BECKHOFF New Automation Technology

Manual | EN

TS6250-0030

TwinCAT 2 Modbus TCP Server CE

Supplement | Communication

2023-08-10 | Version: 1.2

Table of contents

1 Foreword

1.1 Notes on the documentation

This description is only intended for the use of trained specialists in control and automation engineering who are familiar with applicable national standards.
It is essential that the documentation and the following notes and explanations are followed when installing and commissioning the components.
It is the duty of the technical personnel to use the documentation published at the respective time of each installation and commissioning.

The responsible staff must ensure that the application or use of the products described satisfy all the requirements for safety, including all the relevant laws, regulations, guidelines and standards.

Disclaimer

The documentation has been prepared with care. The products described are, however, constantly under development.
We reserve the right to revise and change the documentation at any time and without prior announcement.
No claims for the modification of products that have already been supplied may be made on the basis of the data, diagrams and descriptions in this documentation.

Trademarks

Beckhoff®, TwinCAT®, TwinCAT/BSD®, TC/BSD®, EtherCAT®, EtherCAT G®, EtherCAT G10®, EtherCAT P®, Safety over EtherCAT®, TwinSAFE®, XFC®, XTS® and XPlanar® are registered trademarks of and licensed by Beckhoff Automation GmbH.
Other designations used in this publication may be trademarks whose use by third parties for their own purposes could violate the rights of the owners.

Patent Pending

The EtherCAT Technology is covered, including but not limited to the following patent applications and patents:
EP1590927, EP1789857, EP1456722, EP2137893, DE102015105702
with corresponding applications or registrations in various other countries.

EtherCAT® is a registered trademark and patented technology, licensed by Beckhoff Automation GmbH, Germany

Copyright

1.2 For your safety

Safety regulations

Read the following explanations for your safety.
Always observe and follow product-specific safety instructions, which you may find at the appropriate places in this document.

Exclusion of liability

All the components are supplied in particular hardware and software configurations which are appropriate for the application. Modifications to hardware or software configurations other than those described in the documentation are not permitted, and nullify the liability of Beckhoff Automation GmbH & Co. KG.

Personnel qualification

This description is only intended for trained specialists in control, automation, and drive technology who are familiar with the applicable national standards.

Signal words

The signal words used in the documentation are classified below. In order to prevent injury and damage to persons and property, read and follow the safety and warning notices.

Personal injury warnings

⚠ DANGER
Hazard with high risk of death or serious injury.

⚠ WARNING
Hazard with medium risk of death or serious injury.

⚠ CAUTION
There is a low-risk hazard that could result in medium or minor injury.

Warning of damage to property or environment

NOTICE
The environment, equipment, or data may be damaged.

Information on handling the product

This information includes, for example:
recommendations for action, assistance or further information on the product.

1.3 Notes on information security

The products of Beckhoff Automation GmbH & Co. KG (Beckhoff), insofar as they can be accessed online, are equipped with security functions that support the secure operation of plants, systems, machines and networks. Despite the security functions, the creation, implementation and constant updating of a holistic security concept for the operation are necessary to protect the respective plant, system, machine and networks against cyber threats. The products sold by Beckhoff are only part of the overall security concept. The customer is responsible for preventing unauthorized access by third parties to its equipment, systems, machines and networks. The latter should be connected to the corporate network or the Internet only if appropriate protective measures have been set up.

In addition, the recommendations from Beckhoff regarding appropriate protective measures should be observed. Further information regarding information security and industrial security can be found in our https://www.beckhoff.com/secguide.

Beckhoff products and solutions undergo continuous further development. This also applies to security functions. In light of this continuous further development, Beckhoff expressly recommends that the products are kept up to date at all times and that updates are installed for the products once they have been made available. Using outdated or unsupported product versions can increase the risk of cyber threats.

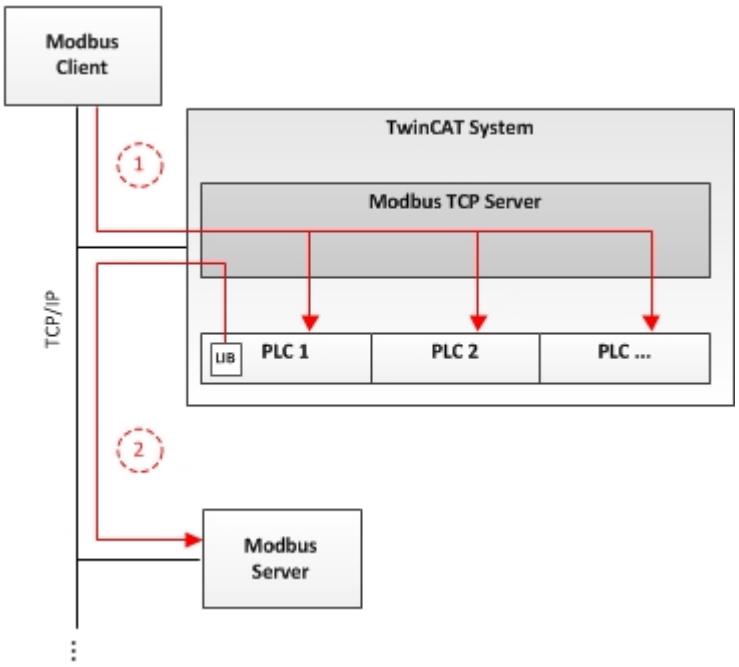
To stay informed about information security for Beckhoff products, subscribe to the RSS feed at https://www.beckhoff.com/secinfo.

2 Overview

The TwinCAT Modbus TCP server enables to communicate over a network connection (TCP/IP) with the Modbus protocol.

Modbus is an open standard in industrial communication which will be maintained by the independent Modbus Organization.

The protocol is based on a client/server-architecture. Therefore the product can be used as client or as server:



Server functionality:

(1) The TwinCAT Modbus TCP server enables to access the TwinCAT PLC. The Modbus register and I/O's are then mapped to TwinCAT PLC areas.

Client functionality: [▶ 18]

(2) The supplied PLC-library allows to communicate with other Modbus devices to request data (e.g. measured values, states) and control them.

3 System requirements

Technical Data	TwinCAT Modbsu TCP Server
Target system	Windows CE (x86-ARM compatible)
Min. TwinCAT Version	2.8.0
Min. TwinCAT Level	TwinCAT PC

4 Installation

This part of the documentation gives a step-by-step explaination of the TwinCAT OPC-UA setup process for Windows XP based operating systems. The following topics are part of this document:

- Downloading the setup file
- Starting the installation
- After the installation

Downloading the setup file

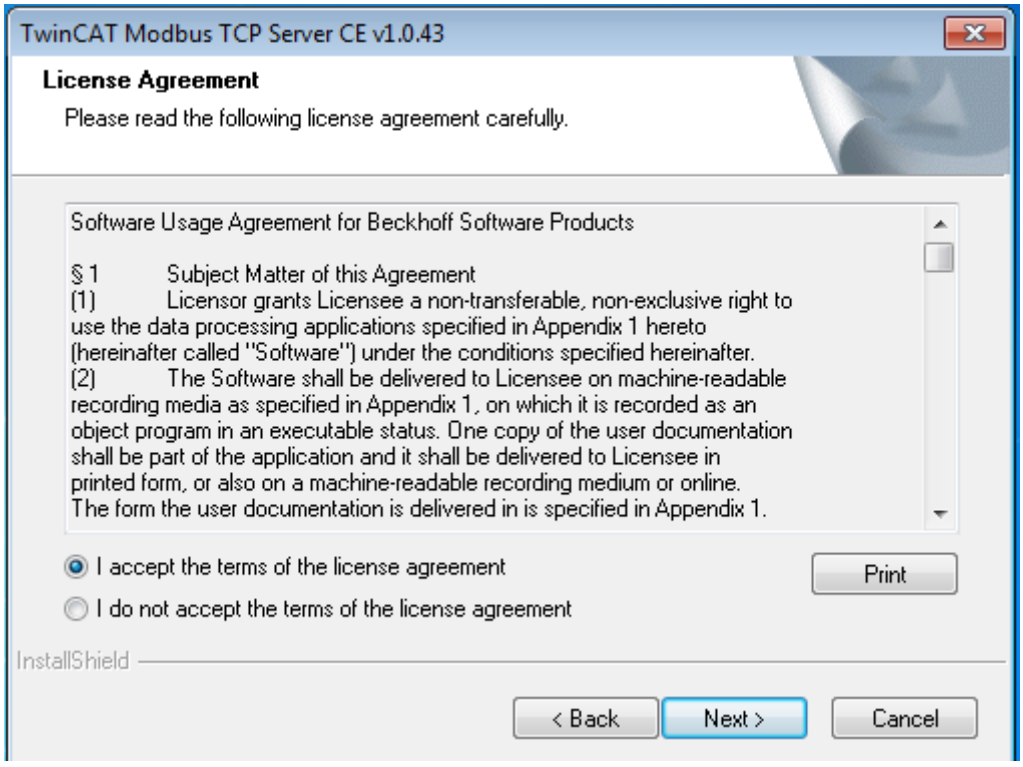
Like many other TwinCAT Supplement products, Modbus TCP is available for download in the Beckhoff download finder. The download represents the most current version, which can be licensed either as a 30-Day Demo or as a full version. To download the setup file, please perform the following steps:

- Open a connection to Beckhoff download finder.
- Select TS6250-0030 TwinCAT Modus Server and download the file via the download-cart.

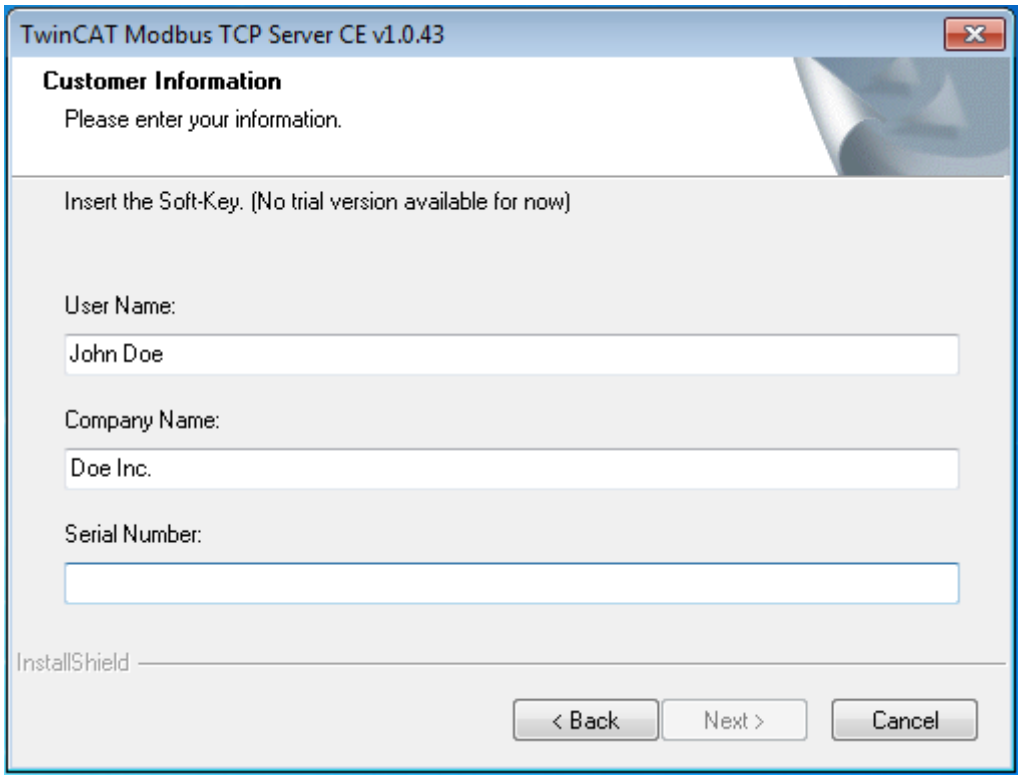
Starting the installation

To install the Supplement, please perform the following steps:

- Double-click the downloaded setup file**. Please note:** Under Windows 7 32-bit/64-bit, please start the installation with "Run as Administrator" by right-clicking the setup file and selecting the corresponding option in the context menu.
- Select an installation language
- Click on "Next" and accept the license agreement



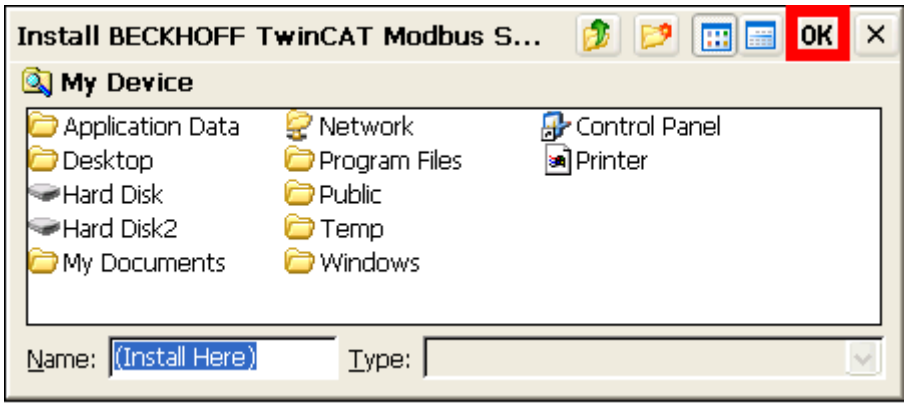
- Enter your user information. All fields are mandatory.



- Click on ""Install" to start installation
- At the end of the setup process, you will find the CAB-installation-files in this folder: **/TwinCAT/CE/TCModbusTCP/Install**

Windows CE Installation

- Copy the appropriate CAB file to your target CE-system (e.g., USB stick, network share)
- use TcModbusTcpSvrCe.ARMV4I.cab for ARM devices (e.g., CX9000) and TcModbusTcpSvrCe.I586 for x86 devices (e.g., CX5000)
- Execute the cab file on the target and accept the following dialog with OK.



Note: The installer selects as target folder /Hard Disk/System by default.

- After the installation you must restart your CE device.

After the installation

The Supplement "TwinCAT Modbus TCP" is automatically configured by setup and provide the default mapping to the TwinCAT PLC.

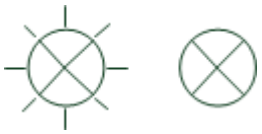
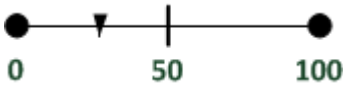
Also see about this

5 Configuration

The server can receive Modbus functions via TCP/IP.

Modbus areas

The Modbus specification defines these four Modbus-areas:

Modbus-areas	Data type	Access	Example
digital inputs (Discrete Inputs)	1 Bit	Read only	
digital outputs (Coils)	1 Bit	Read / write	
Input registers	16 Bit	Read only	
Output registers	16 Bit	Read / write	

After the installation the modbus areas are mapped to the PLC areas. Check the article about the default-mapping [▶ 14].

The mapping can be changed by the configuration file [▶ 13] TcModbusSrv.xml.

ADS-Access

If you want to access the specific modbus areas, you have to add these global variables to your PLC project.

```
VAR_GLOBAL
    mb_Input_Coils        : ARRAY [0..255] OF BOOL;
    mb_Output_Coils       : ARRAY [0..255] OF BOOL;
    mb_Input_Registers    : ARRAY [0..255] OF WORD;
    mb_Output_Registers   : ARRAY [0..255] OF WORD;
END_VAR
```

5.1 XML config

If you want to change the default mapping [▶ 14], you have to create and edit the configuration file **TcModbusSrv.xml** in the location **/Hard Disk/System**.

Hint: The mapping will be enabled after reboot.

Example for a simple mapping:

```
<Configuration>
    <Port>502</Port>
    <IpAddr/>
    <Mapping>
        <InputRegisters>
            <MappingInfo>
              <!-- Port 801 = PLC1 TC2 -->
                <AdsPort>801</AdsPort>
                <StartAddress>0</StartAddress>
                <EndAddress>32767</EndAddress>
            <!-- IndexGroup 61472 = 0xF020 -> physical plc input register %I -->
                <IndexGroup>61472</IndexGroup>
                <IndexOffset>0</IndexOffset>
            </MappingInfo>
```

```
            <MappingInfo>
                <AdsPort>801</AdsPort>
            <!-- Modbus input registers -->
                <StartAddress>32768</StartAddress>
                <EndAddress>33023</EndAddress>
                <VarName>.mb_Input_Registers</VarName>
            </MappingInfo>
        </InputRegisters>;
        <OutputRegisters/>
        <InputCoils/>
        <OutputCoils/>
    </Mapping>
</Configuration>
```

This samples maps the input register (IndexGroup 0xF020) of the first TwinCAT2 runtime (port = 801) to the Modbus input registers.

Hint: It is possible to map by variablename or IndexGroup/Offset(better performance).

More XML in the standard configuratuin can be found under Default Modbus-Mapping [▶ 14].

5.2 Default-Configuration

The standard settings are shown in the following table:

Modbus areas	Modbus address	ADS area	
Digital inputs	0x0000 - 0x7FFF	**Index group**	**Index offset**
		0xF021 - process image of the physical inputs (bit access)	0x0
	0x8000 - 0x80FF	**Name of the variables in the PLC program**	**Data type**
		.mb_Input_Coils	ARRAY [0..255] OF BOOL
Digital outputs (coils)	0x0000 - 0x7FFF	**Index group**	**Index offset**
		0xF031 - process image of the physical outputs (bit access)	0x0
	0x8000 - 0x80FF	**Name of the variables in the PLC program**	**Data type**
		.mb_Output_Coils	ARRAY [0..255] OF BOOL
Input registers	0x0000 - 0x7FFF	**Index group**	**Index offset**
		0xF020 - process image of the physical inputs	0x0
	0x8000 - 0x80FF	**Name of the variables in the PLC program**	**Data type**
		.mb_Input_Registers	ARRAY [0..255] OF WORD
Output registers	0x0000 - 0x2FFF	**Index group**	**Index offset**
		0xF030 - process image of the physical outputs	0x0
	0x3000 - 0x5FFF	0x4020 - PLC memory area	0x0
	0x6000 - 0x7FFF	0x4040 - PLC data area	0x0
	0x8000 - 0x80FF	**Name of the variables in the PLC program**	**Data type**
		.mb_Output_Registers	ARRAY [0..255] OF WORD

The server enables the access to the physical process image and maps the plc data area.

Default XML

The default configuration with optional comments for a better explanation:

```
Configuration>
    <!-- Modbus TCP port, default = 502-->
    <Port>502</Port>
    <!-- optional IP configuration for Modbus TCP server-->
    <IpAddr/>
    <Mapping>
    <InputCoils>
        <MappingInfo>
        <!-- AdsPort: TwinCAT2 PLC1 = 801, PLC2 = 811...-->
        <AdsPort>801</AdsPort>
        <StartAddress>0</StartAddress>
        <EndAddress>32767</EndAddress>
        <!-- IndexGroup 61473 = 0xF021 -> physical plc inputs %IX -->
        <IndexGroup>61473</IndexGroup>
        <!-- Bit offset-->
        <IndexOffset>0</IndexOffset>
        </MappingInfo>
        <MappingInfo>
        <AdsPort>801</AdsPort>
        <!-- Modbus input coils -->
        <StartAddress>32768</StartAddress>
        <EndAddress>33023</EndAddress>
        <VarName>.mb_Input_Coils</VarName>
        </MappingInfo>
    </InputCoils>
    <OutputCoils>
        <MappingInfo>
        <AdsPort>801</AdsPort>
        <EndAddress>32767</EndAddress>
        <!-- IndexGroup 61489 = 0xF031 -> physical plc outputs %QX -->
        <IndexGroup>61489</IndexGroup>
        <!-- Bit offset-->
        <IndexOffset>0</IndexOffset>
        </MappingInfo>
        <MappingInfo>
        <AdsPort>801</AdsPort>
        <!-- Modbus output coils-->
        <StartAddress>32768</StartAddress>
        <EndAddress>33023</EndAddress>
        <VarName>.mb_Output_Coils</VarName>
        </MappingInfo>
    </OutputCoils>
    <InputRegisters>
        <MappingInfo>
        <AdsPort>801</AdsPort>
        <StartAddress>0</StartAddress>
        <EndAddress>32767</EndAddress>
        <!-- IndexGroup 61472 = 0xF020 -> physical plc input register %I -->
        <IndexGroup>61472</IndexGroup>
        <!-- Byte offset-->
        <IndexOffset>0</IndexOffset>
        </MappingInfo>
        <MappingInfo>
        <AdsPort>801</AdsPort>
        <!-- Modbus input registers -->
        <StartAddress>32768</StartAddress>
        <EndAddress>33023</EndAddress>
        <VarName>.mb_Input_Registers</VarName>
        </MappingInfo>
    </InputRegisters>
    <OutputRegisters>
        <MappingInfo>
        <AdsPort>801</AdsPort>
        <StartAddress>0</StartAddress>
        <EndAddress>12287</EndAddress>
        <!-- IndexGroup 61488 = 0xF030 -> physical plc output register %Q -->
        <IndexGroup>61488</IndexGroup>
        <!-- Byte offset-->
        <IndexOffset>0</IndexOffset>
        </MappingInfo>
        <MappingInfo>
        <AdsPort>801</AdsPort>
        <StartAddress>12288</StartAddress>
        <EndAddress>24575</EndAddress>
        <!-- IndexGroup 16416 = 0x4020 -> plc memory area %M -->
        <IndexGroup>16416</IndexGroup>
```

```
        <!-- Byte offset-->
        <IndexOffset>0</IndexOffset>
        </MappingInfo>
        <MappingInfo>
        <AdsPort>801</AdsPort>
        <StartAddress>24576</StartAddress>
        <EndAddress>32767</EndAddress>
        <!-- IndexGroup 16448 = 0x4040 -> plc data area -->
        <IndexGroup>16448</IndexGroup>
        <!-- Byte offset-->
        <IndexOffset>0</IndexOffset>
        </MappingInfo>
        <MappingInfo>
        <AdsPort>801</AdsPort>
        <!-- Modbus output registers -->
        <StartAddress>32768</StartAddress>
        <EndAddress>33023</EndAddress>
        <VarName>.mb_Output_Registers</VarName>
        </MappingInfo>
    </OutputRegisters>
    </Mapping>
</Configuration>
```

6 Modbus ADS Diagnose Interface

The following information can be requested via ADS:

Index Group	Index Offset	Access	Datatype	Description	Minimale Modbus Server Version
0x2000	0	ADS Read	UINT32	**GetConnectedClientCount** Return amount of connected Modbus clients	1.0.50
0x2000	1	ADS Read	UINT32	**GetModbusRequestCount** Return amount of received Modbus requests	1.0.50
0x2000	2	ADS Read	UINT32	**GetModbusResponseCount** Return amount of sent Modbus responses	1.0.50

7 PLC libraries

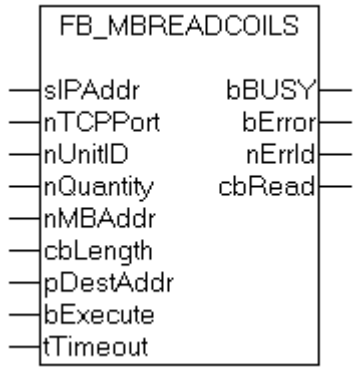
7.1 TcModbusSrv

The defined modbus functions are implemented in the PLC library TcModbusSrv.lib.

Modbus TCP function	Function code	PLC block
Read Coils	1	FB_MBReadCoils [▶ 18]
Read Discrete Inputs	2	FB_MBReadInputs [▶ 20]
Read Registers	3	FB_MBReadRegs [▶ 22]
Read Input Registers	4	FB_MBReadInputRegs [▶ 24]
Write Single Coil	5	FB_MBWriteSingleCoil [▶ 26]
Write Single Register	6	FB_MBWriteSingleReg [▶ 28]
Write Multiple Coils	15	FB_MBWriteCoils [▶ 29]
Write Multiple Registers	16	FB_MBWriteRegs [▶ 31]
Read/Write Multiple Registers	23	FB_MBReadWriteRegs [▶ 33]
Diagnostic	8	FB_MBDiagnose [▶ 35]

Modbus UDP function	Function code	PLC block
Read Coils	1	FB_MBUdpReadCoils [▶ 37]
Read Discrete Inputs	2	FB_MBUdpReadInputs [▶ 38]
Read Registers	3	FB_MBUdpReadRegs [▶ 40]
Read Input Registers	4	FB_MBUdpReadInputRegs [▶ 41]
Write Single Coil	5	FB_MBUdpWriteSingleCoil [▶ 42]
Write Single Register	6	FB_MBUdpWriteSingleReg [▶ 43]
Write Multiple Coils	15	FB_MBUdpWriteCoils [▶ 44]
Write Multiple Registers	16	FB_MBUdpWriteRegs [▶ 46]
Read/Write Multiple Registers	23	FB_MBUdpReadWriteRegs [▶ 47]
Diagnostic	8	FB_MBUdpDiagnose [▶ 49]

7.1.1 FB_MBReadCoils (Modbus function 1)



This function is used for reading 1 to 2048 digital outputs (coils). One digital output corresponds to one bit of the read data bytes.

VAR_INPUT

```
VAR_INPUT
    sIPAddr         : STRING(15);
    nTCPPort        : UINT:= MODBUS_TCP_PORT;
    nUnitID         : BYTE:=16#FF;
    nQuantity       : WORD;
    nMBAddr         : WORD;
    cbLength        : UDINT;
    pDestAddr       : UDINT;
    bExecute        : BOOL;
    tTimeout        : TIME;
END_VAR
```

sIPAddr : Is a string containing the IP address of the target device.

nTCPPort : Port number of the target device.

nUnitID: Identification number of a serial sub-network device. If a device is addressed directly via TCP/IP, this value must be 16#FF.

nQuantity : Number of digital inputs (data bits) to be read. The value of *nQuantity* must be > 0.

nMBAddr : Start address of the digital inputs to be read (bit offset).

cbLength : Contains the max. byte size of the destination buffer into which the data are to be read. The minimum buffer byte size must be: *(nQuantity + 7) / 8*.

pDestAddr : Contains the address of the destination buffer into which the data are to be read. The buffer can be a single variable, an array or a structure, whose address can be found with the ADR operator.

bExecute: The function block is activated by a rising edge at this input.

tTimeout: States the length of the timeout that may not be exceeded by execution of the ADS command.

VAR_OUTPUT

```
VAR_OUTPUT
    bBUSY       : BOOL;
    bError      : BOOL;
    nErrId      : UDINT;
    cbRead      : UDINT;
END_VAR
```

bBusy : When the function block is activated this output is set. It remains set until an acknowledgement is received.

bError : If an ADS error should occur during the transfer of the command, then this output is set once the bBusy output is reset.

nErrId : Supplies the ADS error number when the bError output is set.

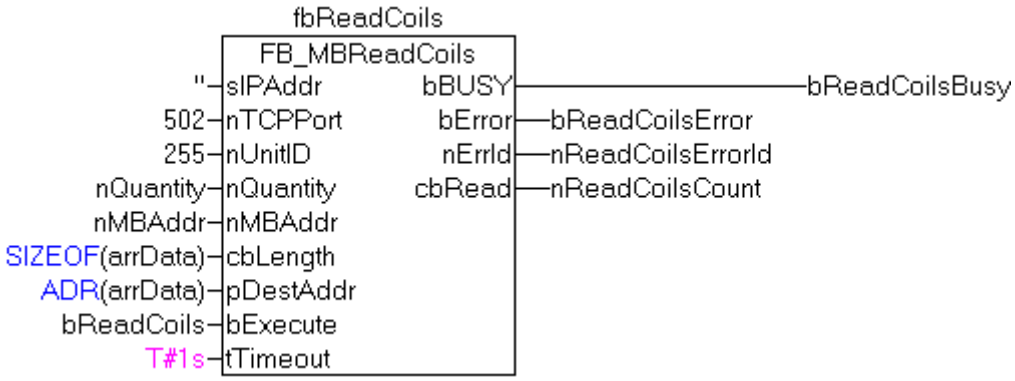
cbRead: Contains the number of bytes currently read.

Function specific ADS error code	Possible reason
0x8001	Modbus function not implemented
0x8002	Invalid address or length
0x8003	Invalid parameters: - wrong number of registers
0x8004	Modbus server error

Example of calling the block in FBD:

```
PROGRAM Test
VAR
    fbReadCoils         : FB_MBReadCoils;
    bReadCoils          : BOOL;
    bReadCoilsBusy      : BOOL;
    bReadCoilsError     : BOOL;
    nReadCoilsErrorId   : UDINT;
    nReadCoilsCount     : UDINT;
```

```
    nQuantity           : WORD := 10;
    nMBAddr             : WORD := 5;
    arrData             : ARRAY [1..2] OF BYTE;
END_VAR
```



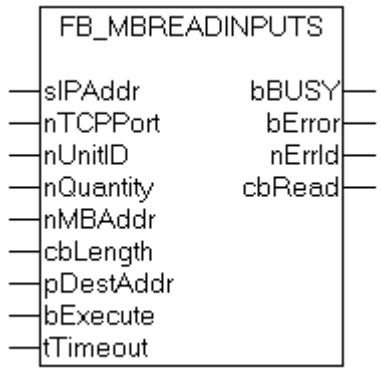
After a rising edge of "bExecute" and successful execution of the ReadCoils command, the content of digital outputs 6 - 15 is written into the arrData array:

Digital outputs	Array offset	Status
6-13	1	0x54 The status of output 13 is the MSB of this byte (left) The status of output 6 is the LSB of this byte (right)
14-15	2	0x02 Since only 10 outputs are to be read, the remaining bits (3-8) are set to 0.

Requirements

Development environment	Target system type	PLC libraries to be linked
TwinCAT v2.8.0	PC (i386)	TcModbusSrv.Lib

7.1.2 FB_MBReadInputs (Modbus function 2)



This function is used for reading 1 to 2048 digital inputs. One digital input corresponds to one bit of the read data bytes.

VAR_INPUT

```
VAR_INPUT
    sIPAddr         : STRING(15);
    nTCPPort        : UINT:= MODBUS_TCP_PORT;
```

```
    nUnitID         : BYTE:=16#FF;
    nQuantity       : WORD;
    nMBAddr         : WORD;
    cbLength        : UDINT;
    pDestAddr       : UDINT;
    bExecute        : BOOL;
    tTimeout        : TIME;
END_VAR
```

sIPAddr: Is a string containing the IP address of the target device.

nTCPPort: Port number of the target device.

nUnitID: Identification number of a serial sub-network device. If a device is addressed directly via TCP/IP, this value must be 16#FF.

nQuantity: Number of digital inputs (data bits) to be read. The *value of nQuantity* must be > 0.

nMBAddr: Start address of the digital inputs to be read (bit offset).

cbLength: Contains the max. byte size of the destination buffer. The minimum buffer byte size must be: *(nQuantity + 7) / 8*.

pDestAddr: Contains the address of the destination buffer into which the data are to be read. The buffer can be a single variable, an array or a structure, whose address can be found with the ADR operator.

bExecute: The function block is activated by a rising edge at this input.

tTimeout: States the length of the timeout that may not be exceeded by execution of the ADS command.

VAR_OUTPUT

```
VAR_OUTPUT
    bBUSY       : BOOL;
    bError      : BOOL;
    nErrId      : UDINT;
    cbRead      : UDINT;
END_VAR
```

bBusy : When the function block is activated this output is set. It remains set until an acknowledgement is received.

bError : If an ADS error should occur during the transfer of the command, then this output is set once the bBusy output is reset.

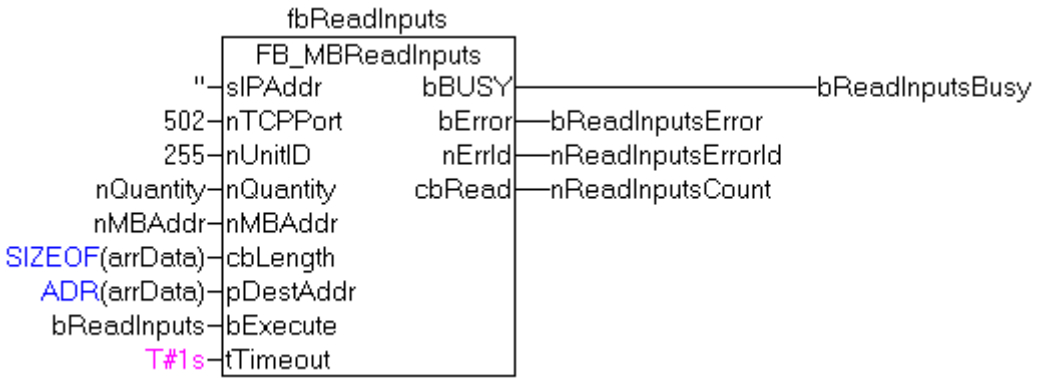
nErrId : Supplies the ADS error number when the bError output is set.

cbRead: Contains the number of bytes currently read.

Function specific ADS error code	Possible reason
0x8001	Modbus function not implemented
0x8002	Invalid address or length
0x8003	Invalid parameters: - wrong number of registers
0x8004	Modbus server error

Example of calling the block in FBD:

```
PROGRAM Test
VAR
    fbReadInputs        : FB_MBReadInputs;
    bReadInputs         : BOOL;
    bReadInputsBusy     : BOOL;
    bReadInputsError    : BOOL;
    nReadInputsErrorId  : UDINT;
    nReadInputsCount    : UDINT;
    nQuantity           : WORD := 20;
    nMBAddr             : WORD := 29;
    arrData             : ARRAY [1..3] OF BYTE;
END_VAR
```



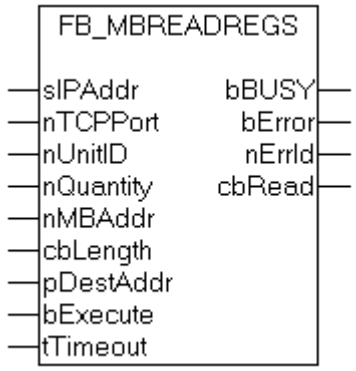
After a rising edge of "bExecute" and successful execution of the ReadInputs command, the content of digital inputs 30 - 49 is written into the arrData array:

Digital outputs	Array offset	Status
29-36	1	0x34 The status of inputs 36 is the MSB of this byte (left) The status of inputs 29 is the LSB of this byte (right)
37-44	2	0x56 The status of inputs 44 is the MSB of this byte (left) The status of inputs 37 is the LSB of this byte (right)
45-49	3	0x07 Since only 20 outputs are to be read, the remaining bits (5-8) are set to 0.

Requirements

Development environment	Target system type	PLC libraries to be linked
TwinCAT v2.8.0	PC (i386)	TcModbusSrv.Lib

7.1.3 FB_MBReadRegs (Modbus function 3)



This function is used for reading 1 to 128 output registers (16 bit). The first byte contains the lower eight bits and the second byte the upper eight bits.

VAR_INPUT

```
VAR_INPUT
    sIPAddr          : STRING(15);
    nTCPPort         : UINT:= MODBUS_TCP_PORT;
    nUnitID          : BYTE:=16#FF;
    nQuantity        : WORD;
```

```
    nMBAddr         : WORD;
    cbLength        : UDINT;
    pDestAddr       : UDINT;
    bExecute        : BOOL;
    tTimeout        : TIME;
END_VAR
```

sIPAddr: Is a string containing the IP address of the target device.

nTCPPort: Port number of the target device.

nUnitID: Identification number of a serial sub-network device. If a device is addressed directly via TCP/IP, this value must be 16#FF.

nQuantity: Number of output registers (data words) to be read. The value of *nQuantity* must be > 0.

nMBAddr: Start address of the output registers to be read (word offset).

cbLength: Contains the max. byte size of the destination buffer. The minimum buffer byte size must be: *nQuantity * 2*.

pDestAddr: Contains the address of the destination buffer into which the data are to be read. The buffer can be a single variable, an array or a structure, whose address can be found with the ADR operator.

bExecute: The function block is activated by a rising edge at this input.

tTimeout: States the length of the timeout that may not be exceeded by execution of the ADS command.

VAR_OUTPUT

```
VAR_OUTPUT
    bBUSY       : BOOL;
    bError      : BOOL;
    nErrId      : UDINT;
    cbRead      : UDINT;
END_VAR
```

bBusy : When the function block is activated this output is set. It remains set until an acknowledgement is received.

bError : If an ADS error should occur during the transfer of the command, then this output is set once the bBusy output is reset.

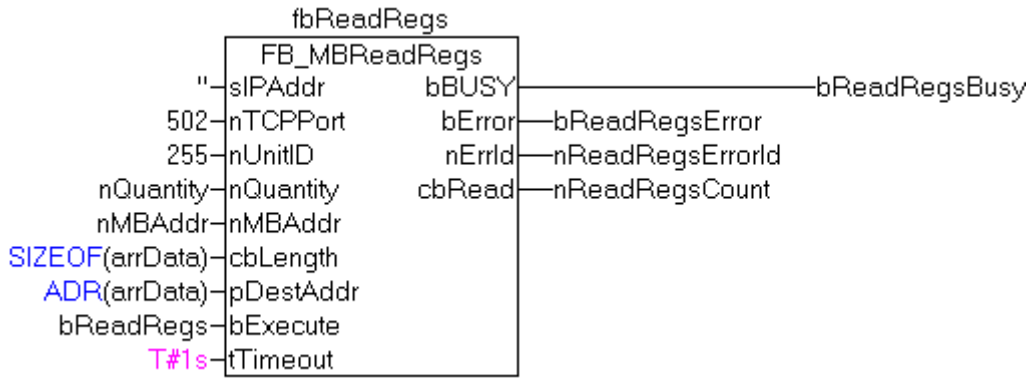
nErrId : Supplies the ADS error number when the bError output is set.

cbRead: Contains the number of bytes currently read.

Function specific ADS error code	Possible reason
0x8001	Modbus function not implemented
0x8002	Invalid address or length
0x8003	Invalid parameters: - wrong number of registers
0x8004	Modbus server error

Example of calling the block in FBD:

```
PROGRAM Test
VAR
    fbReadRegs          : FB_MBReadRegs;
    bReadRegs           : BOOL;
    bReadRegsBusy       : BOOL;
    bReadRegsError      : BOOL;
    nReadRegsErrorId    : UDINT;
    nReadRegsCount      : UDINT;
    nQuantity           : WORD:=2;
    nMBAddr             : WORD:=24;
    arrData             : ARRAY [1..2] OF WORD;
END_VAR
```



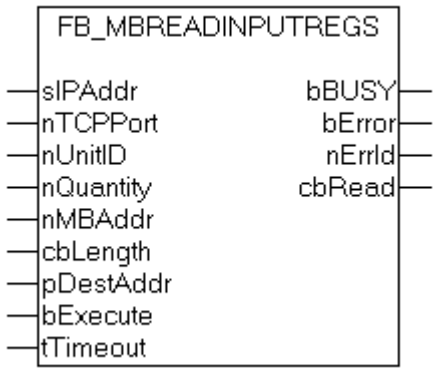
After a rising edge of "bExecute" and successful execution of the ReadRegs command, the content of registers 25 and 26 is in the arrData array:

Register	Array offset	Status
25	1	0x1234 (as byte 0x34 0x12)
26	2	0x5563 (as byte 0x63 0x55)

Requirements

Development environment	Target system type	PLC libraries to be linked
TwinCAT v2.8.0	PC (i386)	TcModbusSrv.Lib

7.1.4 FB_MBReadInputRegs (Modbus function 4)



This function is used for reading 1 to 128 input registers (16 bit). Endian

VAR_INPUT

```
VAR_INPUT
    sIPAddr         : STRING(15);
    nTCPPort        : UINT:= MODBUS_TCP_PORT;
    nUnitID         : BYTE:=16#FF;
    nQuantity       : WORD;
    nMBAddr         : WORD;
    cbLength        : UDINT;
    pDestAddr       : UDINT;
    bExecute        : BOOL;
    tTimeout        : TIME;
END_VAR
```

sIPAddr: Is a string containing the IP address of the target device.

nTCPPort: Port number of the target device.

nUnitID: Identification number of a serial sub-network device. If a device is addressed directly via TCP/IP, this value must be 16#FF.

nQuantity: Number of input registers (data words) to be read. The value of *nQuantity* must be > 0.

nMBAddr: Start address of the input register to be read (word offset).

cbLength: Contains the max. byte size of the destination buffer. The minimum buffer byte size must be: *nQuantity * 2*.

pDestAddr: Contains the address of the destination buffer into which the data are to be read. The buffer can be a single variable, an array or a structure, whose address can be found with the ADR operator.

bExecute: The function block is activated by a rising edge at this input.

tTimeout: States the length of the timeout that may not be exceeded by execution of the ADS command.

VAR_OUTPUT

```
VAR_OUTPUT
    bBUSY       : BOOL;
    bError      : BOOL;
    nErrId      : UDINT;
    cbRead      : UDINT;
END_VAR
```

bBusy : When the function block is activated this output is set. It remains set until an acknowledgement is received.

bError : If an ADS error should occur during the transfer of the command, then this output is set once the bBusy output is reset.

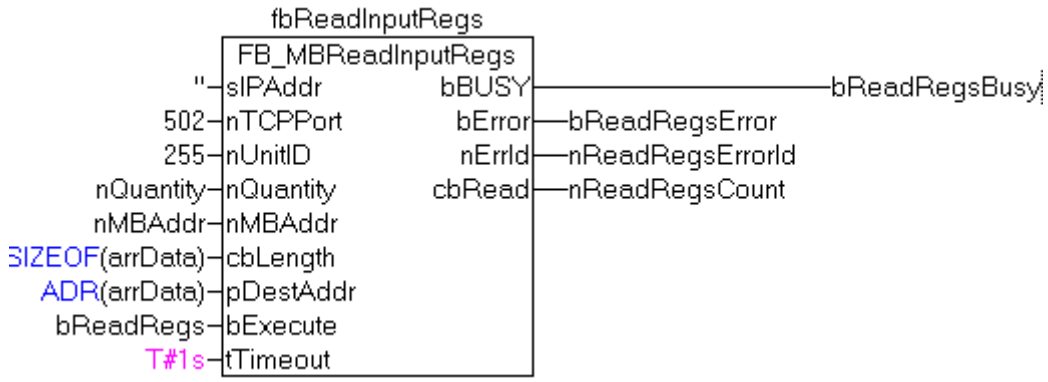
nErrId : Supplies the ADS error number when the bError output is set.

cbRead: Contains the number of bytes currently read.

Function specific ADS error code	Possible reason
0x8001	Modbus function not implemented
0x8002	Invalid address or length
0x8003	Invalid parameters: - wrong number of registers
0x8004	Modbus server error

Example of calling the block in FBD:

```
PROGRAM Test
VAR
    fbReadInputRegs      : FB_MBReadInputRegs;
    bReadRegs            : BOOL;
    bReadRegsBusy        : BOOL;
    bReadRegsError       : BOOL;
    nReadRegsErrorId     : UDINT;
    nReadRegsCount       : UDINT;
    nQuantity            : WORD := 3;
    nMBAddr              : WORD:= 2;
    arrData              : ARRAY [1..3] OF WORD;
END_VAR
```



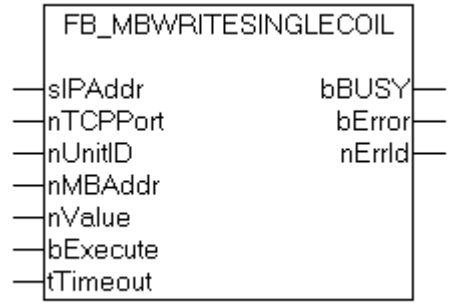
After a rising edge of "bExecute" and successful execution of the ReadRegs command, the content of registers 3-5 is in the arrData array:

Register	Array offset	Status
3	1	0x4543 (as byte 0x43 0x45)
4	2	0x5234 (as byte 0x34 0x52)
5	3	0x1235 (as byte 0x35 0x12)

Requirements

Development environment	Target system type	PLC libraries to be linked
TwinCAT v2.8.0	PC (i386)	TcModbusSrv.Lib

7.1.5 FB_MBWriteSingleCoil (Modbus function 5)



This function is used for writing a single digital output (coil). Bit access is used.

VAR_INPUT

```
VAR_INPUT
    sIPAddr         : STRING(15);
    nTCPPort        : UINT:= MODBUS_TCP_PORT;
    nUnitID         : BYTE:=16#FF;
    nMBAddr         : WORD;
    nValue          : WORD;
    bExecute        : BOOL;
    tTimeout        : TIME;
END_VAR
```

sIPAddr: Is a string containing the IP address of the target device.

nTCPPort: Port number of the target device.

nUnitID: Identification number of a serial sub-network device. If a device is addressed directly via TCP/IP, this value must be 16#FF.

nMBAddr: Address of the digital output (bit offset).

nValue: Value to be written into the digital output. The value 16#FF00 switches the output on, 16#0000 switches it off.

bExecute: The function block is activated by a rising edge at this input.

tTimeout: States the length of the timeout that may not be exceeded by execution of the ADS command.

VAR_OUTPUT

```
VAR_OUTPUT
    bBUSY       : BOOL;
    bError      : BOOL;
    nErrId      : UDINT;
END_VAR
```

bBusy : When the function block is activated this output is set. It remains set until an acknowledgement is received.

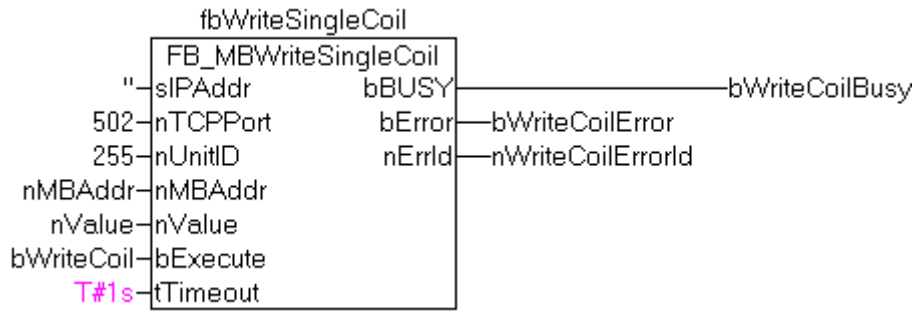
bError : If an ADS error should occur during the transfer of the command, then this output is set once the bBusy output is reset.

nErrId : Supplies the ADS error number when the bError output is set.

Function specific ADS error code	Possible reason
0x8001	Modbus function not implemented
0x8002	Invalid address or length
0x8003	Invalid parameters: - wrong number of registers
0x8004	Modbus server error

Example of calling the block in FBD:

```
PROGRAM Test
VAR
    fbWriteSingleCoil       : FB_MBWriteSingleCoil;
    bWriteCoil              : BOOL;
    bWriteCoilBusy          : BOOL;
    bWriteCoilError         : BOOL;
    nWriteCoilErrorId       : UDINT;
    nMBAddr                 : WORD := 3;
    nValue                  : WORD := 16#FF00;
END_VAR
```

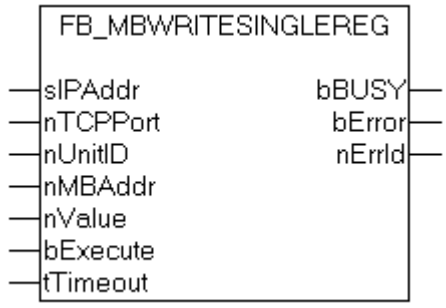


After a rising edge of "bExecute" and successful execution of the WriteSingleCoil command, digital output 4 is switched on.

Requirements

Development environment	Target system type	PLC libraries to be linked
TwinCAT v2.8.0	PC (i386)	TcModbusSrv.Lib

7.1.6 FB_MBWriteSingleReg (Modbus function 6)



This function is used for writing an individual output register. 16 bit access is used.

VAR_INPUT

```
VAR_INPUT
    sIPAddr         : STRING(15);
    nTCPPort        : UINT:= MODBUS_TCP_PORT;
    nUnitID         : BYTE:=16#FF;
    nMBAddr         : WORD;
    nValue          : WORD;
    bExecute        : BOOL;
    tTimeout        : TIME;
END_VAR
```

sIPAddr: Is a string containing the IP address of the target device.

nTCPPort: Port number of the target device.

nUnitID: Identification number of a serial sub-network device. If a device is addressed directly via TCP/IP, this value must be 16#FF.

nMBAddr: Address of the output register (word offset).

nValue: Value to be written into the register (word value).

bExecute: The function block is activated by a rising edge at this input.

tTimeout: States the length of the timeout that may not be exceeded by execution of the ADS command.

VAR_OUTPUT

```
VAR_OUTPUT
    bBUSY       : BOOL;
    bError      : BOOL;
    nErrId      : UDINT;
END_VAR
```

bBusy : When the function block is activated this output is set. It remains set until an acknowledgement is received.

bError : If an ADS error should occur during the transfer of the command, then this output is set once the bBusy output is reset.

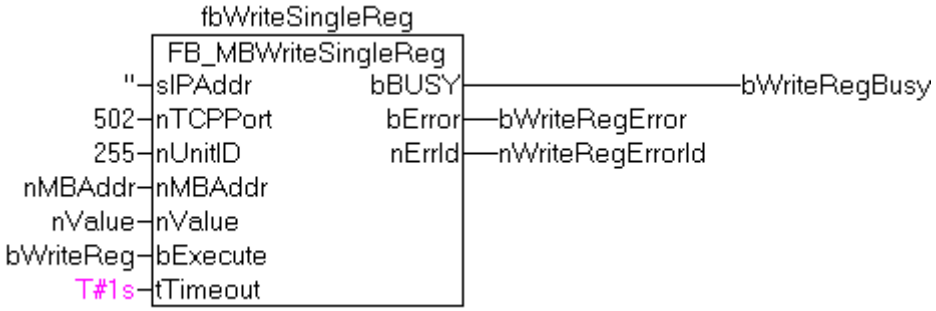
nErrId : Supplies the ADS error number when the bError output is set.

Function specific ADS error code	Possible reason
0x8001	Modbus function not implemented

Function specific ADS error code	Possible reason
0x8002	Invalid address or length
0x8003	Invalid parameters: - wrong number of registers
0x8004	Modbus server error

Example of calling the block in FBD:

```
PROGRAM Test
VAR
    fbWriteSingleReg    : FB_MBWriteSingleReg;
    bWriteReg           : BOOL;
    bWriteRegBusy       : BOOL;
    bWriteRegError      : BOOL;
    nWriteRegErrorId    : UDINT;
    nMBAddr             : WORD := 4;
    nValue              : WORD := 16#1234;
END_VAR
```

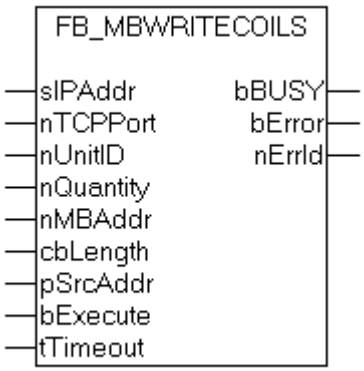


After a rising edge of "bExecute" and successful execution of the WriteSingleReg command, the value 16#1234 is written into register 5.

Requirements

Development environment	Target system type	PLC libraries to be linked
TwinCAT v2.8.0	PC (i386)	TcModbusSrv.Lib

7.1.7 FB_MBWriteCoils (Modbus function 15)



This function is used for writing 1 to 2048 digital outputs (coils). One digital output corresponds to one bit of the write data bytes.

VAR_INPUT

```
VAR_INPUT
    sIPAddr     : STRING(15);
    nTCPPort    : UINT:= MODBUS_TCP_PORT;
    nUnitID     : BYTE:=16#FF;
    nQuantity   : WORD;
    nMBAddr     : WORD;
    cbLength    : UDINT;
    pSrcAddr    : UDINT;
    bExecute    : BOOL;
    tTimeout    : TIME;
END_VAR
```

sIPAddr: Is a string containing the IP address of the target device.

nTCPPort: Port number of the target device.

nUnitID: Identification number of a serial sub-network device. If a device is addressed directly via TCP/IP, this value must be 16#FF.

nQuantity: Number of digital outputs to be written (data bits). *nQuantity* must be > 0.

nMBAddr: Start address of the digital outputs to be written (bit offset).

cbLength: Contains the max. byte size of the source buffer containing the data to be written. The minimum buffer byte size must be: *(nQuantity + 7) / 8*.

pSrcAddr: Contains the address of the source buffer containing the data to be written. The buffer can be a single variable, an array or a structure, whose address can be found with the ADR operator.

bExecute: The function block is activated by a rising edge at this input.

tTimeout: States the length of the timeout that may not be exceeded by execution of the ADS command.

VAR_OUTPUT

```
VAR_OUTPUT
    bBUSY       : BOOL;
    bError      : BOOL;
    nErrId      : UDINT;
END_VAR
```

bBusy : When the function block is activated this output is set. It remains set until an acknowledgement is received.

bError : If an ADS error should occur during the transfer of the command, then this output is set once the bBusy output is reset.

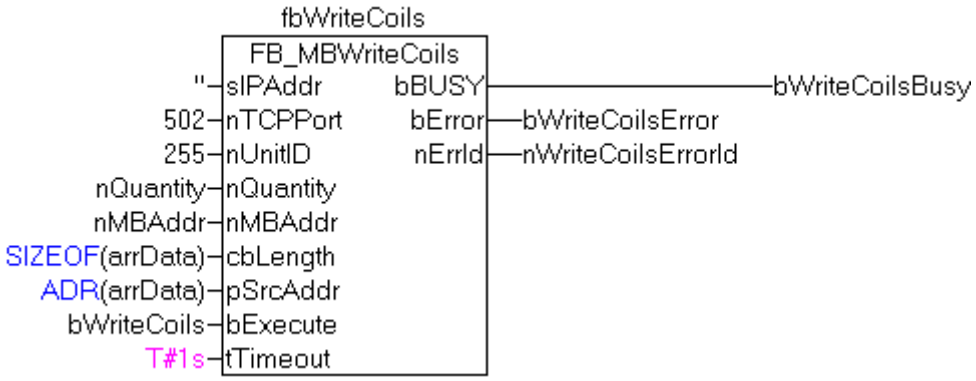
nErrId : Supplies the ADS error number when the bError output is set.

Function specific ADS error code	Possible reason
0x8001	Modbus function not implemented
0x8002	Invalid address or length
0x8003	Invalid parameters: - wrong number of registers
0x8004	Modbus server error

Example of calling the block in FBD:

```
PROGRAM Test
VAR
    fbWriteCoils        : FB_MBWriteCoils;
    bWriteCoils         : BOOL;
    bWriteCoilsBusy     : BOOL;
    bWriteCoilsError    : BOOL;
    nWriteCoilsErrorId  : UDINT;
    nWriteCoilsCount    : UDINT;
    nQuantity           : WORD := 10;
```

```
    nMBAddr             : WORD := 14;
    arrData             : ARRAY [1..2] OF BYTE := 16#75,16#03;
END_VAR
```



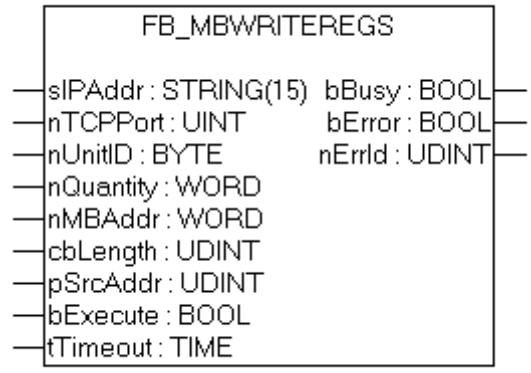
After a rising edge of "bExecute" and successful execution of the ReadCoils command, the content of the arrData array is written to digital outputs 15 - 24:

Bit	0	1	1	1	0	1	0	1	0	0	0	0	0	0	1	1
Output	22	21	20	19	18	17	16	15	X	X	X	X	X	X	24	23

Requirements

Development environment	Target system type	PLC libraries to be linked
TwinCAT v2.8.0	PC (i386)	TcModbusSrv.Lib

7.1.8 FB_MBWriteRegs(Modbus function 16)



This function is used for writing 1 to 128 output registers (16 bit).

VAR_INPUT

```
VAR_INPUT
    sIPAddr     : STRING(15);
    nTCPPort    : UINT:= MODBUS_TCP_PORT;
    nUnitID     : BYTE:=16#FF;
    nQuantity   : WORD;
    nMBAddr     : WORD;
    cbLength    : UDINT;
    pSrcAddr    : UDINT;
    bExecute    : BOOL;
    tTimeout    : TIME;
END_VAR
```

sIPAddr: Is a string containing the IP address of the target device.

nTCPPort: Port number of the target device.

nUnitID: Identification number of a serial sub-network device. If a device is addressed directly via TCP/IP, this value must be 16#FF.

nQuantity: Number of output registers (data words) to be written.

nMBAddr: Start address of the output registers to be written (word offset).

cbLength: Contains the max. byte size of the source buffer. The minimum buffer byte size must be: *nQuantity * 2*.

pSrcAddr: Contains the address of the source buffer containing the data to be written. The buffer can be a single variable, an array or a structure, whose address can be found with the ADR operator.

bExecute The function block is activated by a rising edge at this input.

tTimeout: States the length of the timeout that may not be exceeded by execution of the ADS command.

VAR_OUTPUT

```
VAR_OUTPUT
    bBUSY       : BOOL;
    bError      : BOOL;
    nErrId      : UDINT;
END_VAR
```

bBusy: When the function block is activated this output is set. It remains set until an acknowledgement is received.

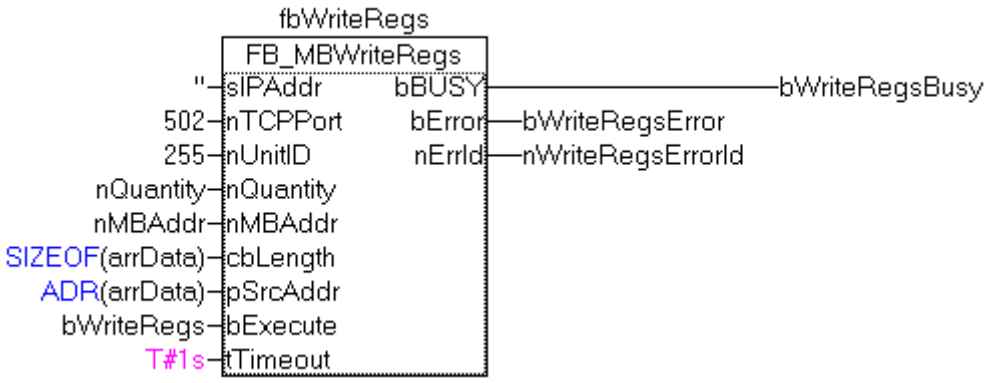
bError: If an ADS error should occur during the transfer of the command, then this output is set once the bBusy output is reset.

nErrId: Supplies the ADS error number when the bError output is set.

Function specific ADS error code	Possible reason
0x8001	Modbus function not implemented
0x8002	Invalid address or length
0x8003	Invalid parameters: - wrong number of registers
0x8004	Modbus server error

Example of calling the block in FBD:

```
PROGRAM Test
VAR
    fbWriteRegs         : FB_MBWriteRegs;
    bWriteRegs          : BOOL;
    bWriteRegsBusy      : BOOL;
    bWriteRegsError     : BOOL;
    nWriteRegsErrorId   : UDINT;
    nWriteRegsCount     : UDINT;
    nQuantity           : WORD := 3;
    nMBAddr             : WORD := 4;
    arrData             : ARRAY [1..3] OF WORD;
END_VAR
```

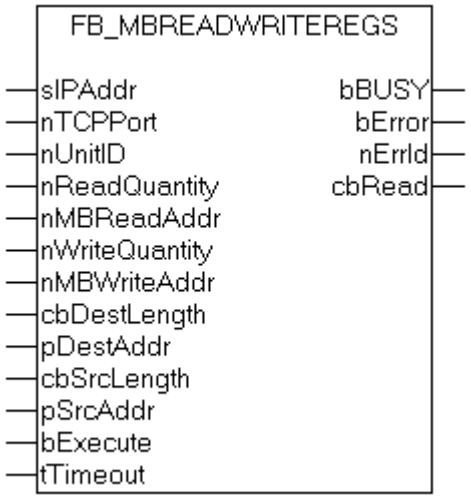


After a rising edge of "bExecute" and successful execution of the ReadRegs command, the content of the arrData array is written to registers 5-7.

Requirements

Development environment	Target system type	PLC libraries to be linked
TwinCAT v2.8.0	PC (i386)	TcModbusSrv.Lib

7.1.9 FB_MBReadWriteRegs (Modbus function 23)



This function first reads 1 to 128 output registers (16 bit) and then writes 1 to 128 output registers (16 bit).

VAR_INPUT

```
VAR_INPUT
    sIPAddr          : STRING(15);
    nTCPPort         : UINT:= MODBUS_TCP_PORT;
    nUnitID          : BYTE:=16#FF;
    nReadQuantity    : WORD;
    nMBReadAddr      : WORD;
    nWriteQuantity   : WORD;
    nMBWriteAddr     : WORD;
    cbDestLength     : UDINT;
    pDestAddr        : UDINT;
    cbSrcLength      : UDINT;
    pSrcAddr         : UDINT;
    bExecute         : BOOL;
    tTimeout         : TIME;
END_VAR
```

sIPAddr : Is a string containing the IP address of the target device.

nTCPPort : Port number of the target device.

nUnitID: Identification number of a serial sub-network device. If a device is addressed directly via TCP/IP, this value must be 16#FF.

nReadQuantity : Number of output registers (data words) to be read. The value of *nReadQuantity* must be > 0.

nMBReadAddr : Start address of the output registers to be read (word offset).

nWriteQuantity : Number of output registers (data words) to be written. The value of *nWriteQuantity* must be > 0.

nMBWriteAddr : Start address of the output registers to be written (word offset).

cbDestLength : Contains the max. byte size of the destination buffer. The minimum destination buffer byte size must be *nReadQuantity * 2*.

pDestAddr : Contains the address of the destination buffer into which the data are to be read. The buffer can be a single variable, an array or a structure, whose address can be found with the ADR operator.

cbSrcLength : Contains the max. byte size of the source buffer. The minimum source buffer byte size must be *nWriteQuantity * 2*.

pSrcAddr : Contains the address of the source buffer containing the data to be written. The buffer can be a single variable, an array or a structure, whose address can be found with the ADR operator.

bExecute: The function block is activated by a rising edge at this input.

tTimeout: States the length of the timeout that may not be exceeded by execution of the ADS command.

VAR_OUTPUT

```
VAR_OUTPUT
    bBUSY       : BOOL;
    bError      : BOOL;
    nErrId      : UDINT;
    cbRead      : UDINT;
END_VAR
```

bBusy : When the function block is activated this output is set. It remains set until an acknowledgement is received.

bError : If an ADS error should occur during the transfer of the command, then this output is set once the bBusy output is reset.

nErrId : Supplies the ADS error number when the bError output is set.

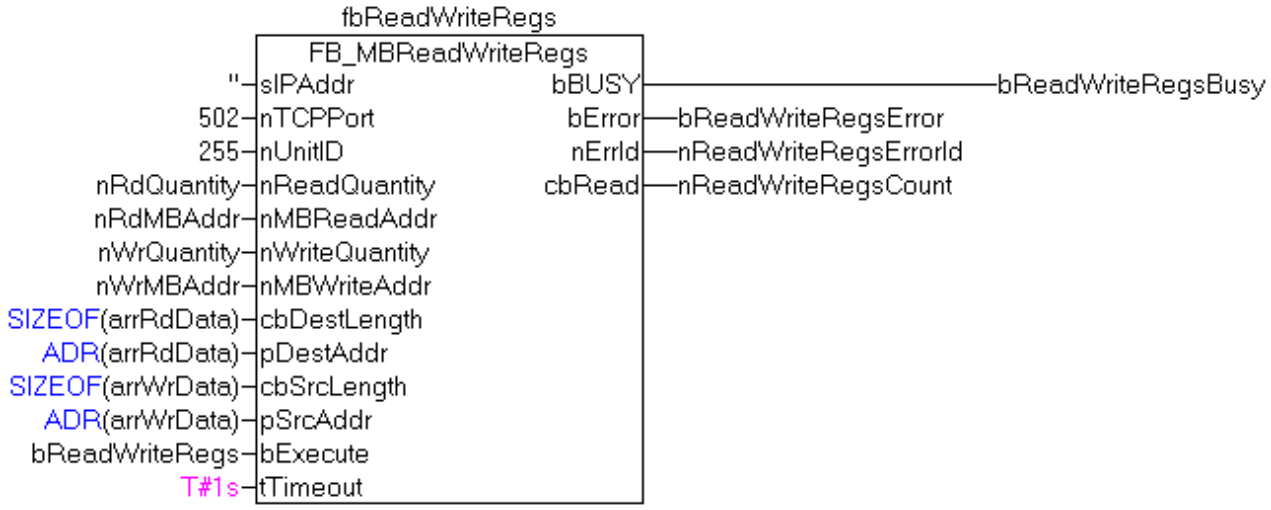
cbRead: Contains the number of bytes currently read.

Function specific ADS error code	Possible reason
0x8001	Modbus function not implemented
0x8002	Invalid address or length
0x8003	Invalid parameters: - wrong number of registers
0x8004	Modbus server error

Example of calling the block in FBD:

```
PROGRAM Test
VAR
    fbReadWriteRegs         : FB_MBReadWriteRegs;
    bReadWriteRegs          : BOOL;
    bReadWriteRegsBusy      : BOOL;
    bReadWriteRegsError     : BOOL;
    nReadWriteRegsErrorId   : UDINT;
    nReadWriteRegsCount     : UDINT;
    nRdQuantity             : WORD;
    nRdMBAddr               : WORD;
```

```
    nWrQuantity             : WORD;
    nWrMBAddr               : WORD;
    arrRdData               : ARRAY [1..9] OF WORD;
    arrWrData               : ARRAY [1..9] OF WORD;
END_VAR
```

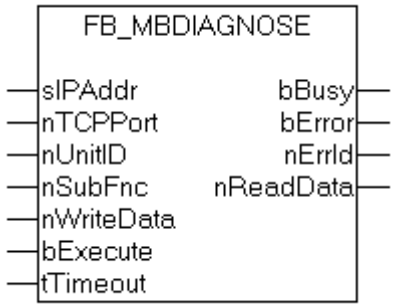


After a rising edge of "bExecute" and successful execution of the ReadWriteRegs command, arrRdData contains the read register data, and the data from arrWrData are written to the registers.

Requirements

Development environment	Target system type	PLC libraries to be linked
TwinCAT v2.8.0	PC (i386)	TcModbusSrv.Lib

7.1.10 FB_MBDiagnose (Modbus function 8)



The diagnosis function provides a series of tests for checking the communication system between the master and the slave and for examining a variety of internal error states within the slave.

VAR_INPUT

```
VAR_INPUT
    sIPAddr         : STRING(15);
    nTCPPort        : UINT:= MODBUS_TCP_PORT;
    nUnitID         : BYTE:=16#FF;
    nSubFnc         : WORD;
    nWriteData      : WORD;
    bExecute        : BOOL;
    tTimeout        : TIME;
END_VAR
```

sIPAddr : Is a string containing the IP address of the target device.

nTCPPort : Port number of the target device.

nUnitID: Identification number of a serial sub-network device. If a device is addressed directly via TCP/IP, this value must be 16#FF.

nSubFnc : The sub-function to be executed.

nWriteData: The data word to be written.

bExecute: The function block is activated by a rising edge at this input.

tTimeout: States the length of the timeout that may not be exceeded by execution of the ADS command.

VAR_OUTPUT

```
VAR_OUTPUT
    bBusy       : BOOL;
    bError      : BOOL;
    nErrId      : UDINT;
    nReadData   : WORD;
END_VAR
```

bBusy : When the function block is activated this output is set. It remains set until an acknowledgement is received.

bError : If an ADS error should occur during the transfer of the command, then this output is set once the bBusy output is reset.

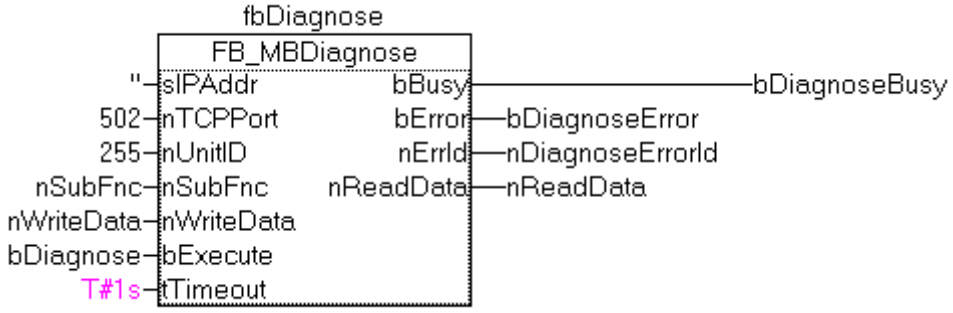
nErrId : Supplies the ADS error number when the bError output is set.

nReadData: Supplies the read data word.

Function specific ADS error code	Possible reason
0x8001	Modbus function not implemented
0x8002	Invalid address or length
0x8003	Invalid parameters: - wrong number of registers
0x8004	Modbus server error

Example of calling the block in FBD:

```
PROGRAM Test
VAR
    fbDiagnose          : FB_MBDiagnose;
    bDiagnose           : BOOL;
    bDiagnoseBusy       : BOOL;
    bDiagnoseError      : BOOL;
    nDiagnoseErrorId    : UDINT;
    nSubFnc             : WORD;
    nReadData           : WORD;
    nWriteData          : WORD;
END_VAR
```



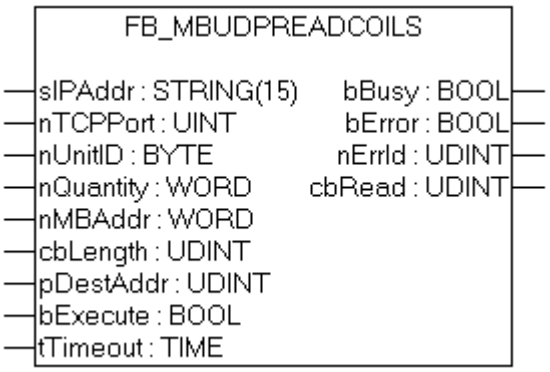
After rising edge of "bExecute" and successful execution of the diagnosis command, nReadData contains the read data word.

Requirements

Development environment	Target system type	PLC libraries to be linked
TwinCAT v2.8.0	PC (i386)	TcModbusSrv.Lib

7.1.11 UDP

7.1.11.1 FB_MBUdpReadCoils (Modbus function 1)



This function is used for reading 1 to 2048 digital outputs (coils). One digital output corresponds to one bit of the read data bytes.

VAR_INPUT

```
VAR_INPUT
    sIPAddr         : STRING(15);
    nTCPPort        : UINT:= MODBUS_TCP_PORT;
    nUnitID         : BYTE:=16#FF;
    nQuantity       : WORD;
    nMBAddr         : WORD;
    cbLength        : UDINT;
    pDestAddr       : UDINT;
    bExecute        : BOOL;
    tTimeout        : TIME;
END_VAR
```

sIPAddr : Is a string containing the IP address of the target device.

nTCPPort : Port number of the target device.

nUnitID: Identification number of a serial sub-network device. If a device is addressed directly via TCP/IP, this value must be 16#FF.

nQuantity : Number of digital inputs (data bits) to be read. The value of *nQuantity* must be > 0.

nMBAddr : Start address of the digital inputs to be read (bit offset).

cbLength : Contains the max. byte size of the destination buffer into which the data are to be read. The minimum buffer byte size must be: *(nQuantity + 7) / 8*.

pDestAddr : Contains the address of the destination buffer into which the data are to be read. The buffer can be a single variable, an array or a structure, whose address can be found with the ADR operator.

bExecute: The function block is activated by a rising edge at this input.

tTimeout: States the length of the timeout that may not be exceeded by execution of the ADS command.

VAR_OUTPUT

```
VAR_OUTPUT
    bBUSY       : BOOL;
    bError      : BOOL;
    nErrId      : UDINT;
    cbRead      : UDINT;
END_VAR
```

bBusy : When the function block is activated this output is set. It remains set until an acknowledgement is received.

bError : If an ADS error should occur during the transfer of the command, then this output is set once the bBusy output is reset.

nErrId : Supplies the ADS error number when the bError output is set.

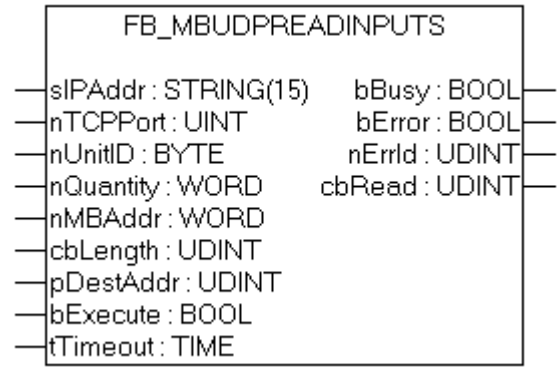
cbRead: Contains the number of bytes currently read.

Function specific ADS error code	Possible reason
0x8001	Modbus function not implemented
0x8002	Invalid address or length
0x8003	Invalid parameters: - wrong number of registers
0x8004	Modbus server error

Requirements

Development environment	Target system type	PLC libraries to be linked
TwinCAT v2.8.0	PC (i386)	TcModbusSrv.Lib

7.1.11.2 FB_MBUdpReadInputs (Modbus function 2)



This function is used for reading 1 to 2048 digital inputs. One digital input corresponds to one bit of the read data bytes.

VAR_INPUT

```
VAR_INPUT
    sIPAddr         : STRING(15);
    nTCPPort        : UINT:= MODBUS_TCP_PORT;
    nUnitID         : BYTE:=16#FF;
    nQuantity       : WORD;
    nMBAddr         : WORD;
    cbLength        : UDINT;
    pDestAddr       : UDINT;
    bExecute        : BOOL;
    tTimeout        : TIME;
END_VAR
```

sIPAddr: Is a string containing the IP address of the target device.

nTCPPort: Port number of the target device.

nUnitID: Identification number of a serial sub-network device. If a device is addressed directly via TCP/IP, this value must be 16#FF.

nQuantity: Number of digital inputs (data bits) to be read. The *value of nQuantity* must be > 0.

nMBAddr: Start address of the digital inputs to be read (bit offset).

cbLength: Contains the max. byte size of the destination buffer. The minimum buffer byte size must be: *(nQuantity + 7) / 8*.

pDestAddr: Contains the address of the destination buffer into which the data are to be read. The buffer can be a single variable, an array or a structure, whose address can be found with the ADR operator.

bExecute: The function block is activated by a rising edge at this input.

tTimeout: States the length of the timeout that may not be exceeded by execution of the ADS command.

VAR_OUTPUT

```
VAR_OUTPUT
    bBUSY       : BOOL;
    bError      : BOOL;
    nErrId      : UDINT;
    cbRead      : UDINT;
END_VAR
```

bBusy : When the function block is activated this output is set. It remains set until an acknowledgement is received.

bError : If an ADS error should occur during the transfer of the command, then this output is set once the bBusy output is reset.

nErrId : Supplies the ADS error number when the bError output is set.

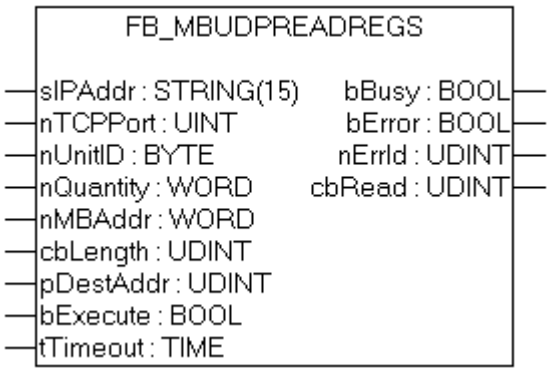
cbRead: Contains the number of bytes currently read.

Function specific ADS error code	Possible reason
0x8001	Modbus function not implemented
0x8002	Invalid address or length
0x8003	Invalid parameters: - wrong number of registers
0x8004	Modbus server error

Requirements

Development environment	Target system type	PLC libraries to be linked
TwinCAT v2.8.0	PC (i386)	TcModbusSrv.Lib

7.1.11.3 FB_MBUdpReadRegs (Modbus function 3)



This function is used for reading 1 to 128 output registers (16 bit). The first byte contains the lower eight bits and the second byte the upper eight bits.

VAR_INPUT

```
VAR_INPUT
    sIPAddr         : STRING(15);
    nTCPPort        : UINT:= MODBUS_TCP_PORT;
    nUnitID         : BYTE:=16#FF;
    nQuantity       : WORD;
    nMBAddr         : WORD;
    cbLength        : UDINT;
    pDestAddr       : UDINT;
    bExecute        : BOOL;
    tTimeout        : TIME;
END_VAR
```

sIPAddr: Is a string containing the IP address of the target device.

nTCPPort: Port number of the target device.

nUnitID: Identification number of a serial sub-network device. If a device is addressed directly via TCP/IP, this value must be 16#FF.

nQuantity: Number of output registers (data words) to be read. The value of *nQuantity* must be > 0.

nMBAddr: Start address of the output registers to be read (word offset).

cbLength: Contains the max. byte size of the destination buffer. The minimum buffer byte size must be: *nQuantity * 2*.

pDestAddr: Contains the address of the destination buffer into which the data are to be read. The buffer can be a single variable, an array or a structure, whose address can be found with the ADR operator.

bExecute: The function block is activated by a rising edge at this input.

tTimeout: States the length of the timeout that may not be exceeded by execution of the ADS command.

VAR_OUTPUT

```
VAR_OUTPUT
    bBUSY       : BOOL;
    bError      : BOOL;
    nErrId      : UDINT;
    cbRead      : UDINT;
END_VAR
```

bBusy : When the function block is activated this output is set. It remains set until an acknowledgement is received.

bError : If an ADS error should occur during the transfer of the command, then this output is set once the bBusy output is reset.

nErrId : Supplies the ADS error number when the bError output is set.

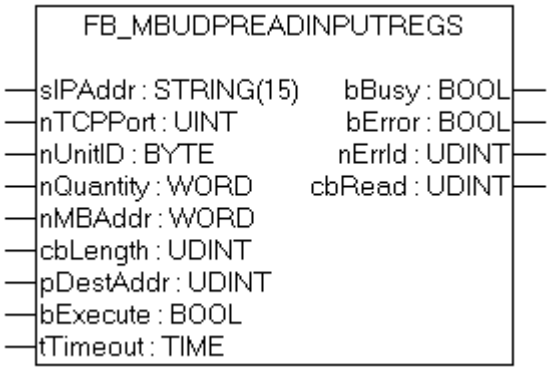
cbRead: Contains the number of bytes currently read.

Function specific ADS error code	Possible reason
0x8001	Modbus function not implemented
0x8002	Invalid address or length
0x8003	Invalid parameters: - wrong number of registers
0x8004	Modbus server error

Requirements

Development environment	Target system type	PLC libraries to be linked
TwinCAT v2.8.0	PC (i386)	TcModbusSrv.Lib

7.1.11.4 FB_MBUdpReadInputRegs (Modbus function 4)



This function is used for reading 1 to 128 input registers (16 bit). Endian

VAR_INPUT

```
VAR_INPUT
    sIPAddr          : STRING(15);
    nTCPPort         : UINT:= MODBUS_TCP_PORT;
    nUnitID          : BYTE:=16#FF;
    nQuantity        : WORD;
    nMBAddr          : WORD;
    cbLength         : UDINT;
    pDestAddr        : UDINT;
    bExecute         : BOOL;
    tTimeout         : TIME;
END_VAR
```

sIPAddr: Is a string containing the IP address of the target device.

nTCPPort: Port number of the target device.

nUnitID: Identification number of a serial sub-network device. If a device is addressed directly via TCP/IP, this value must be 16#FF.

nQuantity: Number of input registers (data words) to be read. The value of *nQuantity* must be > 0.

nMBAddr: Start address of the input register to be read (word offset).

cbLength: Contains the max. byte size of the destination buffer. The minimum buffer byte size must be: *nQuantity * 2*.

pDestAddr: Contains the address of the destination buffer into which the data are to be read. The buffer can be a single variable, an array or a structure, whose address can be found with the ADR operator.

bExecute: The function block is activated by a rising edge at this input.

tTimeout: States the length of the timeout that may not be exceeded by execution of the ADS command.

VAR_OUTPUT

```
VAR_OUTPUT
    bBUSY       : BOOL;
    bError      : BOOL;
    nErrId      : UDINT;
    cbRead      : UDINT;
END_VAR
```

bBusy : When the function block is activated this output is set. It remains set until an acknowledgement is received.

bError : If an ADS error should occur during the transfer of the command, then this output is set once the bBusy output is reset.

nErrId : Supplies the ADS error number when the bError output is set.

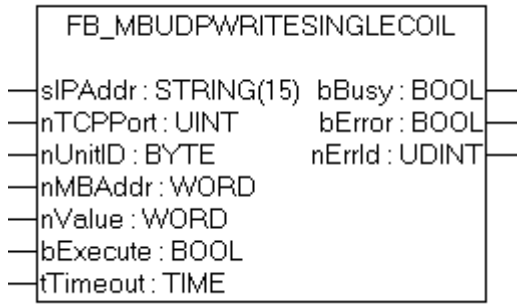
cbRead: Contains the number of bytes currently read.

Function specific ADS error code	Possible reason
0x8001	Modbus function not implemented
0x8002	Invalid address or length
0x8003	Invalid parameters: - wrong number of registers
0x8004	Modbus server error

Requirements

Development environment	Target system type	PLC libraries to be linked
TwinCAT v2.8.0	PC (i386)	TcModbusSrv.Lib

7.1.11.5 FB_MBUdpWriteSingleCoil (Modbus function 5)



This function is used for writing a single digital output (coil). Bit access is used.

VAR_INPUT

```
VAR_INPUT
    sIPAddr         : STRING(15);
    nTCPPort        : UINT:= MODBUS_TCP_PORT;
    nUnitID         : BYTE:=16#FF;
    nMBAddr         : WORD;
    nValue          : WORD;
    bExecute        : BOOL;
    tTimeout        : TIME;
END_VAR
```

sIPAddr: Is a string containing the IP address of the target device.

nTCPPort: Port number of the target device.

nUnitID: Identification number of a serial sub-network device. If a device is addressed directly via TCP/IP, this value must be 16#FF.

nMBAddr: Address of the digital output (bit offset).

nValue: Value to be written into the digital output. The value 16#FF00 switches the output on, 16#0000 switches it off.

bExecute: The function block is activated by a rising edge at this input.

tTimeout: States the length of the timeout that may not be exceeded by execution of the ADS command.

VAR_OUTPUT

```
VAR_OUTPUT
    bBUSY       : BOOL;
    bError      : BOOL;
    nErrId      : UDINT;
END_VAR
```

bBusy : When the function block is activated this output is set. It remains set until an acknowledgement is received.

bError : If an ADS error should occur during the transfer of the command, then this output is set once the bBusy output is reset.

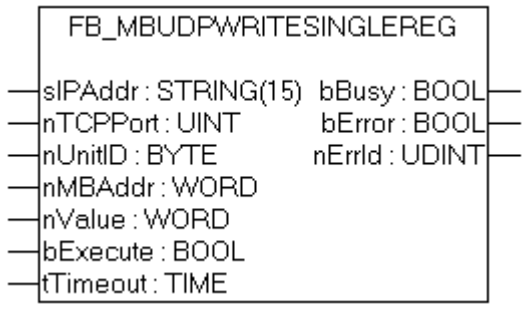
nErrId : Supplies the ADS error number when the bError output is set.

Function specific ADS error code	Possible reason
0x8001	Modbus function not implemented
0x8002	Invalid address or length
0x8003	Invalid parameters: - wrong number of registers
0x8004	Modbus server error

Requirements

Development environment	Target system type	PLC libraries to be linked
TwinCAT v2.8.0	PC (i386)	TcModbusSrv.Lib

7.1.11.6 FB_MBUdpWriteSingleReg (Modbus function 6)



This function is used for writing an individual output register. 16 bit access is used.

VAR_INPUT

```
VAR_INPUT
    sIPAddr         : STRING(15);
    nTCPPort        : UINT:= MODBUS_TCP_PORT;
    nUnitID         : BYTE:=16#FF;
    nMBAddr         : WORD;
    nValue          : WORD;
```

```
    bExecute        : BOOL;
    tTimeout        : TIME;
END_VAR
```

sIPAddr: Is a string containing the IP address of the target device.

nTCPPort: Port number of the target device.

nUnitID: Identification number of a serial sub-network device. If a device is addressed directly via TCP/IP, this value must be 16#FF.

nMBAddr: Address of the output register (word offset).

nValue: Value to be written into the register (word value).

bExecute: The function block is activated by a rising edge at this input.

tTimeout: States the length of the timeout that may not be exceeded by execution of the ADS command.

VAR_OUTPUT

```
VAR_OUTPUT
    bBUSY       : BOOL;
    bError      : BOOL;
    nErrId      : UDINT;
END_VAR
```

bBusy : When the function block is activated this output is set. It remains set until an acknowledgement is received.

bError : If an ADS error should occur during the transfer of the command, then this output is set once the bBusy output is reset.

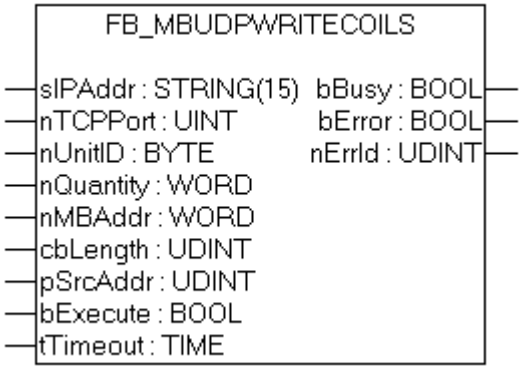
nErrId : Supplies the ADS error number when the bError output is set.

Function specific ADS error code	Possible reason
0x8001	Modbus function not implemented
0x8002	Invalid address or length
0x8003	Invalid parameters: - wrong number of registers
0x8004	Modbus server error

Requirements

Development environment	Target system type	PLC libraries to be linked
TwinCAT v2.8.0	PC (i386)	TcModbusSrv.Lib

7.1.11.7 FB_MBUdpWriteCoils (Modbus function 15)



This function is used for writing 1 to 2048 digital outputs (coils). One digital output corresponds to one bit of the write data bytes.

VAR_INPUT

```
VAR_INPUT
    sIPAddr         : STRING(15);
    nTCPPort        : UINT:= MODBUS_TCP_PORT;
    nUnitID         : BYTE:=16#FF;
    nQuantity       : WORD;
    nMBAddr         : WORD;
    cbLength        : UDINT;
    pSrcAddr        : UDINT;
    bExecute        : BOOL;
    tTimeout        : TIME;
END_VAR
```

sIPAddr: Is a string containing the IP address of the target device.

nTCPPort: Port number of the target device.

nUnitID: Identification number of a serial sub-network device. If a device is addressed directly via TCP/IP, this value must be 16#FF.

nQuantity: Number of digital outputs to be written (data bits). *nQuantity* must be > 0.

nMBAddr: Start address of the digital outputs to be written (bit offset).

cbLength: Contains the max. byte size of the source buffer containing the data to be written. The minimum buffer byte size must be: *(nQuantity + 7) / 8*.

pSrcAddr: Contains the address of the source buffer containing the data to be written. The buffer can be a single variable, an array or a structure, whose address can be found with the ADR operator.

bExecute: The function block is activated by a rising edge at this input.

tTimeout: States the length of the timeout that may not be exceeded by execution of the ADS command.

VAR_OUTPUT

```
VAR_OUTPUT
    bBUSY       : BOOL;
    bError      : BOOL;
    nErrId      : UDINT;
    cbRead      : UDINT;
END_VAR
```

bBusy : When the function block is activated this output is set. It remains set until an acknowledgement is received.

bError : If an ADS error should occur during the transfer of the command, then this output is set once the bBusy output is reset.

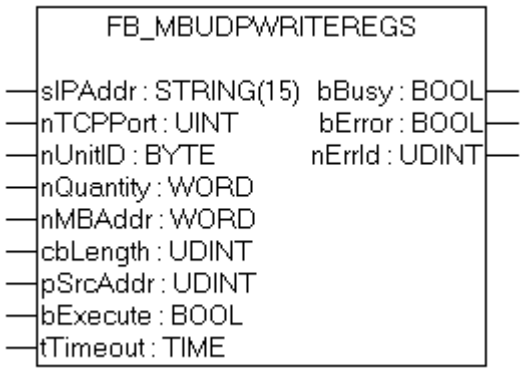
nErrId : Supplies the ADS error number when the bError output is set.

Function specific ADS error code	Possible reason
0x8001	Modbus function not implemented
0x8002	Invalid address or length
0x8003	Invalid parameters: - wrong number of registers
0x8004	Modbus server error

Requirements

Development environment	Target system type	PLC libraries to be linked
TwinCAT v2.8.0	PC (i386)	TcModbusSrv.Lib

7.1.11.8 FB_MBUdpWriteRegs (Modbus function 16)



This function is used for writing 1 to 128 output registers (16 bit).

VAR_INPUT

```
VAR_INPUT
    sIPAddr         : STRING(15);
    nTCPPort        : UINT:= MODBUS_TCP_PORT;
    nUnitID         : BYTE:=16#FF;
    nQuantity       : WORD;
    nMBAddr         : WORD;
    cbLength        : UDINT;
    pSrcAddr        : UDINT;
    bExecute        : BOOL;
    tTimeout        : TIME;
END_VAR
```

sIPAddr: Is a string containing the IP address of the target device.

nTCPPort: Port number of the target device.

nUnitID: Identification number of a serial sub-network device. If a device is addressed directly via TCP/IP, this value must be 16#FF.

nQuantity: Number of output registers (data words) to be written.

nMBAddr: Start address of the output registers to be written (word offset).

cbLength: Contains the max. byte size of the source buffer. The minimum buffer byte size must be: *nQuantity * 2*.

pSrcAddr: Contains the address of the source buffer containing the data to be written. The buffer can be a single variable, an array or a structure, whose address can be found with the ADR operator.

bExecute The function block is activated by a rising edge at this input.

tTimeout: States the length of the timeout that may not be exceeded by execution of the ADS command.

VAR_OUTPUT

```
VAR_OUTPUT
    bBUSY       : BOOL;
    bError      : BOOL;
    nErrId      : UDINT;
END_VAR
```

bBusy: When the function block is activated this output is set. It remains set until an acknowledgement is received.

bError: If an ADS error should occur during the transfer of the command, then this output is set once the bBusy output is reset.

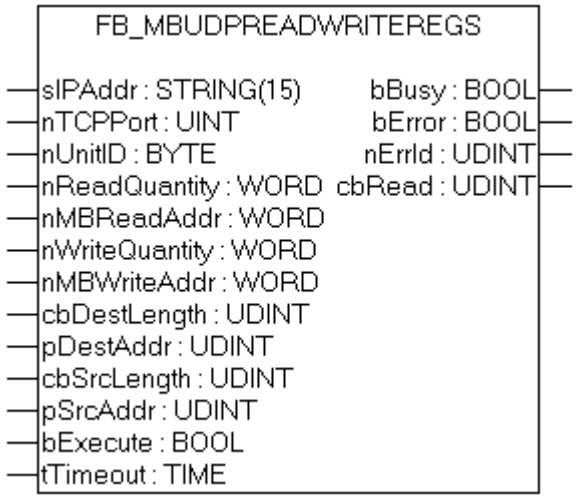
nErrId: Supplies the ADS error number when the bError output is set.

Function specific ADS error code	Possible reason
0x8001	Modbus function not implemented
0x8002	Invalid address or length
0x8003	Invalid parameters: - wrong number of registers
0x8004	Modbus server error

Requirements

Development environment	Target system type	PLC libraries to be linked
TwinCAT v2.8.0	PC (i386)	TcModbusSrv.Lib

7.1.11.9 FB_MBUdpReadWriteRegs (Modbus function 23)



This function first reads 1 to 128 output registers (16 bit) and then writes 1 to 128 output registers (16 bit).

VAR_INPUT

```
VAR_INPUT
    sIPAddr         : STRING(15);
    nTCPPort        : UINT:= MODBUS_TCP_PORT;
    nUnitID         : BYTE:=16#FF;
    nReadQuantity   : WORD;
    nMBReadAddr     : WORD;
    nWriteQuantity  : WORD;
    nMBWriteAddr    : WORD;
    cbDestLength    : UDINT;
    pDestAddr       : UDINT;
    cbSrcLength     : UDINT;
    pSrcAddr        : UDINT;
    bExecute        : BOOL;
    tTimeout        : TIME;
END_VAR
```

sIPAddr : Is a string containing the IP address of the target device.

nTCPPort : Port number of the target device.

nUnitID: Identification number of a serial sub-network device. If a device is addressed directly via TCP/IP, this value must be 16#FF.

nReadQuantity : Number of output registers (data words) to be read. The value of *nReadQuantity* must be > 0.

nMBReadAddr : Start address of the output registers to be read (word offset).

nWriteQuantity : Number of output registers (data words) to be written. The value of *nWriteQuantity* must be > 0.

nMBWriteAddr : Start address of the output registers to be written (word offset).

cbDestLength : Contains the max. byte size of the destination buffer. The minimum destination buffer byte size must be *nReadQuantity * 2*.

pDestAddr : Contains the address of the destination buffer into which the data are to be read. The buffer can be a single variable, an array or a structure, whose address can be found with the ADR operator.

cbSrcLength : Contains the max. byte size of the source buffer. The minimum source buffer byte size must be *nWriteQuantity * 2*.

pSrcAddr : Contains the address of the source buffer containing the data to be written. The buffer can be a single variable, an array or a structure, whose address can be found with the ADR operator.

bExecute: The function block is activated by a rising edge at this input.

tTimeout: States the length of the timeout that may not be exceeded by execution of the ADS command.

VAR_OUTPUT

```
VAR_OUTPUT
    bBUSY       : BOOL;
    bError      : BOOL;
    nErrId      : UDINT;
    cbRead      : UDINT;
END_VAR
```

bBusy : When the function block is activated this output is set. It remains set until an acknowledgement is received.

bError : If an ADS error should occur during the transfer of the command, then this output is set once the bBusy output is reset.

nErrId : Supplies the ADS error number when the bError output is set.

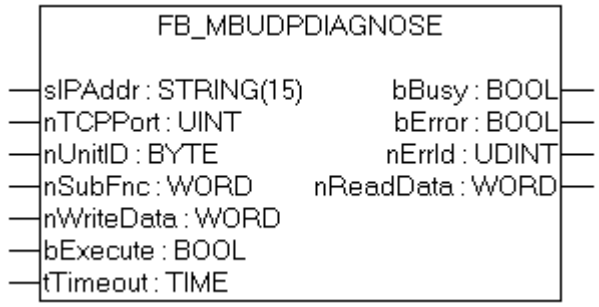
cbRead: Contains the number of bytes currently read.

Function specific ADS error code	Possible reason
0x8001	Modbus function not implemented
0x8002	Invalid address or length
0x8003	Invalid parameters: - wrong number of registers
0x8004	Modbus server error

Requirements

Development environment	Target system type	PLC libraries to be linked
TwinCAT v2.8.0	PC (i386)	TcModbusSrv.Lib

7.1.11.10 FB_MBUdpDiagnose (Modbus function 8)



The diagnosis function provides a series of tests for checking the communication system between the master and the slave and for examining a variety of internal error states within the slave.

VAR_INPUT

```
VAR_INPUT
    sIPAddr         : STRING(15);
    nTCPPort        : UINT:= MODBUS_TCP_PORT;
    nUnitID         : BYTE:=16#FF;
    nSubFnc         : WORD;
    nWriteData      : WORD;
    bExecute        : BOOL;
    tTimeout        : TIME;
END_VAR
```

sIPAddr : Is a string containing the IP address of the target device.

nTCPPort : Port number of the target device.

nUnitID: Identification number of a serial sub-network device. If a device is addressed directly via TCP/IP, this value must be 16#FF.

nSubFnc : The sub-function to be executed.

nWriteData: The data word to be written.

bExecute: The function block is activated by a rising edge at this input.

tTimeout: States the length of the timeout that may not be exceeded by execution of the ADS command.

VAR_OUTPUT

```
VAR_OUTPUT
    bBusy       : BOOL;
    bError      : BOOL;
    nErrId      : UDINT;
    nReadData   : WORD;
END_VAR
```

bBusy : When the function block is activated this output is set. It remains set until an acknowledgement is received.

bError : If an ADS error should occur during the transfer of the command, then this output is set once the bBusy output is reset.

nErrId : Supplies the ADS error number when the bError output is set.

nReadData: Supplies the read data word.

Function specific ADS error code	Possible reason
0x8001	Modbus function not implemented
0x8002	Invalid address or length
0x8003	Invalid parameters:

Function specific ADS error code	Possible reason
	- wrong number of registers
0x8004	Modbus server error

Requirements

Development environment	Target system type	PLC libraries to be linked
TwinCAT v2.8.0	PC (i386)	TcModbusSrv.Lib

8 Samples

8.1 Digital IO access

This sample explains the access to a TwinCAT system via Modbus.

The default mapping [▶ 14] of the TwinCAT Modbus TCP mapps the digital output (coils) to the physical outputs of the PLC.

```
PROGRAM MAIN
VAR
    Q00 AT%QX0.0            : BOOL;
    Q01 AT%QX0.1            : BOOL;
    Q02 AT%QX0.2            : BOOL;
    Q03 AT%QX0.3            : BOOL;
    Q04 AT%QX0.4            : BOOL;
    Q05 AT%QX0.5            : BOOL;
    Q06 AT%QX0.6            : BOOL;
    Q07 AT%QX0.7            : BOOL;

    fbWriteCoils            : FB_MBWriteCoils;
    bWrite                  : BOOL;
    nValue                  : INT;
END_VAR
```

```
IF NOT bWrite THEN
    nValue := nValue + 1;
    bWrite := TRUE;
    fbWriteCoils.nQuantity := 8;
    fbWriteCoils.cbLength := SIZEOF(nValue);
    fbWriteCoils.pSrcAddr := ADR(nValue);
    fbWriteCoils.tTimeout := T#5s;
    fbWriteCoils(bExecute:=TRUE);
ELSE
    IF NOT fbWriteCoils.bBUSY THEN
        bWrite :=FALSE;
    END_IF
    fbWriteCoils(bExecute:=FALSE);
END_IF
```

The counter nValue will be written to physical outputs of the plc (Q00-Q07) by a rising edge of bWrite.

The bit ordering is explained in this table:

Bit	8 MSB	7	6	5	4	3	2	1 LSB
Output	7	6	5	4	3	2	1	0

MSB = Most significant bit

LSB = Least significant bit

https://infosys.beckhoff.com/content/1033/tcmodbussrvce/Resources/11381105547/.zip

8.2 Multiple register access

his sample explains the access to the register of a TwinCAT system via Modbus.

The Modbus address **0x3000** is mapped by the default-configuration [▶ 14] to the memory area of the plc (ADS-Indexgroup 0x4020)

After calling bWriteRegs, the array **arrValue** is written in the flag area and thus in the variable M0.

```
PROGRAM MAIN
VAR
   ipAddr          : STRING(15) := '';
   M0 AT%MB0       : ARRAY [0..3] OF WORD;
   nValue          : ARRAY [0..3] OF WORD := 0,10,100,1000;
   fbWriteRegs     : FB_MBWriteRegs;
   bWriteRegs      : BOOL;
END_VAR
```

```
IF NOT bWriteRegs THEN
    nValue[0]:= nValue[0]+1;
    nValue[1]:= nValue[1]+1;
    nValue[2]:= nValue[2]+1;
    nValue[3]:= nValue[3]+1;
    bWriteRegs :=TRUE;
    fbWriteRegs.sIPAddr :=ipAddr;
    fbWriteRegs.nQuantity := 4;
    fbWriteRegs.nMBAddr := 16#3000;
    fbWriteRegs.cbLength := SIZEOF(nValue);
    fbWriteRegs.pSrcAddr := ADR(nValue);
    fbWriteRegs.tTimeout := T#5s;
    fbWriteRegs(bExecute:=TRUE);
ELSE
    IF NOT fbWriteRegs.bBUSY THEN
    bWriteRegs :=FALSE;
END_IF
    fbWriteRegs(bExecute:=FALSE);
END_IF
```

https://infosys.beckhoff.com/content/1033/tcmodbussrvce/Resources/11381106955/.zip

9 Return codes

Hex	Dezimal	Source
0x00000000-0x00007800	0-30720	TwinCAT System return codes
0x00008000-0x00008FFF	32768-33023	Internal TwinCAT Modbus TCP
0x80070000-0x8007FFFF	2147942400-2147483647	Returncode - 0x80070000 = Win32 System Returncode

TwinCAT Modbus TCP return code

Function specific ADS return code	Possible reason
0x8001	Modbus function not implemented
0x8002	Invalid address or length
0x8003	Invalid parameters: - wrong number of registers
0x8004	Modbus server error

More Information:
www.beckhoff.com/ts6250

Beckhoff Automation GmbH & Co. KG
Hülshorstweg 20
33415 Verl
Germany
Phone: +49 5246 9630
info@beckhoff.com
www.beckhoff.com