BECKHOFF Automation: Foreword

Notes on the documentation

This description is only intended for the use of trained specialists in control and automation engineering who are familiar with the applicable national standards.
It is essential that the following notes and explanations are followed when installing and commissioning these components.

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.
For that reason the documentation is not in every case checked for consistency with performance data, standards or other characteristics.
In the event that it contains technical or editorial errors, we retain the right to make alterations at any time and without warning.
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®, EtherCAT®, Safety over EtherCAT®, TwinSAFE® and XFC® 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 TwinCAT Technology is covered, including but not limited to the following patent applications and patents:
EP0851348, US6167425 with corresponding applications or registrations in various other countries.

Copyright

TwinCAT MODBUS TCP server

Overview

The MODBUS ADS server enables communication with a MODBUS client or a MODBUS server via a MODBUS/TCP connection. The server can therefore be used both as MODBUS client or MODBUS server:

- MODBUS client functionality: PLC blocks can be used to communicate with a MODBUS device via ADS, in order to read or write registers or digital inputs/outputs.
- MODBUS server functionality:The server can receive Modbus functions via TCP/IP. The Modbus register and Modbus inputs/outputs are then mapped to TwinCAT PLC areas.

Product components

- TcModbusSrv.Lib (implements MODBUS client functions);
- TwinCAT Modbus TCP Server (TwinCAT Server);

Installation

Windows NT (NT4, W2K, XP, XPe, Vista, W7)

The PLC libraries are copied into ..\TwinCAT\PLC\Lib folder. The TwinCAT Modbus TCP Server Server is entered in the TwinCAT Server list.
The server is automatically started when TwinCAT is started and stopped when TwinCAT is stopped.

Windows CE

If you have CE version, please do the following steps:

- Install the product on your programming PC. The PLC libraries are copied into ..\TwinCAT\PLC\Lib folder.
- X86 CPU (CX1000, CX1020)
  - The folder **...\TwinCAT\CE\TCModbusTCP\Install** contains a Cabinet-File for the CE runtime system.
  - Copy the file: **TcModbusTcpSvrCe.I586.CAB** in a folder of the CE runtime system.
- ARM CPU (CX9000):
  - The folder **...\TwinCAT\CE\TCModbusTCP\Install** contains a Cabinet-File for the CE runtime system.
  - Copy the file: **TcModbusTcpSvrCe.ARMV4I.CAB** in a folder of the CE runtime system.
- On CE system: Install (double click to Cabinet-File) CE components.
- Please suspend the CE device once after installation via "Start-> Suspend". The TwinCAT Modbus TCP Server starts with CE operating

system.

TwinCAT MODBUS TCP server : Modbus client functionality

Overview

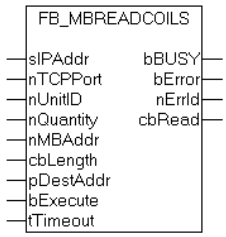
The Modbus ADS server enables communication with a Modbus device (Modbus server) via ADS and a MODBUS/TCP connection. The functions defined in the Modbus protocol are realised as PLC blocks in the library TcModbusSrv.lib. The blocks communicate with the server internally via ADS. The registers or digital inputs/outputs of a Modbus device can be read or written via the Modbus functions.

Modbus function	**Function code**	**PLC block**
Read Coils	1	FB_MBReadCoils
Read Discrete Inputs	2	FB_MBReadInputs
Read Registers	3	FB_MBReadRegs
Read Input Registers	4	FB_MBReadInputRegs
Write Single Coil	5	FB_MBWriteSingleCoil
Write Single Register	6	FB_MBWriteSingleReg
Write Multiple Coils	15	FB_MBWriteCoils
Write Multiple Registers	16	FB_MBWriteRegs
Read/Write Multiple Registers	23	FB_MBReadWriteRegs
Diagnostic	8	FB_MBDiagnose

Note: Only PLC runtime 1 ist accessable though MODBUS.

TwinCAT PLC Library: TcModbusSrv

FUNCTION_BLOCK FB_MBReadCoils (Modbus function 1)



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

VAR_INPUT

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

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

nTCPPort : Port number of the target device.

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

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

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

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

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

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

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

VAR_OUTPUT

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

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

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

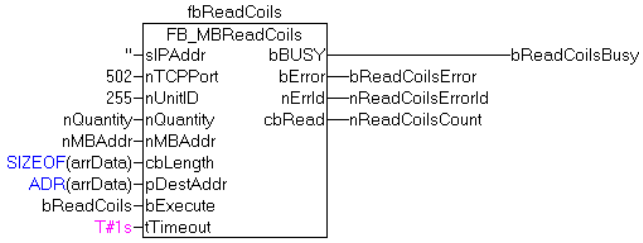
nErrId : Supplies the ADS error number when the bError output is set.

cbRead: Contains the number of bytes currently read.

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

Example of calling the block in FBD:

```
PROGRAM Test
VAR
        fbReadCoils     : FB_MBReadCoils;
        bReadCoils      : BOOL;
        bReadCoilsBusy  : BOOL;
        bReadCoilsError : BOOL;
        nReadCoilsErrorId       : UDINT;
        nReadCoilsCount : UDINT;
        nQuantity       : WORD := 10;
        nMBAddr         : WORD := 5;
        arrData         : ARRAY [1..2] OF BYTE;
END_VAR
```



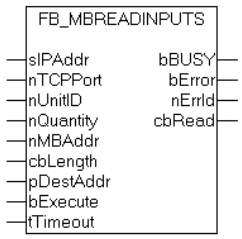
After a rising edge of "bExecute" and successful execution of the ReadCoils command, the content of digital outputs 6 - 15 is written into the arrData array:

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

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

TwinCAT PLC Library: TcModbusSrv

FUNCTION_BLOCK FB_MBReadInputs (Modbus function 2)



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

VAR_INPUT

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

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

nTCPPort: Port number of the target device.

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

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

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

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

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

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

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

VAR_OUTPUT

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

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

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

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

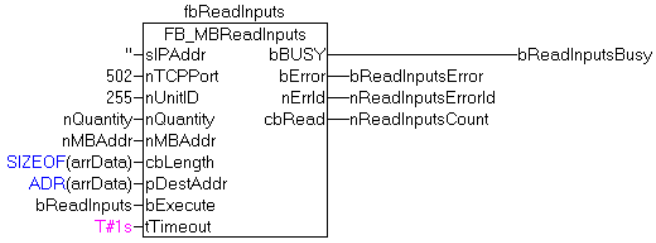
cbRead: Contains the number of bytes currently read.

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

0x8002	Invalid address or length
0x8003	Invalid parameters: - wrong number of registers
0x8004	Modbus server error

Example of calling the block in FBD:

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



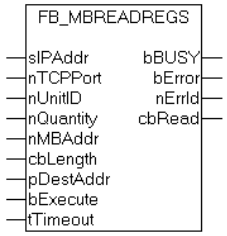
After a rising edge of "bExecute" and successful execution of the ReadInputs command, the content of digital inputs 30 - 49 is written into the arrData array:

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

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

TwinCAT PLC Library: TcModbusSrv

FUNCTION_BLOCK FB_MBReadRegs (Modbus function 3)



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

VAR_INPUT

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

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

nTCPPort: Port number of the target device.

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

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

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

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

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

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

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

VAR_OUTPUT

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

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

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

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

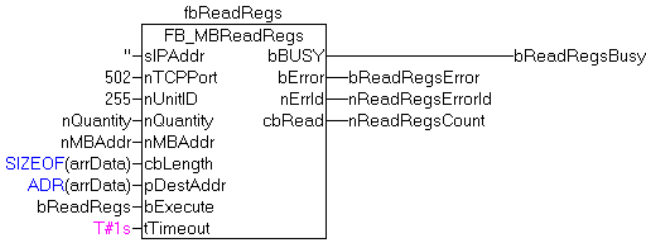
cbRead: Contains the number of bytes currently read.

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

Example of calling the block in FBD:

```
PROGRAM Test
VAR
        fbReadRegs      : FB_MBReadRegs;
        bReadRegs       : BOOL;
        bReadRegsBusy   : BOOL;
        bReadRegsError  : BOOL;
        nReadRegsErrorId        : UDINT;
        nReadRegsCount  : UDINT;
        nQuantity       : WORD:=2;
        nMBAddr         : WORD:=24;
```

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



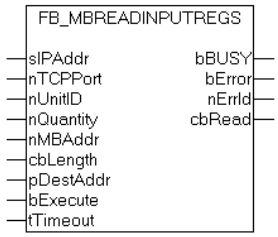
After a rising edge of "bExecute" and successful execution of the ReadRegs command, the content of registers 25 and 26 is located in the arrData array:

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

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

TwinCAT PLC Library: TcModbusSrv

FUNCTION_BLOCK FB_MBReadInputRegs (Modbus function 4)



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

VAR_INPUT

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

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

nTCPPort: Port number of the target device.

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

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

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

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

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

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

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

VAR_OUTPUT

```
VAR_OUTPUT
```

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

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

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

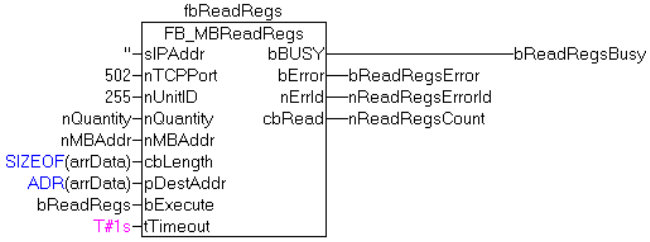
nErrId : Supplies the ADS error number when the bError output is set.

cbRead: Contains the number of bytes currently read.

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

Example of calling the block in FBD:

```
PROGRAM Test
VAR
        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
```

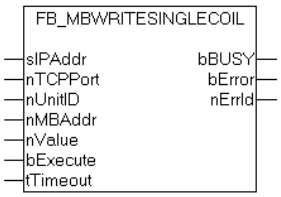


After a rising edge of "bExecute" and successful execution of the ReadRegs command, the content of registers 3-5 is located in the arrData array:

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)

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

FUNCTION_BLOCK FB_MBWriteSingleCoil (Modbus function 5)



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

VAR_INPUT

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

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

nTCPPort: Port number of the target device.

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

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

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

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

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

VAR_OUTPUT

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

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

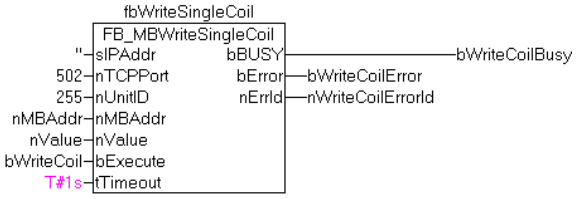
bError : If an ADS error should occur during the transfer of the command, then this output is set once the bBusy output is reset.

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

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

Example of calling the block in FBD:

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

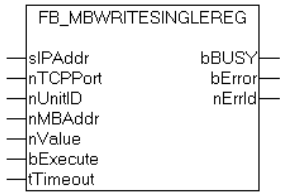


After a rising edge of "bExecute" and successful execution of the WriteSingleCoil command, digital output 4 is switched on.

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

TwinCAT PLC Library: TcModbusSrv

FUNCTION_BLOCK FB_MBWriteSingleReg (Modbus function 6)



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

VAR_INPUT

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

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

nTCPPort: Port number of the target device.

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

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

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

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

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

VAR_OUTPUT

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

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

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

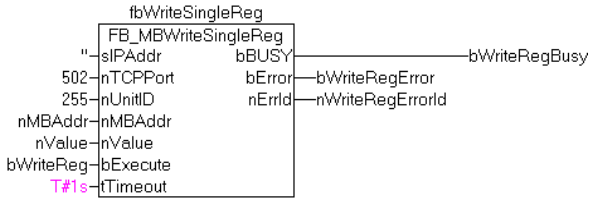
nErrId : Supplies the ADS error number when the bError output is set.

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

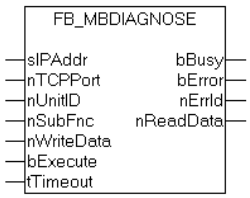


After a rising edge of "bExecute" and successful execution of the WriteSingleReg command, the value 16#1234 is written into register 5.

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

TwinCAT PLC Library: TcModbusSrv

FUNCTION_BLOCK FB_MBDiagnose (Modbus function 8)



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

VAR_INPUT

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

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

nTCPPort : Port number of the target device.

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

nSubFnc : The sub-function to be executed.

nWriteData: The data word to be written.

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

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

VAR_OUTPUT

```
VAR_OUTPUT
        bBusy           : BOOL;
        bError          : BOOL;
```

```
        nErrId          : UDINT;
        nReadData       : WORD;
END_VAR
```

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

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

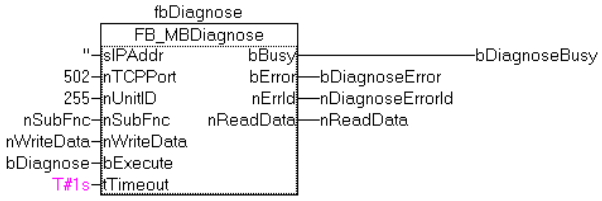
nErrId : Supplies the ADS error number when the bError output is set.

nReadData: Supplies the read data word.

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

Example of calling the block in FBD:

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

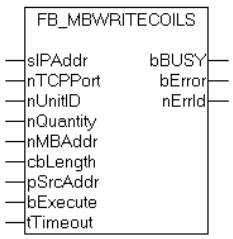


After rising edge of "bExecute" and successful execution of the diagnosis command, nReadData contains the read data word.

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

TwinCAT PLC Library: TcModbusSrv

FUNCTION_BLOCK FB_MBWriteCoils (Modbus function 15)



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

VAR_INPUT

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

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

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

nTCPPort: Port number of the target device.

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

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

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

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

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

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

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

VAR_OUTPUT

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

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

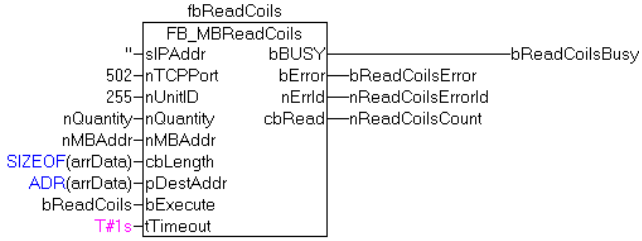
bError : If an ADS error should occur during the transfer of the command, then this output is set once the bBusy output is reset.

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

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

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
```



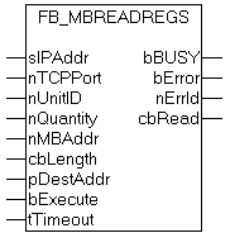
After a rising edge of "bExecute" and successful execution of the ReadCoils command, the content of the arrData array is written to digital outputs 15 - 24:

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

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

TwinCAT PLC Library: TcModbusSrv

FUNCTION_BLOCK FB_MBWriteRegs (Modbus function 16)



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

VAR_INPUT

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

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

nTCPPort: Port number of the target device.

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

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

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

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

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

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

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

VAR_OUTPUT

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

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

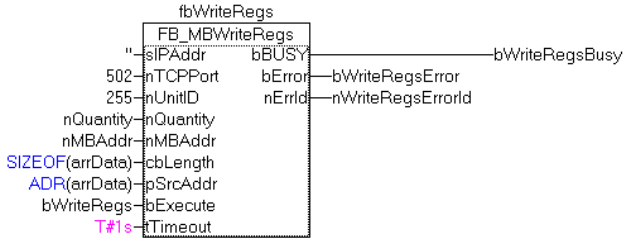
bError: If an ADS error should occur during the transfer of the command, then this output is set once the bBusy output is reset.

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

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

Example of calling the block in FBD:

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

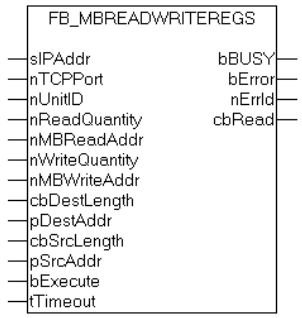


After a rising edge of "bExecute" and successful execution of the ReadRegs command, the content of the arrData array is written to registers 5-7.

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

TwinCAT PLC Library: TcModbusSrv

FUNCTION_BLOCK FB_MBReadWriteRegs (Modbus function 23)



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

VAR_INPUT

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

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

nTCPPort : Port number of the target device.

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

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

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

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

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

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

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

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

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

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

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

VAR_OUTPUT

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

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

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

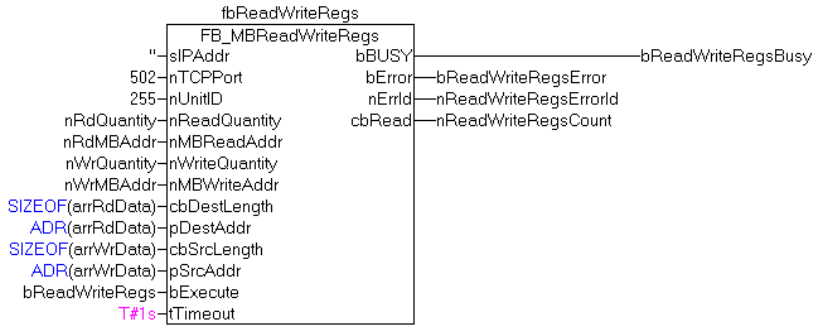
nErrId : Supplies the ADS error number when the bError output is set.

cbRead: Contains the number of bytes currently read.

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

Example of calling the block in FBD:

```
PROGRAM Test
VAR
        fbReadWriteRegs         : FB_MBReadWriteRegs;
        bReadWriteRegs          : BOOL;
        bReadWriteRegsBusy      : BOOL;
        bReadWriteRegsError     : BOOL;
        nReadWriteRegsErrorId   : UDINT;
        nReadWriteRegsCount     : UDINT;
        nRdQuantity             : WORD;
        nRdMBAddr                       : WORD;
        nWrQuantity             : WORD;
        nWrMBAddr               : WORD;
        arrRdData               : ARRAY [1..9] OF WORD;
        arrWrData               : ARRAY [1..9] OF WORD;
END_VAR
```



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

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

TwinCAT MODBUS TCP server : Modbus server functionality

Overview

The server can receive Modbus functions via TCP/IP. The Modbus register and Modbus inputs/outputs are then mapped to PLC areas. The TwinCAT Modbus TCP/IP server configurator is used for configuring this mapping.

TwinCAT MODBUS TCP server : Modbus server functionality

Mapping between Modbus and ADS

In Modbus, the following four addressing areas are defined:

Modbus areas	Data type	Access
Digital inputs (discrete inputs)	1 bit	read only
Digital outputs (coils)	1 bit	read and write
Input register	16 bit	read only
Output register	16 bit	read and write

The individual areas can be addressed with 0 - 0xFFFF. The ADS server maps these addresses to the individual ADS areas. The standard settings are shown in the following table:

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

As shown in the table, the addresses 0x8000-0x80FF are mapped to a PLC variable in all four areas. For this to work, the variables have to be added to the global variables of the PLC program:

```
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
```

Example for writing several registers:

```
PROGRAM Test
VAR
   M1 AT%MB10 : ARRAY [0..3] OF WORD;
   fbWriteRegs : FB_MBWriteRegs;
   bWriteRegs : BOOL;
   arrValue : ARRAY [0..3] OF WORD  := 123,234,567,889;
END_VAR

fbWriteRegs.sIPAddr := '';
fbWriteRegs.nQuantity := 4;
fbWriteRegs.nMBAddr := 16#3005;
fbWriteRegs.cbLength := SIZEOF(arrValue);
fbWriteRegs.pSrcAddr := ADR(arrValue);
```

```
fbReadCoils.tTimeout := T#1s;
fbWriteRegs.bExecute := bWriteRegs;
fbWriteRegs();
```

The FB_MBWriteRegs block is used for sending a Modbus command to the server. The parameter sIPAddr should correspond to the local IP address or an empty string. In this example, address 0x3005 was selected as the start address. The ADS index group 0x4020 can now be read from the above table under output register and the respective address. It is the flag area. The byte offset of the flag area can be calculated as follows:

Byte offset of the flag area = (0x3005 - 0x3000) * 2 = 10

It therefore corresponds to variable M1 AT%MB10. Calling up bWriteRegs therefore writes the arrValue array into the flag area and therefore into variable M1. If the address 0x8000 is entered in fbWriteRegs.MBAddr, the data are written into the global variable mb_OutputRegisters.

Example for writing several digital outputs:

```
PROGRAM TestReadCoilsIO
VAR
QX1 AT%QX0.5 : BOOL;
QX2 AT%QX0.6 : BOOL;
QX3 AT%QX0.7 : BOOL;
QX4 AT%QX1.0 : BOOL;
QX5 AT%QX1.1 : BOOL;
QX6 AT%QX1.2 : BOOL;
QX7 AT%QX1.3 : BOOL;
QX8 AT%QX1.4 : BOOL;
Q1 AT%QB0 : ARRAY [0..1] OF BYTE;

fbReadCoils : FB_MBReadCoils;
bReadCoils : BOOL;
nValue : BYTE;
END_VAR

fbReadCoils.sIPAddr := '';
fbReadCoils.nQuantity := 8;
fbReadCoils.nMBAddr := 16#0005;
fbReadCoils.cbLength := SIZEOF(nValue);
fbReadCoils.pDestAddr := ADR(nValue);
fbReadCoils.tTimeout := T#1s;
fbReadCoils.bExecute := bReadCoils;
fbReadCoils();
```

In this example, the FB_MBReadCoils block is used, and 0x0005 is used as the start address. The ADS index group 0xF031 can now be read from the above table under digital outputs and the respective address. It is the process image of the physical outputs (bit access). The address corresponds to index offset 0x0005. After calling up bWriteCoils, the nValue byte is therefore written to bits 5 -12 of this area, and therefore also to variables QX1-QX8.

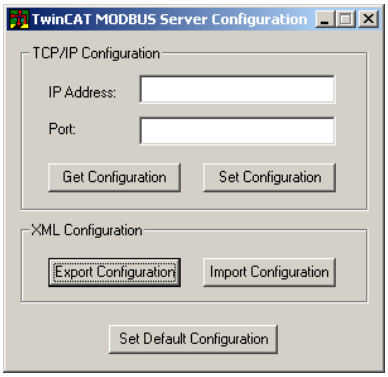
TwinCAT MODBUS TCP server : Modbus server functionality

Configuration

Under CE the configuration cannot be changed.

The TwinCAT Modbus TCP/IP server configurator is used for configuring the Modbus server.
The configurator can be used to set the IP address and the TCP port. It is also possible to change the mapping. To this end, an XML configuration file containing the mapping information can be imported.

It should be noted that it is necessary to stop the TwinCAT system in order to use the configurator. The configuration data is saved in the Default.tps file in the TwinCAT directory. This means that the configuration can be secured with this file, or can be copied to a target computer.



Selection of the IP address and of the TCP ports

The current values can be read from the file Default.tps via the "Get Configuration" button. These are then entered in the respective "IP Address" and "Port" edit fields. The local IP address (empty string) and port 502 are used by default. Changes are saved via "Set Configuration".

Reading the mapping information

The current mapping information from the XML file can be saved via "Export Configuration". The XML file also contains the TCP/IP configuration of the server.

Example for simple mapping:

```
<Configuration>
   <Port>502</Port>
   <IpAddr>172.16.3.217</IpAddr>
   <Mapping>
      <InputRegisters>
         <MappingInfo>
            <StartAddress>0</StartAddress>
            <EndAddress>4095</EndAddress>
            <IndexGroup>61472</IndexGroup>
            <IndexOffset>0</IndexOffset>
         </MappingInfo>
      </InputRegisters>
      <OutputRegisters/>
      <InputCoils/>
      <OutputCoils/>
   </Mapping>
</Configuration>
```

In the example, 502 is used as the port and 172.16.3.217 as the IP address. Only one mapping area is defined. The Modbus input registers 0 to 4095 are mapped to the ADS index group 61472 (= 0xF020 - process image of the physical inputs) with index offset 0.

Reading the mapping information

The mapping information can be imported from an XML file via "Import Configuration". The simplest way is to call "Export Configuration" first, edit the XML file thus created, and then re-import.

The above example could thus be modified as follows:

```
<Configuration>
   <Port>502</Port>
   <IpAddr>172.16.3.217</IpAddr>
   <Mapping>
      <InputRegisters>
         <MappingInfo>
            <StartAddress>0</StartAddress>
            <EndAddress>4095</EndAddress>
            <IndexGroup>61472</IndexGroup>
            <IndexOffset>0</IndexOffset>
         </MappingInfo>
      </InputRegisters>
      <OutputRegisters>
         <MappingInfo>
            <StartAddress>0</StartAddress>
            <EndAddress>4095</EndAddress>
            <IndexGroup>61488</IndexGroup>
            <IndexOffset>0</IndexOffset>
         </MappingInfo>
      </OutputRegisters>
      <InputCoils/>
       <OutputCoils/>
   </Mapping>
</Configuration>
```

The new mapping information is available once this file has been imported via "Import Configuration". In addition, the Modbus output registers 0 to 4095 are mapped to ADS index group 61488 (= 0xF030 - process image of the physical outputs) with index offset 0. A variable name can be used as an alternative to specifying the ADS area via an address. For registers, type WORD or ARRAY OF WORD has to be used, for digital inputs/outputs type BOOL or ARRAY OF BOOL.