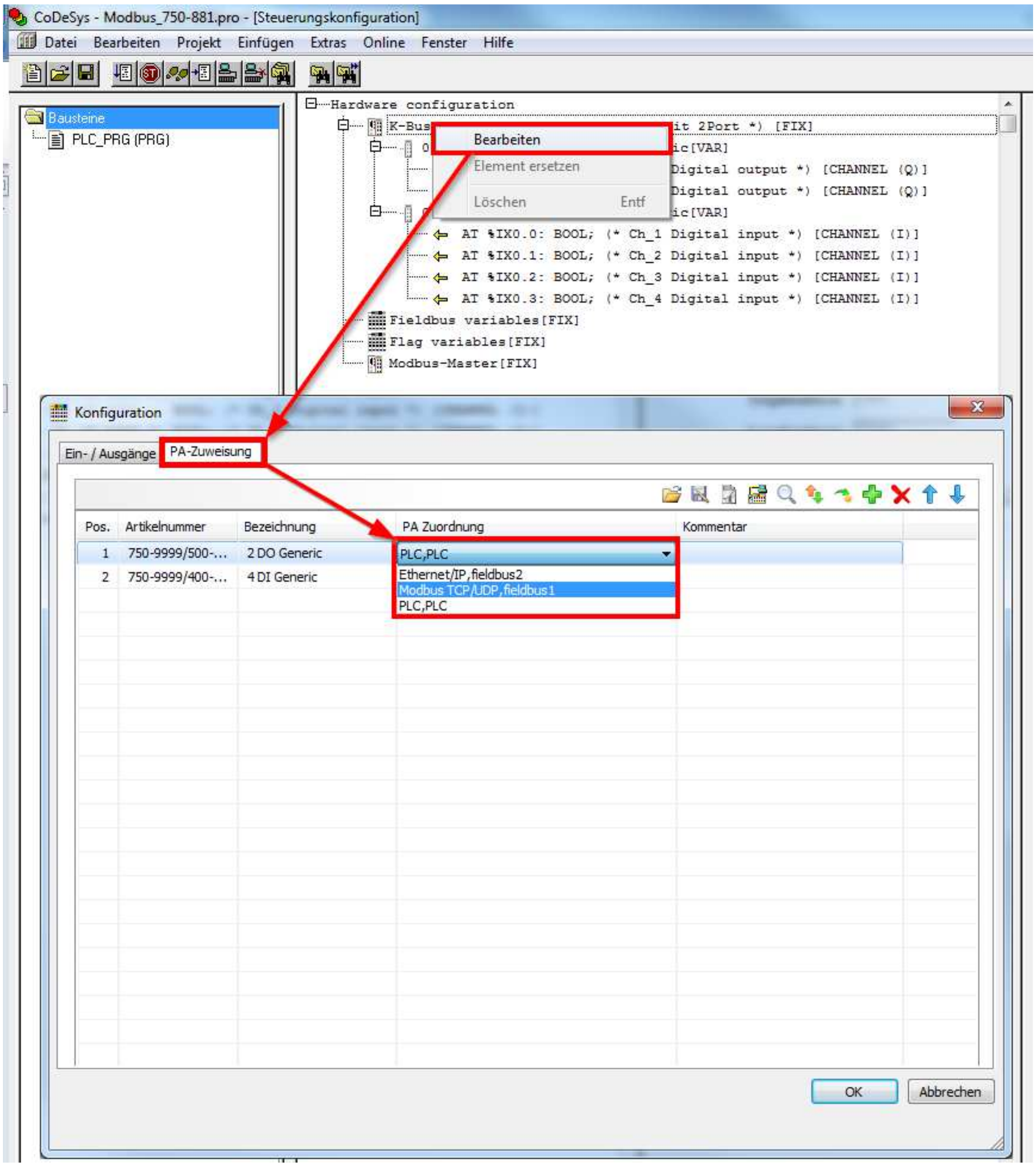
For fieldbus couplers, control of write permissions to process images is simple:

– Physical inputs can only be read.

– Physical outputs can be read and written.

In general, the same applies to programmable fieldbus controllers.
However, only the PLC program has write permission via the outputs.

If a physical output is changed via MODBUS when a PLC program is running, the controller configuration must be adjusted in the programming environment.


CoDeSys - Modbus_750-881.pro - [Steuerungskonfiguration]
Datei Bearbeiten Projekt Einfügen Extras Online Fenster Hilfe
Bausteine
PLC_PRG (PRG)
Bearbeiten
Element ersetzen
Löschen Entf
Konfiguration
Ein- / Ausgänge
PA-Zuweisung
OK
Abbrechen

```
Hardware configuration
  K-Bus                              it 2Port *) [FIX]
    0                                ic[VAR]
                                     Digital output *) [CHANNEL (Q)]
                                     Digital output *) [CHANNEL (Q)]
                                     ic[VAR]
      AT %IX0.0: BOOL; (* Ch_1 Digital input *) [CHANNEL (I)]
      AT %IX0.1: BOOL; (* Ch_2 Digital input *) [CHANNEL (I)]
      AT %IX0.2: BOOL; (* Ch_3 Digital input *) [CHANNEL (I)]
      AT %IX0.3: BOOL; (* Ch_4 Digital input *) [CHANNEL (I)]
  Fieldbus variables[FIX]
  Flag variables[FIX]
  Modbus-Master[FIX]
```

Pos.	Artikelnummer	Bezeichnung	PA Zuordnung	Kommentar
1	750-9999/500-...	2 DO Generic	PLC,PLC Ethernet/IP,fieldbus2 Modbus TCP/UDP,fieldbus1 PLC,PLC	
2	750-9999/400-...	4 DI Generic		

The output must be explicitly assigned to the MODBUS.

The write permission can be assigned to the PLC program, MODBUS protocol or Ethernet/IP protocol.

The write permission assignment is stored in the "/etc/EA-config.xml" file. If the actual node configuration differs from that which is configured, all I/O modules are assigned to the MODBUS protocol and the I/O LED displays error code 6 with error argument 9.

Error code 6: Node configuration error
Error argument 9: Error mapping the I/O modules to a fieldbus
Remedy:
1) Check the "/etc/EA-Config.xml" file for your fieldbus controller.
2) Delete the "/etc/EA-Config.xml" file, e.g., Online -> Reset (original).

In addition to physical inputs and outputs, the programmable fieldbus controllers have a non-volatile flag area of typically 24 kB, as well as PFC-IN and PFC-OUT areas, each 256 words (512 bytes).

Note

PFC200 (750-820x)!
The PFC200 (750-820x) has 104 kB flags und 1000 word PFC-IN and PFC-OUT.

The flag area can be read and written via the MODBUS protocol and via the PLC program.

The main field of application for the PFC-IN and the PFC-OUT area is the implementation of interfaces to other controls via the MODBUS protocol.
It is only possible to write data to the PFC-IN area via the MODBUS protocol. From the perspective of the PLC, the PFC-IN area are local inputs that can only be read.
The PFC-OUT area resembles physical outputs, data can only be written to it via the PLC program.

3.1 Example: FC15(Force multiple coils)

The MODBUS service FC15 “Force multiple coils” allows the user to change up to 512 digital outputs with one telegram.

750-881: MODBUS vs. IEC 61131 Addresses for FC15			
MODBUS Address		IEC 61131 Address	Description
[dec]	[hex]		
0 ... 511	0x0000 ... 0x00FF	Physical-Output-Area (1)	First 512 digital outputs

Since this is a “digital” MODBUS service, the complex I/O modules are ignored when calculating the MODBUS address. The MODBUS address corresponds to the channel number of the digital output.

The MODBUS address of the third channel of the first digital output module 750-504 is “2”. (MODBUS addresses begin with “zero”.)

3.2 Example: FC22 (Mask write)

The MODBUS service FC22 “Mask write” is a register service that makes it possible to specifically change bits in a register.
In addition to the MODBUS address of the register, an AND mask and OR mask is transferred and sent to the MODBUS slave.

Determining the MODBUS address starts with evaluating the node configuration or creating the output process image. As shown in the previous section, the determination can be made “on foot” or for programmable fieldbus controllers with support from the CODESYS controller configuration.

Both methods should output “%QX4.2” as the IEC address of the third digital output of the first 750-504. The third digital output of the first 750-504 has address “4” in the MODBUS register.

The mask registers operate according to the following rule:

```
Result = (Content AND AndMask) OR (OrMask AND (NOT AndMask))
```

To set the third output and to leave all other outputs unchanged, use “0xFFFB” as AND mask and “0x0004” as OR mask.

To set the third output and to leave all other outputs unchanged, use “0xFFFB” as AND mask and “0x0000” as OR mask.

3.3 Example: FC16(Write multiple register)

The MODBUS service FC16 “Write multiple register” is a register service that allows changing up to 120 registers with a single telegram.
“NumberOfPoints” not only sends the MODBUS address of the register to the Modbus slave, but also the number of registers to be changed and the data itself.

Determining the MODBUS address starts with evaluating the node configuration or creating the output process image.
As shown in the previous section, the determination can be made “on foot” or for programmable fieldbus controllers with support from the CODESYS controller configuration.

Both methods should output “%QX4.2” as the IEC address of the third digital output of the first 750-504. The MODBUS address can now be looked up in the address assignment table of the 750-881.

750-881: MODBUS vs. IEC 61131 Addresses for FC6, FC16, FC22 and FC23

MODBUS Address		**IEC 61131 Address**	**Description**
[dec]	**[hex]**		
0 ... 255	0x0000 ... 0x00FF	%QW0 ... %QW255	Physical output area (1) First 256 Words of physical output data

The address assignment table is interpreted as follows: The output word %QW0 can be accessed via MODBUS address 0.
This means that “4” would be the MODBUS address for %QW4.

With this function, it is not possible to set the digital output independently of the other output, which is not disadvantage when using the “state machine” in the PLC program. That way, an output pattern can be defined and the state of the installation/machine is always known.

If only the third channel of the 750-504 is used, it is possible to set the digital output with date 0x0004 and to reset it with date 0x0000.

4 WAGO Controller as MODBUS Master

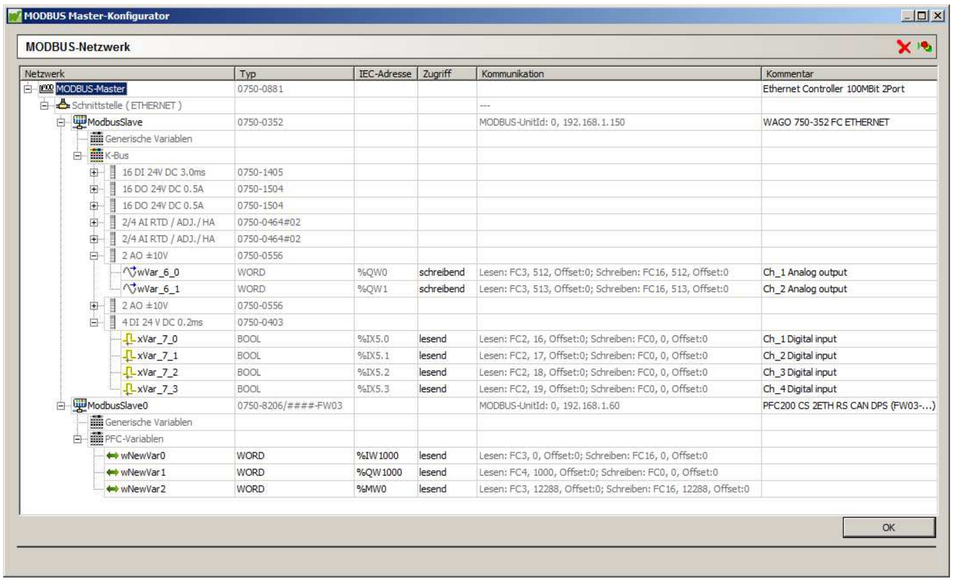
The programmable fieldbus controller can be used as a MODBUS master and MODBUS slave. The CODESYS programming environment can be used to configure the MODBUS master functionality. Generally, there are two approaches as described in the following sections.

4.1 MODBUS Master Configurator

The MODBUS Master Configurator is an extension application of the CODESYS 2.3 programming environment and is used directly in the programming environment. The MODBUS Master Configurator is part of the WAGO-I/O-*PRO* software (759-333) version 2.3.9.40 or higher.

The MODBUS Master Configurator makes configuring a MODBUS network simple. The documentation for the MODBUS Master Configurator contains a corresponding compatibility list.

The dialog streamlines network creation in the MODBUS Master Configurator.
An node scan can be used to configure all MODBUS-enabled WAGO devices in the network. The connected I/O modules are identified and all data points determined.



Netzwerk	Typ	IEC-Adresse	Zugriff	Kommunikation	Kommentar
MODBUS-Master	0750-0881				Ethernet Controller 100MBit 2Port
Schnittstelle (ETHERNET)				---	
ModbusSlave	0750-0352			MODBUS-UnitId: 0, 192.168.1.150	WAGO 750-352 FC ETHERNET
Generische Variablen					
K-Bus					
16 DI 24V DC 3.0ms	0750-1405				
16 DO 24V DC 0.5A	0750-1504				
16 DO 24V DC 0.5A	0750-1504				
2/4 AI RTD / ADJ./ HA	0750-0464#02				
2/4 AI RTD / ADJ./ HA	0750-0464#02				
2 AO ±10V	0750-0556				
wVar_6_0	WORD	%QW0	schreibend	Lesen: FC3, 512, Offset:0; Schreiben: FC16, 512, Offset:0	Ch_1 Analog output
wVar_6_1	WORD	%QW1	schreibend	Lesen: FC3, 513, Offset:0; Schreiben: FC16, 513, Offset:0	Ch_2 Analog output
2 AO ±10V	0750-0556				
4 DI 24 V DC 0.2ms	0750-0403				
xVar_7_0	BOOL	%IX5.0	lesend	Lesen: FC2, 16, Offset:0; Schreiben: FC0, 0, Offset:0	Ch_1 Digital input
xVar_7_1	BOOL	%IX5.1	lesend	Lesen: FC2, 17, Offset:0; Schreiben: FC0, 0, Offset:0	Ch_2 Digital input
xVar_7_2	BOOL	%IX5.2	lesend	Lesen: FC2, 18, Offset:0; Schreiben: FC0, 0, Offset:0	Ch_3 Digital input
xVar_7_3	BOOL	%IX5.3	lesend	Lesen: FC2, 19, Offset:0; Schreiben: FC0, 0, Offset:0	Ch_4 Digital input
ModbusSlave0	0750-8206/####-FW03			MODBUS-UnitId: 0, 192.168.1.60	PFC200 CS 2ETH RS CAN DPS (FW03-...)
Generische Variablen					
PFC-Variablen					
wNewVar0	WORD	%IW1000	lesend	Lesen: FC3, 0, Offset:0; Schreiben: FC16, 0, Offset:0	
wNewVar1	WORD	%QW1000	lesend	Lesen: FC4, 1000, Offset:0; Schreiben: FC0, 0, Offset:0	
wNewVar2	WORD	%MW0	lesend	Lesen: FC3, 12288, Offset:0; Schreiben: FC16, 12288, Offset:0	

For a detailed description of the MODBUS Master Configurator, please refer to the WAGO homepage:

http://www.wago.de/download.esm?file=%5Cdownload%5C00286500_0.pdf&name=m07590333_xxxxxxxx_0_en.pdf

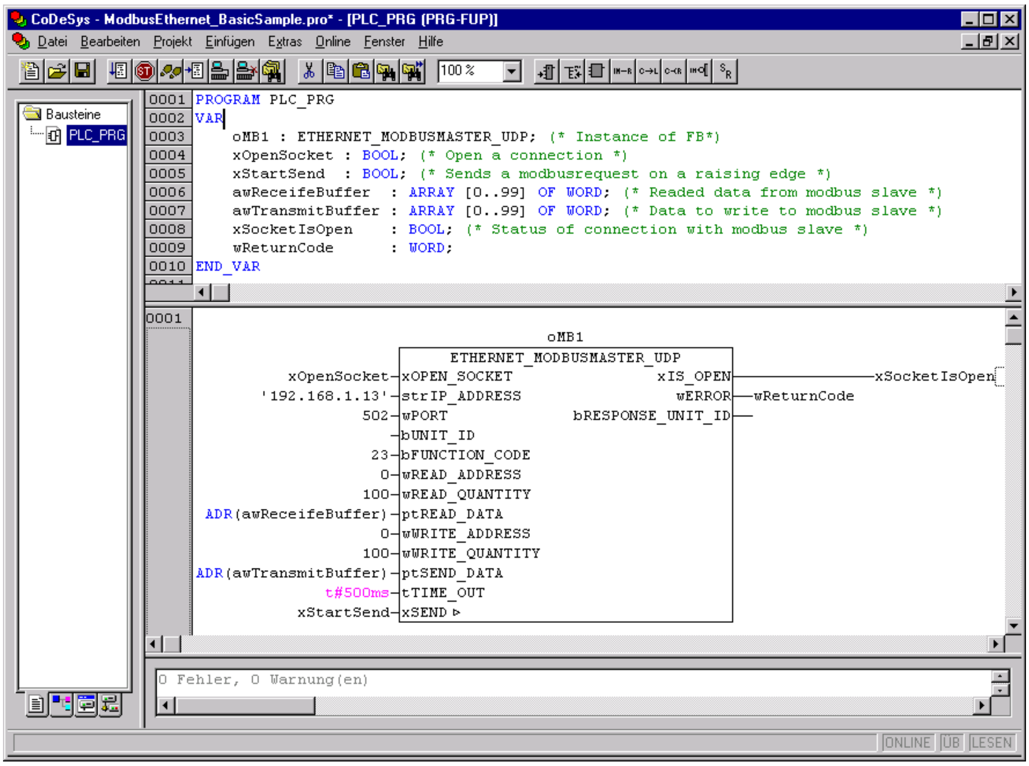
4.2 CODESYS Libraries

Pre-configured libraries can also be used to implement the MODBUS master functionality. This method offers fewer convenience functions (no node scan possible), but provides the library functions that the MODBUS Master Configurator does not offer (e.g., function code 23).

The following libraries are available for the programmable fieldbus controllers:

Library	Depending on the system library	Supported target system
WagoLibMODBUS_IP_01.lib	SysLibSockets.lib	750-841 750-88x 750-820x 758-87x
MODBUSEthernet_04.lib	Ethernet.lib	750-841 750-842 750-843

The following minimal project uses function code “23” to write 100 words to the MODBUS slave with IP address “192.168.1.13”. In addition, 100 words from the MODBUS slave are read in the same telegram.



```
PROGRAM PLC_PRG
VAR
     oMB1 : ETHERNET_MODBUSMASTER_UDP; (* Instance of FB*)
     xOpenSocket : BOOL; (* Open a connection *)
     xStartSend  : BOOL; (* Sends a modbusrequest on a raising edge *)
     awReceifeBuffer   : ARRAY [0..99] OF WORD; (* Readed data from modbus slave *)
     awTransmitBuffer  : ARRAY [0..99] OF WORD; (* Data to write to modbus slave *)
     xSocketIsOpen     : BOOL; (* Status of connection with modbus slave *)
     wReturnCode       : WORD;
END_VAR
```

```
                                              oMB1
                             ETHERNET_MODBUSMASTER_UDP
          xOpenSocket-xOPEN_SOCKET                  xIS_OPEN-----------xSocketIsOpen
       '192.168.1.13'-strIP_ADDRESS                   wERROR--wReturnCode
                  502-wPORT                bRESPONSE_UNIT_ID--
                     -bUNIT_ID
                   23-bFUNCTION_CODE
                    0-wREAD_ADDRESS
                  100-wREAD_QUANTITY
 ADR(awReceifeBuffer)-ptREAD_DATA
                    0-wWRITE_ADDRESS
                  100-wWRITE_QUANTITY
ADR(awTransmitBuffer)-ptSEND_DATA
              t#500ms-tTIME_OUT
           xStartSend-xSEND ▹
```

0 Fehler, 0 Warnung(en)

Operating instructions for the example can be found in the example program.

5 PC Application as MODBUS Master

To create Windows applications, WAGO makes available two MODBUS Master implementations under item number 759-312.

– WagoModbusNet.cs, a C# .NET code class.

– MBT.dll, a procedural 32-bit DLL
(does not run on 64-bit operating systems)

While “WagoModbusNet” requires .NET Framework 2 or higher, “MBT.dll” can be used in nearly all programming languages.

5.1 WagoModbusNet

With “WagoModbusNet” 759-312, WAGO makes a C# coded .NET code class library available that encapsulates the function of a MODBUS master.

“WagoModbusNet” can be used in all versions higher than or equal to VisualStudio2005.
In C# projects, just add the “WagoModbusNet.cs” to the project.
For all other .NET languages such as “vb.net”, add a reference to the “WagoModbusNet.dll” to the project.

The following classes are available:
- wmnModbusMasterTCP
- wmnModbusMasterUDP
- wmnModbusMaster-RTU(Serial)
- wmnModbusMaster-ASCII(Serial).

The commands FC1, FC2, FC3, FC4, FC5, FC6, FC11, FC15, FC16, FC22, and FC23 are supported by the Open MODBUS TCP protocol V1.3.

You can find examples on the CD with item No. 759-312.

5.2 MBT.dll

With “MBT.dll” 759-312, WAGO provides a procedural DLL that implements the MODBUS TCP protocol.

The “MBT.dll” supports 32-bit operating systems such as Windows 95, Windows 98, Windows NT 4.0 (abSP5), Windows 2000 and Windows XP. Windows 95 requires an update to “Windows Socket 2.0”.

TCP or UDP can be selected as the transport protocol. WAGO recommends using UDP as the transport protocol since it allows for better timeout handling.

The “MBT.dll” can be used in many programming languages. On the included CD-ROM, you can find examples for VBA(Excel), VB6, LabView, C, VC++ 6, Delphi, vb.net and C#.

The commands FC1, FC2, FC3, FC4, FC7, FC15 and FC16 are supported by Open MODBUS TCP protocol V1.3.

The “MBT.dll” does not have to be installed or registered.
You just have to copy the DLL to the Windows default directory “\system32”. If you select a different directory, you have to add the path to the “MBT.dll” for the environment variables in the Windows control panel.

The MBT.dll provides the following functions:
MBTInit(); MBTExit()
MBTConnect(); MBTDisconnect()
MBTReadRegisters();MBTWriteRegisters()
MBTReadCoils();MBTWriteCoils()
MBTSwapWord(); MBTSwapDWord()

All functions of the MBT library have return values in HRESULT format. The functions of the socket APIs do not return values in this format. The MBT library converts these return values using the HRESULT_FROM_WIN32. macro. In the following description, this is identified by “HR from”.

In a program, “MBTInit()” should be called once. The function provides the required resources and initializes the DLL. “MBTConnect()” establishes a connection to a remote MODBUS slave (server). Data is exchanged via the functions “MBTWriteRegisters()”, “MBTReadRegisters()”, “MBTWriteCoils()” and “MBTReadCoils()”. Once all data is exchanged, the connection is disabled using the function “MBTDisconnect()”. Now, the connection can be established again or the program can be terminated. To make sure that the resources are released, the function “MBTExit()” should be executed once after program termination and also after program abort.

6 Appendix A: The MODBUS Protocol

The MODBUS protocol developed in 1979 is an open “Internet Draft Standard” of the IETF (Internet Engineering Task Force) today.
The tried and tested original MODBUS services as well as the object model have not been modified and have been adopted by TCP/IP as a transmission medium. Communication occurs via the well-known port 502, which is reserved for MODBUS.
The MODBUS family thus consists of the classic MODBUS RTU and MODBUS ASCII (asynchronous transmission via RS-232 or RS-485) and MODBUS TCP (connection-oriented client-server communication via ETHERNET).

WAGO Kontakttechnik GmbH & Co. KG expands the MODBUS family with MODBUS/UDP. This version uses a connectionless, asynchronous client server communication via ETHERNET.
MODBUS/UDP solves a problem, which arises when a MODBUS slave (server) is not available (e.g., interrupted power supply). MODBUS TCP provides retransmission mechanisms of the TCP stack, which causes the MODBUS master(client) to realize very late that the remote station is not available.
With MODBUS UDP, the time-out monitoring is done on the application layer (OSI layer 7) and can hence react immediately to a missing response telegram. For this reason, we recommend using the MODBUS UDP version, if possible.

MODBUS communication occurs via service calls. The MODBUS master (client) sends a request telegram to Port 502 of the MODBUS slave (server). The MODBUS slave returns the result of the service call in a response telegram to the MODBUS master.

The most important elements of a MODBUS telegram are:

Item	Description
Function Code (FC)	Service identification: Read or write operation in bits or WORDs
Address	Operation start address
Count	Number of bits or WORDs (bytes) depending on the service
[Data]	Process data

First, the service ID or function code (FC) determines whether it is a read or a write operation, then it determines the basic data type to be used with the operation. Therefore, the meaning of the parameters “Address” and “Count” is also dependent on the function code. Consequently, “Address :=3” can stand for the fourth bit or word in the input or output process image.

The MODBUS protocol is based on the following basic data types:

Data Type	Length	Description
Discrete Inputs	1 bit	Digital inputs
Coils	1 bit	Digital Outputs
Input Register	16 bits	Analog input data
Holding Register	16 bits	Analog output data

One or more function codes are defined for every basic data type.

FC	Name	Description
FC1	Read coils	Re-read several digital outputs
FC2	Read inputs discrete	Read several digital inputs
FC3	Read holding registers	Read several analog inputs (and outputs)
FC4	Read input registers	Read several analog inputs (and outputs)
FC5	Write coil	Write a single digital output
FC6	Write single register	Write a single analog output
FC11	Get comm event counter	Communication event counter
FC15	Force multiple coils	Write several digital outputs
FC16	Write multiple registers	Writing several analog outputs
FC23	Read/write multiple registers	Write/read operation to analog inputs/outputs

Although digital and analog process data from WAGO fieldbus couplers and programmable fieldbus controllers is combined in a process image, the first digital output or input is always reached with the “digital” MODBUS services at address 0. That means that the “digital” MODBUS services ignore the complex I/O modules.
On the other hand, however, the status of the digital inputs and outputs can also be determined or changed via the so-called “Register” service.

All WAGO fieldbus couplers and programmable fieldbus controllers do not distinguish between the function codes FC1 and FC2.
Both MODBUS services use the same implementation and allow access to digital input and output modules, as well as the flag area.

All WAGO fieldbus couplers and programmable fieldbus controllers do not distinguish between the function codes FC3 and FC4. Both MODBUS services use the same implementation.

The maximum telegram length is determined by the “BYTE” data type, which can only accept values between 0 and 255. Depending on the MODBUS service, approx. 120 WORDs of user data can be transported.

Please find device-related comparisons of MODBUS addresses and IEC addresses in Appendix B.

6.1 FC1 (Read Coils)

This function reads the content of several input or output bits.

Structure of the request

Example: A request to read bits 0 to 7.

Byte	Field name	Example
Byte 0, 1	Transaction identifier	0x0000
Byte 2, 3	Protocol identifier	0x0000
Byte 4, 5	Length field	0x0006
Byte 6	Unit identifier	0x01 not used
Byte 7	MODBUS function code	0x01
Byte 8, 9	Reference number	0x0000
Byte 10, 11	Bit count	0x0008

Structure of the response

The current values of the bits are entered into the data field. Value 1 = ON, value 0 = OFF. The least significant bit of the first data byte contains the first bit of the request. The other bits follow in ascending order. If the number of inputs is not a multiple of 8, the remaining bits of the last data byte are filled up with zeros.

Byte	Field name	Example
.....		
Byte 7	MODBUS function code	0x01
Byte 8	Byte count	0x01
Byte 9	Bit values	0x12

The status of inputs 0 to 7 is displayed as byte value 0x12 or binary 0001 0010. Input 7 is the most significant bit of this byte, input 0 is the least significant bit. The assignment of 7 to 0 is hence
OFF-OFF-OFF-ON-OFF-OFF-ON-OFF.

Bit:	0	0	0	1	0	0	1	0
Coil:	7	6	5	4	3	2	1	0

Structure of the exception

Byte	Field name	Example
.....		
Byte 7	MODBUS function code	0x81
Byte 8	Exception code	0x01 or 0x02

6.2 FC2 (Read Input Discretes)

This function reads the content of several input bits (digital inputs).

Structure of the request

The request determines the start address and the number of bits to be read. Example: a request to read the bits 0 to 7.

Byte	Field name	Example
Byte 0, 1	Transaction identifier	0x0000
Byte 2, 3	Protocol identifier	0x0000
Byte 4, 5	Length field	0x0006
Byte 6	Unit identifier	0x01 not used
Byte 7	MODBUS function code	0x02
Byte 8, 9	Reference number	0x0000
Byte 10, 11	Bit count	0x0008

Structure of the response

The current values of the bits are entered into the data field. Value 1 = ON, value 0 = OFF. The least significant bit of the first data byte contains the first bit of the request. The other bits follow in ascending order. If the number of inputs is not a multiple of 8, the remaining bits of the last data byte are filled up with zeros.

Byte	Field name	Example
.....		
Byte 7	MODBUS function code	0x02
Byte 8	Byte count	0x01
Byte 9	Bit values	0x12

The state of the inputs 7 to 0 is indicated as byte value 0x12 or binary 0001 0010. Input 7 is the most significant bit of this byte, input 0 is the least significant bit. The assignment of 7 to 0 is hence
OFF-OFF-OFF-ON-OFF-OFF-ON-OFF.

Bit:	0	0	0	1	0	0	1	0
Coil:	7	6	5	4	3	2	1	0

Structure of the exception

Byte	Field name	Example
.....		
Byte 7	MODBUS function code	0x82
Byte 8	Exception code	0x01 or 0x02

6.3 FC3 (Read multiple registers)

This function reads a number of input words (also “Input Registers”).

Structure of the request

The request determines the address of the start word (start register) and the number of registers to be read. Addressing starts with 0.

Example: request to read registers 0 and 1.

Byte	Field name	Example
Byte 0, 1	Transaction identifier	0x0000
Byte 2, 3	Protocol identifier	0x0000
Byte 4, 5	Length field	0x0006
Byte 6	Unit identifier	0x01 not used
Byte 7	MODBUS function code	0x03
Byte 8, 9	Reference number	0x0000
Byte 10, 11	Word count	0x0002

Structure of the response

The register data of the response is entered into the registers (2 bytes per register). The first byte contains the more significant bits, the second byte contains the less significant bits.

Byte	Field name	Example
.....		
Byte 7	MODBUS function code	0x03
Byte 8	Byte count	0x04
Byte 9, 10	Value register 0	0x1234
Byte 11, 12	Value register 1	0x2345

The response shows that register 0 contains the value 0x1234 and register 1 contains the value 0x2345.

Structure of the exception

Byte	Field name	Example
.....		
Byte 7	MODBUS function code	0x83
Byte 8	Exception code	0x01 or 0x02

6.4 FC4 (Read input registers)

This function reads a number of input words (also “Input Registers”).

Structure of the request

The request determines the address of the start word (start register) and the number of registers to be read. Addressing starts with 0.

Example: request to read registers 0 and 1.

Byte	Field name	Example
Byte 0, 1	Transaction identifier	0x0000
Byte 2, 3	Protocol identifier	0x0000
Byte 4, 5	Length field	0x0006
Byte 6	Unit identifier	0x01 not used
Byte 7	MODBUS function code	0x04
Byte 8, 9	Reference number	0x0000
Byte 10, 11	Word count	0x0002

Structure of the response

The register data of the response is entered into the registers (2 bytes per register). The first byte contains the more significant bits, the second byte contains the less significant bits.

Byte	Field name	Example
.....		
Byte 7	MODBUS function code	0x04
Byte 8	Byte count	0x04
Byte 9, 10	Value register 0	0x1234
Byte 11, 12	Value register 1	0x2345

The response shows that register 0 contains the value 0x1234 and register 1 contains the value 0x2345.

Structure of the exception

Byte	Field name	Example
.....		
Byte 7	MODBUS function code	0x84
Byte 8	Exception code	0x01 or 0x02

6.5 FC5 (Write Coil)

This function writes a digital output bit.

Structure of the requests

The request determines the address of the output bit. Addressing starts with 0.

Example: Setting the second output bit (address 1).

Byte	Field name	Example
Byte 0, 1	Transaction identifier	0x0000
Byte 2, 3	Protocol identifier	0x0000
Byte 4, 5	Length field	0x0006
Byte 6	Unit identifier	0x01 not used
Byte 7	MODBUS function code	0x05
Byte 8, 9	Reference number	0x0001
Byte 10	ON/OFF	0xFF
Byte 11		0x00

Structure of the response

Byte	Field name	Example
.....		
Byte 7	MODBUS function code	0x05
Byte 8, 9	Reference number	0x0001
Byte 10	Value	0xFF
Byte 11		0x00

Structure of the exception

Byte	Field name	Example
.....		
Byte 7	MODBUS function code	0x85
Byte 8	Exception code	0x01, 0x02 or 0x03

6.6 FC6 (Write single register)

This function writes a value into a single output word (“output register”).

Structure of the request

Addressing starts with 0. The request determines the address of the first output word to be set. The value to be set is determined in the request data field.

Example: Setting the second output channel (address 0) to value 0x1234.

Byte	Field name	Example
Byte 0, 1	Transaction identifier	0x0000
Byte 2, 3	Protocol identifier	0x0000
Byte 4, 5	Length field	0x0006
Byte 6	Unit identifier	0x01 not used
Byte 7	MODBUS function code	0x06
Byte 8, 9	Reference number	0x0001
Byte 10, 11	Register value	0x1234

Structure of the response

The response is an echo of the request.

Byte	Field name	Example
.....		
Byte 7	MODBUS function code	0x06
Byte 8, 9	Reference number	0x0001
Byte 10, 11	Register value	0x1234

Structure of the exception

Byte	Field name	Example
.....		
Byte 7	MODBUS function code	0x85
Byte 8	Exception code	0x01 or 0x02

6.7 FC11 (Get comm event counter)

This function returns a status word and an event counter from the communication register of the controller. The higher level control system can detect by means of this counter if the controller has accurately processed the messages.

Every time a message is processed successfully, the counter counts up. It does not count up after exception responses or counter requests.

Structure of the request

Byte	Field name	Example
Byte 0, 1	Transaction identifier	0x0000
Byte 2, 3	Protocol identifier	0x0000
Byte 4, 5	Length field	0x0002
Byte 6	Unit identifier	0x01 not used
Byte 7	MODBUS function code	0x0B

Structure of the response

The response contains a 2-byte status word and a 2-byte event counter. The status word consists of zeros.

Byte	Field name	Example
.....		
Byte 7	MODBUS function code	0x10
Byte 8, 9	Status	0x0000
Byte 10, 11	Event count	0x0003

The event counter shows that 3 (0x0003) events were counted.

Structure of the exception

Byte	Field name	Example
.....		
Byte 7	MODBUS function code	0x85
Byte 8	Exception code	0x01 or 0x02

6.8 FC15 (Force Multiple Coils)

This function is used to set multiple output bits to 1 or 0. The maximum number is 256 bits.

Structure of the request

The address of the first bit is 0. The request message specifies the bits to be set. The required state (1 or 0) of the bit is determined by the content of the request data field.

In this example, 16 bits are set starting with address 0. The request contains 2 bytes with the value 0xA5F0, i.e., 1010 0101 1111 0000 binary.

The first byte assigns the 0xA5 to address 7 to 0, 0 being the least significant bit. The next byte assigns 0xF0 to address 15 to 8, 8 being the least significant bit.

Byte	Field name	Example
Byte 0, 1	Transaction identifier	0x0000
Byte 2, 3	Protocol identifier	0x0000
Byte 4, 5	Length field	0x0009
Byte 6	Unit identifier	0x01 not used
Byte 7	MODBUS function code	0x0F
Byte 8, 9	Reference number	0x0000
Byte 10, 11	Bit count	0x0010
Byte 12	Byte count	0x02
Byte 13	Data byte1	0xA5
Byte 14	Data byte2	0xF0

Structure of the response

Byte	Field name	Example
.....		
Byte 7	MODBUS function code	0x0F
Byte 8, 9	Reference number	0x0000
Byte 10, 11	Bit count	0x0010

Structure of the exception

Byte	Field name	Example
.....		
Byte 7	MODBUS function code	0x8F
Byte 8	Exception code	0x01 or 0x02

6.9 FC16 (Write multiple registers)

This function writes values to a number of output words (also “output registers”).

Structure of the requests

The address of the first register is 0.

The request message specifies the registers to be set. The data is transmitted as two bytes per register.

Example: The data in the registers 0 and 1 is written.

Byte	Field name	Example
Byte 0, 1	Transaction identifier	0x0000
Byte 2, 3	Protocol identifier	0x0000
Byte 4, 5	Length field	0x000B
Byte 6	Unit identifier	0x01 not used
Byte 7	MODBUS function code	0x10
Byte 8, 9	Reference number	0x0000
Byte 10, 11	Word count	0x0002
Byte 12	Byte count	0x04
Byte 13, 14	Register value 1	0x1234
Byte 15, 16	Register value 2	0x2345

Structure of the response

Byte	Field name	Example
.....		
Byte 7	MODBUS function code	0x10
Byte 8, 9	Reference number	0x0000
Byte 10, 11	Word count	0x0002

Structure of the exception

Byte	Field name	Example
.....		
Byte 7	MODBUS function code	0x85
Byte 8	Exception code	0x01 or 0x02

6.10 FC22 (Mask Write Register)

This function manipulates bits within a register.

Structure of the requests

Byte	Field name	Example
Byte 0, 1	Transaction identifier	0x0000
Byte 2, 3	Protocol identifier	0x0000
Byte 4, 5	length field	0x0002
Byte 6	Unit identifier	0x01 not used
Byte 7	MODBUS function code	0x16
Byte 8-9	Reference number	0x0000
Byte 10-11	AND mask	0x0000
Byte 12-13	OR mask	0xAAAA

Structure of the response

Byte	Field name	Example
.....		
Byte 7	MODBUS function code	0x10
Byte 8-9	Reference number	0x0000
Byte 10-11	AND mask	0x0000
Byte 12-13	OR mask	0xAAAA

Structure of the exception

Byte	Field name	Example
.....		
Byte 7	MODBUS function code	0x85
Byte 8	Exception code	0x01 or 0x02

6.11 FC23 (Read/Write multiple registers)

This function reads register values and writes values to a number of output words (also “output registers”).

Structure of the requests

Example: the data in register 3 are set to 0x0123 and the values 0x1 and 0x0004 are read from the registers 0 and 5678.

Byte	Field name	Example
Byte 0, 1	Transaction identifier	0x0000
Byte 2, 3	Protocol identifier	0x0000
Byte 4, 5	Length field	0x000F
Byte 6	Unit identifier	0x01 not used
Byte 7	MODBUS function code	0x17
Byte 8-9	Reference number for read	0x0000
Byte 10-11	Word count for read (1-125)	0x0002
Byte 12-13	Reference number for write	0x0003
Byte 14-15	Word count for write (1-100)	0x0001
Byte 16	Byte count (B = 2 x word count for write)	0x02
Byte 17-(B+16)	Register values	0x0123

Structure of the response

Byte	Field name	Example
....		
Byte 7	MODBUS function code	0x17
Byte 8	Byte count (B = 2 x word count for read)	0x04
Byte 9-(B+1)	Register values	0x0004 0x5678

Structure of the exception

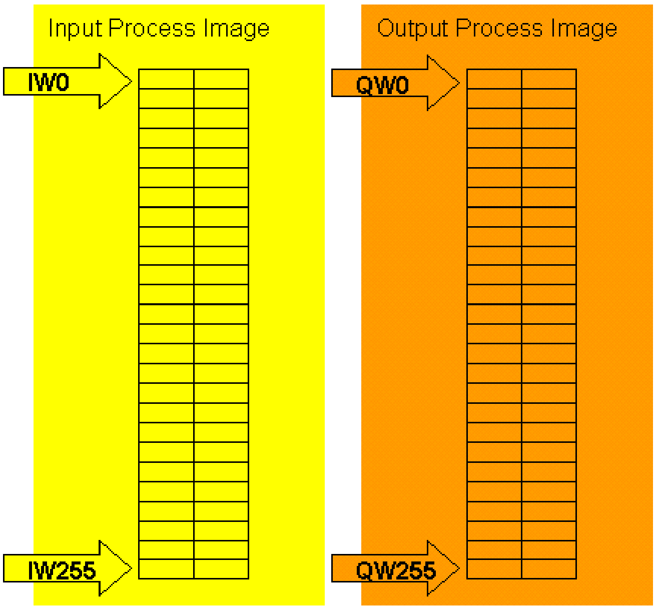
Byte	Field name	Example
.....		
Byte 7	MODBUS function code	0x97
Byte 8	Exception code	0x01 or 0x02

7 Appendix B: Device-specific Process Images

7.1 Fieldbus Coupler 750-342

7.1.1 Process image of the 750-342

The 750-342 can process a maximum of 3 incoming MODBUS TCP connections. The MODBUS connection Watchdog is deactivated when delivered.



In addition to the WAGO basic MODBUS services, the 750-342 also supports the function code FC7 “Read exception status”.

FC	Name	Description
FC1	Read coils	Re-read several digital outputs
FC2	Read inputs discrete	Read several digital inputs
FC3	Read holding registers	Read several analog inputs (and outputs)
FC4	Read input registers	Read several analog inputs (and outputs)
FC5	Write coil	Write a single digital output
FC6	Write single register	Write a single analog output
FC7	Read exception status	Read the first 8 digital outputs
FC11	Get comm event counter	Communication event counter
FC15	Force multiple coils	Write several digital outputs
FC16	Write multiple registers	Writing several analog outputs
FC23	Read/write multiple registers	Read/write operation to analog inputs/outputs

7.1.2 Register Services of the 750-342

Register services are used to determine or change the statuses of complex and digital I/O modules.

7.1.2.1 Read registers with FC3 and FC4:

750-342: MODBUS vs IEC 61131 Addresses for FC3 and FC4

MODBUS Address		IEC 61131 Address	Description
[dec]	[hex]		
0 ... 255	0x0000 ... 0x00FF	%IW0 ... %IW255	Physical input area
256 ... 511	0x0100 ... 0x01FF	-	MODBUS Exception: “Illegal data address”
512 ... 767	0x0200 ... 0x02FF	%QW0 ... %QW255	Physical output area
768 ... 4095	0x0300 ... 0x0FFF	-	MODBUS Exception: “Illegal data address”
4096 ... 12287	0x1000 ... 0x2FFF	-	Configuration register (see manual for details)
12288 ... 65535	0x3000 ... 0xFFFF	-	MODBUS Exception: “Illegal data address”

7.1.2.2 Write registers with FC6 and FC16:

750-342: MODBUS vs IEC 61131 Addresses for FC6 and FC16

MODBUS Address		IEC 61131 Address	Description
[dec]	[hex]		
0 ... 255	0x0000 ... 0x00FF	%QW0 ... %QW255	Physical output area
256 ... 511	0x0100 ... 0x01FF	-	MODBUS Exception: “Illegal data address”
512 ... 767	0x0200 ... 0x02FF	%QW0 ... %QW255	Physical output area
768 ... 4095	0x0300 ... 0x0FFF	-	MODBUS Exception: “Illegal data address”
4096 ... 12287	0x1000 ... 0x2FFF	-	Configuration register (see manual for details)
12288 ... 65535	0x3000 ... 0xFFFF	-	MODBUS Exception: “Illegal data address”

7.1.3 Digital MODBUS services of the 750-342

The digital MODBUS services can only determine or change the state of digital I/O modules. Complex I/O modules are ignored or cannot be accessed.

7.1.3.1 Read coils with FC1 and FC2:

750-342: MODBUS Addresses for FC1 and FC2			
MODBUS Address		Memory Area	Description
[dec]	[hex]		
0 ... 511	0x0000 ... 0x01FF	Physical input area	First 512 digital inputs
512 ... 1023	0x0200 ... 0x03FF	Physical output area	First 512 digital outputs
1024 ... 65535	0x0400 ... 0xFFFF		MODBUS exception: “Illegal data address”

7.1.3.2 Write coils with FC5 and FC15:

750-342: MODBUS Addresses for FC5 and FC15			
MODBUS Address		Memory Area	Description
[dec]	[hex]		
0 ... 511	0x0000 ... 0x01FF	Physical output area	max 512 digital outputs
512 ... 1023	0x0200 ... 0x03FF	Physical output area	max. 512 digital outputs
1024 ... 65535	0x0400 ... 0xFFFF		MODBUS exception: “Illegal data address”

7.1.4 MODBUS Configuration Registers for the 750-342

The configuration registers make it possible to determine and in part change the properties of the 750-342.

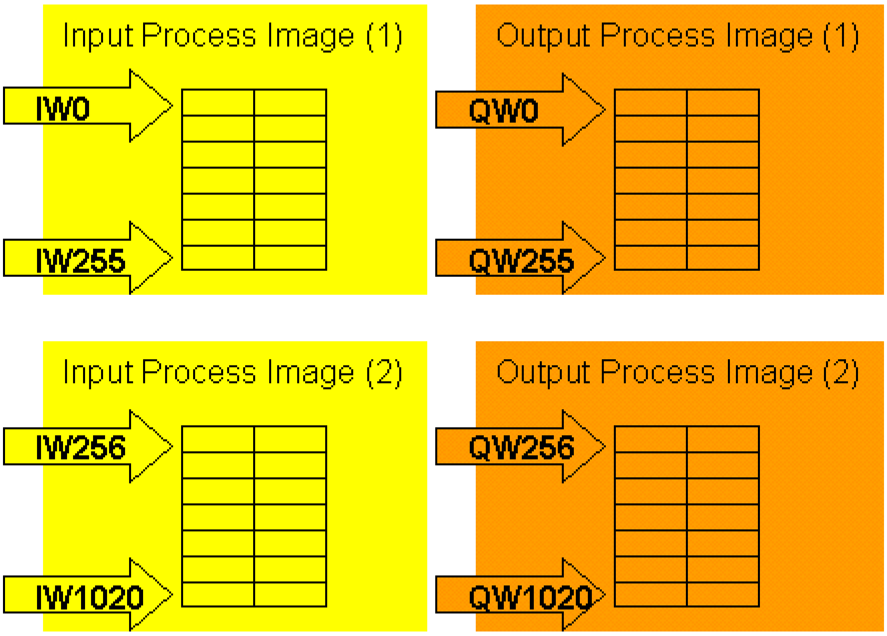
750-342: MODBUS Configuration Register for FC3, FC4, FC6 and FC16				
MODBUS Address		**Length**	**Access**	**Description**
[dec]	**[hex]**	**[Word]**		
4096	0x1000	1	R/W	MODBUS Watchdog Time (multiple of 100 ms)
4097	0x1001	1	R/W	MODBUS Watchdog coding screen 1-16
4098	0x1002	1	R/W	MODBUS Watchdog coding screen 17-32
4099	0x1003	1	R/W	MODBUS Watchdog-Trigger
4100	0x1004	1	R	Minimum trigger time
4101	0x1005	1	R/W	Stop MODBUS Watchdog (0xAAAA and 0x5555)
4102	0x1006	1	R	MODBUS Watchdog status
4103	0x1007	1	R/W	MODBUS Watchdog restart (0x0001)
4104	0x1008	1	R/W	Stop MODBUS Watchdog (0x55AA or 0xAA55)
4105	0x1009	1	R/W	Close MODBUS and HTTP port after time-out
4106	0x100A	1	R/W	Start MODBUS Watchdog in “Modicon Mode”
4107	0x100B	1	W	Save MODBUS Watchdog parameters
4128	0x1020	1	R	LED error code
4129	0x1021	1	R	LED error argument
4130	0x1022	1	R	Number of analog outputs in the process image [Bit]
4131	0x1023	1	R	Number of analog inputs in the process image [Bit]
4132	0x1024	1	R	Number of digital outputs in the process image [Bit]
4133	0x1025	1	R	Number of digital inputs in the process image [Bit]
4135	0x1027	1	R	Execute internal bus cycle
4136	0x1028	1	R/W	IP configuration: BootP(1) or FIX(0)
4137	0x1029	18	R	MODBUS TCP statistics
4144	0x1030	1	R/W	Activate MODBUS connection monitoring
4145	0x1031	3	R	MAC ID or ETHERNET interface
8192	0x2000	1	R	0x0000 (Constant)
8193	0x2001	1	R	0xFFFF (Constant)
8194	0x2002	1	R	0x1234 (Constant)
8195	0x2003	1	R	0xAAAA (Constant)
8196	0x2004	1	R	0x5555 (Constant)
8197	0x2005	1	R	0x7FFF (Constant)
8198	0x2006	1	R	0x8000 (Constant)
8199	0x2007	1	R	0x3FFF (Constant)
8200	0x2008	1	R	0x4000 (Constant)
8208	0x2010	1	R	Firmware release
8209	0x2011	1	R	Series code (750)
8210	0x2012	1	R	Device code (342)
8211	0x2013	1	R	Specific firmware version (0xFFFF)
8212	0x2014	1	R	Specific firmware version (0xFFFF)
8224	0x2020	1 .. 125	R	Short device description
8225	0x2021	8	R	Compile time of the firmware version

750-342: MODBUS Configuration Register for FC3, FC4, FC6 and FC16				
MODBUS Address		Length	Access	Description
[dec]	[hex]	[Word]		
8226	0x2022	8	R	Compile date of the firmware version
8227	0x2023	32	R	Version of the Firmware loader (FWL)
8240	0x2030	65	R	Description of connected IO modules: 0-64
8245	0x2035	1	R/W	Setting process image (Table 0 register 3)
8246	0x2036	1 .. 17	R	Diagnostic information device
8256	0x2040	1	W	Software reset (write 0x55AA or 0xAA55)
8260	0x2044	1	W	Delete MODBUS Configuration file (write 0x55AA)

7.2 Fieldbus Coupler 750-352

7.2.1 Process image of the 750-352

The 750-352 can process a maximum of 15 incoming MODBUS TCP connections. The MODBUS connection Watchdog is deactivated when delivered.



In addition to the WAGO basic MODBUS services, the
750-352 also supports the function code FC22 “Mask write”.

FC	Name	Description
FC1	Read coils	Re-read several digital outputs
FC2	Read inputs discrete	Read several digital inputs
FC3	Read holding registers	Read several analog inputs (and outputs)
FC4	Read input registers	Read several analog inputs (and outputs)
FC5	Write coil	Write a single digital output
FC6	Write single register	Write a single analog output
FC11	Get comm event counter	Communication event counter
FC15	Force multiple coils	Write several digital outputs
FC16	Write multiple registers	Writing several analog outputs
FC22	Mask write	Manipulation of single bits in a register
FC23	Read/write multiple registers	Read/write operation to analog inputs/outputs

7.2.2 Register Services of the 750-352

7.2.2.1 Read registers with FC3 and FC4:

750-352: MODBUS vs. IEC 61131 Addresses for FC3 and FC4			
MODBUS Address		IEC 61131 Address	Description
[dec]	[hex]		
0 ... 255	0x0000 ... 0x00FF	%IW0 ... %IW255	Physical input area (1) First 256 words of physical input data
256 ... 511	0x0100 ... 0x01FF	-	MODBUS exception: “Illegal data address”
512 ... 767	0x0200 ... 0x02FF	%QW0 ... %QW255	Physical output area (1) First 256 words of physical output data
768 ... 4095	0x0300 ... 0x0FFF	-	MODBUS Exception: “ Illegal data address”
4096 ... 12287	0x1000 ... 0x2FFF	-	Configuration register (see manual for details)
12288 ... 24575	0x3000 ... 0x5FFF	-	MODBUS Exception: “ Illegal data address”
24576 ... 25339	0x6000 ... 0x62FB	%IW256 ... %IW1020	Physical input area (2) Additional 764 words physical input data
25340 ... 28671	0x62FC ... 0x6FFF	-	MODBUS Exception: “ Illegal data address”
28672 ... 29435	0x7000 ... 0x72FB	%QW256 ... %QW1020	Physical output area (2) Additional 764 words physical output data
29436 ... 65535	0x72FC ... 0xFFFF	-	MODBUS Exception: “ Illegal data address”

7.2.2.2 Write registers with FC6 and FC16:

750-341: MODBUS vs. IEC 61131 Addresses for FC6 and FC16			
MODBUS Address		IEC 61131 Address	Description
[dec]	[hex]		
0 ... 255	0x0000 ... 0x00FF	%QW0 ... %QW255	Physical output area (1) First 256 words of physical output data
256 ... 511	0x0100 ... 0x01FF	-	MODBUS exception: “Illegal data address”
512 ... 767	0x0200 ... 0x02FF	%QW0 ... %QW255	Physical output area (1) First 256 words of physical output data
768 ... 4095	0x0300 ... 0x0FFF	-	MODBUS Exception: “ Illegal data address”
4096 ... 12287	0x1000 ... 0x2FFF	-	Configuration register (see manual for details)
12288 ... 24575	0x3000 ... 0x5FFF	-	MODBUS Exception: “ Illegal data address”
24576 ... 25339	0x6000 ... 0x62FB	%QW256 ... %QW1020	Physical output area (2) Additional 764 words physical output data
25340 ... 28671	0x62FC ... 0x6FFF	-	MODBUS Exception: “ Illegal data address”
28672 ... 29435	0x7000 ... 0x72FB	%QW256 ... %QW1020	Physical output area (2) Additional 764 words physical output data
29437 ... 65535	0x72FC ... 0xFFFF	-	MODBUS Exception: “ Illegal data address”

7.2.3 Digital MODBUS services of the 750-352

The digital MODBUS services can only determine or change the state of digital I/O modules. Complex I/O modules are ignored or cannot be accessed.

7.2.3.1 Read coils with FC1 and FC2:

750-341: MODBUS Addresses for FC1 and FC2			
MODBUS Address		Memory Area	Description
[dec]	[hex]		
0 ... 511	0x0000 ... 0x01FF	Physical input area (1)	First 512 digital inputs
512 ... 1023	0x0200 ... 0x03FF	Physical output area (1)	First 512 digital outputs
1024 ... 32767	0x0400 ... 0x7FFF	-	MODBUS Exception: “Illegal data address”
32768 ... 34295	0x8000 ... 0x85F7	Physical input area (2)	Starts with the 513th and ends with the 2039th digital input
34296 ... 36863	0x85F8 ... 0x8FFF		MODBUS exception: “Illegal data address”
36864 ... 38391	0x9000 ... 0x95F7	Physical output area (2)	Starts with the 513th and ends with the 2039th digital output
38392 ... 65535	0x95F8 ... 0xFFFF		MODBUS exception: “Illegal data address”

7.2.3.2 Write coils with FC5 and FC15:

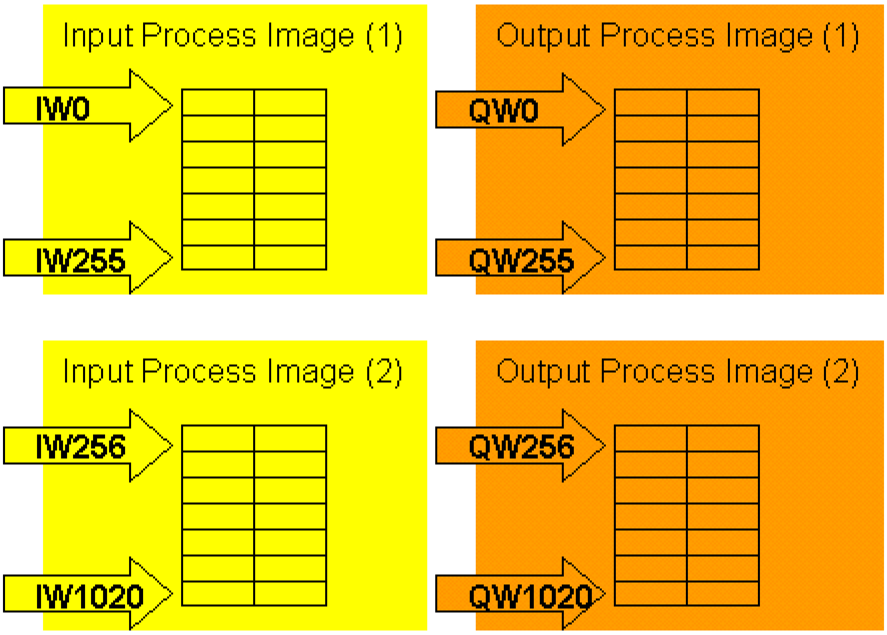
750-341: MODBUS Addresses for FC1 and FC2			
MODBUS Address		Memory Area	Description
[dec]	[hex]		
0 ... 511	0x0000 ... 0x01FF	Physical output area (1)	First 512 digital outputs
512 ... 1023	0x0200 ... 0x03FF	Physical output area (1)	First 512 digital outputs
1024 ... 32767	0x0400 ... 0x7FFF	-	MODBUS Exception: “Illegal data address”
32768 ... 34295	0x8000 ... 0x85F7	Physical output area (2)	Starts with the 513th and ends with the 2039th digital output
34296 ... 36863	0x85F8 ... 0x8FFF		MODBUS exception: “Illegal data address”
36864 ... 38391	0x9000 ... 0x95F7	Physical output area (2)	Starts with the 513th and ends with the 2039th digital output
38392 ... 65535	0x95F8 ... 0xFFFF		MODBUS exception: “Illegal data address”

7.3 Fieldbus Coupler 767-1301

7.3.1 Process Image of 767-1301

The 767-1301 can process a maximum of 15 incoming MODBUS TCP connections. The MODBUS connection Watchdog is deactivated when delivered.

The 767-1301 has two operating modes for MODBUS: V1 and V2.
In the default state, V1 is active, which behaves similar to the MODBUS implementation in the 750 series. For compatibility reasons, V2 is included with firmware Release 1 and can be activated via the Web-Based Management as required.



FC	Name	Description
FC1	Read coils	Re-read several digital outputs
FC2	Read inputs discrete	Read several digital inputs
FC3	Read holding registers	Read several analog inputs (and outputs)
FC4	Read input registers	Read several analog inputs (and outputs)
FC5	Write coil	Write a single digital output
FC6	Write single register	Write a single analog output
FC11	Get comm event counter	Communication event counter
FC15	Force multiple coils	Write several digital outputs
FC16	Write multiple registers	Writing several analog outputs
FC22	Mask write	Manipulation of single bits in a register
FC23	Read/write multiple registers	Read/write operation to analog inputs/outputs

7.3.2 Register Services of 767-1301

7.3.2.1 Read registers with FC3, FC4 and FC23:

767-1301: MODBUS vs. IEC 61131 Addresses for FC3, FC4 and FC23			
MODBUS Address		IEC 61131	Description
[dec]	[hex]	Address	
0 ... 255	0x0000 ... 0x00FF	%IW0 ... %IW255	Physical input area (1) First 256 words of physical input data
256 ... 511	0x0100 ... 0x01FF	-	MODBUS exception: “Illegal data address”
512 ... 767	0x0200 ... 0x02FF	%QW0 ... %QW255	Physical output area (1) First 256 words of physical output data
768 ... 4095	0x0300 ... 0x0FFF	-	MODBUS Exception: “ Illegal data address”
4096 ... 12287	0x1000 ... 0x2FFF	-	Configuration register (see manual for details)
12288 ... 24575	0x3000 ... 0x5FFF	-	MODBUS Exception: “ Illegal data address”
24576 ... 25339	0x6000 ... 0x62FB	%IW256 ... %IW1020	Physical input area (2) Additional 764 words physical input data
25340 ... 28671	0x62FC ... 0x6FFF	-	MODBUS Exception: “ Illegal data address”
28672 ... 29435	0x7000 ... 0x72FB	%QW256 ... %QW1020	Physical output area (2) Additional 764 words physical output data
29436 ... 65535	0x72FC ... 0xFFFF	-	MODBUS Exception: “ Illegal data address”

7.3.2.2 Write registers with FC6, FC16, FC22, FC23:

767-1301: MODBUS vs. IEC 61131 Addresses for FC6, FC16, FC22, FC23			
MODBUS Address		IEC 61131	Description
[dec]	[hex]	Address	
0 ... 255	0x0000 ... 0x00FF	%QW0 ... %QW255	Physical output area (1) First 256 words of physical output data
256 ... 511	0x0100 ... 0x01FF	-	MODBUS exception: “Illegal data address”
512 ... 767	0x0200 ... 0x02FF	%QW0 ... %QW255	Physical output area (1) First 256 words of physical output data
768 ... 4095	0x0300 ... 0x0FFF	-	MODBUS Exception: “ Illegal data address”
4096 ... 12287	0x1000 ... 0x2FFF	-	Configuration register (see manual for details)
12288 ... 24575	0x3000 ... 0x5FFF	-	MODBUS Exception: “ Illegal data address”
24576 ... 25339	0x6000 ... 0x62FB	%QW256 ... %QW1020	Physical output area (2) Additional 764 words physical output data
25340 ... 28671	0x62FC ... 0x6FFF	-	MODBUS Exception: “ Illegal data address”
28672 ... 29435	0x7000 ... 0x72FB	%QW256 ... %QW1020	Physical output area (2) Additional 764 words physical output data
29437 ... 65535	0x72FC ... 0xFFFF	-	MODBUS Exception: “ Illegal data address”

7.3.3 Digital MODBUS services of 767-1301

The digital MODBUS services can only determine or change the state of digital I/O modules. Complex I/O modules are ignored or cannot be accessed.

7.3.3.1 Read coils with FC1 and FC2:

767-1301: MODBUS Addresses for FC1 and FC2			
MODBUS Address		Memory Area	Description
[dec]	[hex]		
0 ... 511	0x0000 ... 0x01FF	Physical input area (1)	First 512 digital inputs
512 ... 1023	0x0200 ... 0x03FF	Physical output area (1)	First 512 digital outputs
1024 ... 32767	0x0400 ... 0x7FFF	-	MODBUS Exception: “Illegal data address”
32768 ... 34295	0x8000 ... 0x85F7	Physical input area (2)	Starts with the 513th and ends with the 2039th digital input
34296 ... 36863	0x85F8 ... 0x8FFF		MODBUS exception: “Illegal data address”
36864 ... 38391	0x9000 ... 0x95F7	Physical output area (2)	Starts with the 513th and ends with the 2039th digital output
38392 ... 65535	0x95F8 ... 0xFFFF		MODBUS exception: “Illegal data address”

7.3.3.2 Write coils with FC5 and FC15:

767-1301: MODBUS Addresses for FC1 and FC2			
MODBUS Address		Memory Area	Description
[dec]	[hex]		
0 ... 511	0x0000 ... 0x01FF	Physical output area (1)	First 512 digital outputs
512 ... 1023	0x0200 ... 0x03FF	Physical output area (1)	First 512 digital outputs
1024 ... 32767	0x0400 ... 0x7FFF	-	MODBUS Exception: “Illegal data address”
32768 ... 34295	0x8000 ... 0x85F7	Physical output area (2)	Starts with the 513th and ends with the 2039th digital output
34296 ... 36863	0x85F8 ... 0x8FFF		MODBUS exception: “Illegal data address”
36864 ... 38391	0x9000 ... 0x95F7	Physical output area (2)	Starts with the 513th and ends with the 2039th digital output
38392 ... 65535	0x95F8 ... 0xFFFF		MODBUS exception: “Illegal data address”

7.3.4 MODBUS configuration register of 767-1301

The configuration registers make it possible to determine and in part change the properties of the 750-1301.

767-1301: MODBUS Configuration Register for FC3, FC4, FC6 and FC16				
MODBUS Address		Length	Access	Description
[dec]	[hex]	[Word]		
4096	0x1000	1	R/W	MODBUS Watchdog Time (multiple of 100 ms)
4097	0x1001	1	R/W	MODBUS Watchdog coding screen 1-16
4098	0x1002	1	R/W	MODBUS Watchdog coding screen 17-32
4099	0x1003	1	R/W	MODBUS Watchdog-Trigger
4100	0x1004	1	R	Minimum trigger time
4101	0x1005	1	R/W	Stop MODBUS Watchdog (0xAAAA and 0x5555)
4102	0x1006	1	R	MODBUS Watchdog status
4103	0x1007	1	R/W	MODBUS Watchdog restart (0x0001)
4104	0x1008	1	R/W	Stop MODBUS Watchdog (0x55AA or 0xAA55)
4105	0x1009	1	R/W	Close MODBUS and HTTP port after time-out
4106	0x100A	1	R/W	Start MODBUS Watchdog in “Modicon Mode”
4107	0x100B	1	W	Save MODBUS Watchdog parameters
4128	0x1020	1 - 2	R	LED error code, LED error argument
4129	0x1021	1	R	LED error argument
4130	0x1022	1 - 4	R	Number of analog outputs in the process image [Bit]
4131	0x1023	1 - 3	R	Number of analog inputs in the process image [Bit]
4132	0x1024	1 - 2	R	Number of digital outputs in the process image [Bit]
4133	0x1025	1	R	Number of digital inputs in the process image [Bit]
4136	0x1028	1	R/W	IP configuration: BootP(1), DHCP(2) or FIX(4)
4137	0x1029	9	R	MODBUS TCP statistics
4138	0x102A	1	R	Number of established MODBUS TCP connections
4144	0x1030	1	R/W	MODBUS TCP time-out (multiple of 100 ms)
4145	0x1031	3	R	MAC ID or ETHERNET interface
8192	0x2000	1	R	0x0000 (Constant)
8193	0x2001	1	R	0xFFFF (Constant)
8194	0x2002	1	R	0x1234 (Constant)
8195	0x2003	1	R	0xAAAA (Constant)
8196	0x2004	1	R	0x5555 (Constant)
8197	0x2005	1	R	0x7FFF (Constant)
8198	0x2006	1	R	0x8000 (Constant)
8199	0x2007	1	R	0x3FFF (Constant)
8200	0x2008	1	R	0x4000 (Constant)
8208	0x2010	1	R	Firmware index
8209	0x2011	1	R	Series designation (767)
8210	0x2012	1	R	Device designation (1301)
8211	0x2013	1	R	Major firmware version
8212	0x2014	1	R	Minor firmware version
8224	0x2020	16	R	Short device description
8225	0x2021	8	R	Compile time of the firmware version

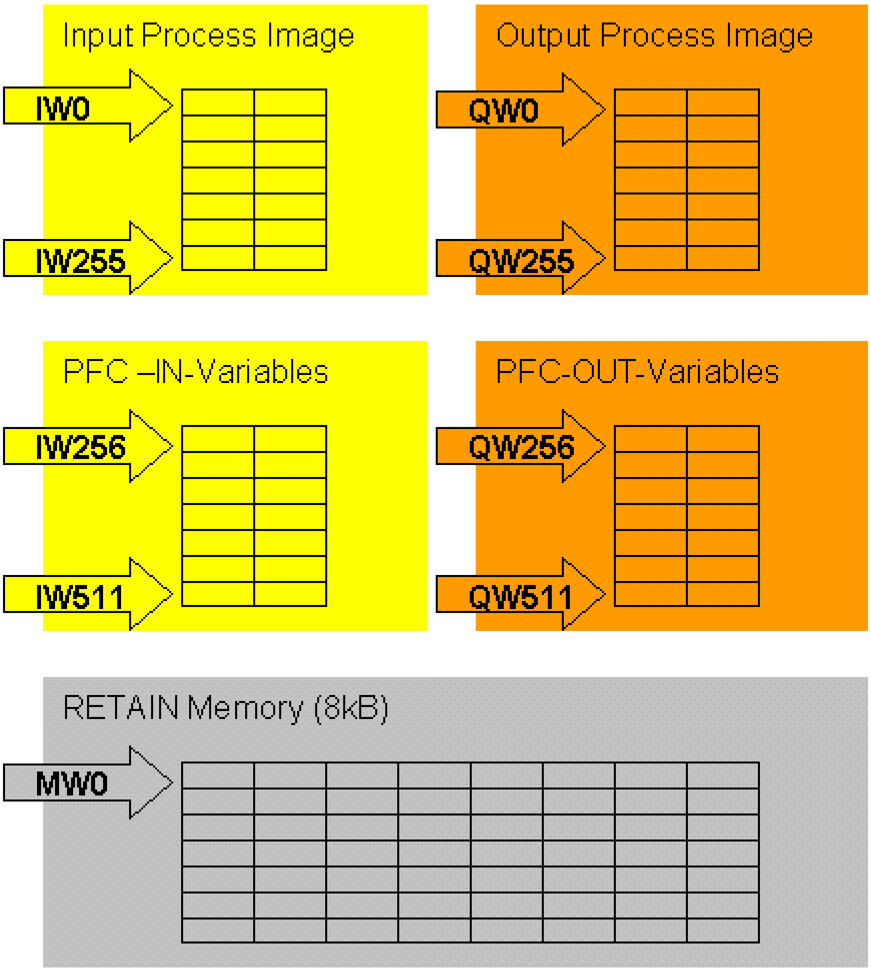
767-1301: MODBUS Configuration Register for FC3, FC4, FC6 and FC16

MODBUS Address		Length	Access	Description
[dec]	[hex]	[Word]		
8226	0x2022	8	R	Compile date of the firmware version
8227	0x2023	32	R	Firmware loader version (FWL)
8240	0x2030	65	R	Description of connected IO modules: 0-64
8256	0x2040	1	W	Software reset (write 0x55AA or 0xAA55)
8257	0x2041	1	W	Formatting the file system
8258	0x2042	1	W	Extracting the file system
8259	0x2043			
8345	0x2099	1	R/W	MODBUS compatibility mode 1 = MODBUS V1 (similar to 750 series) 2 = MODBUS V2

7.4 Programmable Fieldbus Controller 750-842

7.4.1 Process Image of the 750-842

The 750-842 can process a maximum of 3 incoming MODBUS TCP connections at the same time. The MODBUS connection Watchdog is deactivated when delivered. The PLC program can establish a maximum of two TCP connections to remote servers.



The 750-842 supports the following MODBUS service:

FC	Name	Description
FC1	Read coils	Re-read several digital outputs
FC2	Read inputs discrete	Read several digital inputs
FC3	Read holding registers	Reading several analog inputs (and outputs)
FC4	Read input registers	Reading several analog inputs (and outputs)
FC5	Write coil	Write a single digital output
FC6	Write single register	Write a single analog output
FC7	Read exception status	Read the first 8 digital outputs
FC11	Get comm event counter	Communication event counter
FC15	Force multiple coils	Write several digital outputs
FC16	Write multiple registers	Write several analog outputs
FC23	Read/write multiple registers	Read/write operation to analog inputs/outputs

By using the “SET_DIGITAL_INPUT_OFFSET” and “SET_DIGITAL_OUTPUT_OFFSET” function blocks from the “mod_com.lib” library, it is possible to specify the start addresses of the first digital I/O modules that are connected to WAGO ETHERNET controllers. This allows for space for later extensions. The OFFSETs are given in bytes.

Memory area	**MODBUS access**	**PLC access**	**Description**
Physical. Input	read	read	Physical inputs (%IW0 ... %IW255)
Physical output	read/write	read/write	Physical outputs (%QW0 ... %QW255)
PFC IN	read/write	read	Volatile PLC input variables (%IW256 ... %IW511)
PFC OUT	read	read/write	Volatile PLC output variables (%QW256 ... %QW511)
Configuration register	read/(write)	---	Configuration Registers
RETAIN (NOVRAM)	read/write	read/write	8kB residual memory (%MW0 ... %MW4095)

Note that the physical outputs can be changed both via the MODBUS services and the PLC program.

A mnemonic for this behavior could be: “The last one wins”.

In the 750-842, the flag variables and the retain variables share the same area in the NOVRAM.
Overlapping could result in unpredictable behavior.
Only use one of the two types in your CODESYS project.

7.4.2 Register services of the 750-842

7.4.2.1 Read registers with FC3, FC4 and FC23:

750-842: MODBUS vs IEC 61131 Addresses for FC3, FC4 and FC23			
MODBUS Address		**IEC 61131**	**Description**
[dec]	**[hex]**	**Address**	
0 ... 255	0x0000 ... 0x00FF	%IW0 ... %IW255	Physical input area
256 ... 511	0x0100 ... 0x01FF	%QW256 ... %QW511	PFC OUT area Volatile PLC output variables
512 ... 767	0x0200 ... 0x02FF	%QW0 ... %QW255	Physical output area
768 ... 1023	0x0300 ... 0x03FF	%IW256 ... %IW511	PFC IN area Volatile PLC input variables
1024 ... 4095	0x0400 ... 0x0FFF	-	MODBUS exception: “Illegal data address”
4096 ... 12287	0x1000 ... 0x2FFF	-	Configuration register (see manual for details)
12288 ... 16383	0x3000 ... 0x3FFF	%MW0 ... %MW4095	NOVRAM 8kB retain memory
16384 ... 65535	0x4000 ... 0xFFFF	-	MODBUS exception: “Illegal data address”

7.4.2.2 Write registers with FC6, FC16 and FC23:

750-842: MODBUS vs IEC 61131 Addresses for FC6, FC16			
MODBUS Address		**IEC 61131**	**Description**
[dec]	**[hex]**	**Address**	
0 ... 255	0x0000 ... 0x00FF	%QW0 ... %QW255	Physical output area
256 ... 511	0x0100 ... 0x01FF	%IW256 ... %IW511	PFC IN area Volatile PLC input variables
512 ... 767	0x0200 ... 0x02FF	%QW0 ... %QW255	Physical output area
768 ... 1023	0x0300 ... 0x03FF	%IW256 ... %IW511	PFC OUT area Volatile PLC output variables
1024 ... 4095	0x0400 ... 0x0FFF	-	MODBUS exception: “Illegal data address”
4096 ... 8191	0x1000 ... 0x1FFF	-	Configuration register (see manual for details)
8192 ... 12287	0x2000 ... 0x2FFF	-	MODBUS exception: “Illegal data address”
12288 ... 16383	0x3000 ... 0x3FFF	%MW0 ... %MW4095	NOVRAM 8kB retain memory
16384 ... 65535	0x4000 ... 0xFFFF	-	MODBUS exception: “Illegal data address”

7.4.3 Digital MODBUS services of the 750-842

The digital MODBUS services can only determine or change the state of digital I/O modules. Complex I/O modules are ignored or cannot be accessed.

In the PFC-IN and PFC-OUT area, as well as in the flag area (NVRAM), the coil services and register services access the same memory locations.

Since the address space is limited by the data type “WORD”, it is not possible to address all bits in the 8kB flag area via coil services.

7.4.3.1 Read coils with FC1 and FC2:

750-842: MODBUS Addresses for FC1 and FC2			
MODBUS Address		Memory	Description
[dec]	[hex]	Area	
0 ... 511	0x0000 ... 0x01FF	Physical input area	First 512 digital inputs
512 ... 1023	0x0200 ... 0x03FF	Physical output area	First 512 digital outputs
1024 ... 4095	0x0400 ... 0x0FFF	-	MODBUS exception: “Illegal data address”
4096 ... 8191	0x1000 ... 0x1FFF	%QX256.0 ...%QX511.15	PFC OUT area Volatile PLC output variables
8192 ... 12287	0x2000 ... 0x2FFF	%IX256.0 ...%IX511.15	PFC IN area Volatile PLC input variables
12288 ... 65535	0x3000 ... 0xFFFF	%MX0.0 ... %MX3327.15	NOVRAM Retain memory

7.4.3.2 Write coils with FC5 and FC15:

750-842: MODBUS Addresses for FC5 and FC15			
MODBUS Address		Memory	Description
[dec]	[hex]	Area	
0 ... 511	0x0000 ... 0x01FF	Physical output area	max 512 digital outputs
512 ... 1023	0x0200 ... 0x03FF	Physical output area	max 512 digital outputs
1024 ... 4095	0x0400 ... 0x0FFF	-	MODBUS exception: “Illegal data address”
4096 ... 8191	0x1000 ... 0x1FFF	%IX256.0 ...%IX511.15	PFC IN area Volatile PLC input variables
8192 ... 12287	0x2000 ... 0x2FFF	%IX256.0 ...%IX511.15	PFC IN area Volatile PLC input variables
12288 ... 65535	0x3000 ... 0xFFFF	%MX0.0 ... %MX3327.15	NOVRAM Retain memory

7.4.4 MODBUS Configuration Registers for the 750-842

The configuration registers make it possible to determine and in part change the properties of the 750-842.

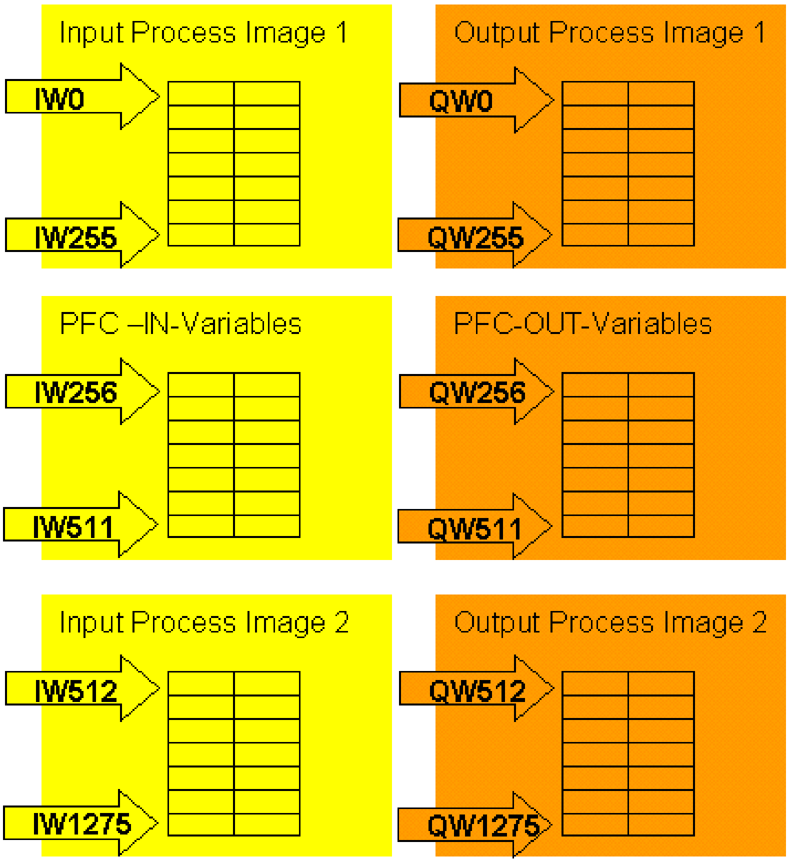
750-842: MODBUS Configuration Register for FC3, FC4, FC6 and FC16				
MODBUS Address		**Length**	**Access**	**Description**
[dec]	**[hex]**	**[Word]**		
4096	0x1000	1	R/W	MODBUS Watchdog Time (multiple of 100 ms)
4097	0x1001	1	R/W	MODBUS Watchdog coding mask 1-16
4098	0x1002	1	R/W	MODBUS Watchdog coding mask 17-32
4099	0x1003	1	R/W	MODBUS Watchdog-Trigger
4100	0x1004	1	R	Minimum trigger time
4101	0x1005	1	R/W	Stop MODBUS Watchdog (0xAAAA and 0x5555)
4102	0x1006	1	R	MODBUS Watchdog status
4103	0x1007	1	R/W	MODBUS Watchdog restart (0x0001)
4104	0x1008	1	R/W	Stop MODBUS Watchdog (0x55AA or 0xAA55)
4105	0x1009	1	R/W	Close MODBUS and HTTP port after time-out
4106	0x100A	1	R/W	Start MODBUS Watchdog in “Modicon Mode”
4107	0x100B	1	W	Save MODBUS Watchdog parameters
4128	0x1020	1	R	LED error code
4129	0x1021	1	R	LED error argument
4130	0x1022	1	R	Number of analog outputs in the process image [Bit]
4131	0x1023	1	R	Number of analog inputs in the process image [Bit]
4132	0x1024	1	R	Number of digital outputs in the process image [Bit]
4133	0x1025	1	R	Number of digital inputs in the process image [Bit]
4135	0x1027	1	R	Execute internal bus cycle
4136	0x1028	1	R/W	IP configuration: BootP(1) or FIX(0)
4137	0x1029	18	R	MODBUS TCP statistics
4144	0x1030	1	R/W	Activate MODBUS connection monitoring
4145	0x1031	3	R	MAC ID or ETHERNET interface
4160	0x1040	1	R/W	Process data interface
8192	0x2000	1	R	0x0000 (Constant)
8193	0x2001	1	R	0xFFFF (Constant)
8194	0x2002	1	R	0x1234 (Constant)
8195	0x2003	1	R	0xAAAA (Constant)
8196	0x2004	1	R	0x5555 (Constant)
8197	0x2005	1	R	0x7FFF (Constant)
8198	0x2006	1	R	0x8000 (Constant)
8199	0x2007	1	R	0x3FFF (Constant)
8200	0x2008	1	R	0x4000 (Constant)
8208	0x2010	1	R	Firmware release
8209	0x2011	1	R	Series code (750)
8210	0x2012	1	R	Device code (842)
8211	0x2013	1	R	Specific firmware version (0xFFFF)
8212	0x2014	1	R	Specific firmware version (0xFFFF)

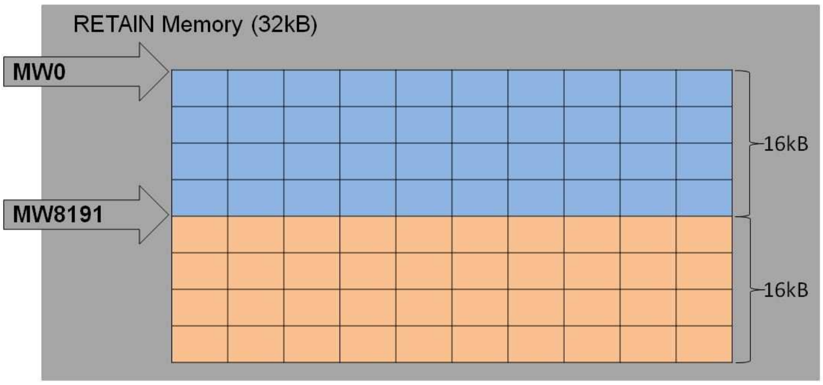
750-842: MODBUS Configuration Register for FC3, FC4, FC6 and FC16				
MODBUS Address		**Length**	**Access**	**Description**
[dec]	**[hex]**	**[Word]**		
8224	0x2020	1 .. 125	R	Short device description
8225	0x2021	8	R	Compile time of the firmware version
8226	0x2022	8	R	Compile date of the firmware version
8227	0x2023	32	R	Firmware loader version (FWL)
8240	0x2030	65	R	Description of connected IO modules: 0-64
8245	0x2035	1	R/W	Setting process image (Table 0 register 3)
8246	0x2036	1 .. 17	R	Diagnostic information device
8256	0x2040	1	W	Software reset (write 0x55AA or 0xAA55)
8260	0x2044	1	W	Delete MODBUS Configuration file (write 0x55AA)

7.5 Programmable Fieldbus Controller 750-88x

7.5.1 Process Image of the 750-88x

The 750-88x series controllers (750-880, 750-881, 750-882, 750-885) can process a maximum of 15 incoming MODBUS TCP connections at the same time. The MODBUS connection Watchdog is deactivated when delivered.

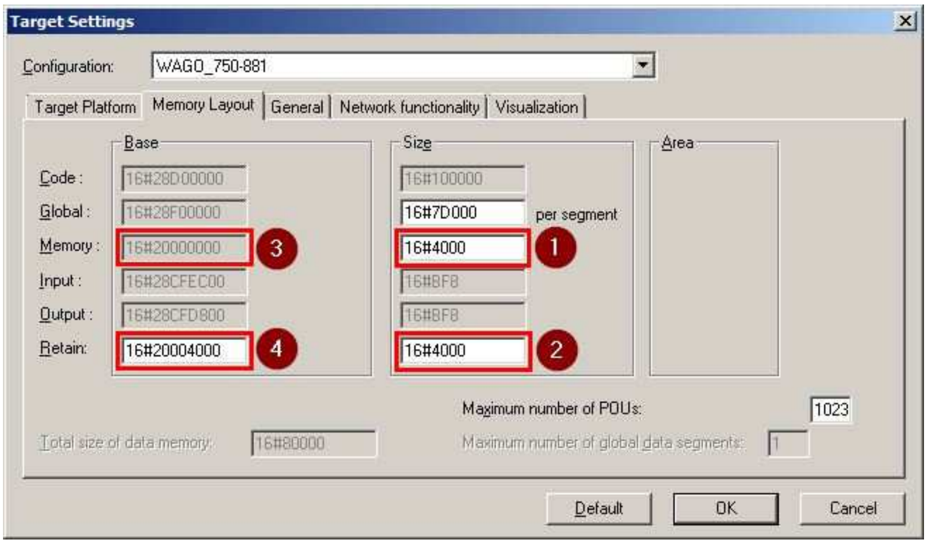




RETAIN memory is non-volatile memory, i.e., in the event of a power failure, all flag memory and variable values explicitly defined with “VAR RETAIN” are retained.

The 32 kByte memory range is normally divided into a 16 kByte addressable range for bit memory (%MW0 ... %MW8191) and a 16 kByte retain area for variables without memory space addressing or for variables that are explicitly defined by “VAR RETAIN”.

If more than 16 kBytes should be used as addressable memory for flags, a corresponding adjustment to memory allocation can be made in CODESYS via the target system settings.



Configuration: WAGO_750-881

	Base	Size
Code :	16#28D00000	16#100000
Global :	16#28F00000	16#7D000 per segment
Memory :	16#20000000	16#4000
Input :	16#28CFEC00	16#BF8
Output :	16#28CFD800	16#BF8
Retain:	16#20004000	16#4000

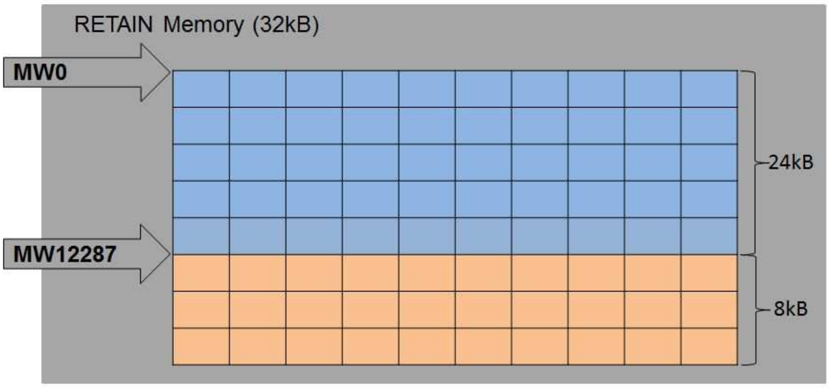
Maximum number of POUs: 1023

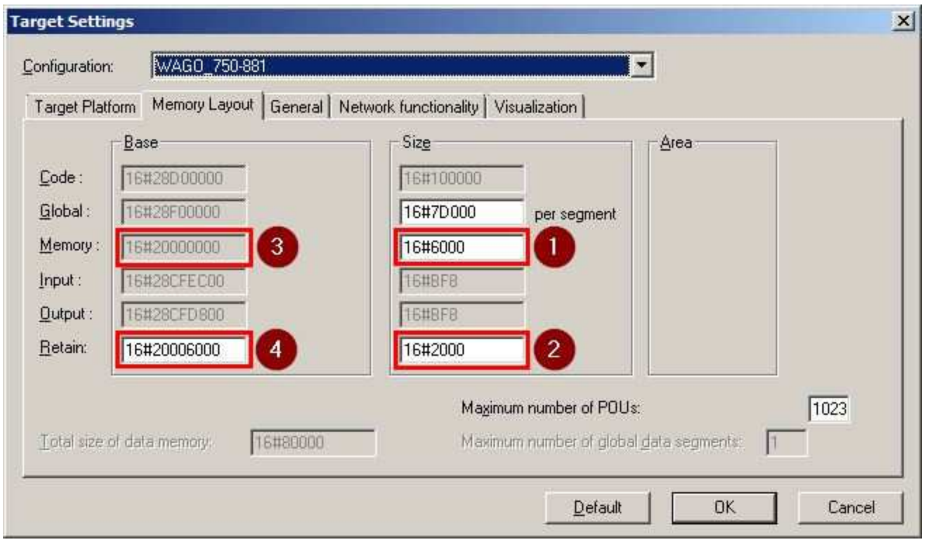
Total size of data memory: 16#80000

Maximum number of global data segments: 1

1	Size of the addressable flag area (16#4000 ⇨16 kByte ⇨8192 flag words)
2	Size of the non-addressable retain memory (16#4000 ⇨16 kByte retain memory)
3	Start address of the addressable flag area (cannot be edited)
4	Start address of the non-addressable retain memory ⇨ Start address flag + flag size ⇨16#20000000 + 16#4000 = 16#20004000

Example configuration with 24 kByte of addressable flag memory:




Target Settings

Configuration: WAGO_750-881

Target Platform | Memory Layout | General | Network functionality | Visualization

	Base	Size
Code :	16#28D00000	16#100000
Global :	16#28F00000	16#7D000 per segment
Memory :	16#20000000 (3)	16#6000 (1)
Input :	16#28CFEC00	16#8F8
Output :	16#28CFD800	16#8F8
Retain:	16#20006000 (4)	16#2000 (2)

Area

Maximum number of POUs: 1023

Total size of data memory: 16#80000

Maximum number of global data segments: 1

1	Size of the addressable flag area (16#6000 ⇨24 kByte ⇨12288 flag words)
2	Size of the non-addressable retain memory (16#2000 ⇨8 kByte retain memory)
3	Start address of the addressable flag area (cannot be edited)
4	Start address of the non-addressable retain memory ⇨ Start address flag + flag size ⇨16#20000000 + 16#6000 = 16#20006000

Information

Do not exceed the size of the retain memory!
The size totals from the sizes specified for the flag area and for the retain area must not exceed the total size of the memory:
Flag size + retain size must always result in 16 kByte (16#8000)!

The 750-88x series controllers support the following MODBUS services:

FC	Name	Description
FC1	Read coils	Re-read several digital outputs
FC2	Read inputs discrete	Read several digital inputs
FC3	Read holding registers	Read several analog inputs (and outputs)
FC4	Read input registers	Read several analog inputs (and outputs)
FC5	Write coil	Write a single digital output
FC6	Write single register	Write a single analog output
FC11	Get comm event counter	Communication event counter
FC15	Force multiple coils	Write several digital outputs
FC16	Write multiple registers	Writing several analog outputs
FC22	Mask write	Manipulation of single bits in a register
FC23	Read/write multiple registers	Read/write operation to analog inputs/outputs

By using the “SET_DIGITAL_INPUT_OFFSET” and “SET_DIGITAL_OUTPUT_OFFSET” function blocks from the “mod_com.lib” library, it is possible to specify the start addresses of the first digital I/O modules that are connected to WAGO ETHERNET controllers. This allows for space for later extensions. The OFFSETs are given in bytes. However, the effectiveness is limited to the PLC. The MODBUS slave ignores the digital offset.

Memory area	MODBUS access	PLC access	Description
Physical. Input(1)	read	read	Physical inputs (%IW0 ... %IW255)
Physical Output(1)	read/[write]$^{*1)}$	read/[write]$^{*1)}$	Physical outputs (%QW0 ... %QW255)
PFC IN	read/write	read	Volatile PLC input variables (%IW256 ... %IW511)
PFC OUT	read	read/write	Volatile PLC output variables (%QW256 ... %QW511)
Configuration register	read/(write)	---	Configuration Registers
NOVRAM Retain memory	read/write	read/write	8 kB retain memory (max 32 kB) (%MW0 ... %MW8191)
Physical. Input(2)	read	read	Physical inputs (%IW512 ... %IW1275)
Physical Output(2)	read/[write]$^{*1)}$	read/[write]$^{*1)}$	Physical outputs (%QW512 ... %QW1275)

[]$^{*1)}$ The “/etc/EA-conf.xml” file specifies the write permission

Another feature of the 750-88x controller is that write permissions can or must be assigned to each I/O module.
The physical outputs can be changed either via MODBUS services or via the PLC program. The “/etc/EA-conf.xml” file specifies the write permissions.
If the file is missing or if the number of configured I/O modules differs from the number of I/O modules actually connected, then write permissions are given to the MODBUS services.
The “/etc/EA-conf.xml” file is created automatically when a CODESYS control configuration is created. It controls the write permissions on the I/O module level.
A mnemonic for this behavior could be: “There can be only one”.

In the 750-88x, the flag variables and the retain variables share the same area in the 32 KB NOVRAM. The default configuration includes 16 kB for flag variables and 16 kB for retain variables. Overlapping, as in the case with 750-842, is not possible.
The partitioning of the 24 kB NOVRAM can be changed in the CODESYS target system settings.

When using “SysLibSocket”, the maximum number of TCP socket connections that can be generated from a PLC program is almost unlimited. In the “Ethernet.lib” the maximum number of TCP socket connections is limited to 5.

7.5.2 Register Services of the 750-88x

7.5.2.1 Read registers with FC3, FC4 and FC23:

750-88x: MODBUS vs IEC 61131 Addresses for FC3, FC4 and FC23			
MODBUS Address		**IEC 61131 Address**	**Description**
[dec]	**[hex]**		
0 ... 255	0x0000 ... 0x00FF	%IW0 ... %IW255	Physical input area (1) First 256 words of physical input data
256 ... 511	0x0100 ... 0x01FF	%QW256 ... %QW511	PFC OUT area Volatile PLC output variables
512 ... 767	0x0200 ... 0x02FF	%QW0 ... %QW255	Physical output area (1) First 256 words of physical output data
768 ... 1023	0x0300 ... 0x03FF	%IW256 ... %IW511	PFC IN area Volatile PLC input variables
1024 ... 4095	0x0400 ... 0x0FFF	-	MODBUS exception: “Illegal data address”
4096 ... 12287	0x1000 ... 0x2FFF	-	Configuration register (see manual for details)
12288 ... 24575	0x3000 ... 0x5FFF	%MW0 ... %MW12287	NOVRAM 24 kB
24576 ... 25339	0x6000 ... 0x62FC	%IW512 ... %IW1275	Physical input area (2) Additional 764 words physical input data
25340 ... 28671	0x62FD ... 0x6FFF	-	MODBUS Exception: “ Illegal data address”
28672 ... 29435	0x7000 ... 0x72FB	%QW512 ... %QW1275	Physical output area (2) Additional 764 words physical output data
29436 ... 32767	0x72FC ... 0x7FFF	-	MODBUS Exception: “ Illegal data address”
32768 ... 36863	0x8000 …0x8FFF	%MW12288 ..%MW16383	NOVRAM 8 kB
36864 ...65535	0x9000 ...0xFFFF		MODBUS exception: “Illegal data address”

7.5.2.2 Write registers with FC6, FC16, FC22 and FC23:

750-88x: MODBUS vs IEC 61131 Addresses for FC6, FC16, FC22 and FC23			
MODBUS Address		**IEC 61131 Address**	**Description**
[dec]	**[hex]**		
0 ... 255	0x0000 ... 0x00FF	%QW0 ... %QW255	Physical output area (1) First 256 words of physical output data
256 ... 511	0x0100 ... 0x01FF	%IW256 ... %IW511	PFC IN area Volatile PLC input variables
512 ... 767	0x0200 ... 0x02FF	%QW0 ... %QW255	Physical output area (1) First 256 words of physical output data
768 ... 1023	0x0300 ... 0x03FF	%IW256 ... %IW511	PFC IN area Volatile PLC input variables
1024 ... 4095	0x0400 ... 0x0FFF	-	MODBUS exception: “Illegal data address”
4096 ... 12287	0x1000 ... 0x2FFF	-	Configuration register (see manual for details)
12288 ... 24575	0x3000 ... 0x5FFF	%MW0 ... %MW12287	NOVRAM 8 kB retain memory (max. 24 kB)
24576 ... 25339	0x6000 ... 0x62FC	%QW512 ... %QW1275	Physical output area (2) Additional 764 words physical output data
25340 ... 28671	0x62FD ... 0x6FFF	-	MODBUS Exception: “ Illegal data address”
28672 ... 29435	0x7000 ... 0x72FC	%QW512 ... %QW1275	Physical output area (2) Additional 764 words physical output data
29436 ... 32767	0x72FD ... 0x7FFF		MODBUS Exception: “ Illegal data address”
32768 ...36863	0x8000 ...0x8FFF	%MW12288 ..%MW16383	NOVRAM 8 kB
36864 ...65535	0x9000 ...0xFFFF		MODBUS exception: “Illegal data address”

7.5.3 Digital MODBUS Services of the 750-88x

The digital MODBUS services can only determine or change the state of digital I/O modules. Complex I/O modules are ignored or cannot be accessed.

7.5.3.1 Read coils with FC1 and FC2:

750-88x: MODBUS Addresses for FC1 and FC2			
MODBUS Address		Memory Area	Description
[dec]	[hex]		
0 … 6143	0x0000 ... 0x01FF	Physical input area (1)	First 512 digital inputs
512 ... 1023	0x0200 ... 0x03FF	Physical output area (1)	First 512 digital outputs
1024 ... 4095	0x0400 ... 0x0FFF	-	MODBUS Exception: “Illegal data address”
4096 ... 8191	0x1000 ... 0x1FFF	%QX256.0 ...%QX511.15	PFC OUT area Volatile PLC output variables
8192 ... 12287	0x2000 ... 0x2FFF	%IX256.0 ...%IX511.15	PFC IN area Volatile PLC input variables
12288 ... 32767	0x3000 ... 0x7FFF	%MX0.0 ... %MX1279.15	NOVRAM Retain area (8 kB default)
32768 ... 34295	0x8000 ... 0x85F7	Physical input area (2)	Starts with the 513th and ends with the 2039th digital input
34296 ... 36863	0x85F8 ... 0x8FFF	-	MODBUS exception: “Illegal data address”
36864 ... 38391	0x9000 ... 0x95F7	Physical output area (2)	Starts with the 513th and ends with the 2039th digital output
38392 ... 65535	0x95F8 ... 0xFFFF	-	MODBUS exception: “Illegal data address”

7.5.3.2 Write coils with FC5 and FC15:

750-88x: MODBUS Addresses for FC5 and FC15			
MODBUS Address		**Memory Area**	**Description**
[dec]	**[hex]**		
0 ... 511	0x0000 ... 0x01FF	Physical output area (1)	First 512 digital outputs
512 ... 1023	0x0200 ... 0x03FF	Physical output area (1)	First 512 digital outputs
1024 ... 4095	0x0400 ... 0x0FFF	-	MODBUS Exception: “Illegal data address”
4096 ... 8191	0x1000 ... 0x1FFF	%IX256.0 ...%IX511.15	PFC IN area Volatile PLC input variables
8192 ... 12287	0x2000 ... 0x2FFF	%IX256.0 ...%IX511.15	PFC IN area Volatile PLC input variables
12288 ... 32767	0x3000 ... 0x7FFF	%MX0.0 ... %MX1279.15	NOVRAM Retain area
32768 ... 34295	0x8000 ... 0x85F7	Physical output area (2)	Starts with the 513th and ends with the 2039th digital output
34296 ... 36863	0x85F8 ... 0x8FFF	-	MODBUS exception: “Illegal data address”
36864 ... 38391	0x9000 ... 0x95F7	Physical output area (2)	Starts with the 513th and ends with the 2039th digital output
38392 ... 65535	0x95F8 ... 0xFFFF	-	MODBUS exception: “Illegal data address”

7.5.4 MODBUS Configuration Registers for the 750-88x

The configuration registers make it possible to determine and in part change the properties of the 750-88x.

750-88x: MODBUS Configuration Register for FC3, FC4, FC6 and FC16

MODBUS Address		Length	Access	Description
[dec]	[hex]	[Word]		
4096	0x1000	1	R/W	MODBUS Watchdog Time (multiple of 100 ms)
4097	0x1001	1	R/W	MODBUS Watchdog coding screen 1-16
4098	0x1002	1	R/W	MODBUS Watchdog coding screen 17-32
4099	0x1003	1	R/W	MODBUS Watchdog-Trigger
4100	0x1004	1	R	Minimum trigger time
4101	0x1005	1	R/W	Stop MODBUS Watchdog (0xAAAA and 0x5555)
4102	0x1006	1	R	MODBUS Watchdog status
4103	0x1007	1	R/W	MODBUS Watchdog restart (0x0001)
4104	0x1008	1	R/W	Stop MODBUS Watchdog (0x55AA or 0xAA55)
4105	0x1009	1	R/W	Close MODBUS and HTTP port after time-out
4106	0x100A	1	R/W	Start MODBUS Watchdog in “Modicon Mode”
4107	0x100B	1	W	Save MODBUS Watchdog parameters
4128	0x1020	1	R	LED error code
4129	0x1021	1	R	LED error argument
4130	0x1022	1	R	Number of analog outputs in the process image [Bit]
4131	0x1023	1	R	Number of analog inputs in the process image [Bit]
4132	0x1024	1	R	Number of digital outputs in the process image [Bit]
4133	0x1025	1	R	Number of digital inputs in the process image [Bit]
4136	0x1028	1	R/W	IP configuration: BootP(1), DHCP(2) or FIX(4)
4137	0x1029	18	R	MODBUS TCP statistics
4138	0x102A	1	R	Number of established MODBUS TCP connections
4139	0x102B	1	W	K-BUS reset
4144	0x1030	1	R/W	Activate MODBUS connection monitoring
4145	0x1031	3	R	MAC ID or ETHERNET interface
4149	0x1035	1	R/W	Time offset RTC (Real Time Clock)
4150	0x1036	1	R/W	Daylight Savings
4151	0x1037	1	1	MODBUS Response Delay (ms)
4176	0x1050	3	R	Diagnostic information of connected IO modules
8192	0x2000	1	R	0x0000 (Constant)
8193	0x2001	1	R	0xFFFF (Constant)
8194	0x2002	1	R	0x1234 (Constant)
8195	0x2003	1	R	0xAAAA (Constant)
8196	0x2004	1	R	0x5555 (Constant)
8197	0x2005	1	R	0x7FFF (Constant)
8198	0x2006	1	R	0x8000 (Constant)
8199	0x2007	1	R	0x3FFF (Constant)
8200	0x2008	1	R	0x4000 (Constant)
8208	0x2010	1	R	Firmware release

750-88x: MODBUS Configuration Register for FC3, FC4, FC6 and FC16				
MODBUS Address		**Length**	**Access**	**Description**
[dec]	**[hex]**	**[Word]**		
8209	0x2011	1	R	Series code (750)
8210	0x2012	1	R	Device code (841)
8211	0x2013	1	R	Major firmware version
8212	0x2014	1	R	Minor firmware version
8224	0x2020	1 .. 125	R	Short device description
8225	0x2021	8	R	Compile time of the firmware version
8226	0x2022	8	R	Compile date of the firmware version
8227	0x2023	32	R	Firmware loader version (FWL)
8240	0x2030	65	R	Description of connected IO modules: 0-64
8241	0x2031	64	R	Description of connected IO modules: 65-129
8242	0x2032	64	R	Description of connected IO modules: 130-194
8243	0x2033	63	R	Description of connected IO modules: 195-255
8245	0x2035	1	R/W	Setting process image (Table 0 register 3)
8246	0x2036	1 .. 17	R	Diagnostic information device
8256	0x2040	1	W	Software reset (write 0x55AA or 0xAA55)
8257	0x2041	1	W	Format Flash
8258	0x2042	1	W	Extract file system
8259	0x2043	1	W	Factory Settings

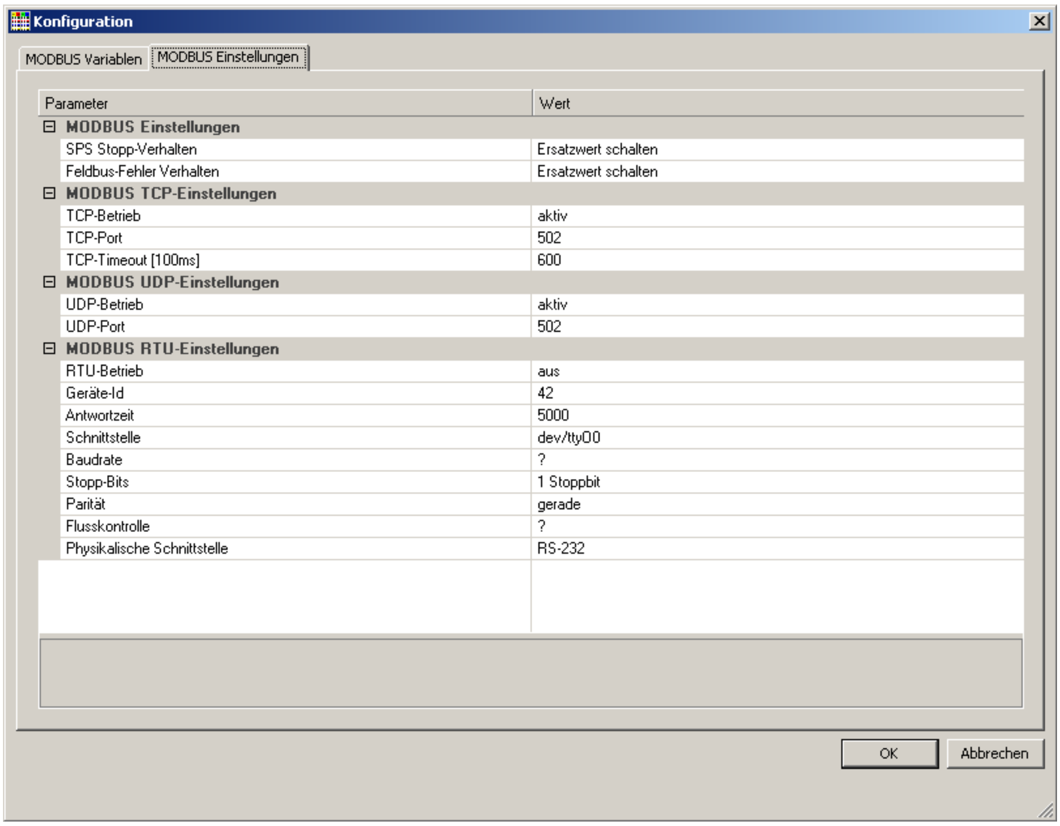
7.6 Programmable Fieldbus Controllers PFC200

This section contains the devices of the PFC200 series (750-8202, 750-8203, 750-8204, 750-8206) with CODESYS 2.3 runtime system.

The MODBUS slave integrated in the PFC200 must be activated and parameterized via the CODESYS controller configuration.

The MODBUS function is only active if a CODESYS project with MODBUS activated is loaded to PFC200.

Direct access to the process image of the I/O modules via MODBUS is not possible.



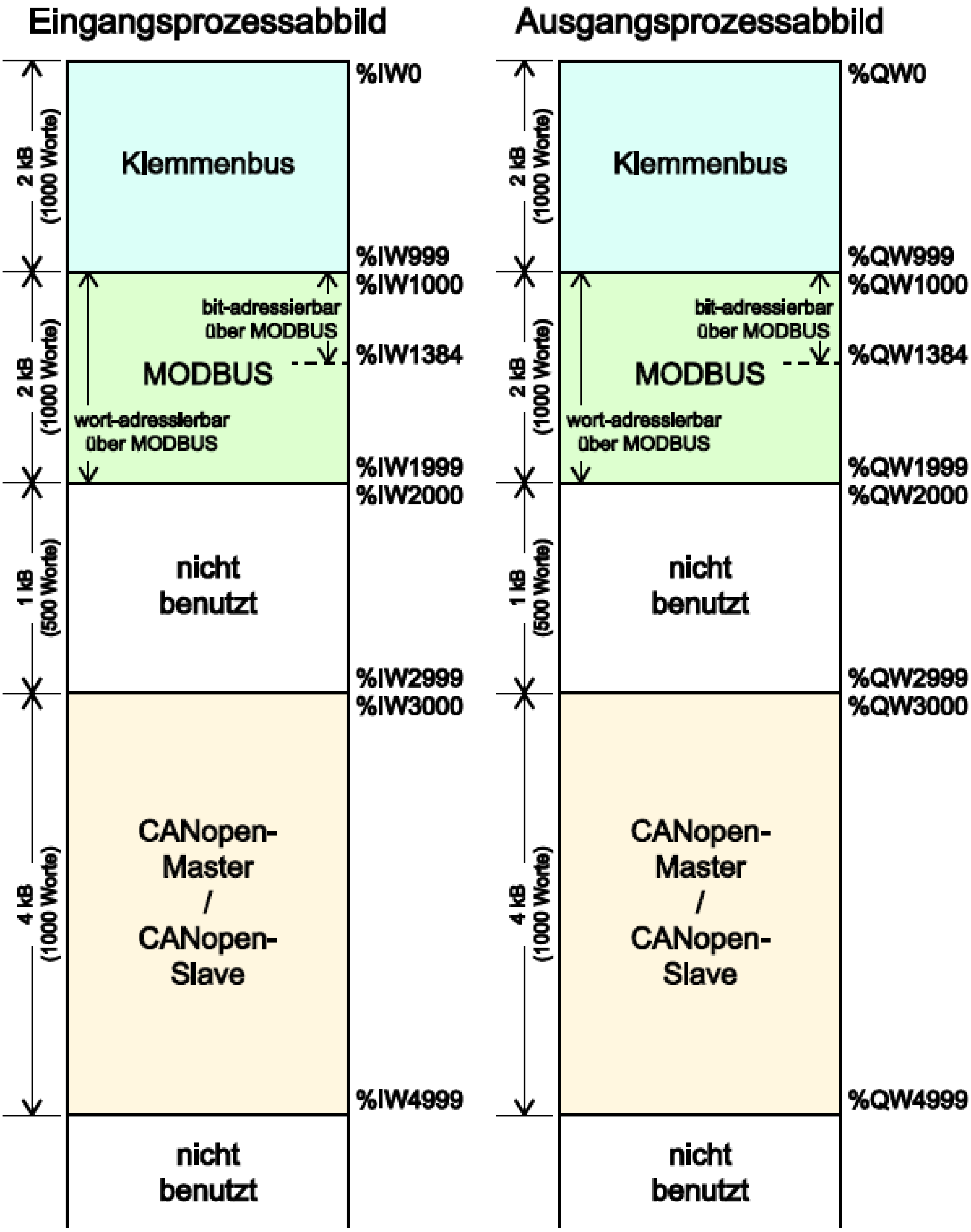
Parameter	Wert
⊟ **MODBUS Einstellungen**	
SPS Stopp-Verhalten	Ersatzwert schalten
Feldbus-Fehler Verhalten	Ersatzwert schalten
⊟ **MODBUS TCP-Einstellungen**	
TCP-Betrieb	aktiv
TCP-Port	502
TCP-Timeout [100ms]	600
⊟ **MODBUS UDP-Einstellungen**	
UDP-Betrieb	aktiv
UDP-Port	502
⊟ **MODBUS RTU-Einstellungen**	
RTU-Betrieb	aus
Geräte-Id	42
Antwortzeit	5000
Schnittstelle	dev/tty00
Baudrate	?
Stopp-Bits	1 Stoppbit
Parität	gerade
Flusskontrolle	?
Physikalische Schnittstelle	RS-232

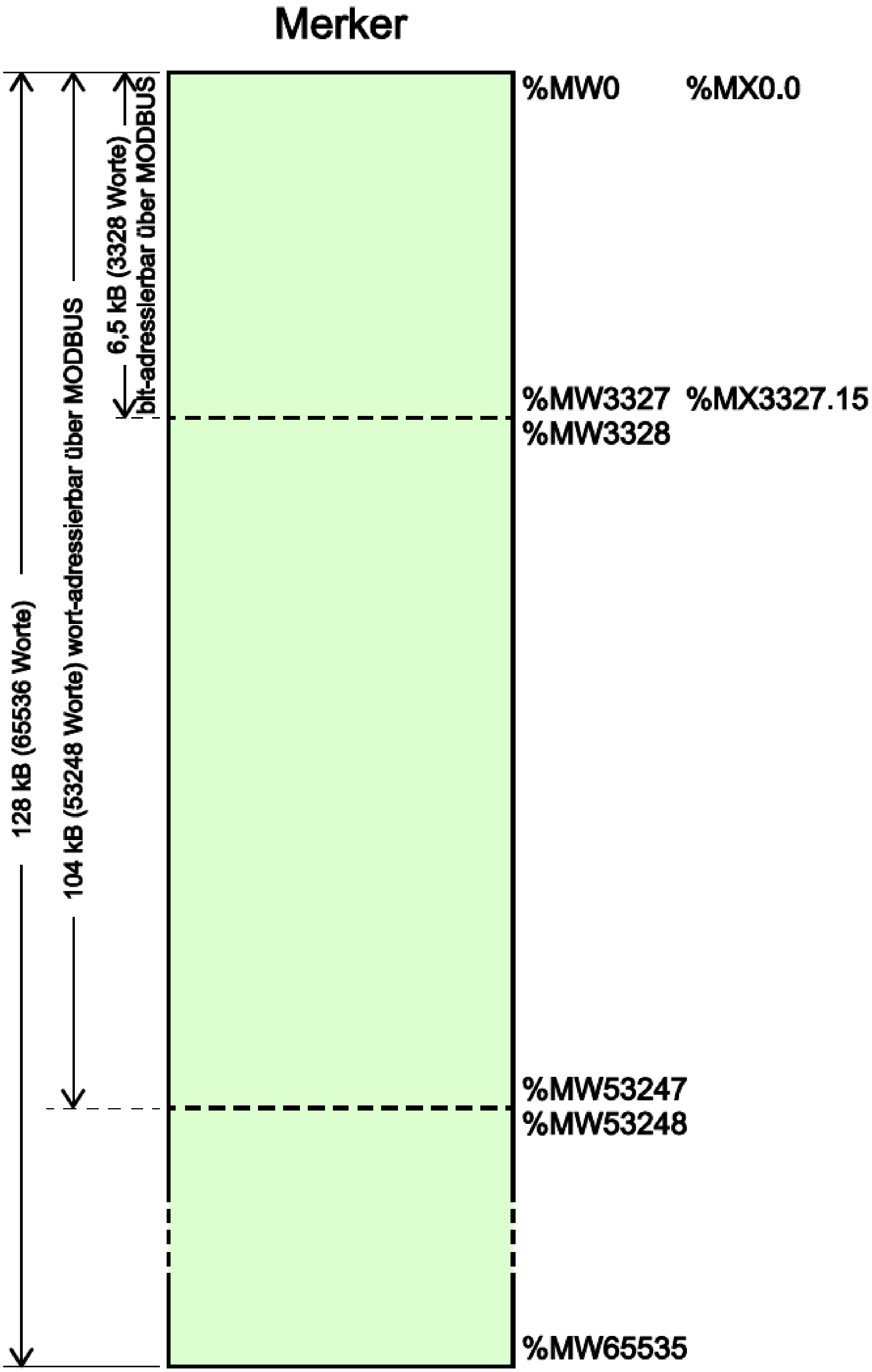
Further information about configuration the PFC200 MODBUS slave is available in the manual for the respective PFC200 model.

Note

I/O modules not directly accessible via MODBUS.
Direct access to the process image of the I/O modules via MODBUS is not possible. Only the PFC variables and flags are accessible via MODBUS. Using a corresponding PLC program, data points of the I/O module can be assigned to the MODBUS area.

7.6.1 Process Image of the PFC200




Merker
%MW0
%MX0.0
%MW3327
%MX3327.15
%MW3328
%MW53247
%MW53248
%MW65535
128 kB (65536 Worte)
104 kB (53248 Worte) wort-adressierbar über MODBUS
6,5 kB (3328 Worte)
bit-adressierbar über MODBUS

7.6.2 Register Services of the PFC200

7.6.2.1 Read registers with FC3, FC4 and FC23:

PFC200: MODBUS vs IEC 61131 Addresses for FC3, FC4 and FC23			
MODBUS Address		**IEC 61131**	**Description**
[dec]	**[hex]**	**Address**	
0 … 999	0x0000 ... 0x03EF	%IW1000 ... %IW1999	1000 PFC input words in the 2 kB input process image
1000 ... 1999	0x03E8 ... 0x07CF	%QW1000 ... %QW1999	1000 PFC output words in the 2 kB output process image
2000 … 4095	0x07D0 … 0x0FFF		MODBUS Exception: “Illegal data address”
4096 … 12287	0x1000 … 0x2FFF		Information and configuration register: Not all MODBUS addresses in this area are valid. Valid MODBUS addresses are described in the Section “Configuration Registers”.
12288 … 65535	0x3000 …0xFFFF	%MW0 …%MW53247	Flag area: 53248 register/word flags (104 kB) in the flag area

7.6.2.2 Write registers with FC6, FC16, FC22 and FC23:

PFC200: MODBUS vs IEC 61131 Addresses for FC6, FC16, FC22 and FC23			
MODBUS Address		**IEC 61131**	**Description**
[dec]	**[hex]**	**Address**	
0 … 999	0x0000 ... 0x03EF	%IW1000 ... %IW1999	1000 PFC input words in the 2 kB input process image
1000 ... 1999	0x03E8 ... 0x07CF		MODBUS exception: “Illegal data address”
2000 … 4095	0x07D0 … 0x0FFF		MODBUS Exception: “Illegal data address”
4096 … 12287	0x1000 … 0x2FFF		Information and configuration register: Not all MODBUS addresses in this area are valid. Valid MODBUS addresses are described in the Section “Configuration Registers”.
12288 … 65535	0x3000 …0xFFFF	%MW0 …%MW53247	Flag area: 53248 register/word flags (104 kB) in the flag area

7.6.3 Digital MODBUS services of the PFC200

The digital MODBUS services allow bitwise access to the first 768 bytes of the process images.

7.6.3.1 Read coils with FC1 and FC2:

PFC200: MODBUS Addresses for FC1 and FC2			
MODBUS Address		Memory Area	Description
[dec]	[hex]		
0 … 6143	0x0000 ... 0x17FF	%IX1000.0 …%IX1383.15	6144 PFC input bit variables in the first 384 words (768 bytes) of the 2 kB input process image
6144 … 12287	0x1800 ... 0x2FFF	%QX1000.0 … 1383.15	6144 PFC output bit variables in the first 384 words (768 bytes) of the 2 kB MODBUS output process image
12288 ... 65535	0x3000 ... 0xFFFF	%MX0.0 … %MX3327.15	Flag area: 53248 bit flags (6.5 kB) in the bit-addressable flag area

7.6.3.2 Write coils with FC5 and FC15:

PFC200: MODBUS Addresses for FC5 and FC15			
MODBUS Address		Memory Area	Description
[dec]	[hex]		
0 … 6143	0x0000 ... 0x17FF	%IX1000.0 …%IX1383.15	6144 PFC input bit variables in the first 384 words (768 bytes) of the 2 kB input process image
6144 … 12287	0x1800 ... 0x2FFF		MODBUS Exception: “Illegal data address”
12288 ... 65535	0x3000 ... 0xFFFF	%MX0.0 … %MX3327.15	Flag area: 53248 bit flags (6.5 kB) in the bit-addressable flag area

7.6.3.3 MODBUS Configuration Registers of the PFC200

The configuration registers make it possible to determine and in part change the properties of the PFC200.

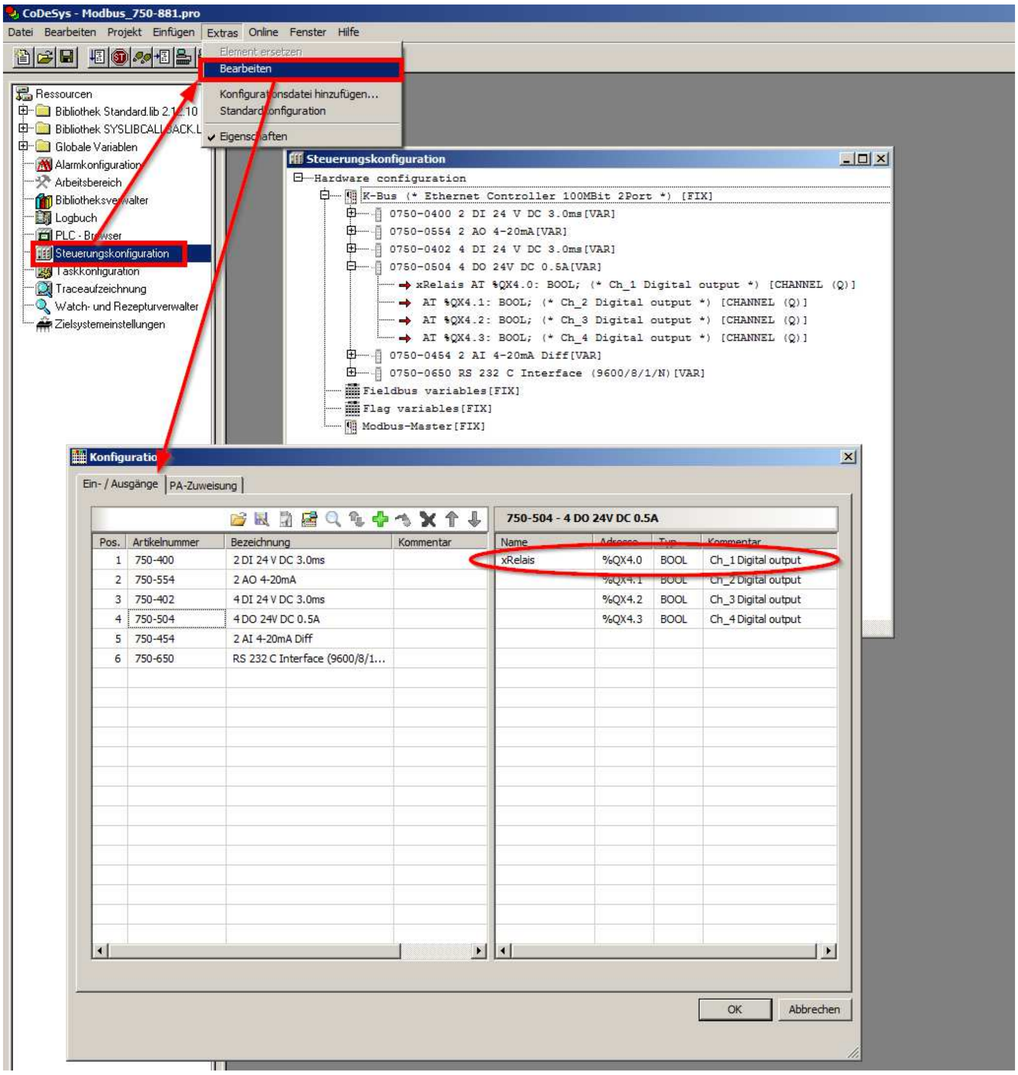
PFC200: MODBUS Configuration Register for FC3, FC4, FC6 and FC16				
MODBUS Address		**Length**	**Access**	**Description**
[dec]	**[hex]**	**[Word]**		
4130	0x1022	1	R	Number of registers in the output process image
4131	0x1023	1	R	Number of registers in the input process image
4132	0x1024	1	R	Number of bits in the output process image
4133	0x1025	1	R	Number of bits in the input process image
4136	0x1028	1	R	IP configuration: BootP(1), DHCP(2) or fixed, coded IP address(4)
4138	0x102A	1	R	Number of established TCP connections
4144	0x1030	1	R/W	MODBUS TCP time-out
4145	0x1031	3	R	MAC ID or ETHERNET interface
4151	0x1037	1	R/W	MODBUS TCP response delay
4160	0x1040	1	R	PLC status
4352	0x1100	1	W	Watchdog Command
4353	0x1101	1	R	Watchdog Status
4354	0x1102	1	R/W	Watchdog time-out
4355	0x1103	1	R/W	Watchdog Config
8192	0x2000	1	R	0x0000 (Constant)
8193	0x2001	1	R	0xFFFF (Constant)
8194	0x2002	1	R	0x1234 (Constant)
8195	0x2003	1	R	0xAAAA (Constant)
8196	0x2004	1	R	0x5555 (Constant)
8197	0x2005	1	R	0x7FFF (Constant)
8198	0x2006	1	R	0x8000 (Constant)
8199	0x2007	1	R	0x3FFF (Constant)
8200	0x2008	1	R	0x4000 (Constant)
8208	0x2010	1	R	Revision (Firmware Index)
8209	0x2011	1	R	Series code
8210	0x2012	1	R	Device code
8211	0x2013	1	R	Major Firmware Version
8212	0x2014	1	R	Minor Firmware Version
8213	0x2015	1	R	MBS Version

8 Appendix C: Useful Tools

8.1 CODESYS Controller Configuration

The CODESYS controller configuration provides convenience functions such as node scan and automatic address calculation.
With the import and export function, external tools can be used to process the controller configuration. That can be useful, for example, when the designations for the data points have to be applied from a CAE program (e.g., Eplan).

The advantage of the controller configuration is automatic calculation of the IEC addresses:



```
Hardware configuration
  K-Bus (* Ethernet Controller 100MBit 2Port *) [FIX]
    0750-0400 2 DI 24 V DC 3.0ms[VAR]
    0750-0554 2 AO 4-20mA[VAR]
    0750-0402 4 DI 24 V DC 3.0ms[VAR]
    0750-0504 4 DO 24V DC 0.5A[VAR]
      xRelais AT %QX4.0: BOOL; (* Ch_1 Digital output *) [CHANNEL (Q)]
       AT %QX4.1: BOOL; (* Ch_2 Digital output *) [CHANNEL (Q)]
       AT %QX4.2: BOOL; (* Ch_3 Digital output *) [CHANNEL (Q)]
       AT %QX4.3: BOOL; (* Ch_4 Digital output *) [CHANNEL (Q)]
    0750-0454 2 AI 4-20mA Diff[VAR]
    0750-0650 RS 232 C Interface (9600/8/1/N)[VAR]
  Fieldbus variables[FIX]
  Flag variables[FIX]
  Modbus-Master[FIX]
```

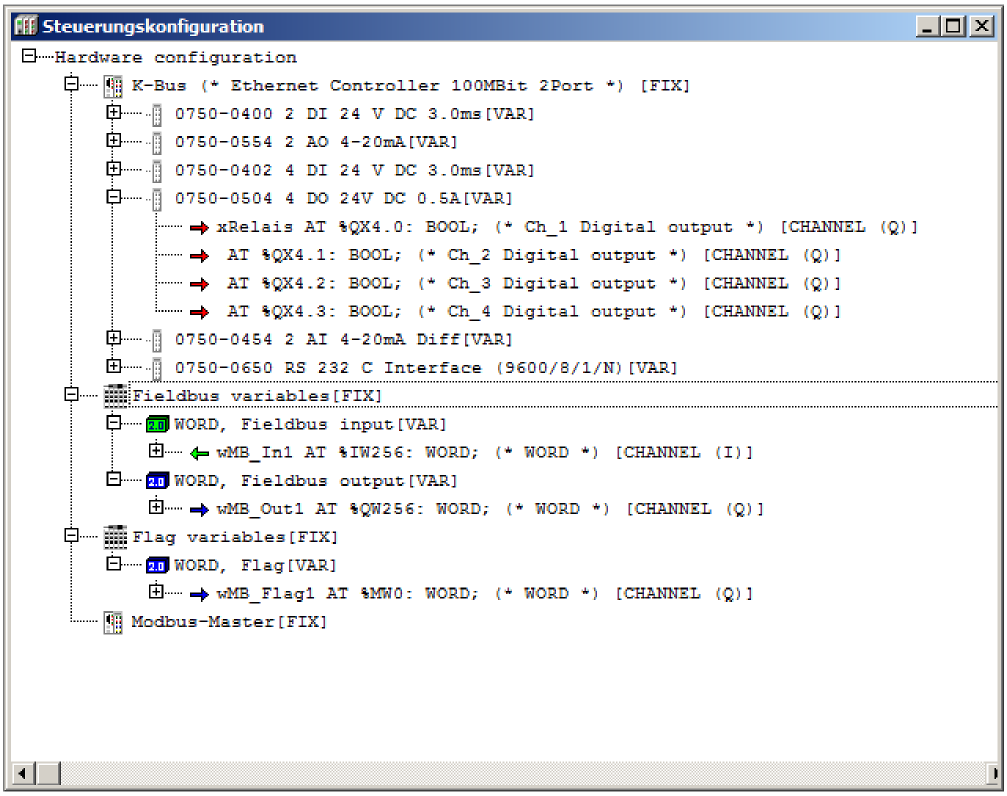
Pos.	Artikelnummer	Bezeichnung	Kommentar
1	750-400	2 DI 24 V DC 3.0ms	
2	750-554	2 AO 4-20mA	
3	750-402	4 DI 24 V DC 3.0ms	
4	750-504	4 DO 24V DC 0.5A	
5	750-454	2 AI 4-20mA Diff	
6	750-650	RS 232 C Interface (9600/8/1…	

750-504 - 4 DO 24V DC 0.5A

Name	Adresse	Typ	Kommentar
xRelais	%QX4.0	BOOL	Ch_1 Digital output
	%QX4.1	BOOL	Ch_2 Digital output
	%QX4.2	BOOL	Ch_3 Digital output
	%QX4.3	BOOL	Ch_4 Digital output

In contract to “manual addressing”, variable names are not bound to an address, but to the channel of the respective I/O module. As a result, the variable name automatically changes if an I/O module is added or removed in the controller configuration.

PFC variables and flags can also be declared via the controller configuration:

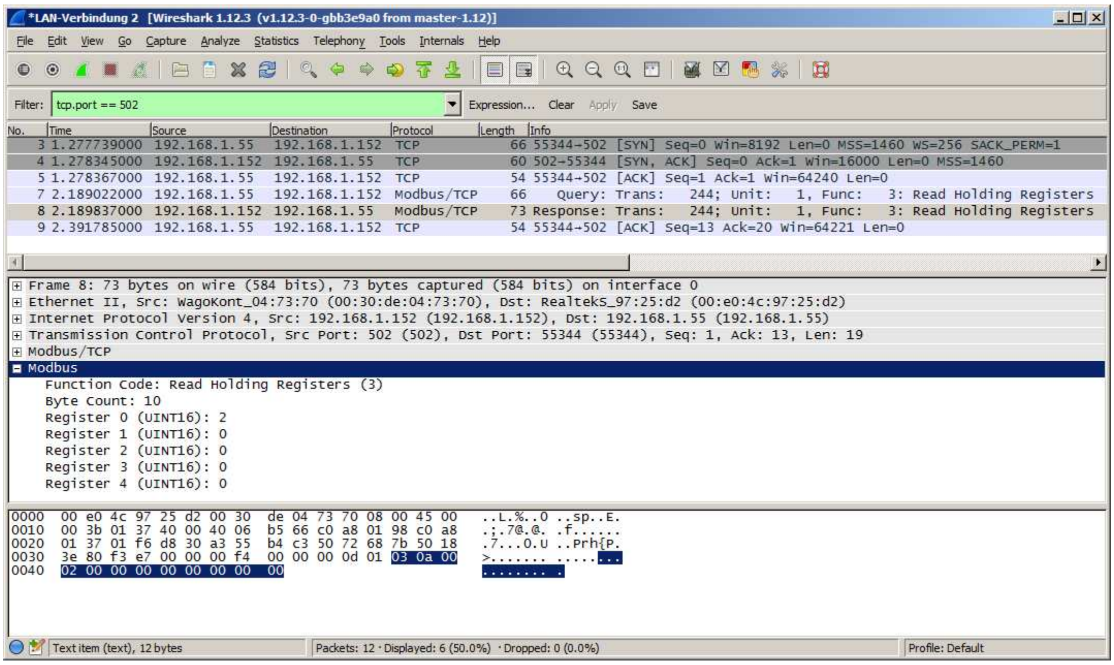


```
Hardware configuration
    K-Bus (* Ethernet Controller 100MBit 2Port *) [FIX]
        0750-0400 2 DI 24 V DC 3.0ms[VAR]
        0750-0554 2 AO 4-20mA[VAR]
        0750-0402 4 DI 24 V DC 3.0ms[VAR]
        0750-0504 4 DO 24V DC 0.5A[VAR]
            xRelais AT %QX4.0: BOOL; (* Ch_1 Digital output *) [CHANNEL (Q)]
             AT %QX4.1: BOOL; (* Ch_2 Digital output *) [CHANNEL (Q)]
             AT %QX4.2: BOOL; (* Ch_3 Digital output *) [CHANNEL (Q)]
             AT %QX4.3: BOOL; (* Ch_4 Digital output *) [CHANNEL (Q)]
        0750-0454 2 AI 4-20mA Diff[VAR]
        0750-0650 RS 232 C Interface (9600/8/1/N)[VAR]
    Fieldbus variables[FIX]
        WORD, Fieldbus input[VAR]
            wMB_In1 AT %IW256: WORD; (* WORD *) [CHANNEL (I)]
        WORD, Fieldbus output[VAR]
            wMB_Out1 AT %QW256: WORD; (* WORD *) [CHANNEL (Q)]
    Flag variables[FIX]
        WORD, Flag[VAR]
            wMB_Flag1 AT %MW0: WORD; (* WORD *) [CHANNEL (Q)]
    Modbus-Master[FIX]
```

8.2 “Wireshark”

A helpful tool at start-up or troubleshooting in the ETHERNET is the open source tool “Wireshark”. The tools is a network sniffer. The tool records the data traffic of a network interface and makes the data available in the form of individual packets.

Below is a screenshot of the “Wireshark” program.



Filter: tcp.port == 502

```
No.  Time         Source         Destination    Protocol    Length Info
   3 1.277739000 192.168.1.55   192.168.1.152  TCP            66 55344→502 [SYN] Seq=0 Win=8192 Len=0 MSS=1460 WS=256 SACK_PERM=1
   4 1.278345000 192.168.1.152  192.168.1.55   TCP            60 502→55344 [SYN, ACK] Seq=0 Ack=1 Win=16000 Len=0 MSS=1460
   5 1.278367000 192.168.1.55   192.168.1.152  TCP            54 55344→502 [ACK] Seq=1 Ack=1 Win=64240 Len=0
   7 2.189022000 192.168.1.55   192.168.1.152  Modbus/TCP     66    Query: Trans:   244; Unit:   1, Func:   3: Read Holding Registers
   8 2.189837000 192.168.1.152  192.168.1.55   Modbus/TCP     73 Response: Trans:   244; Unit:   1, Func:   3: Read Holding Registers
   9 2.391785000 192.168.1.55   192.168.1.152  TCP            54 55344→502 [ACK] Seq=13 Ack=20 Win=64221 Len=0
```

```
⊞ Frame 8: 73 bytes on wire (584 bits), 73 bytes captured (584 bits) on interface 0
⊞ Ethernet II, Src: WagoKont_04:73:70 (00:30:de:04:73:70), Dst: RealtekS_97:25:d2 (00:e0:4c:97:25:d2)
⊞ Internet Protocol Version 4, Src: 192.168.1.152 (192.168.1.152), Dst: 192.168.1.55 (192.168.1.55)
⊞ Transmission Control Protocol, Src Port: 502 (502), Dst Port: 55344 (55344), Seq: 1, Ack: 13, Len: 19
⊞ Modbus/TCP
⊟ Modbus
    Function Code: Read Holding Registers (3)
    Byte Count: 10
    Register 0 (UINT16): 2
    Register 1 (UINT16): 0
    Register 2 (UINT16): 0
    Register 3 (UINT16): 0
    Register 4 (UINT16): 0
```

```
0000  00 e0 4c 97 25 d2 00 30  de 04 73 70 08 00 45 00   ..L.%..0 ..sp..E.
0010  00 3b 01 37 40 00 40 06  b5 66 c0 a8 01 98 c0 a8   .;.7@.@. .f......
0020  01 37 01 f6 d8 30 a3 55  b4 c3 50 72 68 7b 50 18   .7...0.U ..Prh{P.
0030  3e 80 f3 e7 00 00 00 f4  00 00 00 0d 01 03 0a 00   >....... ........
0040  02 00 00 00 00 00 00 00  00                        ........ .
```

Text item (text), 12 bytes | Packets: 12 · Displayed: 6 (50.0%) · Dropped: 0 (0.0%) | Profile: Default

You can also see the response telegram from the example. The filter default “tcp.port == 502” ensures that only the telegrams are displayed on MODBUS port 502.

In contrast to a large group of applications, the recording of a communication is not started via the “File->New” menu item but via the menu item “Capture”. More information at www.wireshark.org.

WAGO Kontakttechnik GmbH & Co. KG

PO Box 2880 · D-32385 Minden
Hansastraße 27 • D-32423 Minden
Phone: +49 (0) 571/8 87 – 0
Fax: +49 (0) 571/8 87 – 1 69
Email: info@wago.com

Online: http://www.wago.com