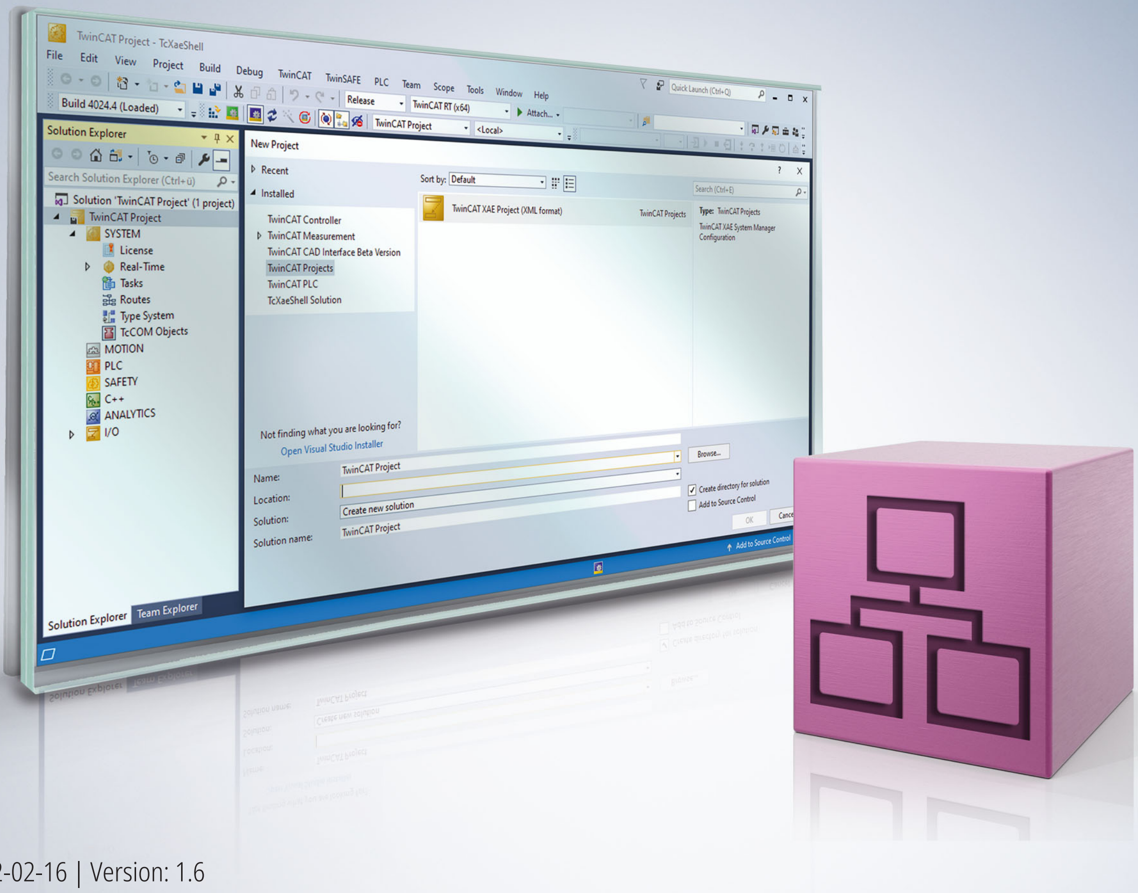
BECKHOFF New Automation Technology

Manual | EN

TF6250

TwinCAT 3 | Modbus TCP



2022-02-16 | Version: 1.6

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®

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

Copyright

1.2 Safety instructions

Safety regulations

Please note the following safety instructions and explanations!
Product-specific safety instructions can be found on following pages or in the areas mounting, wiring, commissioning etc.

Exclusion of liability

All the components are supplied in particular hardware and software configurations 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 engineering who are familiar with the applicable national standards.

Description of symbols

In this documentation the following symbols are used with an accompanying safety instruction or note. The safety instructions must be read carefully and followed without fail!

⚠ DANGER

Serious risk of injury!

Failure to follow the safety instructions associated with this symbol directly endangers the life and health of persons.

⚠ WARNING

Risk of injury!

Failure to follow the safety instructions associated with this symbol endangers the life and health of persons.

⚠ CAUTION

Personal injuries!

Failure to follow the safety instructions associated with this symbol can lead to injuries to persons.

NOTE

Damage to the environment or devices

Failure to follow the instructions associated with this symbol can lead to damage to the environment or equipment.

Tip or pointer

This symbol indicates information that contributes to better understanding.

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.

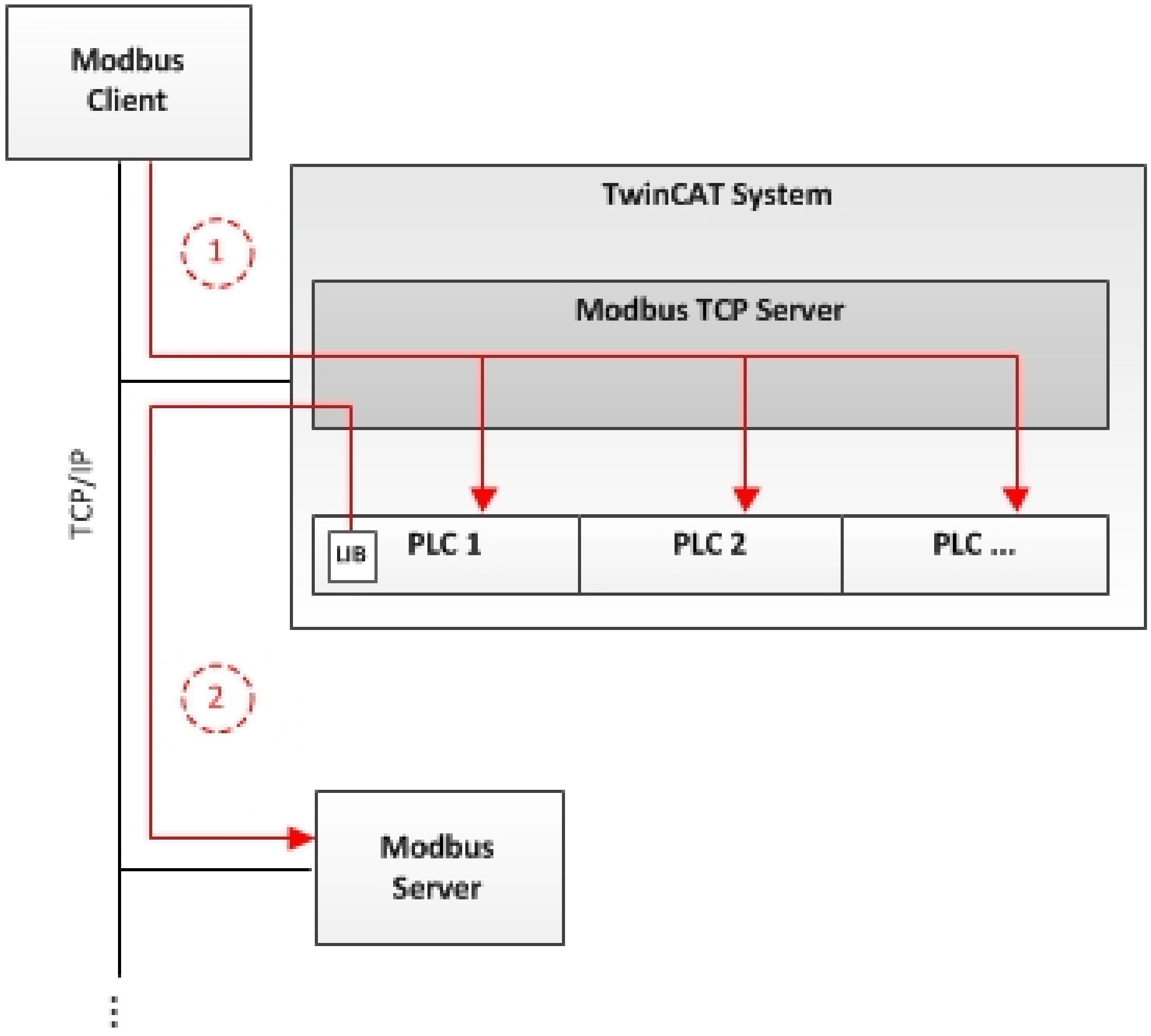
To stay informed about information security for Beckhoff products, subscribe to the RSS feed at https://www.beckhoff.com/secinfo.

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 [▶ 17]:

(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 [▶ 21]:

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

3 Installation

3.1 System Requirements

Technical Data	TF6250 TwinCAT 3 Modbus TCP Server
Target System	Windows NT/2000/XP/Vista/7 PC (x86-compatible)
Min. TwinCAT-Version	3.0.0
Min. TwinCAT-Level	TC1200 TC3 | PLC

Requirements

Development environment	Target system type	PLC libraries to be linked
TwinCAT v3.0.0	PC or CX (x86, ARM)	Tc2_ModbusSrv

3.2 Installation

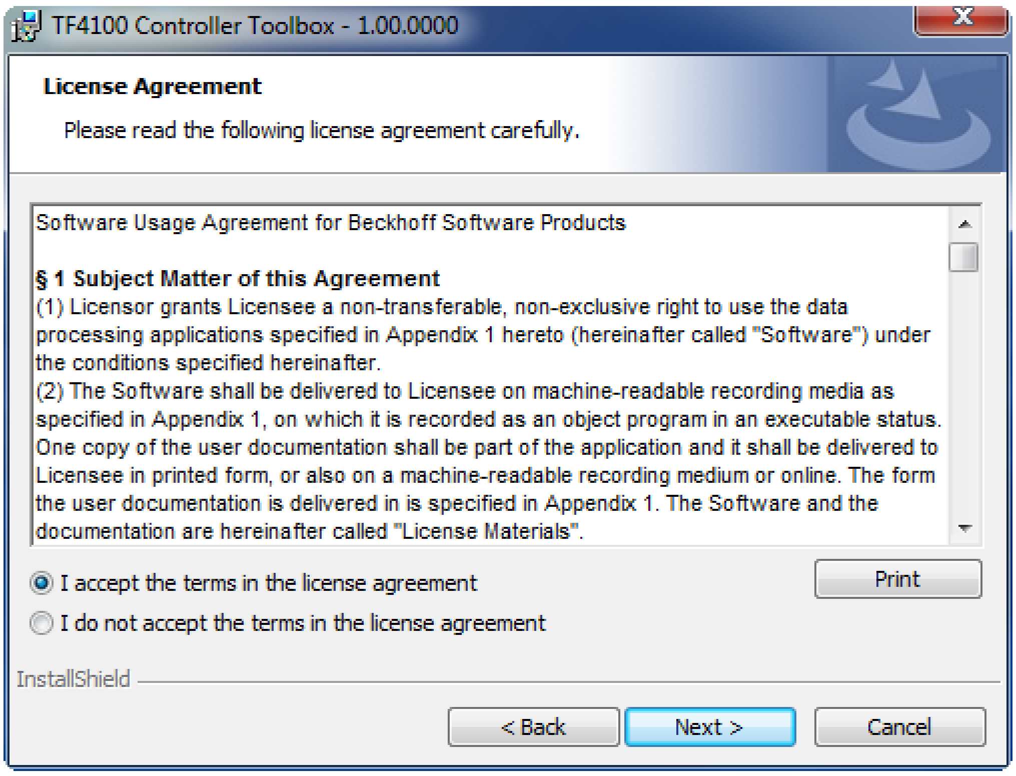
The following section describes how to install the TwinCAT 3 Function for Windows-based operating systems.

✓ The TwinCAT 3 Function setup file was downloaded from the Beckhoff website.

1. Run the setup file as administrator. To do this, select the command **Run as administrator** in the context menu of the file.

   ⇨ The installation dialog opens.

2. Accept the end user licensing agreement and click **Next**.

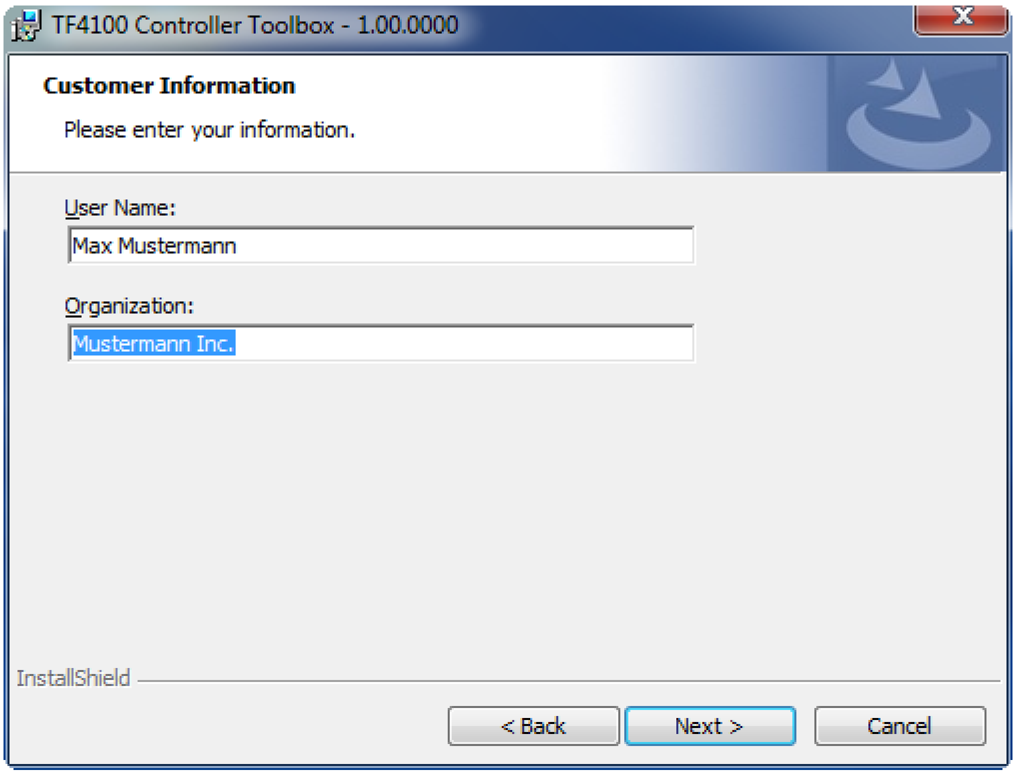

TF4100 Controller Toolbox - 1.00.0000
License Agreement
Please read the following license agreement carefully.
Print
InstallShield
< Back
Next >
Cancel

Software Usage Agreement for Beckhoff Software Products

§ 1 Subject Matter of this Agreement
(1) Licensor grants Licensee a non-transferable, non-exclusive right to use the data processing applications specified in Appendix 1 hereto (hereinafter called "Software") under the conditions specified hereinafter.
(2) The Software shall be delivered to Licensee on machine-readable recording media as specified in Appendix 1, on which it is recorded as an object program in an executable status. One copy of the user documentation shall be part of the application and it shall be delivered to Licensee in printed form, or also on a machine-readable recording medium or online. The form the user documentation is delivered in is specified in Appendix 1. The Software and the documentation are hereinafter called "License Materials".

[✓ I accept the terms in the license agreement]
I do not accept the terms in the license agreement

3. Enter your user data.



TF4100 Controller Toolbox - 1.00.0000

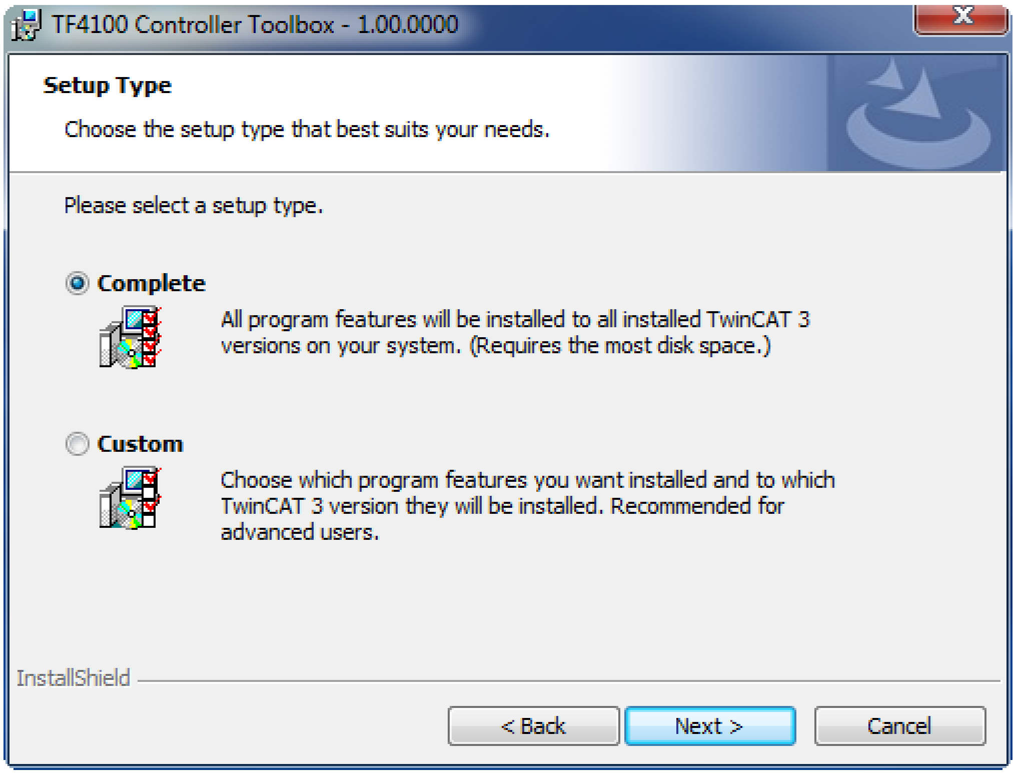
Customer Information

Please enter your information.

User Name: Max Mustermann

Organization: Mustermann Inc.

4. If you want to install the full version of the TwinCAT 3 Function, select **Complete** as installation type. If you want to install the TwinCAT 3 Function components separately, select **Custom**.



TF4100 Controller Toolbox - 1.00.0000

Setup Type

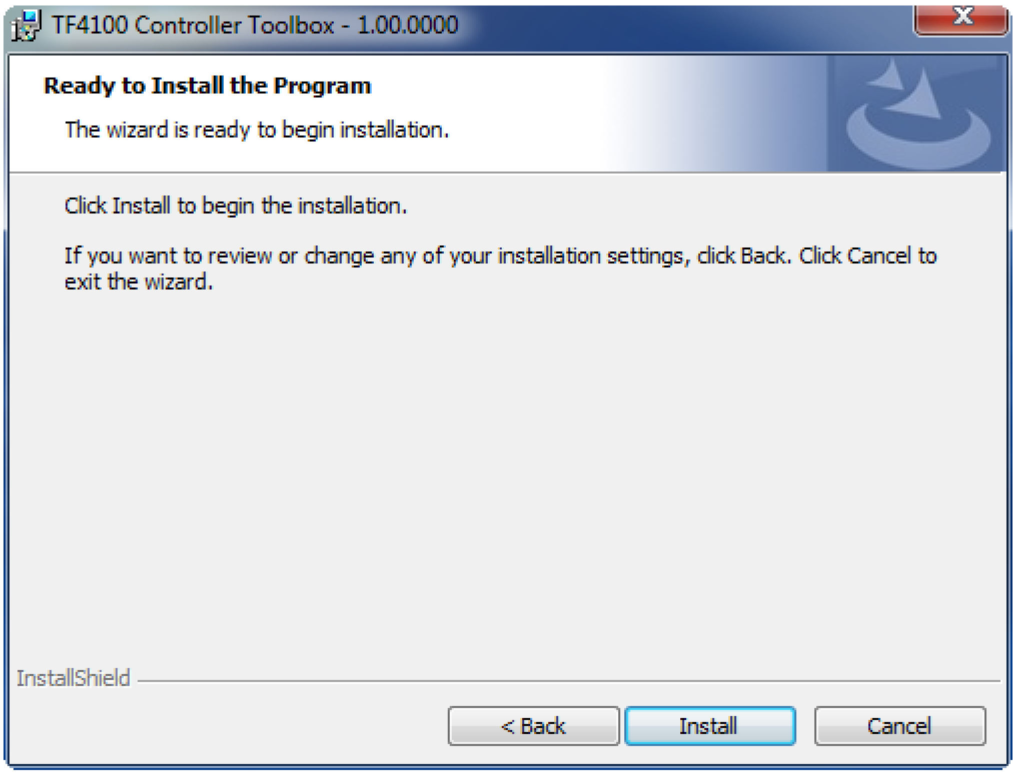
Choose the setup type that best suits your needs.

Please select a setup type.

[✓ Complete]
All program features will be installed to all installed TwinCAT 3 versions on your system. (Requires the most disk space.)

Custom
Choose which program features you want installed and to which TwinCAT 3 version they will be installed. Recommended for advanced users.

5. Select **Next**, then **Install** to start the installation.


TF4100 Controller Toolbox - 1.00.0000

Ready to Install the Program

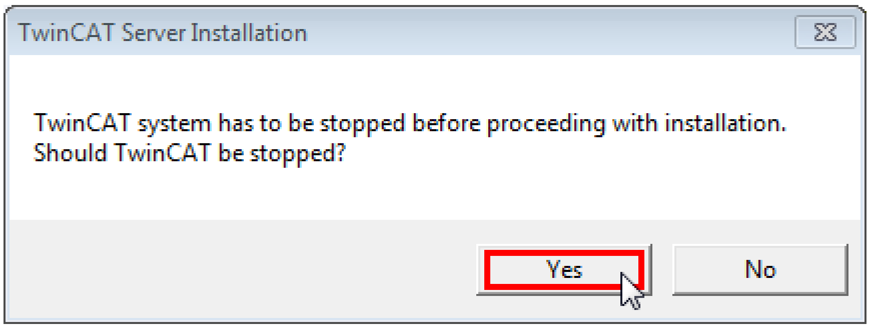
The wizard is ready to begin installation.

Click Install to begin the installation.

If you want to review or change any of your installation settings, click Back. Click Cancel to exit the wizard.

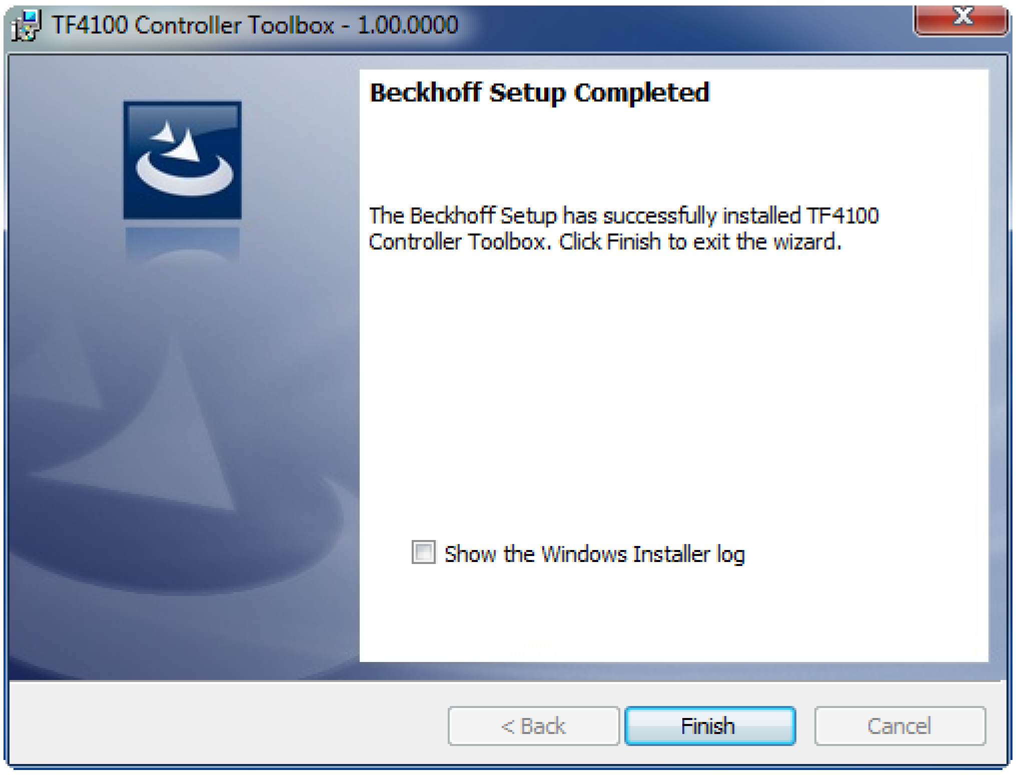
⇨ A dialog box informs you that the TwinCAT system must be stopped to proceed with the installation.

6. Confirm the dialog with **Yes**.


TwinCAT Server Installation

TwinCAT system has to be stopped before proceeding with installation.
Should TwinCAT be stopped?

7. Select **Finish** to exit the setup.



Beckhoff Setup Completed

The Beckhoff Setup has successfully installed TF4100 Controller Toolbox. Click Finish to exit the wizard.

☐ Show the Windows Installer log

⇨ The TwinCAT 3 Function has been successfully installed and can be licensed (see Licensing [▶ 14]).

3.3 Installation Windows CE

The following section describes how to install a TwinCAT 3 function (TFxxx) on a Beckhoff Embedded PC with Windows CE.

1. Download and install the setup file [▶ 12]
2. Transfer the CAB file to the Windows CE device [▶ 13]
3. Run the CAB file on the Windows CE device [▶ 13]

If an older TFxxx version is already installed on the Windows CE device, it can be updated:

- Software upgrade [▶ 13]

Download and install the setup file

The CAB installation file for Windows CE is part of the TFxxx setup. This is made available on the Beckhoff website www.beckhoff.com and automatically contains all versions for Windows XP, Windows 7 and Windows CE (x86 and ARM).

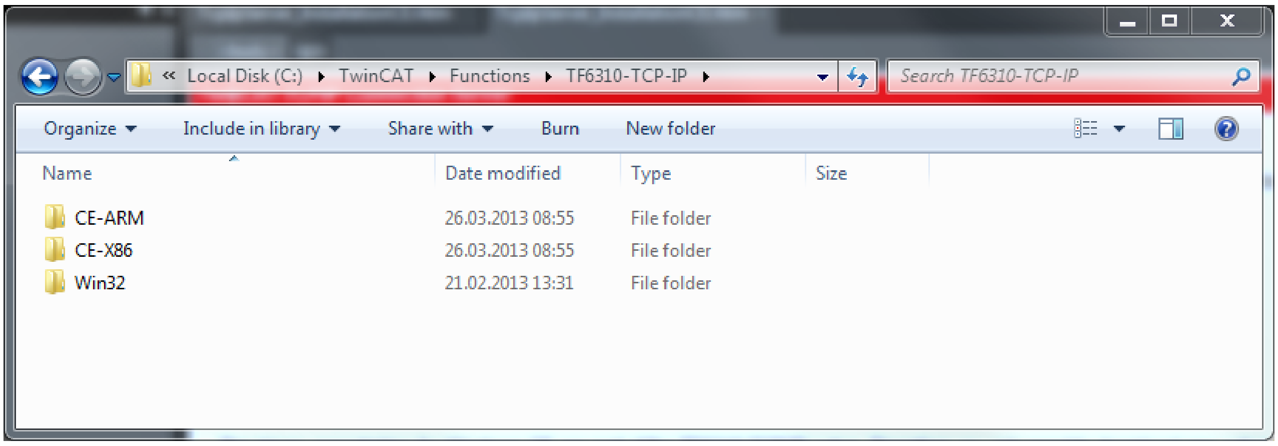
Download the TFxxx setup file and install the TwinCAT 3 function as described in the Installation [▶ 9] section.

After the installation, the installation folder contains three directories (one directory per hardware platform):

- **CE-ARM:** ARM-based Embedded PCs running Windows CE, e.g. CX8090, CX9020
- **CE-X86:** X86-based Embedded PCs running Windows CE, e.g. CX50xx, CX20x0
- **Win32:** Embedded PCs running Windows XP, Windows 7 or Windows Embedded Standard

The CE-ARM and CE-X86 directories contain the CAB files of the TwinCAT 3 function for Windows CE in relation to the respective hardware platform of the Windows CE device.

Example: "TF6310" installation folder



Name	Date modified	Type	Size
CE-ARM	26.03.2013 08:55	File folder	
CE-X86	26.03.2013 08:55	File folder	
Win32	21.02.2013 13:31	File folder	

Transfer the CAB file to the Windows CE device

Transfer the corresponding CAB file to the Windows CE device.

There are various options for transferring the executable file:

- via network shares
- via the integrated FTP server
- via ActiveSync
- via CF/SD cards

Further information can be found in the Beckhoff Information System in the "Operating Systems" documentation (Embedded PC > Operating Systems > CE).

Run the CAB file on the Windows CE device

After transferring the CAB file to the Windows CE device, double-click the file there. Confirm the installation dialog with **OK**. Then restart the Windows CE device.

After restarting the device, the files of the TwinCAT 3 function (TFxxxx) are automatically loaded in the background and are then available.

The software is installed in the following directory on the Windows CE device:
\Hard Disk\TwinCAT\Functions\TFxxxx

Software upgrade

If an older version of the TwinCAT 3 function is already installed on the Windows CE device, carry out the following steps on the Windows CE device to upgrade to a new version:

1. Open the CE Explorer by clicking **Start > Run** and entering "Explorer".
2. Navigate to *\Hard Disk\TwinCAT\Functions\TFxxx\xxxx*.
3. Rename the file *Tc*.exe* to *Tc*.old*.
4. Restart the Windows CE device.
5. Transfer the new CAB file to the Windows CE device.
6. Run the CAB file on the Windows CE device and install the new version.
7. Delete the file *Tc*.old*.
8. Restart the Windows CE device.

⇨ The new version is active after the restart.

3.4 Installing the TwinCAT/BSD

The TwinCAT 3 Function TF6250 - Modbus TCP is available as package `TF6250-Modbus-TCP` for TwinCAT/BSD in the package repository. The package can be installed via the command:

```
doas pkg install TF6250-Modbus-TCP
```

Further information about the Package Server can be found in the TwinCAT/BSD manual.

The installation stores the TwinCAT Modbus TCP Server and its default configuration file `TcModbusSrv.xml` in the following directory:

```
ls /usr/local/etc/TwinCAT/Functions/TF6250-Modbus-TCP
```

After a restart of the system or restart of TwinCAT, the Modbus TCP Server is started and the configuration from `TcModbusSrv.xml` is taken over.

Information about the configuration can be found in the chapter Configuration.

To adapt the configuration file, different text editors are available under TwinCAT/BSD.

Alternatively, you can use a remote access to replace `TcModbusSrv.xml` with an existing configuration file.

In order for the changes to `TcModbusSrv.xml` to be adopted by the TwinCAT Modbus Server, the TwinCAT system must be stopped and restarted. This can be achieved via the following command:

```
doas service TcSystemService restart
```

3.5 Licensing

The TwinCAT 3 function can be activated as a full version or as a 7-day test version. Both license types can be activated via the TwinCAT 3 development environment (XAE).

Licensing the full version of a TwinCAT 3 Function

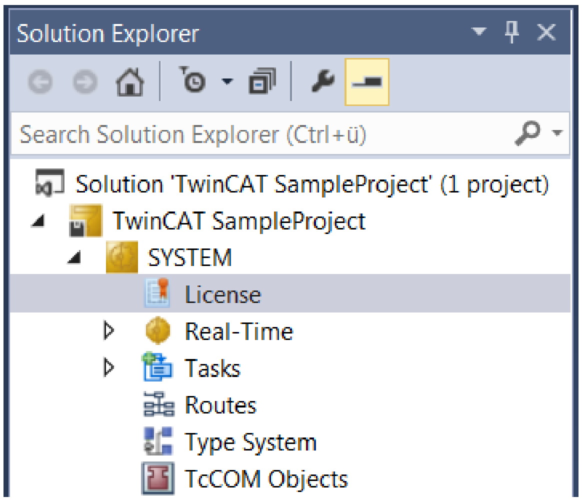
A description of the procedure to license a full version can be found in the Beckhoff Information System in the documentation "TwinCAT 3 Licensing".

Licensing the 7-day test version of a TwinCAT 3 Function

A 7-day test version cannot be enabled for a TwinCAT 3 license dongle.

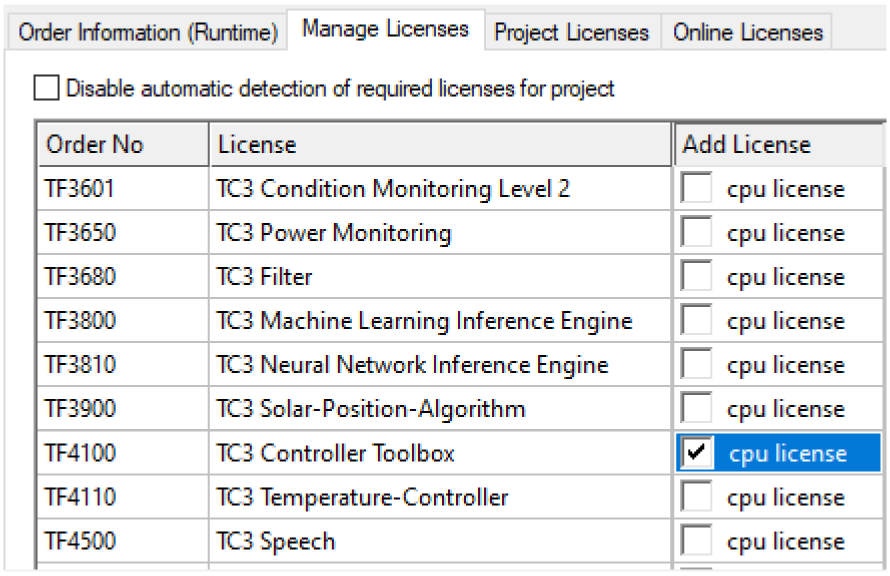
1. Start the TwinCAT 3 development environment (XAE).
2. Open an existing TwinCAT 3 project or create a new project.
3. If you want to activate the license for a remote device, set the desired target system. To do this, select the target system from the **Choose Target System** drop-down list in the toolbar.
   ⇨ The licensing settings always refer to the selected target system. When the project is activated on the target system, the corresponding TwinCAT 3 licenses are automatically copied to this system.

4. In the **Solution Explorer**, double-click **License** in the **SYSTEM** subtree.



⇨ The TwinCAT 3 license manager opens.

5. Open the **Manage Licenses** tab. In the **Add License** column, check the check box for the license you want to add to your project (e.g. "TF4100 TC3 Controller Toolbox").



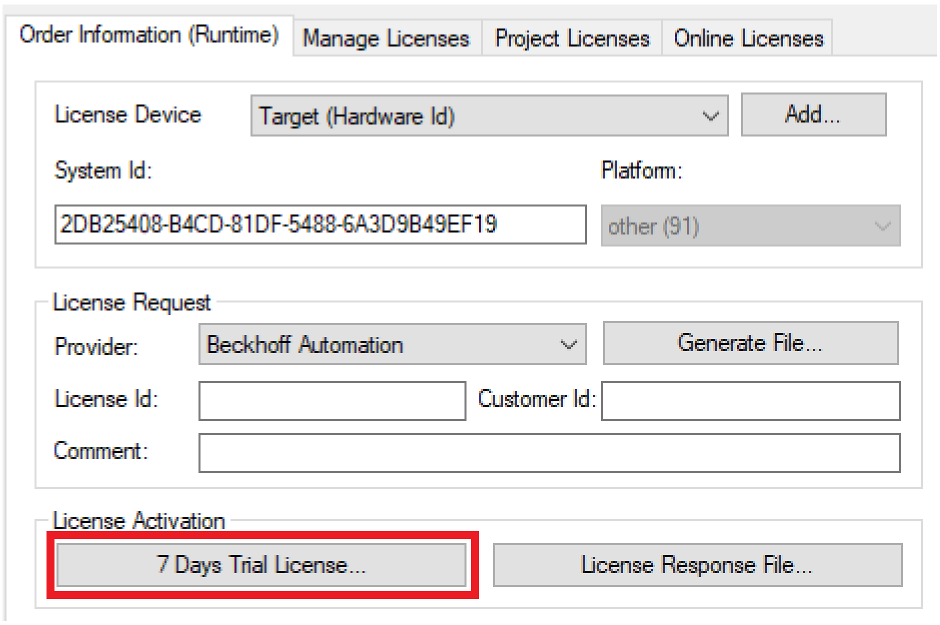
☐ Disable automatic detection of required licenses for project

Order No	License	Add License
TF3601	TC3 Condition Monitoring Level 2	☐ cpu license
TF3650	TC3 Power Monitoring	☐ cpu license
TF3680	TC3 Filter	☐ cpu license
TF3800	TC3 Machine Learning Inference Engine	☐ cpu license
TF3810	TC3 Neural Network Inference Engine	☐ cpu license
TF3900	TC3 Solar-Position-Algorithm	☐ cpu license
TF4100	TC3 Controller Toolbox	✓ cpu license
TF4110	TC3 Temperature-Controller	☐ cpu license
TF4500	TC3 Speech	☐ cpu license

6. Open the **Order Information (Runtime)** tab.

⇨ In the tabular overview of licenses, the previously selected license is displayed with the status “missing”**.**

7. Click **7-Day Trial License...** to activate the 7-day trial license.



License Device: Target (Hardware Id)

System Id: 2DB25408-B4CD-81DF-5488-6A3D9B49EF19

Platform: other (91)

License Request

Provider: Beckhoff Automation

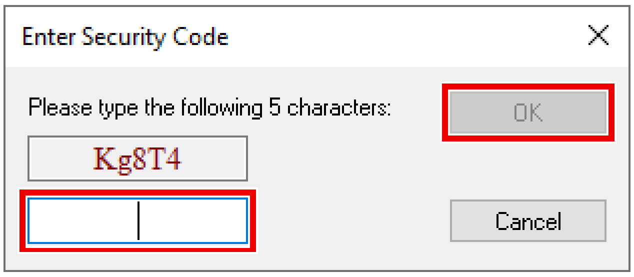
License Id:

Customer Id:

Comment:

License Activation

⇨ A dialog box opens, prompting you to enter the security code displayed in the dialog.



Please type the following 5 characters:

Kg8T4

8. Enter the code exactly as it is displayed and confirm the entry.
9. Confirm the subsequent dialog, which indicates the successful activation.
   ⇨ In the tabular overview of licenses, the license status now indicates the expiry date of the license.
10. Restart the TwinCAT system.

⇨ The 7-day trial version is enabled.

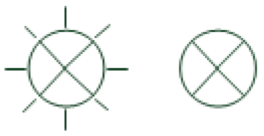
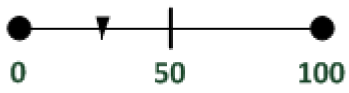
4 Configuration

4.1 Overview

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	
digitale 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 [▶ 19].

The TwinCAT Modbus TCP/IP server configurator [▶ 17] is used for configuring this mapping.

ADS-Access

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

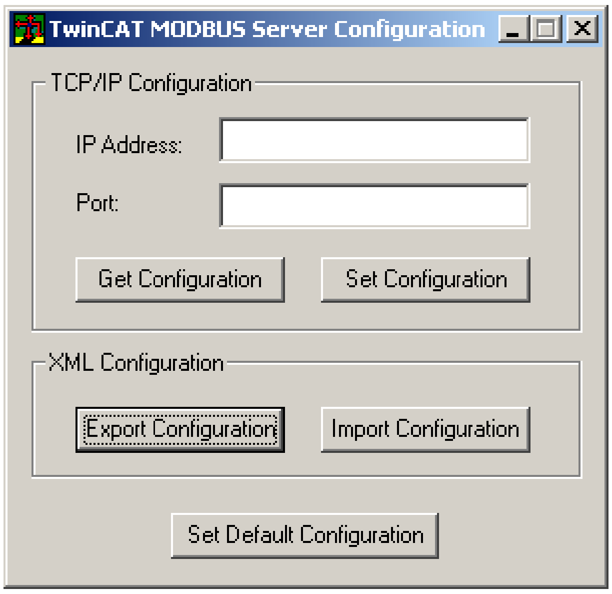
```
VAR_GLOBAL
mb_Input_Coils    :    ARRAY [0..255] OF BOOL;
mb_Output_Coils :    ARRAY [0..255] OF BOOL;
mb_Input_Registers :    ARRAY [0..255] OF WORD;
mb_Output_Registers :    ARRAY [0..255] OF WORD;
END_VAR
```

Requirements

Development environment	Target system type	PLC libraries to be linked
TwinCAT v3.0.0	PC or CX (x86)	Tc2_ModbusSrv

4.2 TwinCAT Modbus TCP Configurator

The configurator is installed per default to the directory **\TwinCAT3\Functions\TF6250-Modbus-TCP**. The tools allow to read and change the actual configuration of TwinCAT Modbus TCP server.



IP Address: IP of the server. If no address is set, the local one is used (default) .

Port: Configured port of the server (default port = 502).

Get Configuration: Read configured IP address and port.

Set Configuration: Set IP address and port.

Export Configuration: Read and save configuration.

Import Configuration: Import new configuration.

Set Default Configuration: Reset to default-settings (use local ip, Port = 502, and default mapping [▶ 19]).

TwinCAT must be stopped if you want to use the configurator, which will be done by the tool.

Export configuration

The configuration is XML-based and can be changed by a text editor. With "Export Configuration" the actual configuration can be stored local as XML-file.

It is easier to edit and activate an exported configuration.

Import Mapping-Informations

With "Import Configuration" a changed configuration can be imported and activated.

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

Windows CE

The standard configuration is in the TcModbusSrv.xml (path: \TwinCAT3\Functions\TF6250-Modbus-TCP\Server). If you change the settings in the file, a restart is necessary.

4.3 Default Configuration

The default mapping is shown in the following table:

Modbus areas	Modbus address	ADS area	
Digital inputs	0x8000 - 0x80FF	**Name of the variables in the PLC program**	**Data type**
		GVL.mb_Input_Coils	ARRAY [0..255] OF BOOL
Digital outputs (coils)	0x8000 - 0x80FF	**Name of the variables in the PLC program**	**Data type**
		GVL.mb_Output_Coils	ARRAY [0..255] OF BOOL
Input registers	0x8000 - 0x80FF	**Name of the variables in the PLC program**	**Data type**
		GVL.mb_Input_Registers	ARRAY [0..255] OF WORD
Output registers	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**
		GVL.mb_Output_Registers	ARRAY [0..255] OF WORD

The server maps the individuals ADS areas and enables the access to the physical process image and maps the PLC data area.

The mapping can be adjusted by the TwinCAT Modbus TCP Configurator [▶ 17].

5 Diagnosis

5.1 Modbus ADS Diagnosis Interface

Modbus ADS diagnosis interface

Via ADS the following information can be monitored:

AMSNetID: AMSNetID of the system. If the local system is used leave empty.

Port: 10500 (AMSPORT_R3_MODBUSSERV)

ADSRead: see ADSREAD in the manual TwinCAT 3 PLC Lib: Tc2_System

index group	index offset	access	data type	description	minimal Modbus server version
0x2000	0	ADS Read	UINT32	**GetConnectedClientCount** returns the number of connected Modbus clients	1.0.50
0x2000	1	ADS Read	UINT32	**GetModbusRequestCount** returns the received Modbus requests	1.0.50
0x2000	2	ADS Read	UINT32	**GetModbusResponseCount** returns the received Modbus answers	1.0.50

6 PLC libraries

6.1 Overview

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

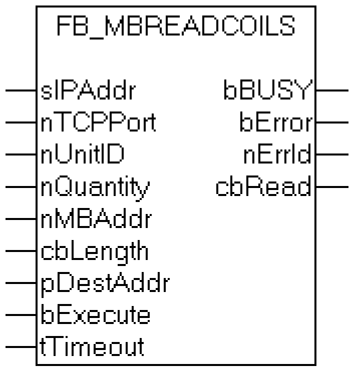
Modbus TCP function	Function code	PLC block
Read Coils	1	FB_MBReadCoils [▸ 21]
Read Inputs	2	FB_MBReadInputs [▸ 23]
Read Registers	3	FB_MBReadRegs [▸ 25]
Read Input Registers	4	FB_MBReadInputRegs [▸ 27]
Write Single Coil	5	FB_MBWriteSingleCoil [▸ 29]
Write Single Register	6	FB_MBWriteSingleReg [▸ 31]
Write Multiple Coils	15	FB_MBWriteCoils [▸ 32]
Write Multiple Registers	16	FB_MBWriteRegs [▸ 34]
Read/Write Multiple Registers	23	FB_MBReadWriteRegs [▸ 36]
Diagnostic	8	FB_MBDiagnose [▸ 38]

Requirements

Development environment	Target system type	PLC libraries to be linked
TwinCAT v3.0.0	PC or CX (x86)	Tc2_ModbusSrv

6.2 Function blocks

6.2.1 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 : POINTER OF BYTE;
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.

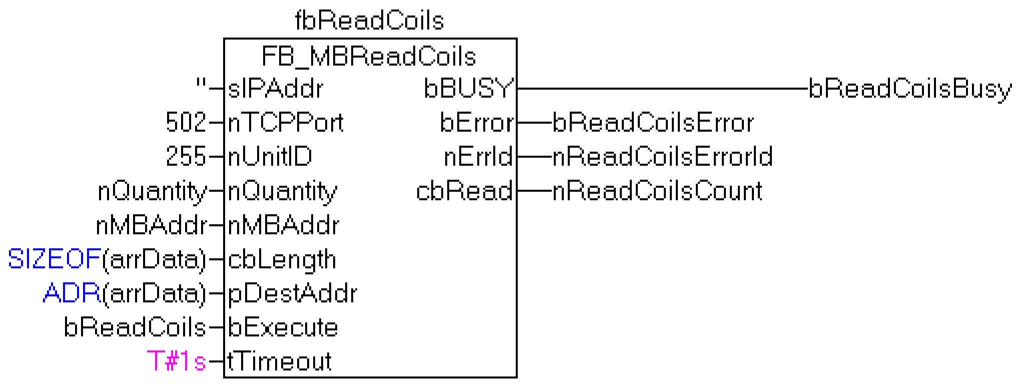
nErrId : Supplies the ADS error number [▸ 56] when the bError output is set.

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
```



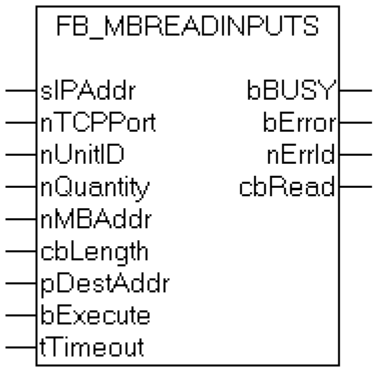
After a rising edge of "bExecute" and successful execution of the ReadCoils command, the content of digital outputs 6 - 15 is written into the arrData array:

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 v3.0.0	PC or CX (x86)	Tc2_ModbusSrv

6.2.2 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   : POINTER OF BYTE;
    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.

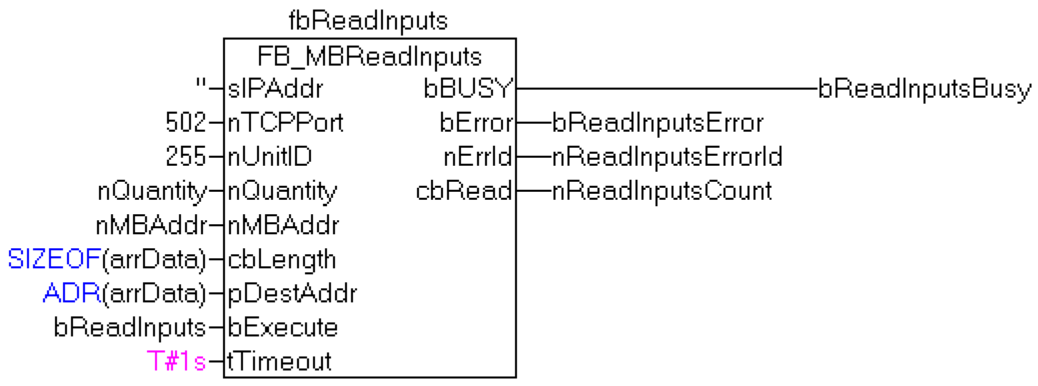
nErrId : Supplies the ADS error number [▸ 56] when the bError output is set.

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
```



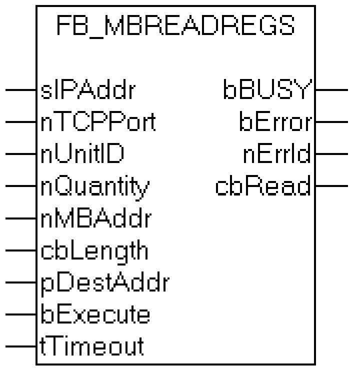
After a rising edge of "bExecute" and successful execution of the ReadInputs command, the content of digital inputs 30 - 49 is written into the arrData array:

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 v3.0.0	PC or CX (x86)	Tc2_ModbusSrv

6.2.3 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  : POINTER OF BYTE;
    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.

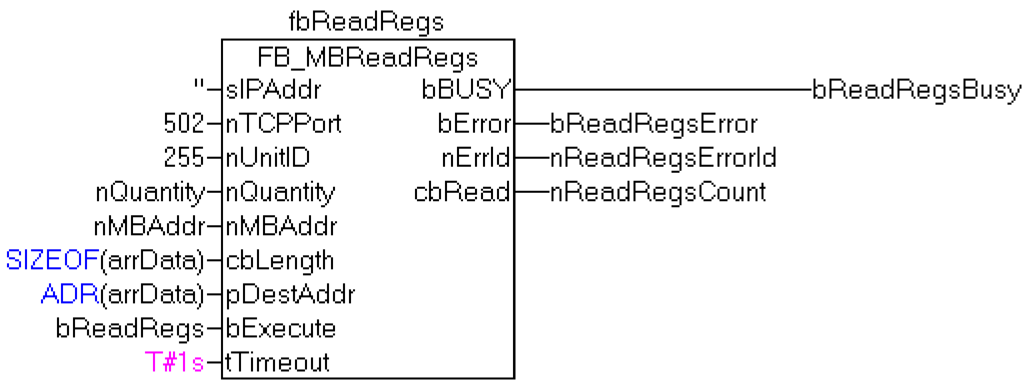
nErrId : Supplies the ADS error number [▶ 56] when the bError output is set.

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
```



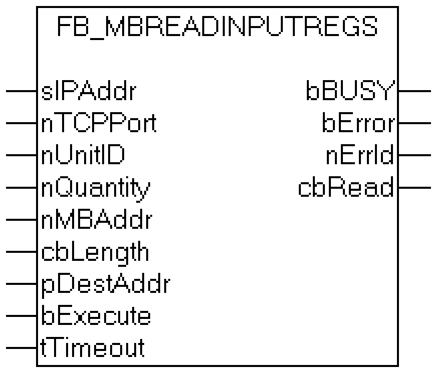
After a rising edge of "bExecute" and successful execution of the ReadRegs command, the content of registers 25 and 26 is located in the arrData array:

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 v3.0.0	PC or CX (x86)	Tc2_ModbusSrv

6.2.4 FB_MBReadInputRegs (Modbus function 4)



This function is used for reading 1 to 128 input registers (16 bit). Observe the byte-order little endian.

VAR_INPUT

```
VAR_INPUT
    sIPAddr    : STRING(15);
    nTCPPort   : UINT:= MODBUS_TCP_PORT;
    nUnitID    : BYTE:=16#FF;
    nQuantity  : WORD;
    nMBAddr    : WORD;
    cbLength   : UDINT;
    pDestAddr  : POINTER OF BYTE;
    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.

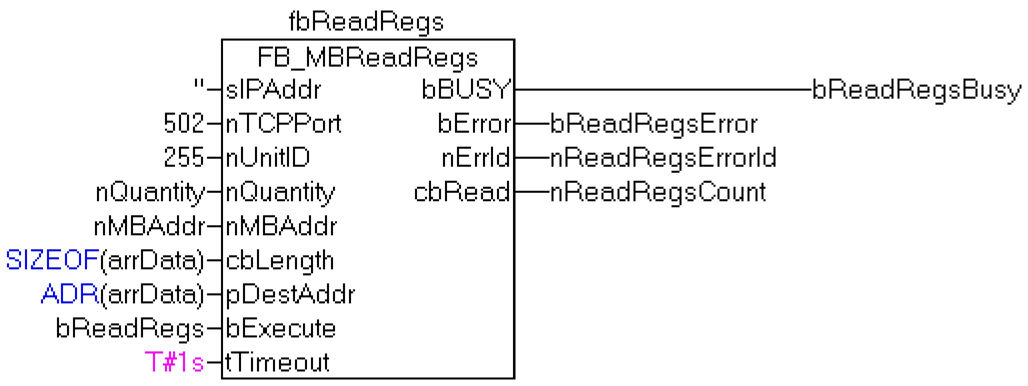
nErrId : Supplies the ADS error number [▶ 56] when the bError output is set.

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 := 3;
    nMBAddr             : WORD:= 2;
    arrData             : ARRAY [1..3] OF WORD;
END_VAR
```



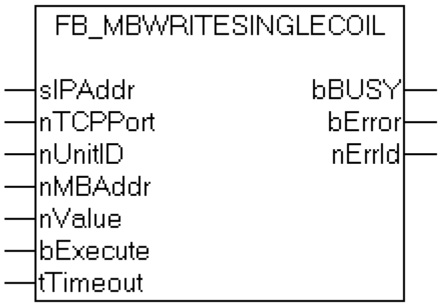
After a rising edge of "bExecute" and successful execution of the ReadRegs command, the content of registers 3-5 is located in the arrData array:

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

Requirements

Development environment	Target system type	PLC libraries to be linked
TwinCAT v3.0.0	PC or CX (x86)	Tc2_ModbusSrv

6.2.5 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.

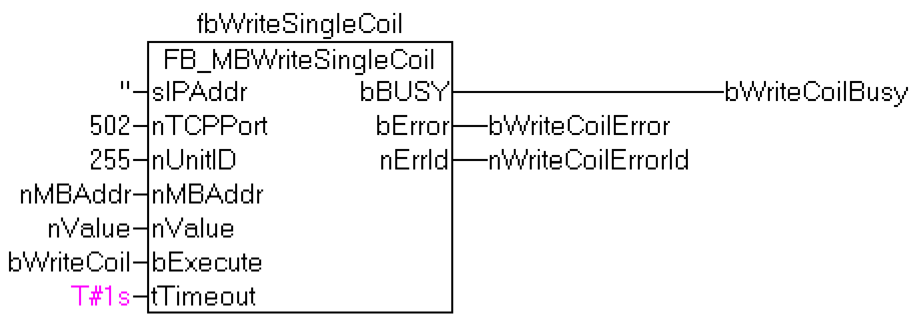
bError : If an ADS error should occur during the transfer of the command, then this output is set once the bBusy output is reset.

nErrId : Supplies the ADS error number [▸ 56] 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
```

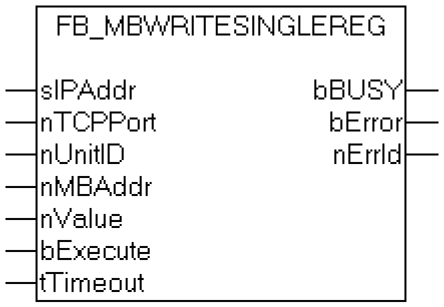


After a rising edge of "bExecute" and successful execution of the WriteSingleCoil command, digital output 4 is switched on.

Requirements

Development environment	Target system type	PLC libraries to be linked
TwinCAT v3.0.0	PC or CX (x86)	Tc2_ModbusSrv

6.2.6 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.

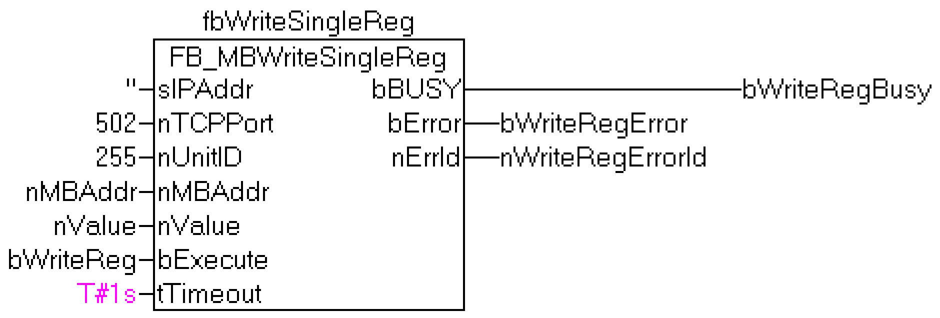
bError : If an ADS error should occur during the transfer of the command, then this output is set once the bBusy output is reset.

nErrId : Supplies the ADS error number [▶ 56] 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
    fbWriteSingleReg      : FB_MBWriteSingleReg;
    bWriteReg             : BOOL;
    bWriteRegBusy         : BOOL;
    bWriteRegError        : BOOL;
    nWriteRegErrorId      : UDINT;
    nMBAddr               : WORD := 4;
    nValue                : WORD := 16#1234;
END_VAR
```

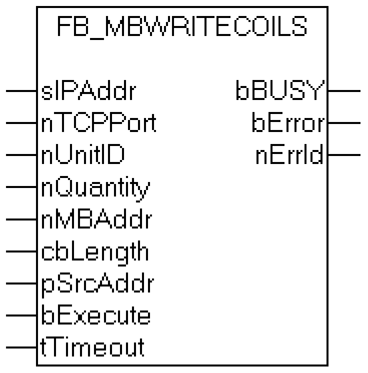


After a rising edge of "bExecute" and successful execution of the WriteSingleReg command, the value 16#1234 is written into register 5.

Requirements

Development environment	Target system type	PLC libraries to be linked
TwinCAT v3.0.0	PC or CX (x86)	Tc2_ModbusSrv

6.2.7 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     : POINTER OF BYTE;
    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.

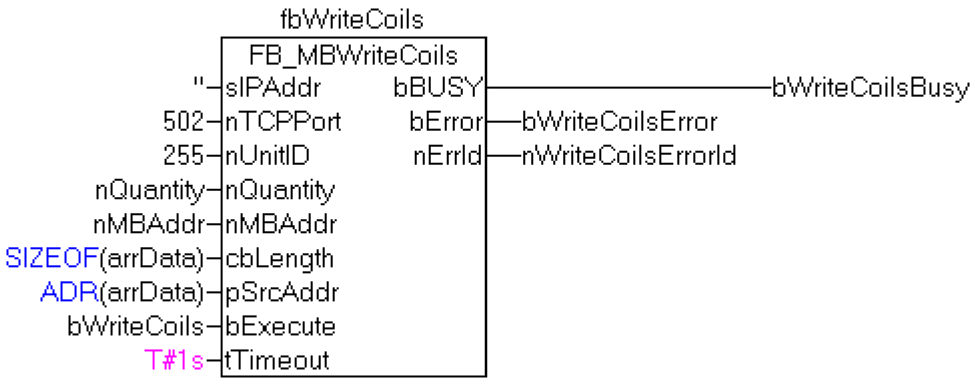
bError : If an ADS error should occur during the transfer of the command, then this output is set once the bBusy output is reset.

nErrId : Supplies the ADS error number [▸ 56] 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
```



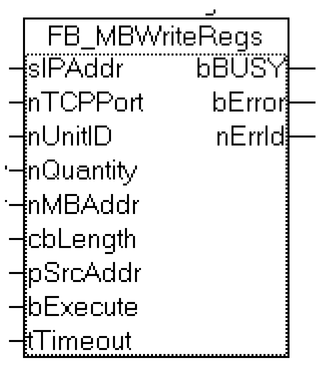
After a rising edge of "bExecute" and successful execution of the ReadCoils command, the content of the arrData array is written to digital outputs 15 - 24:

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 v3.0.0	PC or CX (x86)	Tc2_ModbusSrv

6.2.8 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     : POINTER OF BYTE;
    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.

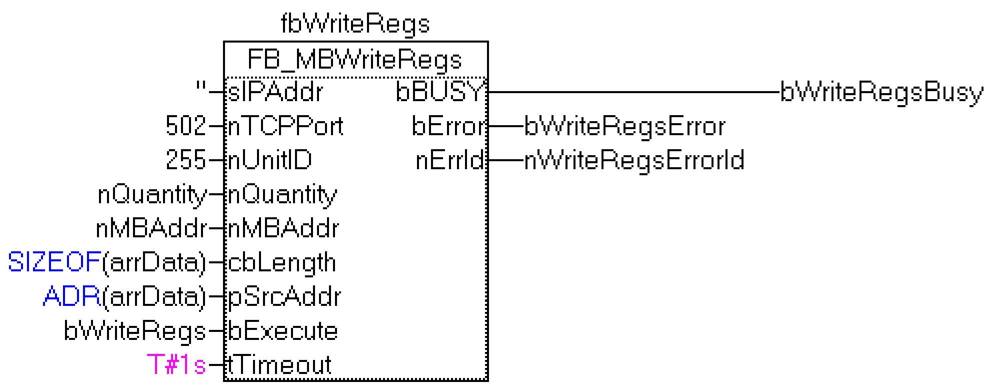
bError: If an ADS error should occur during the transfer of the command, then this output is set once the bBusy output is reset.

nErrId: Supplies the ADS error number [▶ 56] 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
```

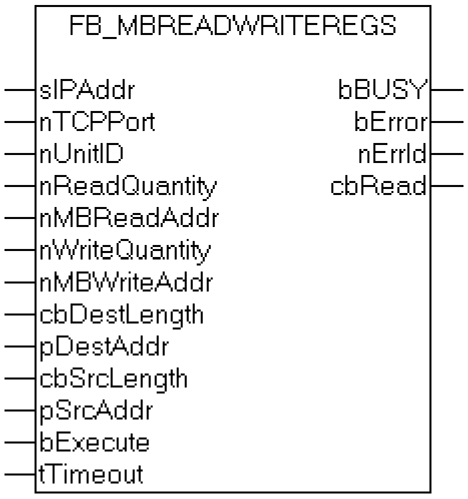


After a rising edge of "bExecute" and successful execution of the ReadRegs command, the content of the arrData array is written to registers 5-7.

Requirements

Development environment	Target system type	PLC libraries to be linked
TwinCAT v3.0.0	PC or CX (x86)	Tc2_ModbusSrv

6.2.9 FB_MBReadWriteRegs (Modbus-Funktion 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           : POINTER OF BYTE;
    cbSrcLength         : UDINT;
    pSrcAddr            : POINTER OF BYTE;
    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.

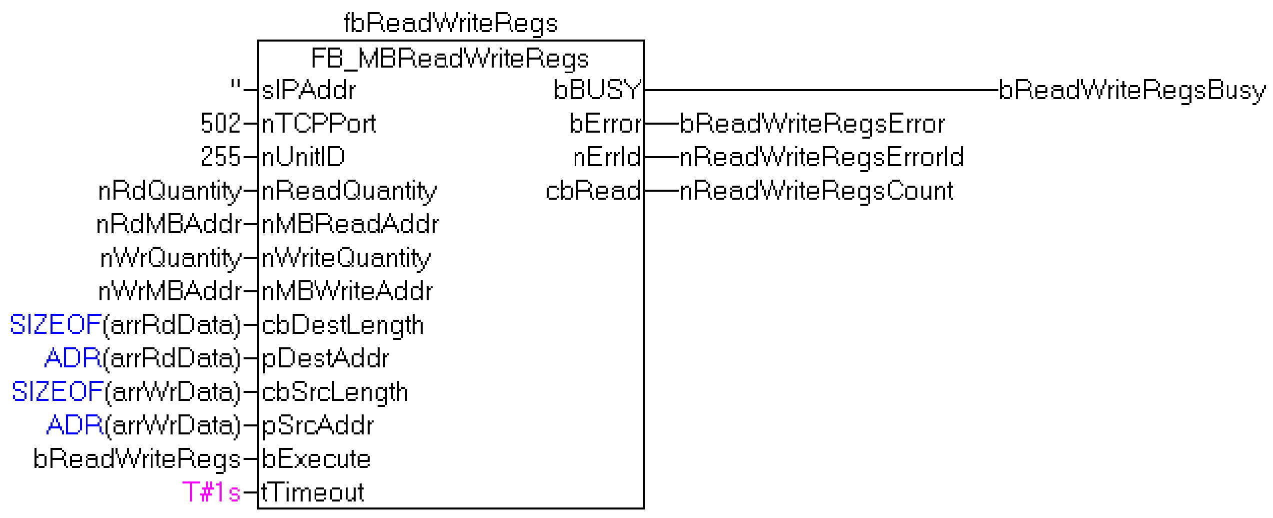
nErrId: Supplies the ADS error number [▸ 56] when the bError output is set.

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
```

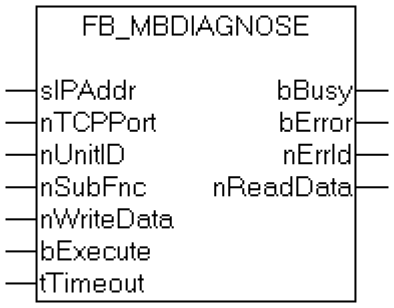


After a rising edge of "bExecute" and successful execution of the ReadWriteRegs command, arrRdData contains the read register data, and the data from arrWrData are written to the registers.

Requirements

Development environment	Target system type	PLC libraries to be linked
TwinCAT v3.0.0	PC or CX (x86)	Tc2_ModbusSrv

6.2.10 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.

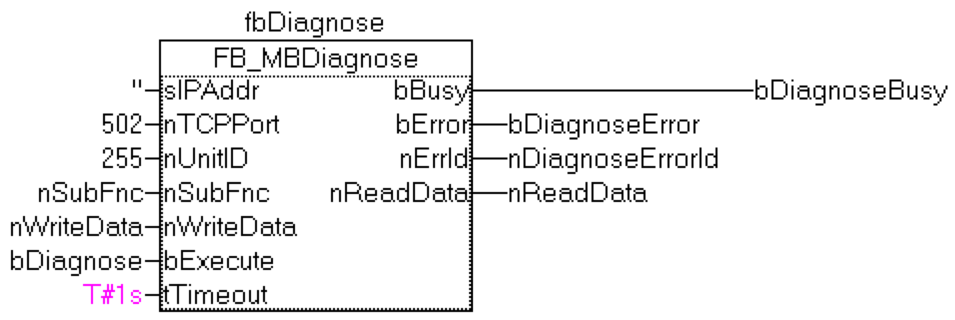
nErrId : Supplies the ADS error number [▸ 56] when the bError output is set.

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
```



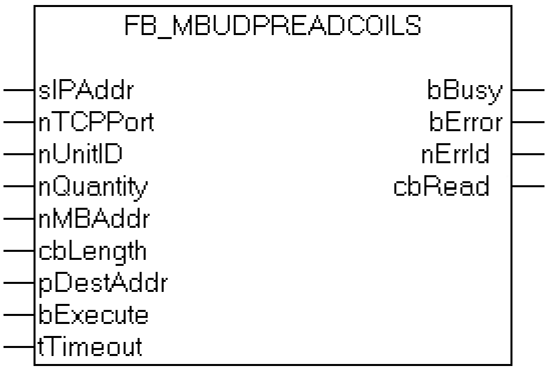
After rising edge of "bExecute" and successful execution of the diagnosis command, nReadData contains the read data word.

Requirements

Development environment	Target system type	PLC libraries to be linked
TwinCAT v3.0.0	PC or CX (x86)	Tc2_ModbusSrv

6.2.11 UDP

6.2.11.1 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 [▸ 56] when the bError output is set.

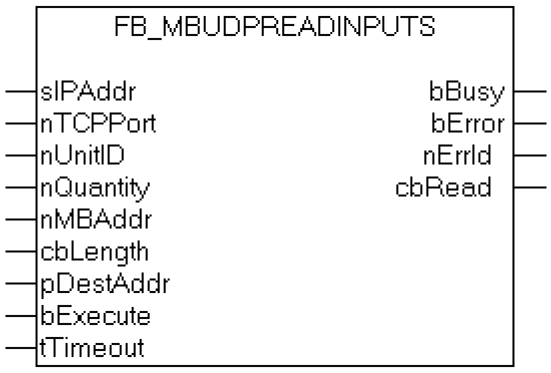
cbRead: Contains the number of bytes currently read.

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 v3.0.0	PC or CX (x86)	Tc2_ModbusSrv

6.2.11.2 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  : POINTER OF BYTE;
    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 [▸ 56] when the bError output is set.

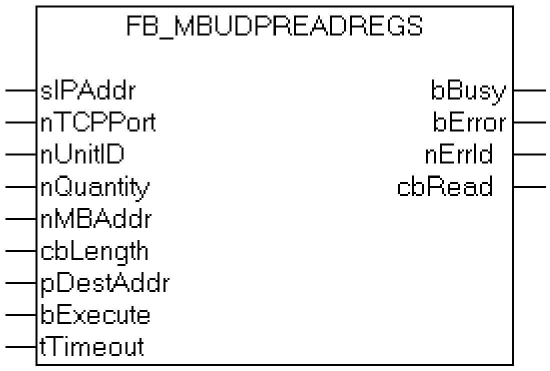
cbRead: Contains the number of bytes currently read.

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 v3.0.0	PC or CX (x86)	Tc2_ModbusSrv

6.2.11.3 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  : POINTER OF BYTE;
    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 [▸ 56] when the bError output is set.

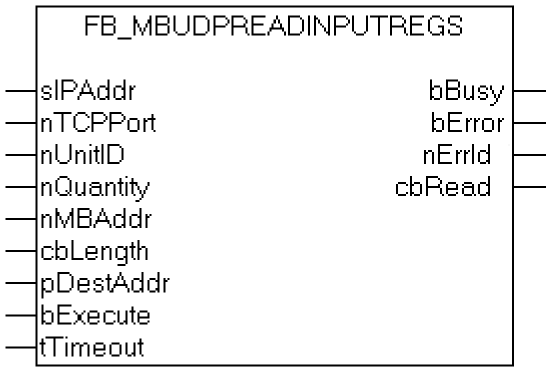
cbRead: Contains the number of bytes currently read.

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 v3.0.0	PC or CX (x86)	Tc2_ModbusSrv

6.2.11.4 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 : POINTER OF BYTE;
    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 [▸ 56] when the bError output is set.

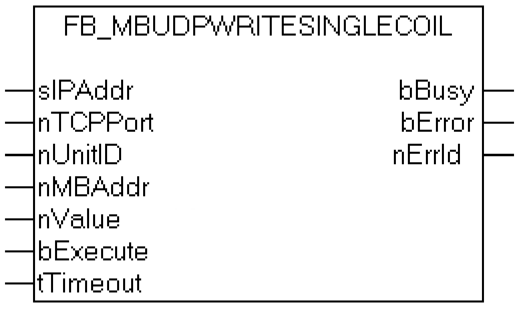
cbRead: Contains the number of bytes currently read.

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 v3.0.0	PC or CX (x86)	Tc2_ModbusSrv

6.2.11.5 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.

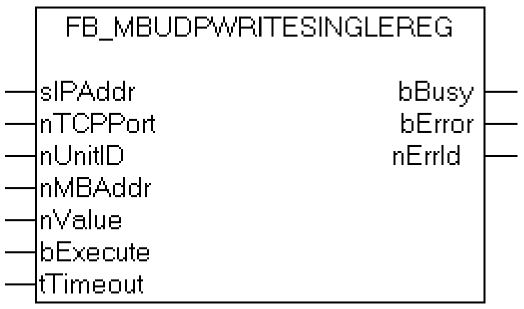
nErrId : Supplies the ADS error number [▶ 56] when the bError output is set.

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 v3.0.0	PC or CX (x86)	Tc2_ModbusSrv

6.2.11.6 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.

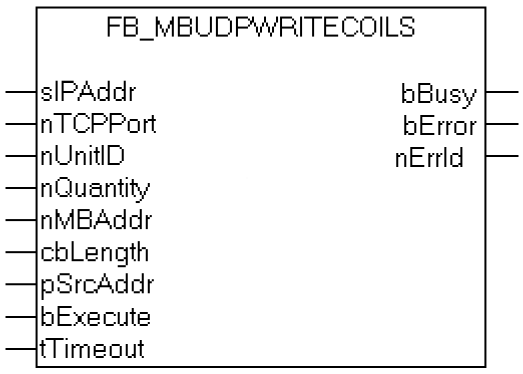
nErrId : Supplies the ADS error number [▸ 56] when the bError output is set.

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 v3.0.0	PC or CX (x86)	Tc2_ModbusSrv

6.2.11.7 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   : POINTER OF BYTE;
    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.

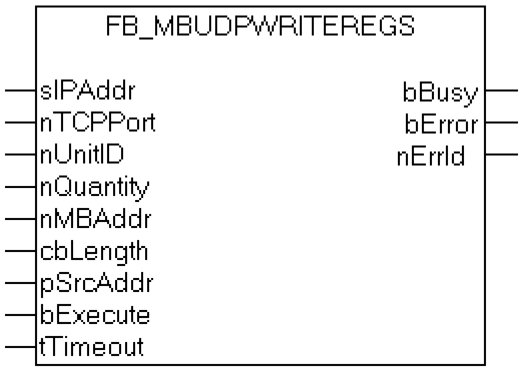
nErrId : Supplies the ADS error number [▶ 56] when the bError output is set.

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 v3.0.0	PC or CX (x86)	Tc2_ModbusSrv

6.2.11.8 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  : POINTER OF BYTE;
    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.

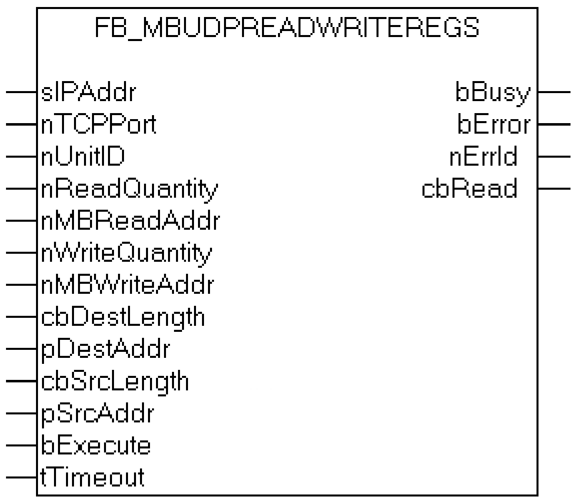
nErrId: Supplies the ADS error number [▶ 56] when the bError output is set.

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 v3.0.0	PC or CX (x86)	Tc2_ModbusSrv

6.2.11.9 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       : POINTER OF BYTE;
    cbSrcLength     : UDINT;
    pSrcAddr        : POINTER OF BYTE;
    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 [▶ 56] when the bError output is set.

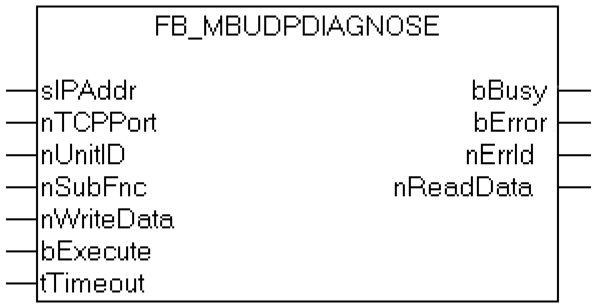
cbRead: Contains the number of bytes currently read.

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 v3.0.0	PC or CX (x86)	Tc2_ModbusSrv

6.2.11.10 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 [▸ 56] 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

Requirements

Development environment	Target system type	PLC libraries to be linked
TwinCAT v3.0.0	PC or CX (x86)	Tc2_ModbusSrv

6.3 Global constants

6.3.1 Library Version

All libraries have a specific version. This version is shown in the PLC library repository too.
A global constant contains the library version information:

Global_Version

```
VAR_GLOBAL CONSTANT
    stLibVersion_Tc2_ModbusSrv : ST_LibVersion;
END_VAR
```

To compare the existing version to a required version the function F_CmpLibVersion (defined in Tc2_System library) is offered.

All other possibilities known from TwinCAT2 libraries to query a library version are obsolete!

7 Samples

7.1 Sample: Digital IO access

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

The default mapping [▶ 19] 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);

ELSEIF 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

Requirements

Development environment	Target system type	PLC libraries to be linked
TwinCAT v3.0.0	PC or CX (x86)	Tc2_ModbusSrv

7.2 Sample: Multiple register access

This sample explains the access to the register of aTwinCAT system via Modbus.

The Modbusaddress **0x3000** is mapped by the default-configuration to the memory area of the plc (ADS-Indexgroup 0x4020)

```
PROGRAM MAIN
VAR
ipAddr      : STRING(15)  := '';
M0 AT%MB0 : ARRAY [0..3] OF WORD;
nValue      : ARRAY [0..3] OF WORD;
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
```

The array arrValue will be written to the memory area of the plc (M0) by a rising edge on bWriteRegs.

Requirements

Development environment	Target system type	PLC libraries to be linked
TwinCAT v3.0.0	PC or CX (x86)	Tc2_ModbusSrv

Also see about this

- Default Configuration [▶ 19]

8 Appendix

8.1 Overview

TwinCAT ADS return code

Hex	Dezimal	Source
0x00000000-0x00007800	0-30720	TwinCAT System return codes
0x00008000-0x000080FF	32768-33023	Internal TwinCAT Modbus TCP
0x80070000-0x8007FFFF	2147942400-2148007935	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

Requirements

Development environment	Target system type	PLC libraries to be linked
TwinCAT v3.0.0	PC or CX (x86)	Tc2_ModbusSrv

8.2 ADS Return Codes

Grouping of error codes: 0x000 [▸ 56]..., 0x500 [▸ 57]..., 0x700 [▸ 58]..., 0x1000 [▸ 60]...

Global error codes

Hex	Dec	HRESULT	Name	Description
0x0	0	0x9811 0000	ERR_NOERROR	No error.
0x1	1	0x9811 0001	ERR_INTERNAL	Internal error.
0x2	2	0x9811 0002	ERR_NORTIME	No real-time.
0x3	3	0x9811 0003	ERR_ALLOCLOCKEDMEM	Allocation locked – memory error.
0x4	4	0x9811 0004	ERR_INSERTMAILBOX	Mailbox full – the ADS message could not be sent. Reducing the number of ADS messages per cycle will help.
0x5	5	0x9811 0005	ERR_WRONGRECEIVEHMSG	Wrong HMSG.
0x6	6	0x9811 0006	ERR_TARGETPORTNOTFOUND	Target port not found – ADS server is not started or is not reachable.
0x7	7	0x9811 0007	ERR_TARGETMACHINENOTFOUND	Target computer not found – AMS route was not found.
0x8	8	0x9811 0008	ERR_UNKNOWNCMDID	Unknown command ID.
0x9	9	0x9811 0009	ERR_BADTASKID	Invalid task ID.
0xA	10	0x9811 000A	ERR_NOIO	No IO.
0xB	11	0x9811 000B	ERR_UNKNOWNAMSCMD	Unknown AMS command.
0xC	12	0x9811 000C	ERR_WIN32ERROR	Win32 error.
0xD	13	0x9811 000D	ERR_PORTNOTCONNECTED	Port not connected.
0xE	14	0x9811 000E	ERR_INVALIDAMSLENGTH	Invalid AMS length.
0xF	15	0x9811 000F	ERR_INVALIDAMSNETID	Invalid AMS Net ID.
0x10	16	0x9811 0010	ERR_LOWINSTLEVEL	Installation level is too low –TwinCAT 2 license error.
0x11	17	0x9811 0011	ERR_NODEBUGINTAVAILABLE	No debugging available.
0x12	18	0x9811 0012	ERR_PORTDISABLED	Port disabled – TwinCAT system service not started.
0x13	19	0x9811 0013	ERR_PORTALREADYCONNECTED	Port already connected.
0x14	20	0x9811 0014	ERR_AMSSYNC_W32ERROR	AMS Sync Win32 error.
0x15	21	0x9811 0015	ERR_AMSSYNC_TIMEOUT	AMS Sync Timeout.
0x16	22	0x9811 0016	ERR_AMSSYNC_AMSERROR	AMS Sync error.
0x17	23	0x9811 0017	ERR_AMSSYNC_NOINDEXINMAP	No index map for AMS Sync available.
0x18	24	0x9811 0018	ERR_INVALIDAMSPORT	Invalid AMS port.
0x19	25	0x9811 0019	ERR_NOMEMORY	No memory.
0x1A	26	0x9811 001A	ERR_TCPSEND	TCP send error.
0x1B	27	0x9811 001B	ERR_HOSTUNREACHABLE	Host unreachable.
0x1C	28	0x9811 001C	ERR_INVALIDAMSFRAGMENT	Invalid AMS fragment.
0x1D	29	0x9811 001D	ERR_TLSSEND	TLS send error – secure ADS connection failed.
0x1E	30	0x9811 001E	ERR_ACCESSDENIED	Access denied – secure ADS access denied.

Router error codes

Hex	Dec	HRESULT	Name	Description
0x500	1280	0x9811 0500	ROUTERERR_NOLOCKEDMEMORY	Locked memory cannot be allocated.
0x501	1281	0x9811 0501	ROUTERERR_RESIZEMEMORY	The router memory size could not be changed.
0x502	1282	0x9811 0502	ROUTERERR_MAILBOXFULL	The mailbox has reached the maximum number of possible messages.
0x503	1283	0x9811 0503	ROUTERERR_DEBUGBOXFULL	The Debug mailbox has reached the maximum number of possible messages.
0x504	1284	0x9811 0504	ROUTERERR_UNKNOWNPORTTYPE	The port type is unknown.
0x505	1285	0x9811 0505	ROUTERERR_NOTINITIALIZED	The router is not initialized.
0x506	1286	0x9811 0506	ROUTERERR_PORTALREADYINUSE	The port number is already assigned.
0x507	1287	0x9811 0507	ROUTERERR_NOTREGISTERED	The port is not registered.
0x508	1288	0x9811 0508	ROUTERERR_NOMOREQUEUES	The maximum number of ports has been reached.
0x509	1289	0x9811 0509	ROUTERERR_INVALIDPORT	The port is invalid.
0x50A	1290	0x9811 050A	ROUTERERR_NOTACTIVATED	The router is not active.
0x50B	1291	0x9811 050B	ROUTERERR_FRAGMENTBOXFULL	The mailbox has reached the maximum number for fragmented messages.
0x50C	1292	0x9811 050C	ROUTERERR_FRAGMENTTIMEOUT	A fragment timeout has occurred.
0x50D	1293	0x9811 050D	ROUTERERR_TOBEREMOVED	The port is removed.

General ADS error codes

Hex	Dec	HRESULT	Name	Description
0x700	1792	0x9811 0700	ADSERR_DEVICE_ERROR	General device error.
0x701	1793	0x9811 0701	ADSERR_DEVICE_SRVNOTSUPP	Service is not supported by the server.
0x702	1794	0x9811 0702	ADSERR_DEVICE_INVALIDGRP	Invalid index group.
0x703	1795	0x9811 0703	ADSERR_DEVICE_INVALIDOFFSET	Invalid index offset.
0x704	1796	0x9811 0704	ADSERR_DEVICE_INVALIDACCESS	Reading or writing not permitted.
0x705	1797	0x9811 0705	ADSERR_DEVICE_INVALIDSIZE	Parameter size not correct.
0x706	1798	0x9811 0706	ADSERR_DEVICE_INVALIDDATA	Invalid data values.
0x707	1799	0x9811 0707	ADSERR_DEVICE_NOTREADY	Device is not ready to operate.
0x708	1800	0x9811 0708	ADSERR_DEVICE_BUSY	Device is busy.
0x709	1801	0x9811 0709	ADSERR_DEVICE_INVALIDCONTEXT	Invalid operating system context. This can result from use of ADS function blocks in different tasks. It may be possible to resolve this through Multi-task data access synchronization in the PLC.
0x70A	1802	0x9811 070A	ADSERR_DEVICE_NOMEMORY	Insufficient memory.
0x70B	1803	0x9811 070B	ADSERR_DEVICE_INVALIDPARM	Invalid parameter values.
0x70C	1804	0x9811 070C	ADSERR_DEVICE_NOTFOUND	Not found (files, ...).
0x70D	1805	0x9811 070D	ADSERR_DEVICE_SYNTAX	Syntax error in file or command.
0x70E	1806	0x9811 070E	ADSERR_DEVICE_INCOMPATIBLE	Objects do not match.
0x70F	1807	0x9811 070F	ADSERR_DEVICE_EXISTS	Object already exists.
0x710	1808	0x9811 0710	ADSERR_DEVICE_SYMBOLNOTFOUND	Symbol not found.
0x711	1809	0x9811 0711	ADSERR_DEVICE_SYMBOLVERSIONIN-VALID	Invalid symbol version. This can occur due to an on-line change. Create a new handle.
0x712	1810	0x9811 0712	ADSERR_DEVICE_INVALIDSTATE	Device (server) is in invalid state.
0x713	1811	0x9811 0713	ADSERR_DEVICE_TRANSMODENOTSUPP	AdsTransMode not supported.
0x714	1812	0x9811 0714	ADSERR_DEVICE_NOTIFYHNDINVALID	Notification handle is invalid.
0x715	1813	0x9811 0715	ADSERR_DEVICE_CLIENTUNKNOWN	Notification client not registered.
0x716	1814	0x9811 0716	ADSERR_DEVICE_NOMOREHDLS	No further handle available.
0x717	1815	0x9811 0717	ADSERR_DEVICE_INVALIDWATCHSIZE	Notification size too large.
0x718	1816	0x9811 0718	ADSERR_DEVICE_NOTINIT	Device not initialized.
0x719	1817	0x9811 0719	ADSERR_DEVICE_TIMEOUT	Device has a timeout.
0x71A	1818	0x9811 071A	ADSERR_DEVICE_NOINTERFACE	Interface query failed.
0x71B	1819	0x9811 071B	ADSERR_DEVICE_INVALIDINTERFACE	Wrong interface requested.
0x71C	1820	0x9811 071C	ADSERR_DEVICE_INVALIDCLSID	Class ID is invalid.
0x71D	1821	0x9811 071D	ADSERR_DEVICE_INVALIDOBJID	Object ID is invalid.
0x71E	1822	0x9811 071E	ADSERR_DEVICE_PENDING	Request pending.
0x71F	1823	0x9811 071F	ADSERR_DEVICE_ABORTED	Request is aborted.
0x720	1824	0x9811 0720	ADSERR_DEVICE_WARNING	Signal warning.
0x721	1825	0x9811 0721	ADSERR_DEVICE_INVALIDARRAYIDX	Invalid array index.
0x722	1826	0x9811 0722	ADSERR_DEVICE_SYMBOLNOTACTIVE	Symbol not active.
0x723	1827	0x9811 0723	ADSERR_DEVICE_ACCESSDENIED	Access denied.
0x724	1828	0x9811 0724	ADSERR_DEVICE_LICENSENOTFOUND	Missing license.
0x725	1829	0x9811 0725	ADSERR_DEVICE_LICENSEEXPIRED	License expired.
0x726	1830	0x9811 0726	ADSERR_DEVICE_LICENSEEXCEEDED	License exceeded.
0x727	1831	0x9811 0727	ADSERR_DEVICE_LICENSEINVALID	Invalid license.
0x728	1832	0x9811 0728	ADSERR_DEVICE_LICENSESYSTEMID	License problem: System ID is invalid.
0x729	1833	0x9811 0729	ADSERR_DEVICE_LICENSENOTIMELIMIT	License not limited in time.
0x72A	1834	0x9811 072A	ADSERR_DEVICE_LICENSEFUTUREISSUE	License problem: Time in the future.
0x72B	1835	0x9811 072B	ADSERR_DEVICE_LICENSETIMETOLONG	License period too long.
0x72C	1836	0x9811 072C	ADSERR_DEVICE_EXCEPTION	Exception at system startup.
0x72D	1837	0x9811 072D	ADSERR_DEVICE_LICENSEDUPLICATED	License file read twice.
0x72E	1838	0x9811 072E	ADSERR_DEVICE_SIGNATUREINVALID	Invalid signature.
0x72F	1839	0x9811 072F	ADSERR_DEVICE_CERTIFICATEINVALID	Invalid certificate.
0x730	1840	0x9811 0730	ADSERR_DEVICE_LICENSEOEMNOT-FOUND	Public key not known from OEM.
0x731	1841	0x9811 0731	ADSERR_DEVICE_LICENSERESTRICTED	License not valid for this system ID.
0x732	1842	0x9811 0732	ADSERR_DEVICE_LICENSEDEMODENIED	Demo license prohibited.
0x733	1843	0x9811 0733	ADSERR_DEVICE_INVALIDFNCID	Invalid function ID.
0x734	1844	0x9811 0734	ADSERR_DEVICE_OUTOFRANGE	Outside the valid range.
0x735	1845	0x9811 0735	ADSERR_DEVICE_INVALIDALIGNMENT	Invalid alignment.

Hex	Dec	HRESULT	Name	Description
0x736	1846	0x9811 0736	ADSERR_DEVICE_LICENSEPLATFORM	Invalid platform level.
0x737	1847	0x9811 0737	ADSERR_DEVICE_FORWARD_PL	Context – forward to passive level.
0x738	1848	0x9811 0738	ADSERR_DEVICE_FORWARD_DL	Context – forward to dispatch level.
0x739	1849	0x9811 0739	ADSERR_DEVICE_FORWARD_RT	Context – forward to real-time.
0x740	1856	0x9811 0740	ADSERR_CLIENT_ERROR	Client error.
0x741	1857	0x9811 0741	ADSERR_CLIENT_INVALIDPARM	Service contains an invalid parameter.
0x742	1858	0x9811 0742	ADSERR_CLIENT_LISTEMPTY	Polling list is empty.
0x743	1859	0x9811 0743	ADSERR_CLIENT_VARUSED	Var connection already in use.
0x744	1860	0x9811 0744	ADSERR_CLIENT_DUPLINVOKEID	The called ID is already in use.
0x745	1861	0x9811 0745	ADSERR_CLIENT_SYNCTIMEOUT	Timeout has occurred – the remote terminal is not responding in the specified ADS timeout. The route setting of the remote terminal may be configured incorrectly.
0x746	1862	0x9811 0746	ADSERR_CLIENT_W32ERROR	Error in Win32 subsystem.
0x747	1863	0x9811 0747	ADSERR_CLIENT_TIMEOUTINVALID	Invalid client timeout value.
0x748	1864	0x9811 0748	ADSERR_CLIENT_PORTNOTOPEN	Port not open.
0x749	1865	0x9811 0749	ADSERR_CLIENT_NOAMSADDR	No AMS address.
0x750	1872	0x9811 0750	ADSERR_CLIENT_SYNCINTERNAL	Internal error in Ads sync.
0x751	1873	0x9811 0751	ADSERR_CLIENT_ADDHASH	Hash table overflow.
0x752	1874	0x9811 0752	ADSERR_CLIENT_REMOVEHASH	Key not found in the table.
0x753	1875	0x9811 0753	ADSERR_CLIENT_NOMORESYM	No symbols in the cache.
0x754	1876	0x9811 0754	ADSERR_CLIENT_SYNCRESINVALID	Invalid response received.
0x755	1877	0x9811 0755	ADSERR_CLIENT_SYNCPORTLOCKED	Sync Port is locked.

RTime error codes

Hex	Dec	HRESULT	Name	Description
0x1000	4096	0x9811 1000	RTERR_INTERNAL	Internal error in the real-time system.
0x1001	4097	0x9811 1001	RTERR_BADTIMERPERIODS	Timer value is not valid.
0x1002	4098	0x9811 1002	RTERR_INVALIDTASKPTR	Task pointer has the invalid value 0 (zero).
0x1003	4099	0x9811 1003	RTERR_INVALIDSTACKPTR	Stack pointer has the invalid value 0 (zero).
0x1004	4100	0x9811 1004	RTERR_PRIOEXISTS	The request task priority is already assigned.
0x1005	4101	0x9811 1005	RTERR_NOMORETCB	No free TCB (Task Control Block) available. The maximum number of TCBs is 64.
0x1006	4102	0x9811 1006	RTERR_NOMORESEMAS	No free semaphores available. The maximum number of semaphores is 64.
0x1007	4103	0x9811 1007	RTERR_NOMOREQUEUES	No free space available in the queue. The maximum number of positions in the queue is 64.
0x100D	4109	0x9811 100D	RTERR_EXTIRQALREADYDEF	An external synchronization interrupt is already applied.
0x100E	4110	0x9811 100E	RTERR_EXTIRQNOTDEF	No external sync interrupt applied.
0x100F	4111	0x9811 100F	RTERR_EXTIRQINSTALLFAILED	Application of the external synchronization interrupt has failed.
0x1010	4112	0x9811 1010	RTERR_IRQLNOTLESSOREQUAL	Call of a service function in the wrong context
0x1017	4119	0x9811 1017	RTERR_VMXNOTSUPPORTED	Intel VT-x extension is not supported.
0x1018	4120	0x9811 1018	RTERR_VMXDISABLED	Intel VT-x extension is not enabled in the BIOS.
0x1019	4121	0x9811 1019	RTERR_VMXCONTROLSMISSING	Missing function in Intel VT-x extension.
0x101A	4122	0x9811 101A	RTERR_VMXENABLEFAILS	Activation of Intel VT-x fails.

TCP Winsock error codes

Hex	Dec	Name	Description
0x274C	10060	WSAETIMEDOUT	A connection timeout has occurred - error while establishing the connection, because the remote terminal did not respond properly after a certain period of time, or the established connection could not be maintained because the connected host did not respond.
0x274D	10061	WSAECONNREFUSED	Connection refused - no connection could be established because the target computer has explicitly rejected it. This error usually results from an attempt to connect to a service that is inactive on the external host, that is, a service for which no server application is running.
0x2751	10065	WSAEHOSTUNREACH	No route to host - a socket operation referred to an unavailable host.
More Winsock error codes: Win32 error codes			

More Information:
www.beckhoff.com/tf6250