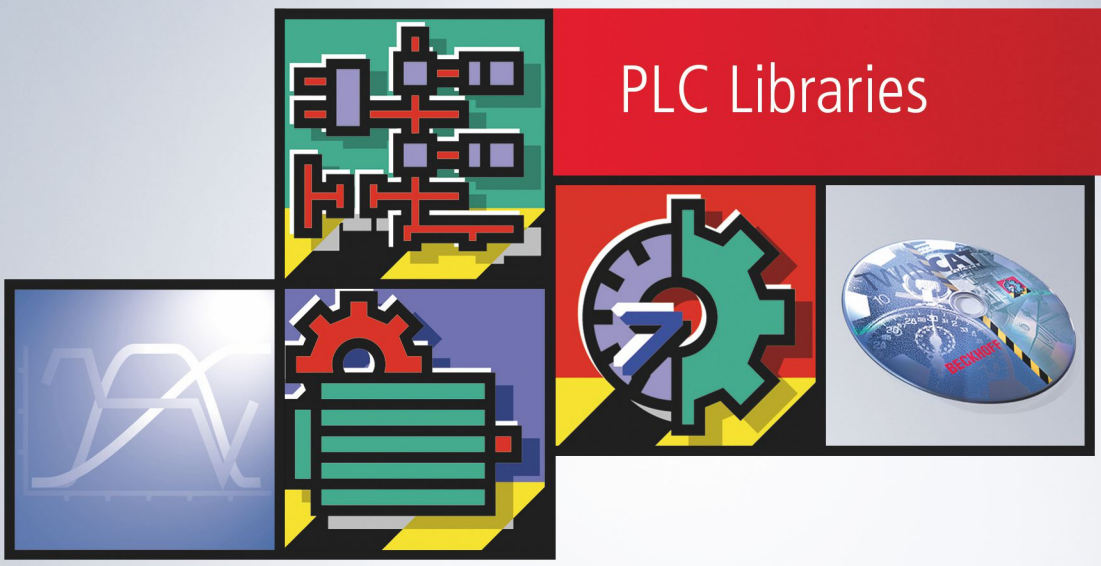
BECKHOFF New Automation Technology

Manual | EN

TX1200

TwinCAT 2 | PLC Library: TcMBus



2022-10-31 | Version: 1.2

Table of contents

1 Foreword

1.1 Notes on the documentation

This description is intended exclusively for trained specialists in control and automation technology who are familiar with the applicable national standards.
For installation and commissioning of the components, it is absolutely necessary to observe the documentation and the following notes and explanations.
The qualified personnel is obliged to always use the currently valid documentation.

The responsible staff must ensure that the application or use of the products described satisfies all 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 notice.
No claims to modify 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 and licensed trademarks of Beckhoff Automation GmbH.
If third parties make use of designations or trademarks used in this publication for their own purposes, this could infringe upon the rights of the owners of the said designations.

Patents

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

EtherCAT®

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

Copyright

1.2 For your safety

Safety regulations

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

Exclusion of liability

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

Personnel qualification

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

Signal words

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

Personal injury warnings

⚠ DANGER

Hazard with high risk of death or serious injury.

⚠ WARNING

Hazard with medium risk of death or serious injury.

⚠ CAUTION

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

Warning of damage to property or environment

NOTICE

The environment, equipment, or data may be damaged.

Information on handling the product

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

1.3 Notes on information security

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

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

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

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

2 Introduction

The MBus library is a comprehensive TwinCAT PLC library for reading M-Bus devices.

The application of this PLC library significantly simplifies the engineering in these areas of building technical equipment.

The function blocks are object-oriented and characterised by a self-contained, more or less complex function.

The input parameters form the interface to the user. The parameters can be used to adapt the function block to its specific task within the associated system.

Thanks to strongly object-oriented encapsulation of complex system functions within the function blocks, comprehensive system programs can be set up with a few function blocks. The blocks are linked to each other via a small number of PLC variables.

The status of all objects is indicated through a large number of different output variables at the function blocks. The simplifies the connection of HMI and visualisation systems.

These features offer the following benefits for system programmers during system setup and for system operators during operation:

- Faster creation of system programs.
- Faster system parameterisation and commissioning.
- Guarantee of a very large range of system functions at all times.
- Improved readability of programs (prerequisite for long-term maintainability and expandability of the systems)
- Improved reusability of templates for systems or system components
- Easier familiarisation of personnel.
- Easier extension of existing systems.
- Programs are easier to document.

3 Target groups

The user of this library requires basic knowledge of the following:

- TwinCAT PLC-Control
- TwinCAT System Manager
- PCs and networks
- Structure and properties of the Beckhoff Embedded PC and its Bus Terminal system
- Technology of M-Bus devices
- Relevant safety regulations for building technical equipment

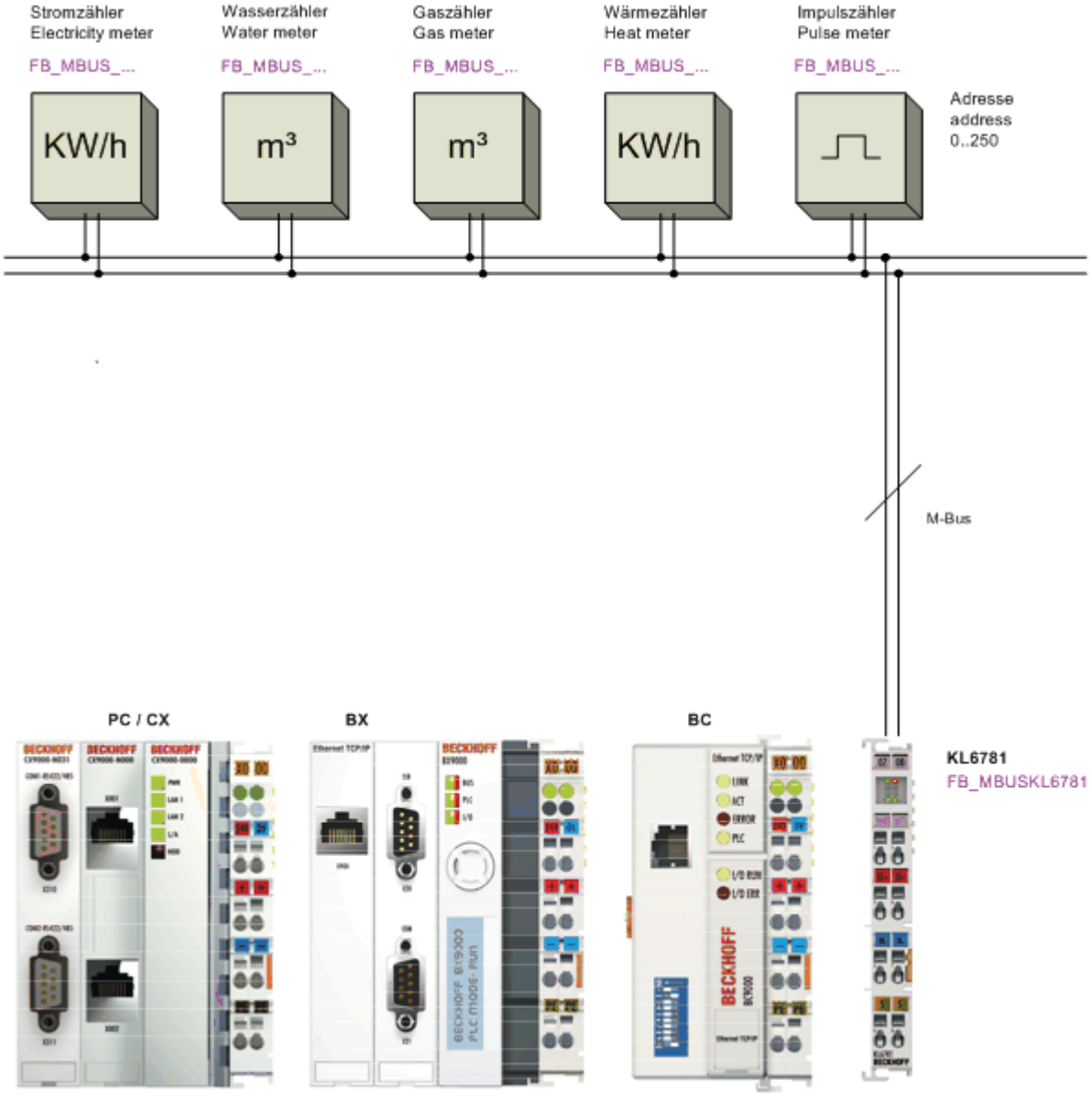
This software library is intended for building automation system partners of Beckhoff Automation GmbH & Co. KG. The system partners operate in the field of building automation and are concerned with the installation, commissioning, expansion, maintenance and service of measurement, control and regulating systems for the technical equipment of buildings.

4 M-Bus

M-Bus = Meter-Bus

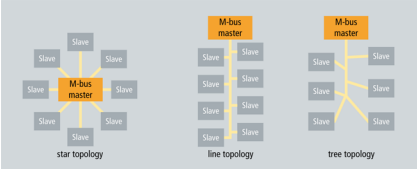
The M-Bus is a fieldbus for the recording consumption data (e.g. energy meters). Further details about M-Bus can be found under www.m-bus.com. The M-Bus is European standard and is described in the EN1434 standard. The data are sent serially from a slave (measuring device) to a master (M-Bus master terminal KL6781). Master and slave are connected via a two-wire cable that is protected against polarity reversal. Up to 40 slaves can be connected to a KL6781 with any topology (star or line). Devices from different manufacturers can be operated on the same bus.

The M-Bus master (KL6781) controls the communication on the bus by requesting data from the slaves. The slaves can respond with a fixed or variable data structure. The M-Bus library only evaluates data with variable data structure (low byte first). The slaves do not communicate with each other. The data have to be requested sequentially from the slaves.

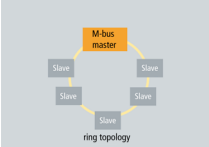


4.1 Topology

Star, line and tree topology



Ring topology



Ring topology not supported

Ring topology is possible for M-Bus, but not recommended and therefore also not supported by Beckhoff.

4.2 Bulletin

4.2.1 Functionality of the function block

Three ways to read out M-Bus counters are offered:

1. Via the variable **tMinSendTime** > t#0s the counter block is read out automatically after the time has elapsed. Internally this variable is set to t#2s.
2. At rising edge of the counter block's variable **bStart**, the counter is read out once.
3. At rising edge of the block FB_MBUS_KL6781() [▶ 29]'s variable *bStart*, all counters are read out once.

If several counter blocks get a start command simultaneously, they are started in order of their calling within the PLC.

The *bReady* variable will be True for one cycle, if the block has received data.

If an error has occurred, **bError** will be True. This Error is described with eError [▶ 216].

If the counter should be read out after a start/restart of the PLC, the variable *bReadInit* is set to True; otherwise to False. This variable is internally set to True per default.

eBaudrate: This variable is internally set to 2400 per default. If the counter should be read out with this baud rate (2400 baud), the variable has not to be set explicitly. At baud rate change the KL6781 is reparameterized automatically. This allows to read out counter with different baud rate in one M-Bus network. The baud rate of the counters are not changed. They have to be able to operate with the given baud rate. Some counters operate with automatic baud rate detection. Please take information about this from the manual of the counter

bSND_NKE: This variable is internally set to True per default. SND_NKE is a special telegram to the slave. This telegram causes an initializing of the receiver. This telegram is important for counters sending several telegrams. After a SND_NKE these counters respond with the first telegram. If True the telegram is sent before the real request, if False the SND_NKE telegram is not sent.

With *bDisabled* = True the execution of the block can be interrupted. A started query of the counter will be finished.

4.2.2 Long set

Data is sent to the M-bus device with a long set. The long set is composed of a maximum of 255 bytes and is transferred to the counter with the FB_MBUS_General_Send [▶ 43] block.

Structure of the protocol:

Byte	Long set	Description	Assignment in the 'FB_MBUS_General_Send' block
1	Start character	68hex	Is added in the block
2	L field	Length of user data plus 3	Is added in the block
3	L field	Length of user data plus 3	Is added in the block
4	Start character	68hex	Is added in the block
5	**C field**	Function field	Is transferred to the **'byC_Field'** input variable
6	**A field**	Primary address of the M-Bus device	Is transferred to the **'usiAddress'** input variable
7	**CI Field**	Identifier field	Is transferred to the **'byCI_Field'** input variable
8..x	**User data (0 – 240)**	User data	Are transferred to the **'arrData'** input variable
x+1	Checksum	Checksum	Is added in the block
x+2	Stop character	16hex	Is added in the block

Only the bytes marked in bold letters need to be transferred to the block.

The user data in the *arrData* array must contain '16hex' as the last character. It is important to ensure that the subsequent bytes are empty.

Example: Changing the primary address at address 14, old address is 0.

```
(*Transfer of user data*)
fbSend.arrData[0] := 16#01;     (*DIF / Data format 8 -bit integer*)
fbSend.arrData[1] := 16#7A;     (*VIF / Change address*)
fbSend.arrData[2] := 14;        (*New address = 14*)
fbSend.arrData[3] := 16#16;     (*Do not transfer stop character/
checksum; they will be calculated in the block*)

fbSend.byC_Field  := 16#53;     (*C field*)
fbSend.byCI_Field := 16#51;     (*CI field*)
fbSend.usiAddress := 0;         (*Old address*)

fbSend(iComId := 1,             (*Block call*)
       bStart := bStart,
       bInit  := TRUE);
```

Sending is started with the *bStart* variable. The *bStart* variable is reset in the block after sending is complete.

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055566347/.zip:

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055569163/.zip:

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055569163/.zip:

Documents about this

- demogeneralbc.zip (Resources/zip/12055567755.zip)

4.2.3 Primary address

The counters are addressed via the primary address. This can be set on the device, via the manufacturer's software or with the FB_MBUS_ChangeAdr [▶ 30] or FB_MBUS_General_Send [▶ 43] block.

All M-Bus devices must have a unique address (0-255).

Address 0 - 250 Addresses of the devices

Address 251 Not used at present

Address 252 Not used at present

Address 253 Use of secondary addressing [▶ 15]

Address 254 Send to all M-Bus devices with response (E5 hexadecimal). If several devices are connected, all of them will respond, leading to data collision. Therefore, this address should only be used if only one device is connected.

Address 255 Send to all M-Bus devices without response.

4.2.4 Secondary address

The secondary address like also the primary address is used to identify the device. The advantage of the secondary addressing is among other things a bigger number of addresses (slaves). Only by the identification number 100 million different values can be build. Furthermore the assignment of primary addresses is not applicable.

According to M-Bus standard a secondary address has the following structure:
Ident-No.: 4 Byte / 8-digit BCD device identification number
Herstellerkürzel : 2 Byte / vendor short symbol
Version: 1 Byte / Generation number of the vendor
Medium: 1 Byte / Medium

If the secondary address should be used, the primary address is set to 253.

The secondary address is given to the function block via the structure "stSecAdr" (ST_MBUS_SecAdr [▶ 223]).

Vendor short symbol, version and Medium are each internally set to 16#FF by default. So these values have not to be set explicitly.

Request example:

```
stSecAdr1.udiIdNumber    := 16#12345678;
stSecAdr1.uiManufacturer := 16#FFFF;
stSecAdr1.usiMedium      := 16#FF;
stSecAdr1.usiVersion     := 16#FF;

fbmeter(usiAddress := 253,
    stSecAdr.udiIdNumber := stSecAdr1,
    stCom := stComKL6781_1);
```

or also

```
fbmeter.stSecAdr.udiIdNumber := 16#12345678;
fbmeter(usiAddress := 253,
    stCom := stComKL6781_1);
```

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055570571/.zip:

5 Integration into TwinCAT

5.1 KL6781 - Linking to the System Manager

1. Link the PLC program and click with the right mouse button on the data structure.

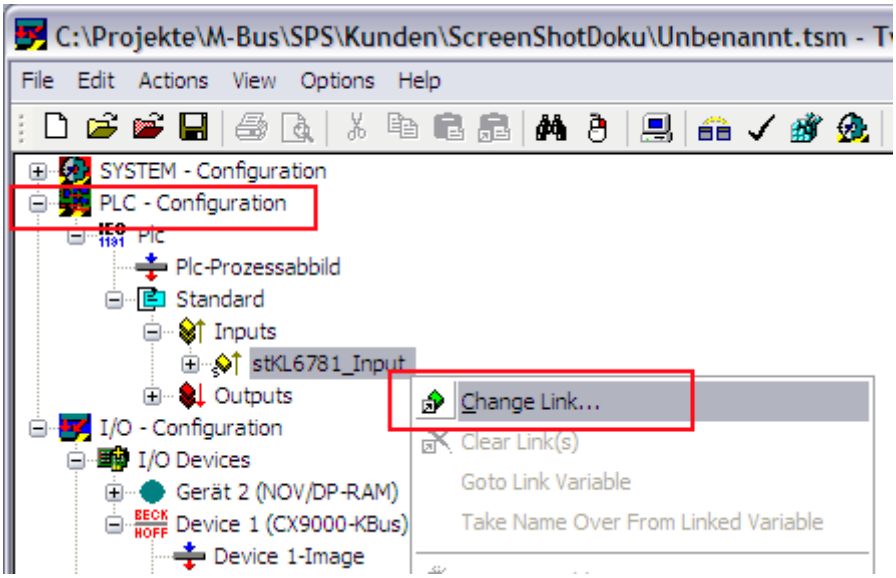


Figure 1

2. Select "All Types" and "Continuous" (see Figure 2).

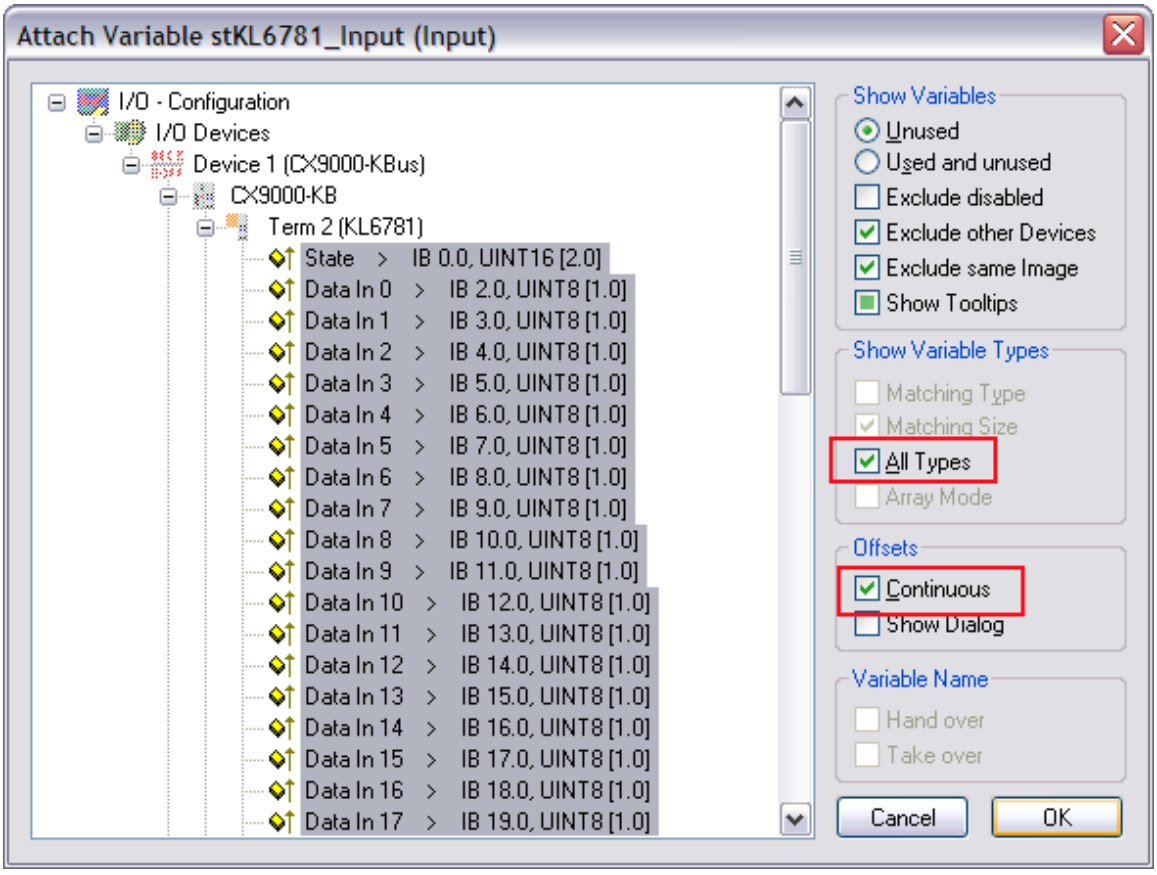


Figure 2

3. Click with the mouse on the first variable of the M-Bus master terminal KL6781 ‘Status’. Then press the <SHIFT> key, and hold it down.
4. Move the mouse pointer over the last variable of the KL6781 ‘Data In 21’ and click again with the left mouse button.
5. Now release the <SHIFT> key again. All the terminal's data should now be highlighted (see Figure 2).
6. Press button OK.
7. Check the links. To do this go onto the KL6781 and open it. All the terminal's data should now be marked by a small arrow (see Figure 3). If that is the case, then proceed in exactly the same way with the outputs.

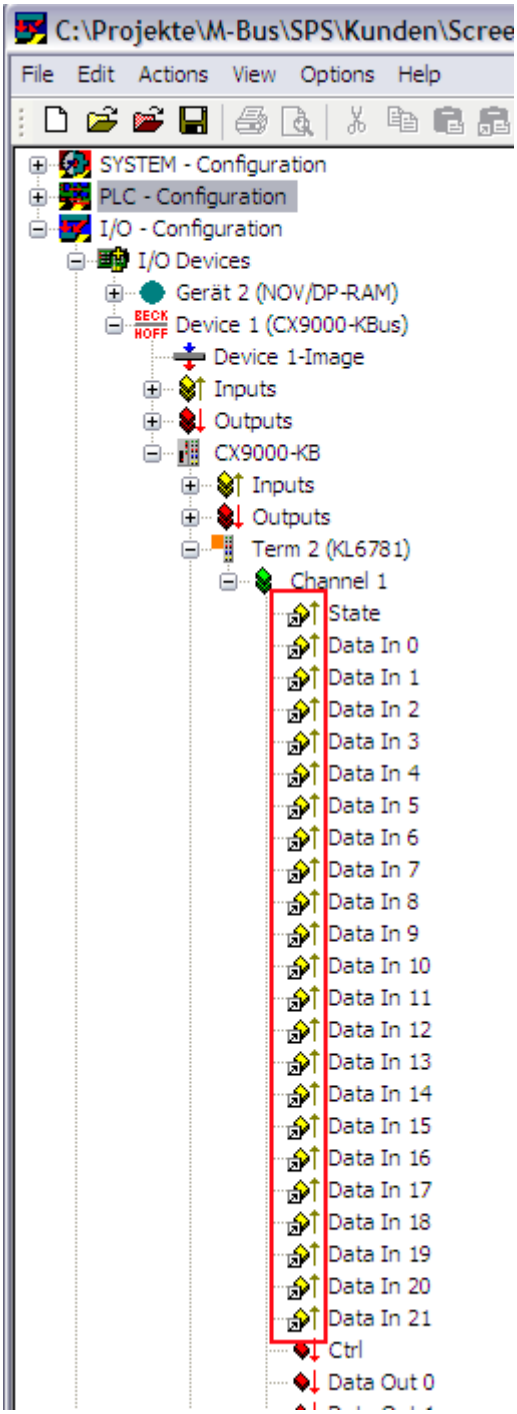


Figure 3

5.2 Integration in TwinCAT (CX9020)

This example describes how a simple PLC program for M-Bus can be written in TwinCAT and how it is linked with the hardware. The task is to read a counter with four digital inputs.

Unpacking the example files https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/

12055571979/.zip

Hardware

Setting up the components

The following hardware is required:

- 1x Embedded PC CX9020
- 1x M-Bus master terminal KL6781
- 1x end terminal KL9010

Set up the hardware and the M-Bus components as described in the associated documentation.

This example assumes that the counter address is known.

Software

Creation of the PLC program

Create a new PLC project for PC-based systems (ARM) and add the *TcMBus.lib* library.

Next, generate the following global variables:

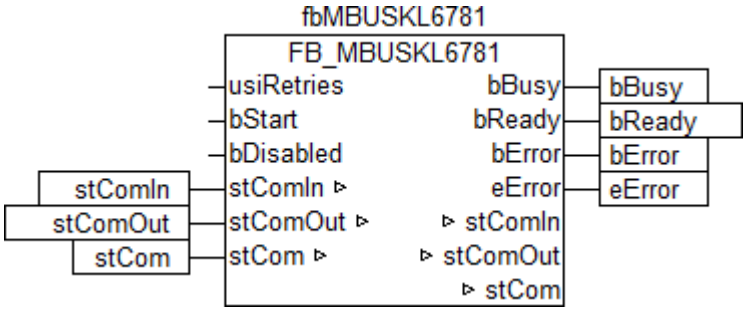
```
VAR_GLOBAL
    stComIn     AT%I* : ST_KL6781inData22B;
    stComOut    AT%Q* : ST_KL6781outData22B;
    stCom             : ST_MBUS_Communication;
END_VAR
```

stComIn: Input variable [▶ 220] for the M-Bus terminal.

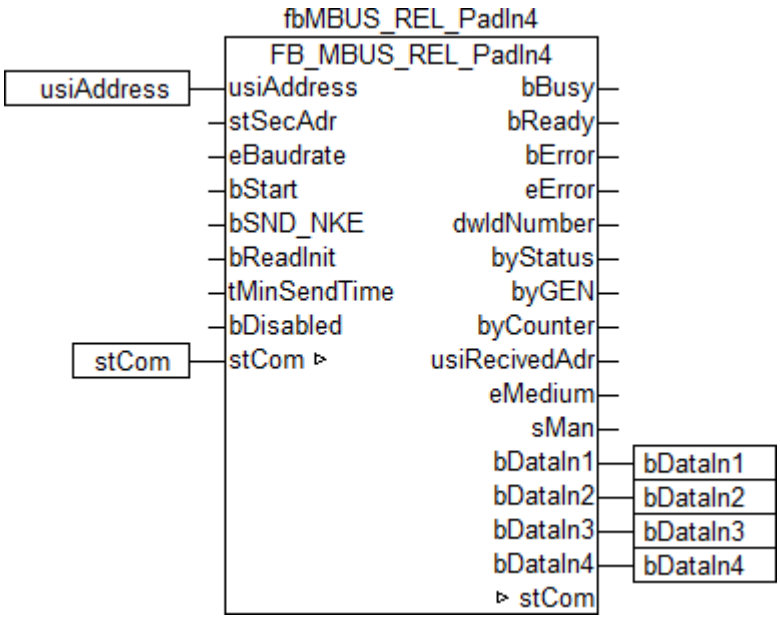
stComOut: Output variable [▶ 221] for the M-Bus terminal.

stCom: Required for communication with M-Bus.

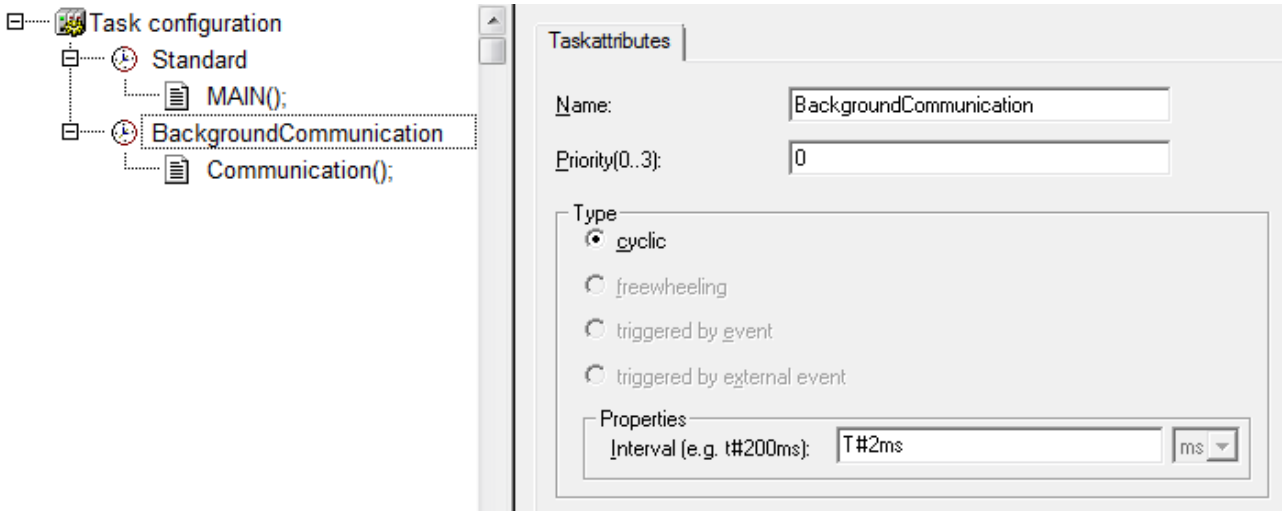
Then create a program (CFC) for background communication with M-Bus. The FB_MBUSKL6781() [▶ 29] block is called in this program. Make sure to link the communication block with *stComIn*, *stComOut* and *stCom*.



Create a MAIN program (CFC) in which the block FB_MBUS_REL_PadIn4() [▶ 175] is called up. Link the input *usiAddress* of the counter block with the local variable *usiAddress* and *stCom* with the global variable *stCom*.



Go to the task configuration and give the task a lower interval time. More detailed information can be found in the FB_MBUSKL6781() [▶ 29] block description.

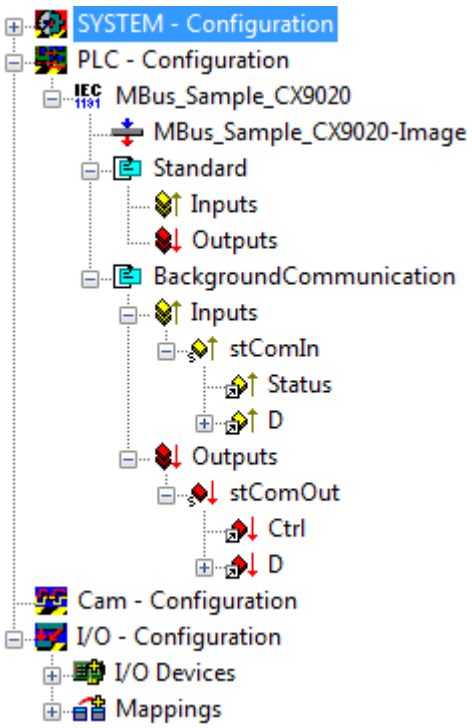


Load the project to the CX as the boot project and save it.

Configuration in the System Manager

Create a new TwinCAT System Manager project, select the CX as the target system, and search for the associated hardware.

Add the PLC program created above under PLC configuration. The two tasks are listed when the PLC project is expanded in the tree view. However, since the variables *stComIn* and *stComOut* are to be processed faster, move them to the background communication task via drag & drop.



Now link the global variables of the PLC program with the Bus Terminal inputs and outputs, create the allocations, and activate the configuration. Then start the device in run mode.

Your CX is now ready for use.

After starting the PLC, the current values are regularly read by the counter.

5.3 Integration into TwinCAT (BC9191)

This example describes how a simple PLC program for M-Bus can be written in TwinCAT and how it is linked with the hardware. The task is to read a counter with four digital inputs.

Unpacking the example files https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/

12055573387/.zip

Hardware

Setting up the components

The following hardware is required:

- 1x Bus Terminal Controller BC9191
- 1x potential feed terminal 24V DC
- 1x M-Bus master terminal KL6781
- 1x end terminal KL9010

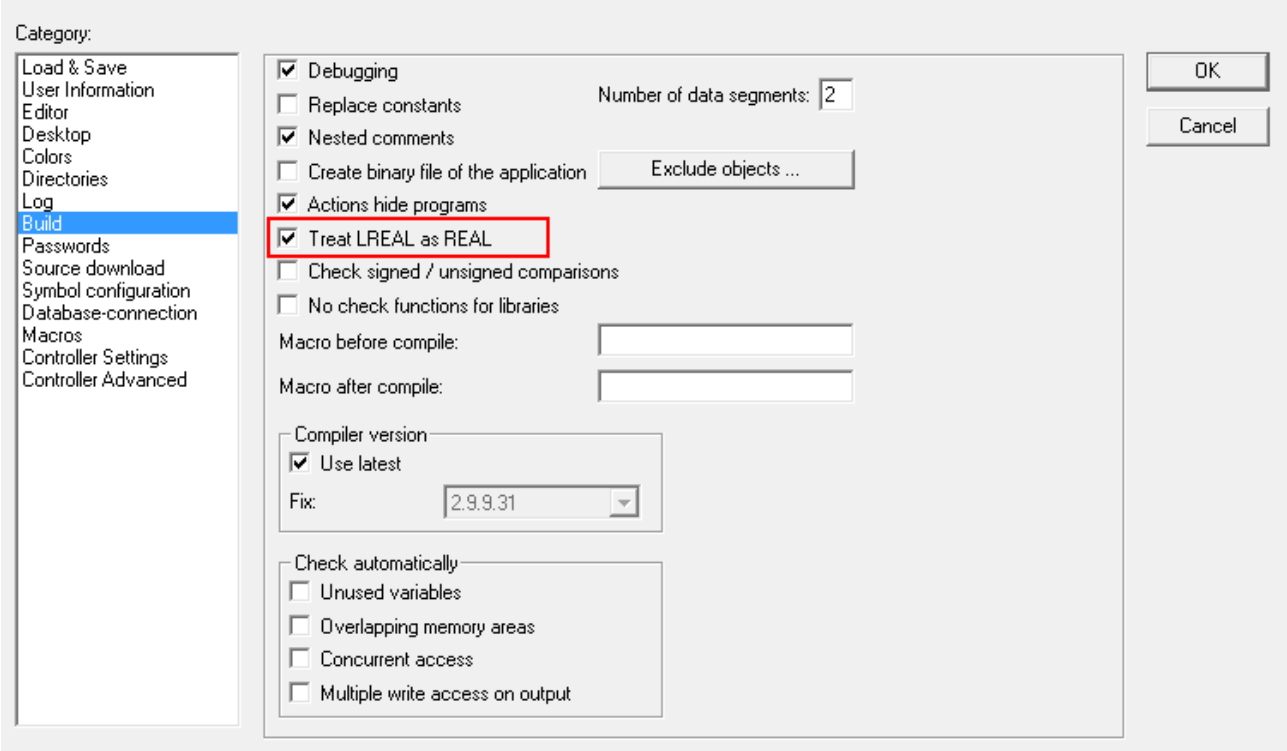
Set up the hardware and the M-Bus components as described in the associated documentation.

This example assumes that the counter address is known.

Software

Creation of the PLC program

Create a new PLC project for BC-based systems (BCxx50 via AMS) and add the library *TcMBus.lbx*. Then navigate to *Project→Options... →Build* and select *TreatLREAL as REAL*.



Next, generate the following global variables:

```
VAR_GLOBAL
    stComIn     AT%I* : ST_KL6781inData22B;
    stComOut    AT%Q* : ST_KL6781outData22B;
    stCom             : ST_MBUS_Communication;
END_VAR
```

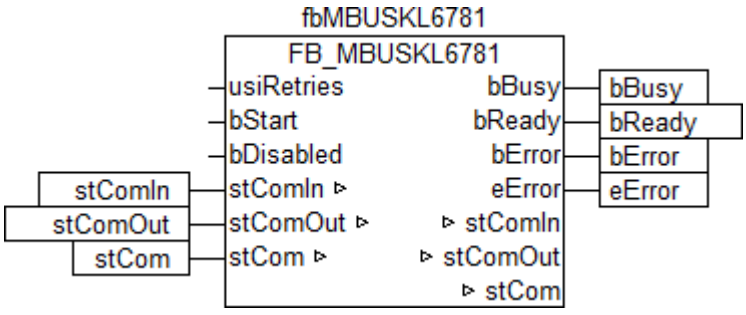
stComIn: Input variable [▶ 220] for the M-Bus terminal.

stComOut: Output variable [▶ 221] for the M-Bus terminal.

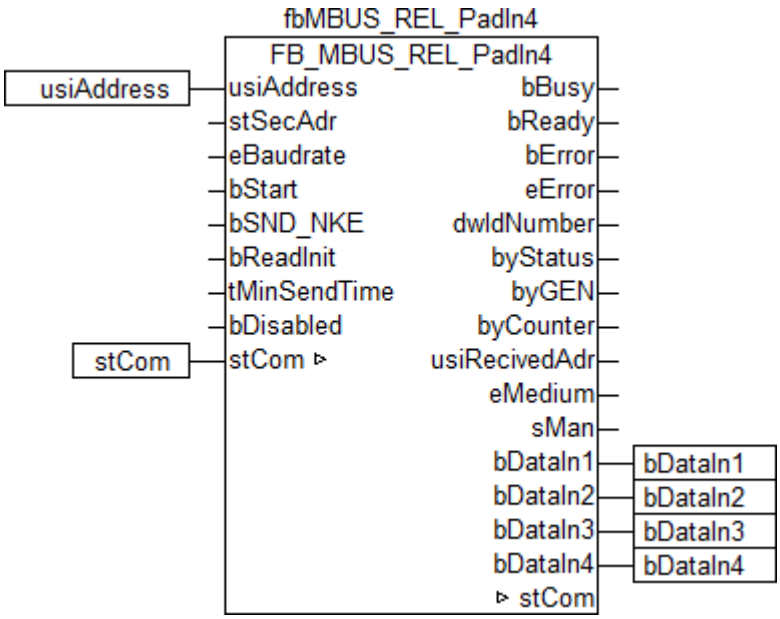
stCom: Required for communication with M-Bus.

Since BC devices can only process one task, communication with M-Bus cannot run separately.

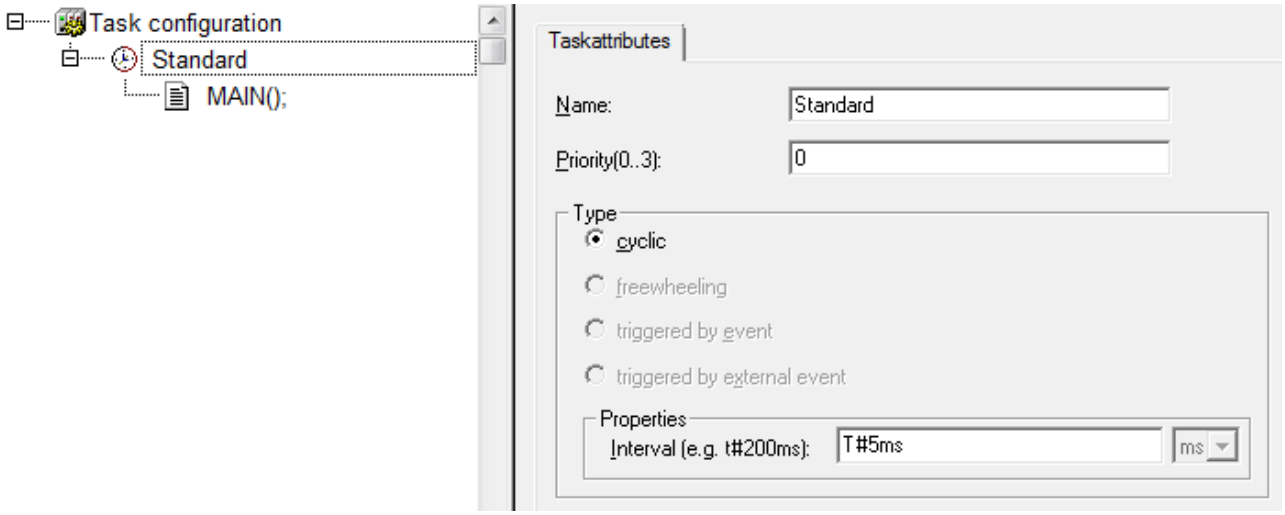
Therefore, create a MAIN program (CFC) in which the FB_MBUSKL6781() [▶ 29] and FB_MBUS_REL_PadIn4() [▶ 175] function blocks are called. Make sure to link the communication block with *stComIn*, *stComOut* and *stCom*.



Link the input *usiAddress* of the counter block with the local variable *usiAddress* and *stCom* with the global variable *stCom*.



Go to the task configuration and give the task a lower interval time. More detailed information can be found in the FB_MBUSKL6781() [▶ 29] block description.

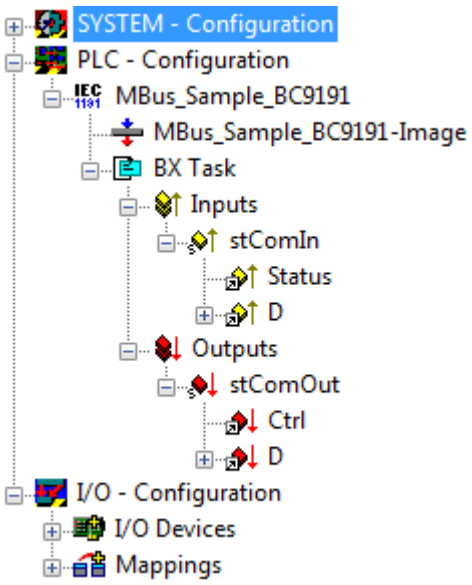


Now load the project as a boot project to the BC and save it.

Configuration in the System Manager

Create a new TwinCAT System Manager project, select the BC as the target system, and search for the associated hardware.

Add the PLC program created above under PLC configuration.



Now link the global variables of the PLC program with the Bus Terminal inputs and outputs, create the allocations, and activate the configuration. Then start the device in run mode.

Your BC is now ready for use.

After starting the PLC, the current values are regularly read by the counter.

6 Programming

The vendor-specific function blocks only output a selection of the most common data. These data are described on the respective pages under "VAR_OUT". If more or all data are required, the function blocks FB_MBUS_General [▶ 31], FB_MBUS_General_Ext [▶ 35] or FB_MBUS_General_Param [▶ 41] from the General [▶ 30] folder should be used. Not that these function blocks do not run on BC or BX systems. The function block FB_MBUS_General_Send [▶ 43] can be used to send data to the device (e.g. setting of the primary address).

Vendor	Type	Device	Function block
General	Communication with KL6781		FB_MBUSKL6781 [▶ 29]
General [▶ 30]	Electricity meter	all electricity meters	FB_MBUS_General_Electricity [▶ 34]
	Heat meter	all heat meters	FB_MBUS_General_Heat [▶ 39]
	Water meter	all water meters	FB_MBUS_General_Water [▶ 44]
	Raw data of the first telegram	all	FB_MBUS_RawData [▶ 46]
	max. 40 values from the first telegram	all	FB_MBUS_General [▶ 31]
	all telegrams for all values	all	FB_MBUS_General_Ext [▶ 35]
	Values parameterizable	all	FB_MBUS_General_Param [▶ 41]
	Universal send blocks	all	FB_MBUS_General_Send [▶ 43]
	Scan function block	all	FB_MBUS_Scan [▶ 48]
	Change primary address	all	FB_MBUS_ChangeAdr [▶ 30]
ABB [▶ 49]	Electricity meter	DELTAplus DZ+	FB_MBUS_ABB_DZ [▶ 50]
Actaris [▶ 52]	Heat meter	CF-Echo II	FB_MBUS_ACW_CF [▶ 52]
	Arithmetic unit	CF-51	FB_MBUS_ACW_CF [▶ 52]
	Arithmetic unit	CF-55	FB_MBUS_ACW_CF [▶ 52]
	Water meter	MB +M	FB_MBUS_ACW_PlusM [▶ 55]
Aquametro [▶ 56]	Water meter	SAPHIR	FB_MBUS_AMT_SAPHIR [▶ 66]
	Heat meter	CALEC MB	FB_MBUS_AMT_CALEC [▶ 61]
	Heat meter	CALEC ST, version C4	FB_MBUS_AMT_CALEC_STC4 [▶ 64]
	Heat meter	AMTRON	FB_MBUS_AMT_AMTRON [▶ 59]
	Pulse collector	AMBUS	FB_MBUS_AMT_AMBUS [▶ 57]

Vendor	Type	Device	Function block
	Heat meter	AMTRON SONIC D	FB_MBUS_HYD_Sharky [▶ 117], FB_MBUS_HYD_Sharky_00 [▶ 120]
Berg [▶ 68]	Electricity meter	DZ+	FB_MBUS_BEC_DZ [▶ 70]
	Electricity meter	DCMi	FB_MBUS_BEC_DCMi [▶ 68]
Brunata [▶ 72]	Heat meter	HGQ / HGS	FB_MBUS_BHG_HGx [▶ 73]
	Heat meter	Optuna H (775)	FB_MBUS_HYD_Sharky [▶ 117], FB_MBUS_HYD_Sharky_00 [▶ 120]
Carlo Gavazzi [▶ 75]	Energy meter	EM24	FB_MBUS_GAV_EM24 [▶ 75]
Cynox [▶ 77]	Pulse counter	MCount2C	FB_MBUS_CYN_MCount2C [▶ 78]
Elster [▶ 80]	Gas meter	Encoder Z6	FB_MBUS_ELS_EncoderZ6 [▶ 80]
elvaco [▶ 82]	Temperature and humidity sensors	CMa10 & CMa20	FB_MBUS_ELV_CMa10_20 [▶ 82]
EMH [▶ 84]	Electricity meter	DIZ	FB_MBUS_EMH_DIZ [▶ 85]
	Electricity meter	EIZ-E	FB_MBUS_EMH_EIZE [▶ 87]
	Electricity meter	EIZ-G	FB_MBUS_EMH_EIZG [▶ 89]
	Electricity meter	MIZ	FB_MBUS_EMH_MIZ [▶ 91]
EMU [▶ 93]	Electricity meter	EMU32x7	FB_MBUS_EMU_32x7 [▶ 93]
	Electricity meter	EMU32x7	FB_MBUS_EMU_32x7_Option 8 [▶ 96]
	Electricity meter	Allrounder 3/5	FB_MBUS_EMU_3_5_Allrounder [▶ 99]
	Electricity meter	DHZ 5/63	FB_MBUS_EMU_DHZ_5_63 [▶ 102]
Engelmann [▶ 103]	Heat meter	Sensostar 2C	FB_MBUS_EFF_SensoStar2C [▶ 104]
Gossen Metrawatt [▶ 106]	Electricity meter	U128x	FB_MBUS_GMC_Electricity [▶ 107]
	Electricity meter	U138x	FB_MBUS_GMC_Electricity [▶ 107]

Vendor	Type	Device	Function block
GWF [▶ 109]	Water meter		FB_MBUS_GWF_Coder [▶ 109]
	Gas meter	S1	FB_MBUS_GWF_Coder [▶ 109]
	Gas meter	Z1	FB_MBUS_GWF_Coder [▶ 109]
Hydrometer [▶ 111]	2 pulse inputs	HYDRO-PORT Pulse	FB_MBUS_HYD_PortPulse [▶ 115]
	2 analog inputs 1 temperature sensor	HYDRO-PORT Analog	FB_MBUS_HYD_PortAnalog [▶ 113]
	Water meter	Flypper	FB_MBUS_HYD_Flypper [▶ 111]
	Heat meter	Sharky 773	FB_MBUS_HYD_Sharky [▶ 117], FB_MBUS_HYD_Sharky_00 [▶ 120]
	Heat meter	Sharky 775	FB_MBUS_HYD_Sharky [▶ 117], FB_MBUS_HYD_Sharky_00 [▶ 120]
ista [▶ 123]	Water meter	domaqua® m	FB_MBUS_IST_Istameter [▶ 123]
	Water meter	istameter® m	FB_MBUS_IST_Istameter [▶ 123]
	Water meter	istameter III	FB_MBUS_IST_IstameterIII [▶ 125]
	Pulse counter	pulsonic II	FB_MBUS_IST_PulsonicII [▶ 127]
	Heat meter	sensonic II	FB_MBUS_IST_SensonicII [▶ 129]
Itron [▶ 131]	Energy meter	Integral-V UltraLite	FB_MBUS_ITR_IntegralVUltraLite [▶ 132]
Janitza [▶ 134]	Electricity meter	UMG96S	FB_MBUS_JAN_UMG96S [▶ 134]
Kamstrup [▶ 137]	Electricity meter	Kamstrup 162	FB_MBUS_KAM_KamstrupE [▶ 138]
	Electricity meter	Kamstrup 351	FB_MBUS_KAM_KamstrupE [▶ 138]
	Electricity meter	Kamstrup 382	FB_MBUS_KAM_KamstrupE [▶ 138]
	Heat/cold meter	Maxical III	FB_MBUS_KAM_Maxical_III [▶ 140]
	Heat/cold meter	Multical 401	FB_MBUS_KAM_Multical [▶ 142]
	Heat/cold meter	Multical 402	FB_MBUS_KAM_Multical402 [▶ 144]

Vendor	Type	Device	Function block
	Water meter	Multical 41	FB_MBUS_KAM_Multical41 [▶ 147]
	Heat/cold meter	Multical 601	FB_MBUS_KAM_Multical601 [▶ 149]
KUNDO [▶ 151]	Heat/cold meter	Compact WMZ G20	FB_MBUS_KST_G20 [▶ 152]
	Heat/cold meter	Compact WMZ G21	FB_MBUS_KST_G20 [▶ 152]
	External M-Bus module	him1s	FB_MBUS_KST_him1 [▶ 154]
	External M-Bus module	him1plus	FB_MBUS_KST_him1 [▶ 154]
	Pulse input	him1plus	FB_MBUS_KST_him1Puls [▶ 156]
Landis & Gyr [▶ 158]	Heat/cold meter	ULTRAHEAT 2WR5	FB_MBUS_LUG_Heat [▶ 158]
	Heat/cold meter	ULTRAHEAT 2WR6	FB_MBUS_LUG_Heat [▶ 158]
	Heat/cold meter	ULTRAHEAT UH50	FB_MBUS_LUG_Heat [▶ 158]
Metrima [▶ 160]	Heat meter	F22 (default values)	FB_MBUS_SVM_F22 [▶ 161]
		F22 (with additional output values)	FB_MBUS_SVM_F22_Ext [▶ 163]
NZR [▶ 165]	Electricity meter	EcoCount "S"	FB_MBUS_TIP_SINUS85M [▶ 207]
	Pulse memory module	IC-M2	FB_MBUS_NZR_ICM2 [▶ 166]
	Pulse memory module	IC-M2C	FB_MBUS_NZR_ICM2 [▶ 166]
	Water meter	Modularis 2	FB_MBUS_NZR_Modularis2 [▶ 168]
OPTEC [▶ 169]	Electricity meter	ECS Type 2	FB_MBUS_OPT_ECSType2 [▶ 170]
Relay [▶ 172]	1-4 analog inputs	AnDi 1-4	FB_MBUS_REL_AnDi [▶ 173]
	4 digital inputs	PadIn 4	FB_MBUS_REL_PadIn4 [▶ 175]
	1-way pulse adapter	PadPuls M1	FB_MBUS_REL_PadPulsM1 [▶ 177]
	1-way pulse adapter	PadPuls M1C	FB_MBUS_REL_PadPulsM1 [▶ 177]
	2-way pulse adapter	PadPuls M2	FB_MBUS_REL_PadPulsM2 [▶ 179]
	2-way pulse adapter	PadPuls M2C	FB_MBUS_REL_PadPulsM2 [▶ 179]
	4-way pulse adapter	PadPuls M4	FB_MBUS_REL_PadPulsM4 [▶ 181]
	4-way pulse adapter	PadPuls M4L	FB_MBUS_REL_PadPulsM4 [▶ 181]

Vendor	Type	Device	Function block
Saia-Burgess [▶ 183]	Electricity meter	ALD1	FB_MBUS_SBC_ALD1 [▶ 184]
	Electricity meter	ALE3	FB_MBUS_SBC_ALE3 [▶ 186]
	Electricity meter	AWD3	FB_MBUS_SBC_ALE3 [▶ 186]
SANEXT [▶ 189]	Heat meter	Sanext Combi	FB_MBUS_ZRM_zelsiusZR [▶ 189]
Schlumberger [▶ 191]	Heat meter	Integral-Mk MaXX	FB_MBUS_SLB_MK_MaXX [▶ 194]
	Heat meter	CF Echo I	FB_MBUS_SLB_CFEchoI [▶ 192]
Schneider Electric [▶ 201]	Electricity meter	iEM3135	FB_MBUS_SEC_iEM3135 [▶ 201]
Sensus [▶ 196]	Heat/cold meter	PolluStat E	FB_MBUS_SEN_Pollu [▶ 197]
	Heat/cold meter	PolluTherm	FB_MBUS_SEN_Pollu [▶ 197]
	Heat/cold meter	PolluCom E	FB_MBUS_SEN_Pollu [▶ 197]
	Water meter		FB_MBUS_SEN_Water [▶ 199]
Sontex [▶ 204]	Heat/cold meter	Supercal 531	FB_MBUS_SON_Supercal531 [▶ 204]
TIP [▶ 207]	Electricity meter	SINUS 85 M	FB_MBUS_TIP_SINUS85M [▶ 207]
Zenner [▶ 210]	Arithmetic unit	multidataWR3	FB_MBUS_ZRM_multidataWR3 [▶ 211]
	Heat meter	zelsiusZR	FB_MBUS_ZRM_zelsiusZR [▶ 189]

6.1 General information

Installation

The "TcSMI.lib/.lb6/.lbx" libraries are installed by default from TwinCAT 2.11 Build 2229 (R3 and x64 Engineering).

Library name

This library replaces the "TcKL6781.lib/.lb6./.lbx".

Hardware documentation in the Beckhoff Information System: KL6781 - M-Bus master terminal

Incompatibility

The TcMBus library is not compatible to the versions older than V2.0.0 and also not usable in the same PLC program. From version V2.0.0 the level converters of the company Relay (e.g. PW3, PW20 or PW60) are no longer supported.

Further libraries required

For PC systems (x86) and Embedded PCs (CXxxxx):

- Standard.lib

For Bus Terminal Controllers from the BCxx00 series:

- Standard.lb6

For Bus Terminal Controllers of the BCxx50, BCxx20, BC9191 and BXxx00 series:

- Standard.lbx

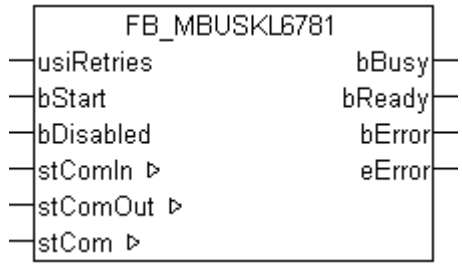
Memory usage

Some of the PLC program memory is already used up by integrating the library. Depending on the application program, therefore, the remaining memory may not be sufficient.

Rounding errors

M-Bus devices may supply very large values (the DWord value range may be exceeded). They are therefore output in string format. Conversions to Real format may lead to inaccuracies/invalid values. Conversions to LReal format are therefore preferable. However, this is not possible for controllers from the BC/BX series. If the values have to be provided in a number format, controllers from the BC/BX series are unsuitable, if the values exceed the DWord value range.

6.2 FB_MBUSKL6781



This function block is used for reading M-Bus devices via a serial KL6781 - M-Bus master terminal.

The block can only be used in conjunction with at least one meter block.

An instance of this block is required for each serial KL6781 terminal.

The minimum cycle time [▶ 231] for reading the serial interface depends on the baud rate. Guide value for 2400 baud: 10 ms max. For programs with a higher cycle time, the blocks be called in a faster task (see also here [▶ 233]).

VAR_INPUT

```
usiRetries      : USINT;
bStart          : BOOL;
bDisabled       : BOOL := FALSE;
```

usiRetries: Number of retries on error. If *usiRetries* is not assigned or set to 0, the value is set to 3 internally.

bStart: Positive edge on this input, all meters is read out once.

bDisabled: TRUE = disable the function block

VAR_OUTPUT

```
bBusy       : BOOL;
bReady      : BOOL;
bError      : BOOL;
eError      : E_MBUS_ERROR;
```

bBusy: The *bBusy* output is TRUE while the meter is being read.

bReady: The *bReady* output is TRUE for one cycle, once meter reading is completed.

bError: The *bError* output becomes TRUE as soon as an error occurs. The error is described via the variable *eError*.

eError: The *eError* output issues an error code when an error occurs (see E_MBUS_ERROR [▶ 216]).

VAR_IN_OUT

```
stComIn     : ST_KL6781inData22B;
stComOut    : ST_KL6781outData22B;
stCom       : ST_MBUS_Communication;
```

stComIn: Is linked to the terminal in the System Manager [▶ 220].

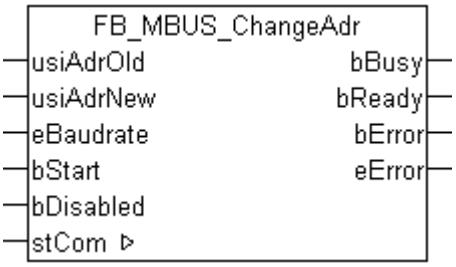
stComOut: Is linked to the terminal in the System Manager [▶ 221].

stCom: About this structure [▶ 221], the block is connected to the meter function blocks.

6.3 General function blocks

Type	Device	Block
Electricity meter	all electricity meter	FB_MBUS_General_Electricity [▶ 34]
Heat meter	all heat meter	FB_MBUS_General_Heat [▶ 39]
Water meter	all water meter	FB_MBUS_General_Water [▶ 44]
Raw data	all	FB_MBUS_RawData [▶ 46]
max. 40 values from the first telegram	all	FB_MBUS_General [▶ 31]
all telegrams, all values	all	FB_MBUS_General_Ext [▶ 35]
parameterized values	all	FB_MBUS_General_Param [▶ 41]
universal send	all	FB_MBUS_General_Send [▶ 43]
scan	all	FB_MBUS_Scan [▶ 48]
Change address	all	FB_MBUS_ChangeAdr [▶ 30]

6.3.1 FB_MBUS_ChangeAdr



This function block can be used to change the primary address.

The function block can only be executed together with the function block FB_MBUSKL6781() [▶ 29].

VAR_INPUT

```
usiAdrOld       : USINT;
usiAdrNew       : USINT;
eBaudrate       : E_MBUS_Baudrate := eMBUS_Baud2400;
bStart          : BOOL;
bDisabled       : BOOL := FALSE;
```

usiAdrOld: Old primary address [▶ 14] of the counter.

usiAdrNew: New primary address [▶ 15] of the counter.

eBaudrate: 300, 600, 1200, 2400, 4800, 9600 baud [▶ 216].

bStart: Positive edge of this input, the primary address [▶ 14] of the meter is changed.

bDisabled: TRUE = disable the function block.

VAR_OUTPUT

```
bBusy       : BOOL;
bReady      : BOOL;
bError      : BOOL;
eError      : E_MBUS_ERROR;
```

bBusy: The *bBusy* output is TRUE while the meter is being read.

bReady: The *bReady* output is TRUE for one cycle, once meter reading is completed.

bError: The *bError* output becomes TRUE as soon as an error occurs. The error is described via the variable *eError*.

eError: The *eError* output issues an error code when an error occurs (see E_MBUS_ERROR [▶ 216]).

VAR_IN_OUT

```
stCom       : ST_MBUS_Communication;
```

stCom: About this structure, the block FB_MBUSKL6781() [▶ 29] is connected to the meter function blocks (see ST_MBUS_Communication [▶ 221]).

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055566347/.zip:

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055569163/.zip:

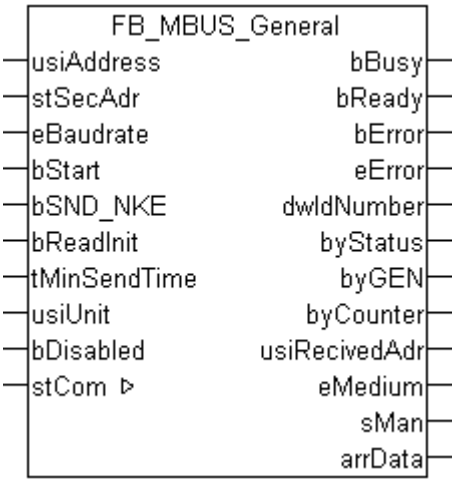
Controller configuration setting: "BCxx50 or BX serial"

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055567755/.zip:

BCxx00 must be set to "Large Model" in the PLC under "Project/Options/Controller Settings".

Controller configuration setting: "BC serial"

6.3.2 FB_MBUS_General



This block is used for reading any M-Bus devices. The variable arrData [▶ 222] supplies a maximum of cMBUS_MaxData [▶ 226] of the first telegram. String values and manufacturer-specific information are not shown correctly.

This function block is **not** suitable for BC/BX.

The function block can only be executed together with the function block FB_MBUSKL6781() [▶ 29].

Functionality of the function block [▶ 13]

VAR_INPUT

```
usiAddress      : USINT;
stSecAdr        : ST_MBUS_SecAdr;
eBaudrate       : E_MBUS_Baudrate := eMBUS_Baud2400;
bStart          : BOOL;
bSND_NKE        : BOOL := TRUE;
bReadInit       : BOOL := TRUE;
tMinSendTime    : TIME := t#2s;
usiUnit         : USINT;
bDisabled       : BOOL := FALSE;
```

usiAddress: Primary address [▶ 14] of the counter, that shall be readout with this module.

stSecAdr: Secondary address [▶ 15] of the counter, that shall be readout with this module.

eBaudrate: 300, 600, 1200, 2400, 4800, 9600 baud [▶ 216].

bStart: Positive edge on this input, the meter is read out once.

bSND_NKE: TRUE to initialize the meter at each reading, and sets the meter on the first telegram (SND_NKE).

bReadInit: After restarting the PLC, the meter is read out once.

tMinSendTime: Standard t#2s. When this time exceeds, the counter is rereadout. At t#0s the counter is not readout and can be readout with bStart manually.

usiUnit: Unit of the energy values. 0=W/J(h) / 1=KW/KJ(h) / 2=MW/MJ(h) / 3=GW/GJ(h).

bDisabled: TRUE = disable the function block.

VAR_OUTPUT

```
bBusy           : BOOL;
bReady          : BOOL;
bError          : BOOL;
eError          : E_MBUS_ERROR;
dwIdNumber      : DWORD;
byStatus        : BYTE;
byGEN           : BYTE;
byCounter       : BYTE;
usiRecivedAdr   : USINT;
eMedium         : E_MBUS_Medium;
sMan            : STRING(3);
arrData         : ARRAY [1..cMBUS_MaxData] OF ST_MBus_Data;
```

bBusy: The *bBusy* output is TRUE while the meter is being read.

bReady: The *bReady* output is TRUE for one cycle, once meter reading is completed.

bError: The *bError* output becomes TRUE as soon as an error occurs. The error is described via the variable *eError*.

eError: The *eError* output issues an error code when an error occurs (see E_MBUS_ERROR [▶ 216]).

dwIdNumber: Serial number of the meter (secondary address).

byStatus: Status of the meter. Please refer to device description for meanings.

byGEN: Meter software version.

byCounter: Transmission counter (number of transmitted RSP_UD).

usiRecivedAdr: Received primary address (0-250).

eMedium: Medium (see E_MBUS_Medium [▶ 219]).

sMan: Manufacturer code.

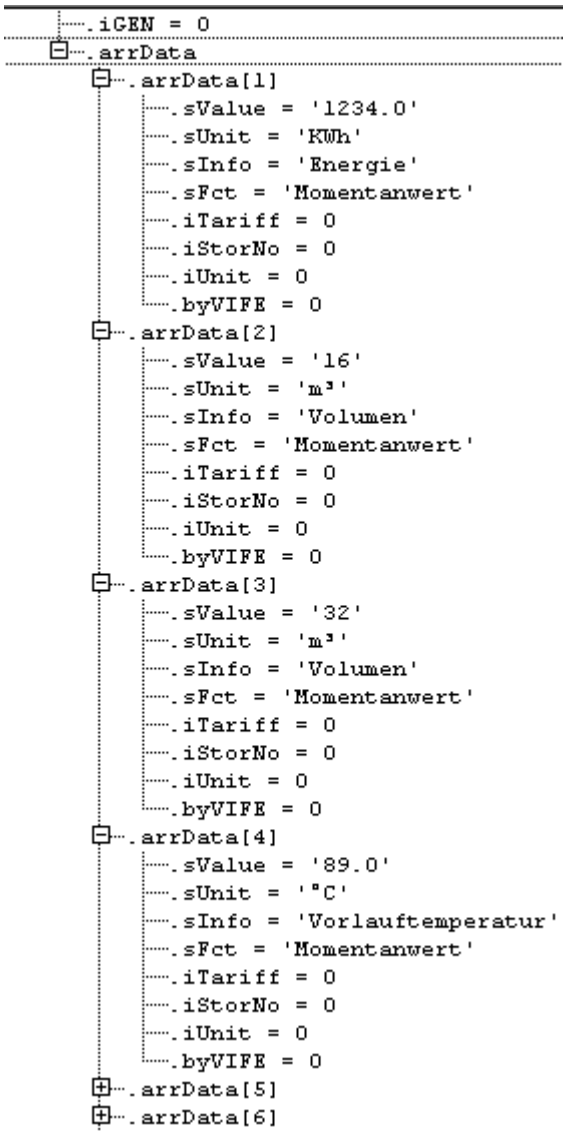
arrData: Maximum cMBUS_MaxData [▶ 226] values of the first telegram (see ST_MBus_Data [▶ 222]). The meaning of the values are explained in the M-Bus protocol of the device.

VAR_IN_OUT

```
stCom          : ST_MBUS_Communication;
```

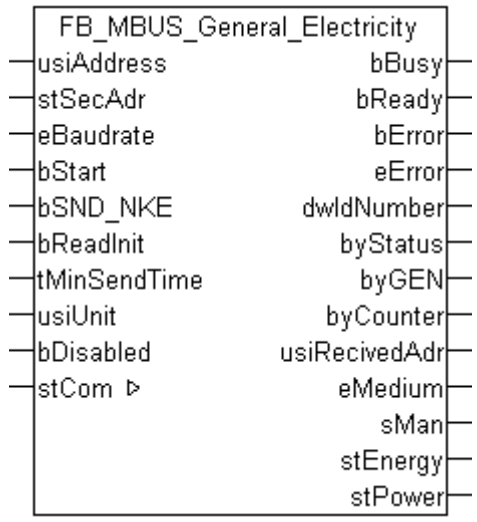
stCom: About this structure, the block FB_MBUSKL6781() [▶ 29] is connected to the meter function blocks (see ST_MBUS_Communication [▶ 221]).

Example view:



https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055566347/.zip:

6.3.3 FB_MBUS_General_Electricity



This function block is used to read electricity meters.

Not all electricity meters automatically send power data. In this case the corresponding structure remains empty.

The function block can only be executed together with the function block FB_MBUSKL6781() [▶ 29].

Functionality of the function block [▶ 13]

VAR_INPUT

```
usiAddress      : USINT;
stSecAdr        : ST_MBUS_SecAdr;
eBaudrate       : E_MBUS_Baudrate := eMBUS_Baud2400;
bStart          : BOOL;
bSND_NKE        : BOOL := TRUE;
bReadInit       : BOOL := TRUE;
tMinSendTime    : TIME := t#2s;
usiUnit         : USINT;
bDisabled       : BOOL := FALSE;
```

usiAddress: Primary address [▶ 14] of the counter, that shall be readout with this module.

stSecAdr: Secondary address [▶ 15] of the counter, that shall be readout with this module.

eBaudrate: 300, 600, 1200, 2400, 4800, 9600 baud [▶ 216].

bStart: Positive edge on this input, the meter is read out once.

bSND_NKE: TRUE to initialize the meter at each reading and sets the meter on the first telegram (SND_NKE).

bReadInit: After restarting the PLC, the meter is read out once.

tMinSendTime: Standard t#2s. When this time exceeds, the counter is rereadout. At t#0s the counter is not readout and can be readout with bStart manually.

usiUnit: Unit of the energy values. 0=W/J(h) / 1=KW/KJ(h) / 2=MW/MJ(h) / 3=GW/GJ(h).

bDisabled: TRUE = disable the function block.

VAR_OUTPUT

```
bBusy           : BOOL;
bReady          : BOOL;
bError          : BOOL;
```

```
eError          : E_MBUS_ERROR;
dwIdNumber      : DWORD;
byStatus        : BYTE;
byGEN           : BYTE;
byCounter       : BYTE;
usiRecivedAdr   : USINT;
eMedium         : E_MBUS_Medium;
sMan            : STRING(3);
stEnergy        : ST_MBus_Info;
stPower         : ST_MBus_Info;
```

bBusy: The *bBusy* output is TRUE while the meter is being read.

bReady: The *bReady* output is TRUE for one cycle, once meter reading is completed.

bError: The *bError* output becomes TRUE as soon as an error occurs. The error is described via the variable *eError*.

eError: The *eError* output issues an error code when an error occurs (see E_MBUS_ERROR [▶ 216]).

dwIdNumber: Serial number of the meter (secondary address).

byStatus: Status of the meter. Please refer to device description for meanings.

byGEN: Meter software version.

byCounter: Transmission counter (number of transmitted RSP_UD).

usiRecivedAdr: Received primary address (0-250).

eMedium: Medium (see E_MBUS_Medium [▶ 219]).

sMan: Manufacturer code.

stEnergy: Meter reading, energy consumption (see ST_MBus_Info [▶ 222]).

stPower: Current energy consumption (see ST_MBus_Info [▶ 222]).

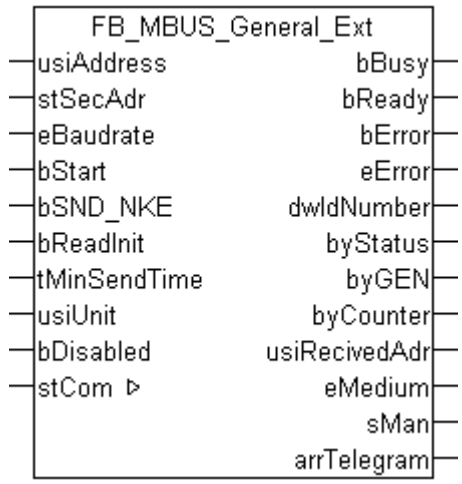
VAR_IN_OUT

```
stCom           : ST_MBUS_Communication;
```

stCom: About this structure, the block FB_MBUSKL6781() [▶ 29] is connected to the meter function blocks (see ST_MBUS_Communication [▶ 221]).

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055566347/.zip:

6.3.4 FB_MBUS_General_Ext



There are devices that send values distributed to several telegrams. With this function block all messages can be read by any device.

The variable *arrTelegram[1..cMBUS_MaxTelegrams].arrData[1..cMBUS_MaxData]* provides a maximum of cMBUS_MaxTelegrams [▶ 226] telegrams maximum cMBUS_MaxData [▶ 226] data.

The number of telegrams to be read can be changed by the constant cMBUS_MaxTelegrams [▶ 226].

The number of data per telegram can be changed by the constant cMBUS_MaxData [▶ 226].

This function block is **not** suitable for BC/BX.

The function block can only be executed together with the function block FB_MBUSKL6781() [▶ 29].

Functionality of the function block [▶ 13]

VAR_INPUT

```
usiAddress      : USINT;
stSecAdr        : ST_MBUS_SecAdr;
eBaudrate       : E_MBUS_Baudrate := eMBUS_Baud2400;
bStart          : BOOL;
bSND_NKE        : BOOL := TRUE;
bReadInit       : BOOL := TRUE;
tMinSendTime    : TIME := t#2s;
usiUnit         : USINT;
bDisabled       : BOOL := FALSE;
```

usiAddress: Primary address [▶ 14] of the counter, that shall be readout with this module.

stSecAdr: Secondary address [▶ 15] of the counter, that shall be readout with this module.

eBaudrate: 300, 600, 1200, 2400, 4800, 9600 baud [▶ 216].

bStart: Positive edge on this input, the meter is read out once.

bSND_NKE: TRUE to initialize the meter at each reading, and sets the meter on the first telegram (SND_NKE).

bReadInit: After restarting the PLC, the meter is read out once.

tMinSendTime: Standard t#2s. When this time exceeds, the counter is rereadout. At t#0s the counter is not readout and can be readout with bStart manually.

usiUnit: Unit of the energy values. 0=W/J(h) / 1=KW/KJ(h) / 2=MW/MJ(h) / 3=GW/GJ(h).

bDisabled: TRUE = disable the function block.

VAR_OUTPUT

```
bBusy           : BOOL;
bReady          : BOOL;
bError          : BOOL;
eError          : E_MBUS_ERROR;
dwIdNumber      : DWORD;
byStatus        : BYTE;
byGEN           : BYTE;
byCounter       : BYTE;
usiRecivedAdr   : USINT;
eMedium         : E_MBUS_Medium;
sMan            : STRING(3);
arrTelegram     : ARRAY [1..cMBUS_MaxTelegrams] OF ST_MBus_Data2;
```

bBusy: The *bBusy* output is TRUE while the meter is being read.

bReady: The *bReady* output is TRUE for one cycle, once meter reading is completed.

bError: The *bError* output becomes TRUE as soon as an error occurs. The error is described via the variable *eError*.

eError: The *eError* output issues an error code when an error occurs (see E_MBUS_ERROR [▶ 216]).

dwIdNumber: Serial number of the meter (secondary address).

byStatus: Status of the meter. Please refer to device description for meanings.

byGEN: Meter software version.

byCounter: Transmission counter (number of transmitted RSP_UD).

usiRecivedAdr: Received primary address (0-250).

eMedium: Medium (see E_MBUS_Medium [▶ 219]).

sMan: Manufacturer code.

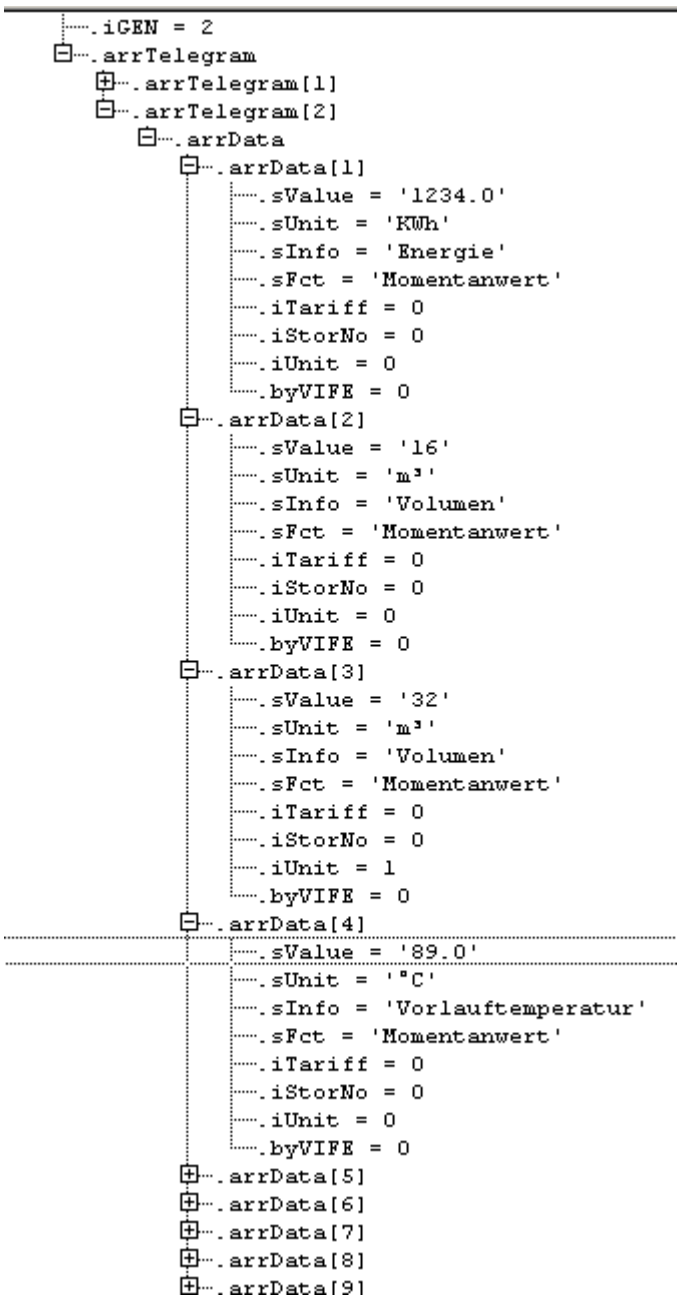
arrTelegram: Maximum cMBUS_MaxTelegrams [▶ 226] telegrams (see ST_MBus_Data2 [▶ 222]). The meaning of the values are explained in the M-Bus protocol of the device.

VAR_IN_OUT

```
stCom          : ST_MBUS_Communication;
```

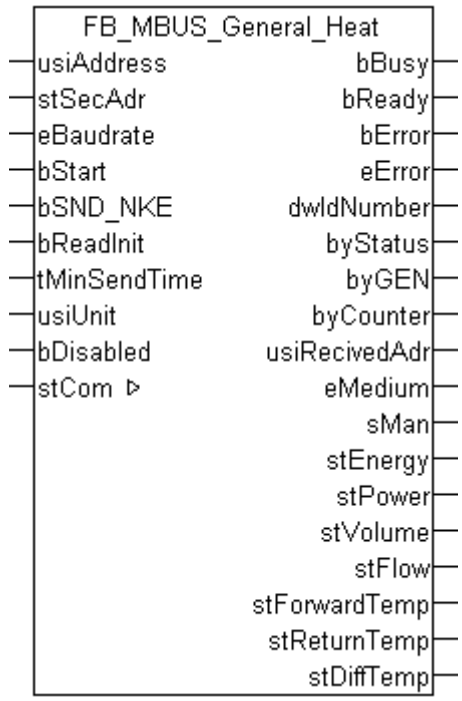
stCom: About this structure, the block FB_MBUSKL6781() [▶ 29] is connected to the meter function blocks (see ST_MBUS_Communication [▶ 221]).

Example view:



https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055566347/.zip:

6.3.5 FB_MBUS_General_Heat



This function block is used to read heat meters.

Many heat meters do not send all values. In this case the corresponding structures remain empty.

The function block can only be executed together with the function block FB_MBUSKL6781() [▶ 29].

Functionality of the function block [▶ 13]

VAR_INPUT

```
usiAddress      : USINT;
stSecAdr        : ST_MBUS_SecAdr;
eBaudrate       : E_MBUS_Baudrate := eMBUS_Baud2400;
bStart          : BOOL;
bSND_NKE        : BOOL := TRUE;
bReadInit       : BOOL := TRUE;
tMinSendTime    : TIME := t#2s;
usiUnit         : USINT;
bDisabled       : BOOL := FALSE;
```

usiAddress: Primary address [▶ 14] of the counter, that shall be readout with this module.

stSecAdr: Secondary address [▶ 15] of the counter, that shall be readout with this module.

eBaudrate: 300, 600, 1200, 2400, 4800, 9600 baud [▶ 216].

bStart: Positive edge on this input, the meter is read out once.

bSND_NKE: TRUE to initialize the meter at each reading, and sets the meter on the first telegram (SND_NKE).

bReadInit: After restarting the PLC, the meter is read out once.

tMinSendTime: Standard t#2s. When this time exceeds, the counter is rereadout. At t#0s the counter is not readout and can be readout with bStart manually.

usiUnit: Unit of the energy values. 0=W/J(h) / 1=KW/KJ(h) / 2=MW/MJ(h) / 3=GW/GJ(h).

bDisabled: TRUE = disable the function block.

VAR_OUTPUT

```
bBusy           : BOOL;
bReady          : BOOL;
bError          : BOOL;
eError          : E_MBUS_ERROR;
dwIdNumber      : DWORD;
byStatus        : BYTE;
byGEN           : BYTE;
byCounter       : BYTE;
usiRecivedAdr   : USINT;
eMedium         : E_MBUS_Medium;
sMan            : STRING(3);
stEnergy        : ST_MBus_Info;
stPower         : ST_MBus_Info;
stVolume        : ST_MBus_Info;
stFlow          : ST_MBus_Info;
stForwardTemp   : ST_MBus_Info;
stReturnTemp    : ST_MBus_Info;
stDiffTemp      : ST_MBus_Info;
```

bBusy: The *bBusy* output is TRUE while the meter is being read.

bReady: The *bReady* output is TRUE for one cycle, once meter reading is completed.

bError: The *bError* output becomes TRUE as soon as an error occurs. The error is described via the variable *eError*.

eError: The *eError* output issues an error code when an error occurs (see E_MBUS_ERROR [▶ 216]).

dwIdNumber: Serial number of the meter (secondary address).

byStatus: Status of the meter. Please refer to device description for meanings.

byGEN: Meter software version.

byCounter: Transmission counter (number of transmitted RSP_UD).

usiRecivedAdr: Received primary address (0-250).

eMedium: Medium (see E_MBUS_Medium [▶ 219]).

sMan: Manufacturer code.

stEnergy: Meter reading, energy consumption (see ST_MBus_Info [▶ 222]).

stPower: Current energy consumption (see ST_MBus_Info [▶ 222]).

stVolume: Meter reading, volume (see ST_MBus_Info [▶ 222]).

stFlow: Current flow (see ST_MBus_Info [▶ 222]).

stForwardTemp: Flow temperature (see ST_MBus_Info [▶ 222]).

stReturnTemp: Return temperature (see ST_MBus_Info [▶ 222]).

stDiffTemp: Temperature difference (see ST_MBus_Info [▶ 222]).

VAR_IN_OUT

```
stCom       : ST_MBUS_Communication;
```

stCom: About this structure, the block FB_MBUSKL6781() [▶ 29] is connected to the meter function blocks (see ST_MBUS_Communication [▶ 221]).

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055566347/.zip:

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055569163/.zip:

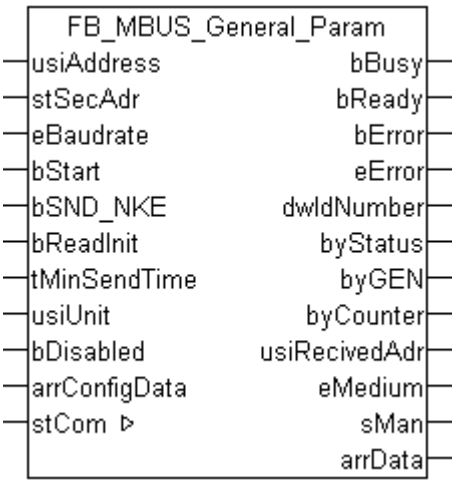
Controller configuration setting: "BCxx50 or BX serial"

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055567755/.zip:

BCxx00 must be set to "Large Model" in the PLC under "Project/Options/Controller Settings".

Controller configuration setting: "BC serial"

6.3.6 FB_MBUS_General_Param



This block is used for reading any M-Bus devices. The variable arrData [▶ 222] supplies cMBUS_MaxDataParam [▶ 226] values.

These values can be parameterised in the input array *arrConfigData*. String values and manufacturer-specific information are not shown correctly.

This function block is not suitable for BC/BX.

The function block can only be executed together with the function block FB_MBUSKL6781(). [▶ 29]

Functionality of the function block [▶ 13]

VAR_INPUT

```
usiAddress      : USINT;
stSecAdr        : ST_MBUS_SecAdr;
eBaudrate       : E_MBUS_Baudrate := eMBUS_Baud2400;
bStart          : BOOL;
bSND_NKE        : BOOL := TRUE;
bReadInit       : BOOL := TRUE;
tMinSendTime    : TIME := t#2s;
usiUnit         : USINT;
bDisabled       : BOOL := FALSE;
arrConfigData       : ARRAY [1..cMBUS_MaxDataParam] OF WORD;
```

usiAddress: Primary address [▶ 14] of the counter, that shall be readout with this module.

stSecAdr: Secondary address [▶ 15] of the counter, that shall be readout with this module.

eBaudrate: 300, 600, 1200, 2400, 4800, 9600 baud [▶ 216].

bStart: Positive edge on this input, the meter is read out once.

bSND_NKE: TRUE to initialize the meter at each reading, and sets the meter on the first telegram (SND_NKE).

bReadInit: After restarting the PLC, the meter is read out once.

tMinSendTime: Standard t#2s. When this time exceeds, the counter is rereadout. At t#0s the counter is not readout and can be readout with bStart manually.

usiUnit: Unit of the energy values. 0=W/J(h) / 1=KW/KJ(h) / 2=MW/MJ(h) / 3=GW/GJ(h).

bDisabled: TRUE = disable the function block.

arrConfigData: maximum of cMBUS_MaxDataParam [▶ 226] input parameters for specifying which values are to be displayed in the output array *arrData*. See example below.

VAR_OUTPUT

```
bBusy           : BOOL;
bReady          : BOOL;
bError          : BOOL;
eError          : E_MBUS_ERROR;
dwIdNumber      : DWORD;
byStatus        : BYTE;
byGEN           : BYTE;
byCounter       : BYTE;
usiRecivedAdr   : USINT;
eMedium         : E_MBUS_Medium;
sMan            : STRING(3);
arrData         : ARRAY [1..cMBUS_MaxDataParam] OF ST_MBus_Data;
```

bBusy: The *bBusy* output is TRUE while the meter is being read.

bReady: The *bReady* output is TRUE for one cycle, once meter reading is completed.

bError: The *bError* output becomes TRUE as soon as an error occurs. The error is described via the variable *eError*.

eError: The *eError* output issues an error code when an error occurs (see E_MBUS_ERROR [▶ 216]).

dwIdNumber: Serial number of the meter (secondary address).

byStatus: Status of the meter. Please refer to device description for meanings.

byGEN: Meter software version.

byCounter: Transmission counter (number of transmitted RSP_UD).

usiRecivedAdr: Received primary address (0-250).

eMedium: Medium (see E_MBUS_Medium [▶ 219]).

sMan: Manufacturer code.

arrData: Maximum cMBUS_MaxDataParam [▶ 226] values. The values can be configured via the input variable *arrConfigData*. The meaning of the values is explained in the M-Bus protocol for the device.

VAR_IN_OUT

```
stCom           : ST_MBUS_Communication;
```

stCom: About this structure, the block FB_MBUSKL6781() [▶ 29] is connected to the meter function blocks (see ST_MBUS_Communication [▶ 221]).

Sample VAR_INPUT *arrConfigData*

arrConfigData[x]:=T T V V; T T= telegram 2 digits (<=99), V V = value 2 digits (maximum cMBUS_MaxData [▶ 226] <=99)

Sample *arrConfigData:*

MBUS.arrConfigData[1]:=0101; (telegram 1, value 1)

MBUS.arrConfigData[2]:=0102; (telegram 1, value 2)

MBUS.arrConfigData[3]:=0309; (telegram 3, value 9)

MBUS.arrConfigData[3]:=1510; (telegram 15, value 10)

MBUS.arrConfigData[4]:=511; (telegram 5, value 11)

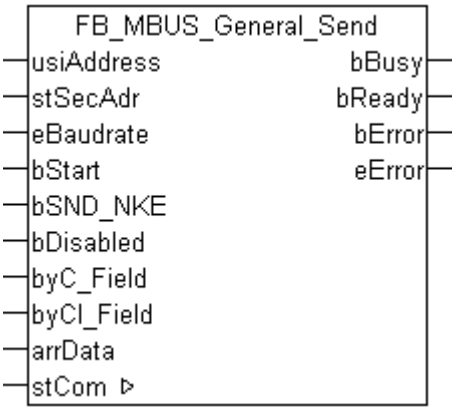
MBUS.arrConfigData[x]:=10; false, too few digits (minimum 3)

MBUS.arrConfigData[x]:=12345; false, too many digits (maximum 4)

Incorrect assignment results in the corresponding field in the output array (arrData) remaining empty (no error message).

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055566347/.zip:

6.3.7 FB_MBUS_General_Send



This function block serves to send data to any M-Bus devices. (for example, the primary address of the meter can be changed with this function block)

The function block can only be executed together with the function block FB_MBUSKL6781() [▶ 29].

Functionality of the function block [▶ 13]

VAR_INPUT

```
usiAddress   : USINT;
stSecAdr     : ST_MBUS_SecAdr;
eBaudrate    : E_MBUS_Baudrate := eMBUS_Baud2400;
bStart       : BOOL;
bSND_NKE     : BOOL := TRUE;
bDisabled    : BOOL := FALSE;
byC_Field    : USINT:=16#53;
byCI_Field   : USINT:=16#51;
arrData      : ARRAY [0..240] OF BYTE;
```

usiAddress: Primary address [▶ 14] of the counter, that shall be readout with this module.

stSecAdr: Secondary address [▶ 15] of the counter, that shall be readout with this module.

eBaudrate: 300, 600, 1200, 2400, 4800, 9600 baud [▶ 216].

bStart: Positive edge on this input, the meter is read out once.

bSND_NKE: TRUE to initialize the meter at each reading, and sets the meter on the first telegram (SND_NKE).

bDisabled: TRUE = disable the function block.

byC_Field: C-Field / Control field.

byCI_Field: CI-Field / Control information field.

arrData: The data to be sent [▶ 14] must be transferred to this variable.

VAR_OUTPUT

```
bBusy       : BOOL;
bReady      : BOOL;
bError      : BOOL;
eError      : E_MBUS_ERROR;
```

bBusy: The *bBusy* output is TRUE while the meter is being read.

bReady: The *bReady* output is TRUE for one cycle, once meter reading is completed.

bError: The *bError* output becomes TRUE as soon as an error occurs. The error is described via the variable *eError*.

eError: The *eError* output issues an error code when an error occurs (see E_MBUS_ERROR [▶ 216]).

VAR_IN_OUT

```
stCom       : ST_MBUS_Communication;
```

stCom: About this structure, the block FB_MBUSKL6781() [▶ 29] is connected to the meter function blocks (see ST_MBUS_Communication [▶ 221]).

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055566347/.zip:

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055569163/.zip:

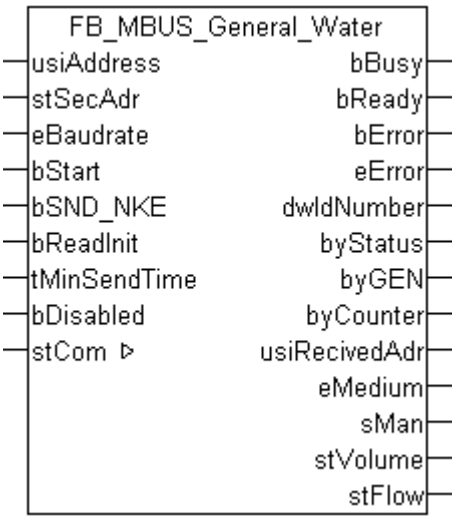
Controller configuration setting: "BCxx50 or BX serial"

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055567755/.zip:

BCxx00 must be set to "Large Model" in the PLC under "Project/Options/Controller Settings".

Controller configuration setting: "BC serial"

6.3.8 FB_MBUS_General_Water



This function block is used to read water meters.

The flow rate is not sent from all water meters. In this case the corresponding structure remains empty.

The function block can only be executed together with the function block FB_MBUSKL6781() [▶ 29].

Functionality of the function block [▶ 13]

VAR_INPUT

```
usiAddress      : USINT;
stSecAdr        : ST_MBUS_SecAdr;
eBaudrate       : E_MBUS_Baudrate := eMBUS_Baud2400;
bStart          : BOOL;
bSND_NKE        : BOOL := TRUE;
bReadInit       : BOOL := TRUE;
tMinSendTime    : TIME := t#2s;
bDisabled       : BOOL := FALSE;
```

usiAddress: Primary address [▶ 14] of the counter, that shall be readout with this module.

stSecAdr: Secondary address [▶ 15] of the counter, that shall be readout with this module.

eBaudrate: 300, 600, 1200, 2400, 4800, 9600 baud [▶ 216].

bStart: Positive edge on this input, the meter is read out once.

bSND_NKE: TRUE to initialize the meter at each reading, and sets the meter on the first telegram (SND_NKE).

bReadInit: After restarting the PLC, the meter is read out once.

tMinSendTime: Standard t#2s. When this time exceeds, the counter is rereadout. At t#0s the counter is not readout and can be readout with bStart manually.

bDisabled: TRUE = disable the function block.

VAR_OUTPUT

```
bBusy           : BOOL;
bReady          : BOOL;
bError          : BOOL;
eError          : E_MBUS_ERROR;
dwIdNumber      : DWORD;
byStatus        : BYTE;
byGEN           : BYTE;
byCounter       : BYTE;
usiRecivedAdr   : USINT;
eMedium         : E_MBUS_Medium;
sMan            : STRING(3);
stVolume        : ST_MBus_Info;
stFlow          : ST_MBus_Info;
```

bBusy: The *bBusy* output is TRUE while the meter is being read.

bReady: The *bReady* output is TRUE for one cycle, once meter reading is completed.

bError: The *bError* output becomes TRUE as soon as an error occurs. The error is described via the variable *eError*.

eError: The *eError* output issues an error code when an error occurs (see E_MBUS_ERROR [▶ 216]).

dwIdNumber: Serial number of the meter (secondary address).

byStatus: Status of the meter. Please refer to device description for meanings.

byGEN: Meter software version.

byCounter: Transmission counter (number of transmitted RSP_UD).

usiRecivedAdr: Received primary address (0-250).

eMedium: Medium (see E_MBUS_Medium [▶ 219]).

sMan: Manufacturer code.

stVolume: Meter reading, volume (see ST_MBus_Info [▶ 222]).

stFlow: Current flow (see ST_MBus_Info [▶ 222]).

VAR_IN_OUT

```
stCom       : ST_MBUS_Communication;
```

stCom: About this structure, the block FB_MBUSKL6781() [▶ 29] is connected to the meter function blocks (see ST_MBUS_Communication [▶ 221]).

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055566347/.zip:

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055569163/.zip:

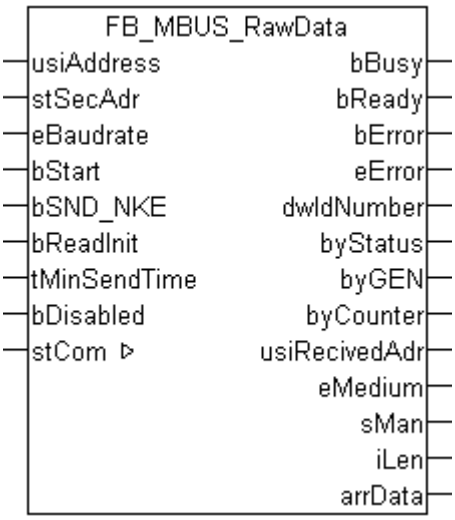
Controller configuration setting: "BCxx50 or BX serial"

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055567755/.zip:

BCxx00 must be set to "Large Model" in the PLC under "Project/Options/Controller Settings".

Controller configuration setting: "BC serial"

6.3.9 FB_MBUS_RawData



This function block is used for reading any M-Bus devices. The variable *arrData* supplies the raw data of the M-Bus device. Only the first telegram is evaluated.

The function block can only be executed together with the function block FB_MBUSKL6781() [▶ 29].

Functionality of the function block [▶ 13]

VAR_INPUT

```
usiAddress      : USINT;
stSecAdr        : ST_MBUS_SecAdr;
eBaudrate       : E_MBUS_Baudrate := eMBUS_Baud2400;
bStart          : BOOL;
bSND_NKE        : BOOL := TRUE;
```

```
bReadInit       : BOOL := TRUE;
tMinSendTime    : TIME := t#2s;
bDisabled       : BOOL := FALSE;
```

usiAddress: Primary address [▶ 14] of the counter, that shall be readout with this module.

stSecAdr: Secondary address [▶ 15] of the counter, that shall be readout with this module.

eBaudrate: 300, 600, 1200, 2400, 4800, 9600 baud [▶ 216].

bStart: Positive edge on this input, the meter is read out once.

bSND_NKE: TRUE to initialize the meter at each reading, and sets the meter on the first telegram (SND_NKE).

bReadInit: After restarting the PLC, the meter is read out once.

tMinSendTime: Standard t#2s. When this time exceeds, the counter is rereadout. At t#0s the counter is not readout and can be readout with bStart manually.

bDisabled: TRUE = disable the function block.

VAR_OUTPUT

```
bBusy           : BOOL;
bReady          : BOOL;
bError          : BOOL;
eError          : E_MBUS_ERROR;
dwIdNumber      : DWORD;
byStatus        : BYTE;
byGEN           : BYTE;
byCounter       : BYTE;
usiRecivedAdr   : USINT;
eMedium         : E_MBUS_Medium;
sMan            : STRING(3);
iLen            : INT;
arrData         : ARRAY [0..259] OF BYTE;
```

bBusy: The *bBusy* output is TRUE while the meter is being read.

bReady: The *bReady* output is TRUE for one cycle, once meter reading is completed.

bError: The *bError* output becomes TRUE as soon as an error occurs. The error is described via the variable *eError*.

eError: The *eError* output issues an error code when an error occurs (see E_MBUS_ERROR [▶ 216]).

dwIdNumber: Serial number of the meter (secondary address).

byStatus: Status of the meter. Please refer to device description for meanings.

byGEN: Meter software version.

byCounter: Transmission counter (number of transmitted RSP_UD).

usiRecivedAdr: Received primary address (0-250).

eMedium: Medium (see E_MBUS_Medium [▶ 219]).

sMan: Manufacturer code.

iLen: Number of transferred bytes.

arrData: Raw data of the first telegram.

VAR_IN_OUT

```
stCom       : ST_MBUS_Communication;
```

stCom: About this structure, the block FB_MBUSKL6781() [▶ 29] is connected to the meter function blocks (see ST_MBUS_Communication [▶ 221]).

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055566347/.zip:

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055569163/.zip:

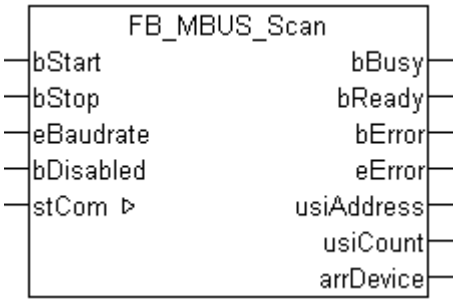
Controller configuration setting: "BCxx50 or BX serial"

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055567755/.zip:

BCxx00 must be set to "Large Model" in the PLC under "Project/Options/Controller Settings".

Controller configuration setting: "BC serial"

6.3.10 FB_MBUS_Scan



This function block can be used to scan the M-Bus bus. All primary addresses (0..250) are queried successively. The array *arrDevice* is used to show certain device information.

Only the primary address is used for scanning.

The primary address [▶ 14] of all devices must be set.

The function block can only be executed together with the function block FB_MBUSKL6781() [▶ 29].

This function block is not compatible with BC / BX devices.

VAR_INPUT

```
bStart      : BOOL;
bStop       : BOOL;
eBaudrate   : E_MBUS_Baudrate := eMBUS_Baud2400;
bDisabled   : BOOL := FALSE;
```

bStart: Positive edge on this input starts the scanning process.

bStop: Positive edge on this input stops the scanning process.

eBaudrate: 300, 600, 1200, 2400, 4800, 9600 baud [▶ 216].

bDisabled: TRUE = disable the function block.

VAR_OUTPUT

```
bBusy       : BOOL;
bReady      : BOOL;
bError      : BOOL;
eError      : E_MBUS_ERROR;
usiAddress  : USINT;
usiCount    : USINT;
arrDevice   : ARRAY [0..250] OF ST_MBus_Scan;
```

bBusy: The *bBusy* output is TRUE while the meter is being read.

bReady: The *bReady* output is TRUE for one cycle, once meter reading is completed.

bError: The *bError* output becomes TRUE as soon as an error occurs. The error is described via the variable *eError*.

eError: The *eError* output issues an error code when an error occurs (see E_MBUS_ERROR [▶ 216]).

usiAddress: Primary address [▶ 14] of the meter.

usiCount: Amount of devices recognized as valid.

arrDevice: Information [▶ 223] about the scanned devices.

VAR_IN_OUT

```
stCom       : ST_MBUS_Communication;
```

stCom: About this structure, the block FB_MBUSKL6781() [▶ 29] is connected to the meter function blocks (see ST_MBUS_Communication [▶ 221]).

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055566347/.zip:

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055569163/.zip:

Controller configuration setting: "BCxx50 or BX serial"

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055567755/.zip:

BCxx00 must be set to "Large Model" in the PLC under "Project/Options/Controller Settings".

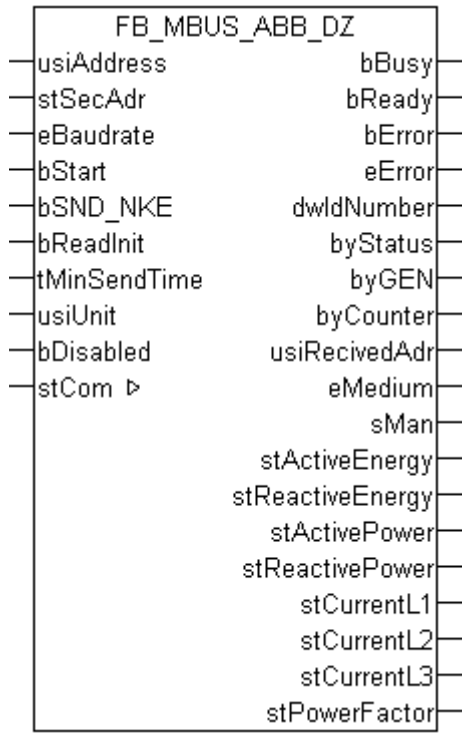
Controller configuration setting: "BC serial"

6.4 ABB

The function blocks only output a selection of the most common data. These data are described on the respective pages under "VAR_OUT". If more or all data are required, the function blocks FB_MBUS_General [▶ 31], FB_MBUS_General_Ext [▶ 35] or FB_MBUS_General_Param [▶ 41] from the General folder should be used. Not that these function blocks do not run on BC or BX systems. The function block FB_MBUS_General_Send [▶ 43] can be used to send data to the device (e.g. setting of the primary address).

Vendor	Type	Device	Function block
ABB	Electricity meter	DELTAplus DZ+	FB_MBUS_ABB_DZ [▶ 50]

6.4.1 FB_MBUS_ABB_DZ



This block is used for reading electricity meters from ABB:

-DELTAplus DZ+

The block can only be used in conjunction with the block FB_MBUSKL6781() [▶ 29].

Functionality of the function block [▶ 13]

VAR_INPUT

```
usiAddress          : USINT;
stSecAdr            : ST_MBUS_SecAdr;
eBaudrate           : E_MBUS_Baudrate := eMBUS_Baud2400;
bStart              : BOOL;
bSND_NKE            : BOOL := TRUE;
bReadInit           : BOOL := TRUE;
tMinSendTime        : TIME := t#2s;
usiUnit             : USINT;
bDisabled           : BOOL := FALSE;
```

usiAddress: Primary address [▶ 14] of the counter, that shall be readout with this module.

stSecAdr: Secondary address [▶ 15] of the counter, that shall be readout with this module.

eBaudrate: 300, 600, 1200, 2400, 4800, 9600 baud [▶ 216].

bStart: Positive edge on this input, the meter is read out once.

bSND_NKE: TRUE to initialize the meter at each reading, and sets the meter on the first telegram (SND_NKE).

bReadInit: After restarting the PLC, the meter is read out once.

tMinSendTime: Standard t#2s. When this time exceeds, the counter is rereadout. At t#0s the counter is not readout and can be readout with bStart manually.

usiUnit: Unit of the energy values. 0=W/J(h) / 1=KW/KJ(h) / 2=MW/MJ(h) / 3=GW/GJ(h).

bDisabled: TRUE = disable the function block.

VAR_OUTPUT

```
bBusy             : BOOL;
bReady            : BOOL;
bError            : BOOL;
eError            : E_MBUS_ERROR;
dwIdNumber        : DWORD;
byStatus          : BYTE;
byGEN             : BYTE;
byCounter         : BYTE;
usiRecivedAdr     : USINT;
eMedium           : E_MBUS_Medium;
sMan              : STRING(3);
stActiveEnergy    : ST_MBus_Info;
stReactiveEnergy  : ST_MBus_Info;
stActivePower     : ST_MBus_Info;
stReactivePower   : ST_MBus_Info;
stCurrentL1       : ST_MBus_Info;
stCurrentL2       : ST_MBus_Info;
stCurrentL3       : ST_MBus_Info;
stPowerFactor     : ST_MBus_Info;
```

bBusy: The *bBusy* output is TRUE while the meter is being read.

bReady: The *bReady* output is TRUE for one cycle once meter reading is completed.

bError: The *bError* output becomes TRUE as soon as an error occurs. The error is described via the variable *eError*.

eError: The *eError* output issues an error code when an error occurs (see E_MBUS_ERROR [▶ 216]).

dwIdNumber: Serial number of the meter (secondary address).

byStatus: Status of the meter. Please refer to device description for meanings.

byGEN: Meter software version.

byCounter: Transmission counter (number of transmitted RSP_UD).

usiRecivedAdr: Received primary address (0-250).

eMedium: Medium (see E_MBUS_Medium [▶ 219]).

sMan: Manufacturer code.

stActiveEnergy: Meter reading, total active energy (see ST_MBus_Info [▶ 222]).

stReactiveEnergy: Meter reading, total reactive energy (see ST_MBus_Info [▶ 222]).

stActivePower: Current consumption, total active power (see ST_MBus_Info [▶ 222]).

stReactivePower: Current consumption, total reactive power (see ST_MBus_Info [▶ 222]).

stCurrentL1: Current L1 (see ST_MBus_Info [▶ 222]).

stCurrentL2: Current L2 (see ST_MBus_Info [▶ 222]).

stCurrentL3: Current L3 (see ST_MBus_Info [▶ 222]).

stPowerFactor: Total power factor (see ST_MBus_Info [▶ 222]).

VAR_IN_OUT

```
stCom          : ST_MBUS_Communication;
```

stCom: About this structure, the block FB_MBUSKL6781() [▶ 29] is connected to the meter function blocks (see ST_MBUS_Communication [▶ 221]).

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055574795/.zip:

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055576203/.zip:

Controller configuration setting: BCxx50 or BX serial

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055577611/.zip:

BCxx00 must be set to "Large Model" in the PLC under "Project/Options/Controller Settings".

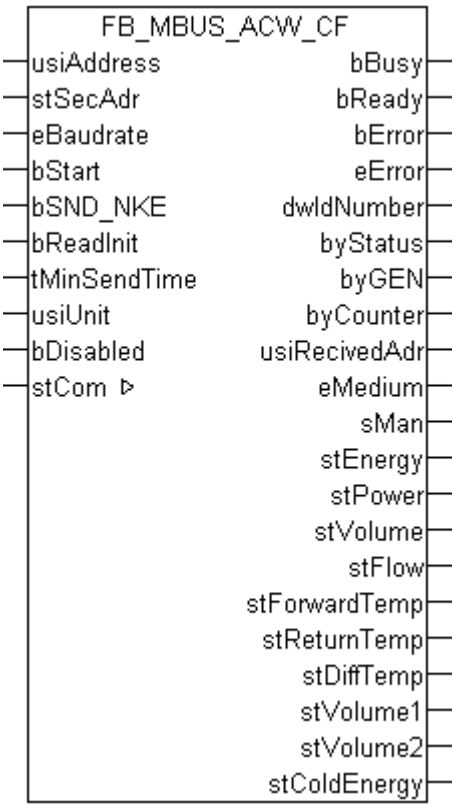
Controller configuration setting: "BC serial"

6.5 Actaris

The function blocks only output a selection of the most common data. These data are described on the respective pages under "VAR_OUT". If more or all data are required, the function blocks FB_MBUS_General [▶ 31], FB_MBUS_General_Ext [▶ 35] or FB_MBUS_General_Param [▶ 41] from the General folder should be used. Not that these function blocks do not run on BC or BX systems. The function block FB_MBUS_General_Send [▶ 43] can be used to send data to the device (e.g. setting of the primary address).

Vendor	Type	Device	Function block
Actaris	Heat meter	CF-Echo II	FB_MBUS_ACW_CF [▶ 52]
	Arithmetic unit	CF-51	
	Arithmetic unit	CF-55	
	Water meter	MB +M	FB_MBUS_ACW_PlusM [▶ 55]

6.5.1 FB_MBUS_ACW_CF



This block is used for reading heat meters from Actaris:

-CF-Echo II

-CF-51

-CF-55

-CF-800

Up to two additional water meters can be connected to this device (optional).

The block can only be used in conjunction with the block FB_MBUSKL6781() [▶ 29].

Functionality of the function block [▶ 13]

VAR_INPUT

```
usiAddress      : USINT;
stSecAdr        : ST_MBUS_SecAdr;
eBaudrate       : E_MBUS_Baudrate := eMBUS_Baud2400;
bStart          : BOOL;
bSND_NKE        : BOOL:=TRUE;
bReadInit       : BOOL:=TRUE;
tMinSendTime    : TIME:=t#2s;
usiUnit         : USINT;
bDisabled       : BOOL:=FALSE;
```

usiAddress: Primary address [▶ 14] of the counter, that shall be readout with this module.

stSecAdr: Secondary address [▶ 15] of the counter, that shall be readout with this module.

eBaudrate: 300, 1200, 2400, 9600 baud [▶ 216].

bStart: Positive edge on this input, the meter is read out once.

bSND_NKE: TRUE to initialize the meter at each reading, and sets the meter on the first telegram (SND_NKE).

bReadInit: After restarting the PLC, the meter is read out once.

tMinSendTime: Standard t#2s. When this time exceeds, the counter is rereadout. At t#0s the counter is not readout and can be readout with bStart manually.

usiUnit: Unit of the energy values. 0=W/J(h) / 1=KW/KJ(h) / 2=MW/MJ(h) / 3=GW/GJ(h).

bDisabled: TRUE = disable the function block.

VAR_OUTPUT

```
bBusy           : BOOL;
bReady          : BOOL;
bError          : BOOL;
eError          : E_MBUS_ERROR;
dwIdNumber      : DWORD;
byStatus        : BYTE;
byGEN           : BYTE;
byCounter       : BYTE;
usiRecivedAdr   : USINT;
eMedium         : E_MBUS_Medium;
sMan            : STRING(3);
stEnergy        : ST_MBus_Info;
stPower         : ST_MBus_Info;
stVolume        : ST_MBus_Info;
stFlow          : ST_MBus_Info;
stForwardTemp   : ST_MBus_Info;
stReturnTemp    : ST_MBus_Info;
stDiffTemp      : ST_MBus_Info;
stVolume1       : ST_MBus_Info;
stVolume2       : ST_MBus_Info;
stColdEnergy    : ST_MBus_Info;
```

bBusy: The *bBusy* output is TRUE while the meter is being read.

bReady: The *bReady* output is TRUE for one cycle, once meter reading is completed.

bError: The *bError* output becomes TRUE as soon as an error occurs. The error is described via the variable *eError*.

eError: The *eError* output issues an error code when an error occurs (see E_MBUS_ERROR [▶ 216]).

dwIdNumber: Serial number of the meter (secondary address).

byStatus: Status of the meter. Please refer to device description for meanings.

byGEN: Meter software version.

byCounter: Transmission counter (number of transmitted RSP_UD).

usiRecivedAdr: Received primary address (0-250).

eMedium: Medium (see E_MBUS_Medium [▶ 219]).

sMan: Manufacturer code.

stEnergy: Meter reading, energy consumption (see ST_MBus_Info [▶ 222]).

stPower: Current energy consumption (see ST_MBus_Info [▶ 222]).

stVolume: Meter reading, volume (see ST_MBus_Info [▶ 222]).

stFlow: Current flow (see ST_MBus_Info [▶ 222]).

stForwardTemp: Flow temperature (see ST_MBus_Info [▶ 222]).

stReturnTemp: Return temperature (see ST_MBus_Info [▶ 222]).

stDiffTemp: Temperature difference (see ST_MBus_Info [▶ 222]).

stVolume1: Meter reading of additional water meter 1 (option) (see ST_MBus_Info [▶ 222]).

stVolume2: Meter reading of additional water meter 2 (option) (see ST_MBus_Info [▶ 222]).

stColdEnergy: Meter reading, cold energy (option) (see ST_MBus_Info [▶ 222]).

VAR_IN_OUT

```
stCom          : ST_MBUS_Communication;
```

stCom: About this structure, the block FB_MBUSKL6781() [▶ 29] is connected to the meter function blocks (see ST_MBUS_Communication [▶ 221]).

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055579019/.zip:

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055580427/.zip:

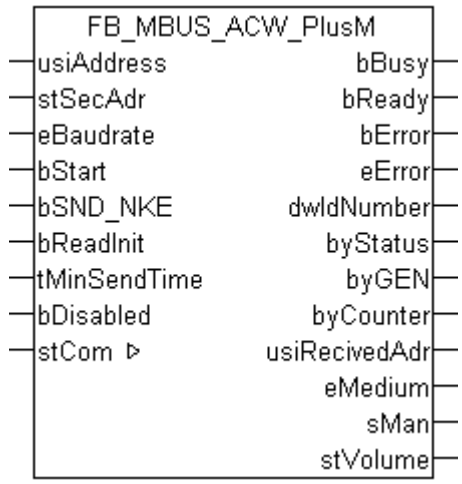
Controller configuration setting: "BCxx50 or BX serial"

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055581835/.zip:

BCxx00 must be set to "Large Model" in the PLC under "Project/Options/Controller Settings".

Controller configuration setting: "BC serial"

6.5.2 FB_MBUS_ACW_PlusM



This function block is used to read water meters from Actaris:

-BM +M

The function block can only be executed together with the function block FB_MBUSKL6781() [▶ 29].

Functionality of the function block [▶ 13]

VAR_INPUT

```
usiAddress      : USINT;
stSecAdr        : ST_MBUS_SecAdr;
eBaudrate       : E_MBUS_Baudrate := eMBUS_Baud2400;
bStart          : BOOL;
bSND_NKE        : BOOL:=TRUE;
bReadInit       : BOOL:=TRUE;
tMinSendTime    : TIME:=t#2s;
bDisabled       : BOOL:=FALSE;
```

usiAddress: Primary address [▶ 14] of the counter, that shall be readout with this module.

stSecAdr: Secondary address [▶ 15] of the counter, that shall be readout with this module.

eBaudrate: 300, 2400 baud [▶ 216].

bStart: Positive edge on this input, the meter is read out once.

bSND_NKE: TRUE to initialize the meter at each reading, and sets the meter on the first telegram (SND_NKE).

bReadInit: After restarting the PLC, the meter is read out once.

tMinSendTime: Standard t#2s. When this time exceeds, the counter is rereadout. At t#0s the counter is not readout and can be readout with bStart manually.

bDisabled: TRUE = disable the function block.

VAR_OUTPUT

```
bBusy           : BOOL;
bReady          : BOOL;
bError          : BOOL;
eError          : E_MBUS_ERROR;
dwIdNumber      : DWORD;
byStatus        : BYTE;
byGEN           : BYTE;
byCounter       : BYTE;
usiRecivedAdr   : USINT;
eMedium         : E_MBUS_Medium;
sMan            : STRING(3);
stVolume        : ST_MBus_Info;
```

bBusy: The *bBusy* output is TRUE while the meter is being read.

bReady: The *bReady* output is TRUE for one cycle, once meter reading is completed.

bError: The *bError* output becomes TRUE as soon as an error occurs. The error is described via the variable *eError*.

eError: The *eError* output issues an error code when an error occurs (see E_MBUS_ERROR [▶ 216]).

dwIdNumber: Serial number of the meter (secondary address).

byStatus: Status of the meter. Please refer to device description for meanings.

byGEN: Meter software version.

byCounter: Transmission counter (number of transmitted RSP_UD).

usiRecivedAdr: Received primary address (0-250).

eMedium: Medium (see E_MBUS_Medium [▶ 219]).

sMan: Manufacturer code.

stVolume: Meter reading, water consumption (see ST_MBus_Info [▶ 222]).

VAR_IN_OUT

```
stCom          : ST_MBUS_Communication;
```

stCom: About this structure, the block FB_MBUSKL6781() [▶ 29] is connected to the meter function blocks (see ST_MBUS_Communication [▶ 221]).

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055579019/.zip:

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055580427/.zip:

Controller configuration setting: "BCxx50 or BX serial"

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055581835/.zip:

BCxx00 must be set to "Large Model" in the PLC under "Project/Options/Controller Settings".

Controller configuration setting: "BC serial"

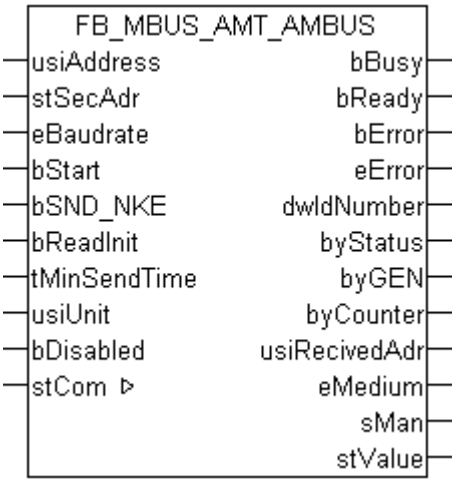
6.6 Aquametro

The function blocks only output a selection of the most common data. These data are described on the respective pages under "VAR_OUT". If more or all data are required, the function blocks FB_MBUS_General [▶ 31], FB_MBUS_General_Ext [▶ 35] or FB_MBUS_General_Param [▶ 41] from the General folder should be used. Not that these function blocks do not run on BC or BX systems. The function block FB_MBUS_General_Send [▶ 43] can be used to send data to the device (e.g. setting of the primary address).

Vendor	Type	Device	Function block
Aquametro	Water meter	SAPHIR	FB_MBUS_AMT_SAPHIR [▶ 66]
	Heat meter	CALEC	FB_MBUS_AMT_CALEC [▶ 61]
	Heat meter	CALEC ST, version C4	FB_MBUS_AMT_CALEC_STC4 [▶ 64]
	Heat meter	AMTRON	FB_MBUS_AMT_AMTRON [▶ 59]
	Pulse collector	AMBUS	FB_MBUS_AMT_AMBUS [▶ 57]
	Heat meter	AMTRON SONIC D	FB_MBUS_HYD_Sharky [▶ 117], FB_MBUS_HYD_Sharky_00 [▶ 120]

Vendor	Type	Device	Function block

6.6.1 FB_MBUS_AMT_AMBUS



This function block is used to read pulse collectors from Aquametro:

-AMBUS IS

The function block can only be executed together with the function block FB_MBUSKL6781() [▶ 29].

Functionality of the function block [▶ 13]

VAR_INPUT

```
usiAddress      : USINT;
stSecAdr        : ST_MBUS_SecAdr;
eBaudrate       : E_MBUS_Baudrate := eMBUS_Baud2400;
bStart          : BOOL;
bSND_NKE        : BOOL := TRUE;
bReadInit       : BOOL := TRUE;
tMinSendTime    : TIME := t#2s;
usiUnit         : USINT;
bDisabled       : BOOL := FALSE;
```

usiAddress: Primary address [▶ 14] of the counter, that shall be readout with this module.

stSecAdr: Secondary address [▶ 15] of the counter, that shall be readout with this module.

eBaudrate: baud [▶ 216].

bStart: Positive edge on this input, the meter is read out once.

bSND_NKE: TRUE to initialize the meter at each reading and sets the meter on the first telegram (SND_NKE).

bReadInit: After restarting the PLC, the meter is read out once.

tMinSendTime: Standard t#2s. When this time exceeds, the counter is rereadout. At t#0s the counter is not readout and can be readout with bStart manually.

usiUnit: Unit of the energy values. 0=W/J(h) / 1=KW/KJ(h) / 2=MW/MJ(h) / 3=GW/GJ(h).

bDisabled: TRUE = disable the function block.

VAR_OUTPUT

```
bBusy           : BOOL;
bReady          : BOOL;
bError          : BOOL;
```

```
eError          : E_MBUS_ERROR;
dwIdNumber      : DWORD;
byStatus        : BYTE;
byGEN           : BYTE;
byCounter       : BYTE;
usiRecivedAdr   : USINT;
eMedium         : E_MBUS_Medium;
sMan            : STRING(3);
stValue         : ST_MBus_Info;
```

bBusy: The *bBusy* output is TRUE while the meter is being read.

bReady: The *bReady* output is TRUE for one cycle, once meter reading is completed.

bError: The *bError* output becomes TRUE as soon as an error occurs. The error is described via the variable *eError*.

eError: The *eError* output issues an error code when an error occurs (see E_MBUS_ERROR [▶ 216]).

dwIdNumber: Serial number of the meter (secondary address).

byStatus: Status of the meter. Please refer to device description for meanings.

byGEN: Meter software version.

byCounter: Transmission counter (number of transmitted RSP_UD).

usiRecivedAdr: Received primary address (0-250).

eMedium: Medium (see E_MBUS_Medium [▶ 219]).

sMan: Manufacturer code.

stValue: Meter reading (see ST_MBus_Info [▶ 222]).

VAR_IN_OUT

```
stCom           : ST_MBUS_Communication;
```

stCom: About this structure, the block FB_MBUSKL6781() [▶ 29] is connected to the meter function blocks (see ST_MBUS_Communication [▶ 221]).

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055583243/.zip:

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055584651/.zip:

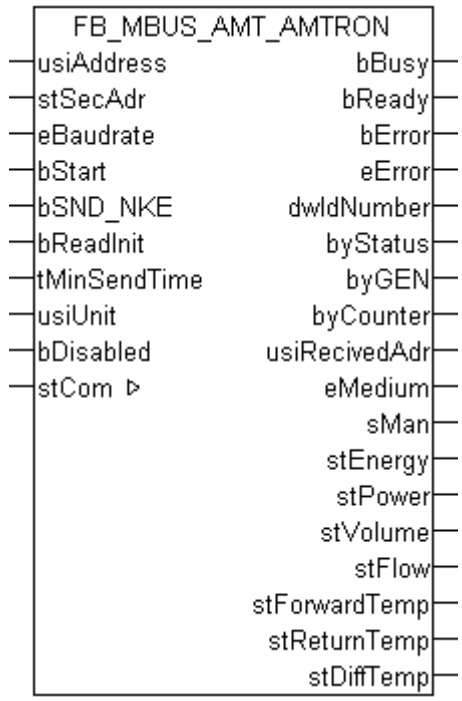
Controller configuration setting: "BCxx50 or BX serial"

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055586059/.zip:

BCxx00 must be set to "Large Model" in the PLC under "Project/Options/Controller Settings".

Controller configuration setting: "BC serial"

6.6.2 FB_MBUS_AMT_AMTRON



This function block is used to read heat meters from Aquametro:

-AMTRON

The function block can only be executed together with the function block FB_MBUSKL6781() [▶ 29].

Functionality of the function block [▶ 13]

VAR_INPUT

```
usiAddress      : USINT;
stSecAdr        : ST_MBUS_SecAdr;
eBaudrate       : E_MBUS_Baudrate := eMBUS_Baud2400;
bStart          : BOOL;
bSND_NKE        : BOOL := TRUE;
bReadInit       : BOOL := TRUE;
tMinSendTime    : TIME := t#2s;
usiUnit         : USINT;
bDisabled       : BOOL := FALSE;
```

usiAddress: Primary address [▶ 14] of the counter, that shall be readout with this module.

stSecAdr: Secondary address [▶ 15] of the counter, that shall be readout with this module.

eBaudrate: 300, 2400 baud [▶ 216].

bStart: Positive edge on this input, the meter is read out once.

bSND_NKE: TRUE to initialize the meter at each reading, and sets the meter on the first telegram (SND_NKE).

bReadInit: After restarting the PLC, the meter is read out once.

tMinSendTime: Standard t#2s. When this time exceeds, the counter is rereadout. At t#0s the counter is not readout and can be readout with bStart manually.

usiUnit: Unit of the energy values. 0=W/J(h) / 1=KW/KJ(h) / 2=MW/MJ(h) / 3=GW/GJ(h).

bDisabled: TRUE = disable the function block.

VAR_OUTPUT

```
bBusy           : BOOL;
bReady          : BOOL;
bError          : BOOL;
eError          : E_MBUS_ERROR;
dwIdNumber      : DWORD;
byStatus        : BYTE;
byGEN           : BYTE;
byCounter       : BYTE;
usiRecivedAdr   : USINT;
eMedium         : E_MBUS_Medium;
sMan            : STRING(3);
stEnergy        : ST_MBus_Info;
stPower         : ST_MBus_Info;
stVolume        : ST_MBus_Info;
stFlow          : ST_MBus_Info;
stForwardTemp   : ST_MBus_Info;
stReturnTemp    : ST_MBus_Info;
stDiffTemp      : ST_MBus_Info;
```

bBusy: The *bBusy* output is TRUE while the meter is being read.

bReady: The *bReady* output is TRUE for one cycle, once meter reading is completed.

bError: The *bError* output becomes TRUE as soon as an error occurs. The error is described via the variable *eError*.

eError: The *eError* output issues an error code when an error occurs (see E_MBUS_ERROR [▶ 216]).

dwIdNumber: Serial number of the meter (secondary address).

byStatus: Status of the meter. Please refer to device description for meanings.

byGEN: Meter software version.

byCounter: Transmission counter (number of transmitted RSP_UD).

usiRecivedAdr: Received primary address (0-250).

eMedium: Medium (see E_MBUS_Medium [▶ 219]).

sMan: Manufacturer code.

stEnergy: Meter reading, energy consumption (see ST_MBus_Info [▶ 222]).

stPower: Current energy consumption (see ST_MBus_Info [▶ 222]).

stVolume: Meter reading, volume (see ST_MBus_Info [▶ 222]).

stFlow: Current flow (see ST_MBus_Info [▶ 222]).

stForwardTemp: Flow temperature (see ST_MBus_Info [▶ 222]).

stReturnTemp: Return temperature (see ST_MBus_Info [▶ 222]).

stDiffTemp: Temperature difference (see ST_MBus_Info [▶ 222]).

VAR_IN_OUT

```
stCom           : ST_MBUS_Communication;
```

stCom: About this structure, the block FB_MBUSKL6781() [▶ 29] is connected to the meter function blocks (see ST_MBUS_Communication [▶ 221]).

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055583243/.zip:

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055584651/.zip:

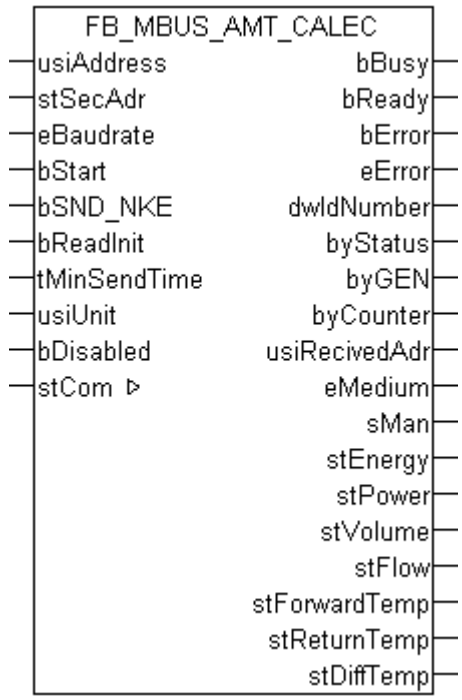
Controller configuration setting: "BCxx50 or BX serial"

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055586059/.zip:

BCxx00 must be set to "Large Model" in the PLC under "Project/Options/Controller Settings".

Controller configuration setting: "BC serial"

6.6.3 FB_MBUS_AMT_CALEC



This block is used for reading heat meters from Aquametro:

-CALEC

The block can only be used in conjunction with the block FB_MBUSKL6781() [▶ 29].

Functionality of the function block [▶ 13]

VAR_INPUT

```
usiAddress      : USINT;
stSecAdr        : ST_MBUS_SecAdr;
eBaudrate       : E_MBUS_Baudrate := eMBUS_Baud2400;
bStart          : BOOL;
bSND_NKE        : BOOL := TRUE;
bReadInit       : BOOL := TRUE;
tMinSendTime    : TIME := t#2s;
usiUnit         : USINT;
bDisabled       : BOOL := FALSE;
```

usiAddress: Primary address [▶ 14] of the counter, that shall be readout with this module.

stSecAdr: Secondary address [▶ 15] of the counter, that shall be readout with this module.

eBaudrate: 300, 2400 baud [▶ 216].

bStart: Positive edge on this input, the meter is read out once.

bSND_NKE: TRUE to initialize the meter at each reading, and sets the meter on the first telegram (SND_NKE).

bReadInit: After restarting the PLC, the meter is read out once.

tMinSendTime: Standard t#2s. When this time exceeds, the counter is rereadout. At t#0s the counter is not readout and can be readout with bStart manually.

usiUnit: Unit of the energy values. 0=W/J(h) / 1=KW/KJ(h) / 2=MW/MJ(h) / 3=GW/GJ(h).

bDisabled: TRUE = disable the function block.

VAR_OUTPUT

```
bBusy           : BOOL;
bReady          : BOOL;
bError          : BOOL;
eError          : E_MBUS_ERROR;
dwIdNumber      : DWORD;
byStatus        : BYTE;
byGEN           : BYTE;
byCounter       : BYTE;
usiRecivedAdr   : USINT;
eMedium         : E_MBUS_Medium;
sMan            : STRING(3);
stEnergy        : ST_MBus_Info;
stPower         : ST_MBus_Info;
stVolume        : ST_MBus_Info;
stFlow          : ST_MBus_Info;
stForwardTemp   : ST_MBus_Info;
stReturnTemp    : ST_MBus_Info;
stDiffTemp      : ST_MBus_Info;
```

bBusy: The *bBusy* output is TRUE while the meter is being read.

bReady: The *bReady* output is TRUE for one cycle, once meter reading is completed.

bError: The *bError* output becomes TRUE as soon as an error occurs. The error is described via the variable *eError*.

eError: The *eError* output issues an error code when an error occurs (see E_MBUS_ERROR [▶ 216]).

dwIdNumber: Serial number of the meter (secondary address).

byStatus: Status of the meter. Please refer to device description for meanings.

byGEN: Meter software version.

byCounter: Transmission counter (number of transmitted RSP_UD).

usiRecivedAdr: Received primary address (0-250).

eMedium: Medium (see E_MBUS_Medium [▶ 219]).

sMan: Manufacturer code.

stEnergy: Meter reading, energy consumption (see ST_MBus_Info [▶ 222]).

stPower: Current energy consumption (see ST_MBus_Info [▶ 222]).

stVolume: Meter reading, volume (see ST_MBus_Info [▶ 222]).

stFlow: Current flow (see ST_MBus_Info [▶ 222]).

stForwardTemp: Flow temperature (see ST_MBus_Info [▶ 222]).

stReturnTemp: Return temperature (see ST_MBus_Info [▶ 222]).

stDiffTemp: Temperature difference (see ST_MBus_Info [▶ 222]).

VAR_IN_OUT

```
stCom           : ST_MBUS_Communication;
```

stCom: About this structure, the block FB_MBUSKL6781() [▶ 29] is connected to the meter function blocks (see ST_MBUS_Communication [▶ 221]).

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055583243/.zip:

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055584651/.zip:

Controller configuration setting: "BCxx50 or BX serial"

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055586059/.zip:

BCxx00 must be set to "Large Model" in the PLC under "Project/Options/Controller Settings".

Controller configuration setting: "BC serial"

VAR_IN_OUT

```
stCom          : ST_MBUS_Communication;
```

stCom: About this structure, the block FB_MBUSKL6781() [▶ 29] is connected to the meter function blocks (see ST_MBUS_Communication [▶ 221]).

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055583243/.zip:

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055584651/.zip:

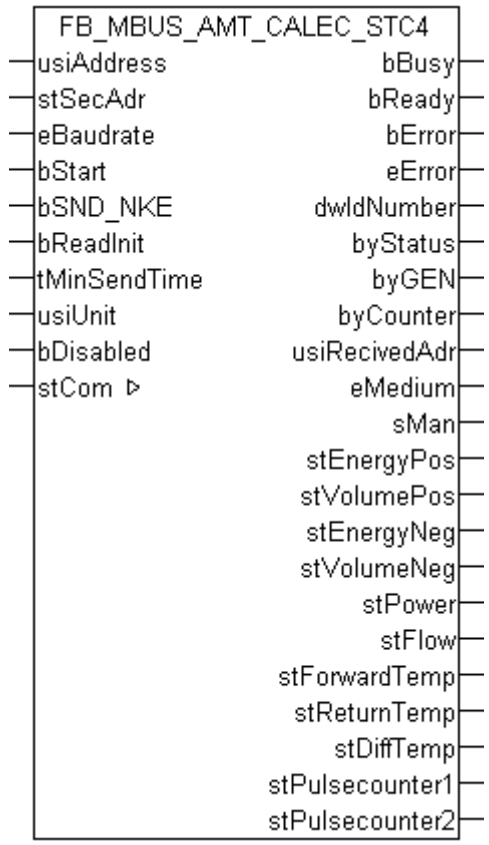
Controller configuration setting: "BCxx50 or BX serial"

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055586059/.zip:

BCxx00 must be set to "Large Model" in the PLC under "Project/Options/Controller Settings".

Controller configuration setting: "BC serial"

6.6.4 FB_MBUS_AMT_CALEC_STC4



This function block is used to read heat meters from Aquametro:

-CALEC ST, version C4

The function block can only be executed together with the function block FB_MBUSKL6781() [▶ 29].

Functionality of the function block [▶ 13]

VAR_INPUT

```
usiAddress       : USINT;
stSecAdr         : ST_MBUS_SecAdr;
eBaudrate        : E_MBUS_Baudrate := eMBUS_Baud2400;
bStart           : BOOL;
bSND_NKE         : BOOL := TRUE;
bReadInit        : BOOL := TRUE;
tMinSendTime     : TIME := t#2s;
usiUnit          : USINT;
bDisabled        : BOOL := FALSE;
```

usiAddress: Primary address [▶ 14] of the counter, that shall be readout with this module.

stSecAdr: Secondary address [▶ 15] of the counter, that shall be readout with this module.

eBaudrate: 300, 2400 baud [▶ 216].

bStart: Positive edge on this input, the meter is read out once.

bSND_NKE: TRUE to initialize the meter at each reading, and sets the meter on the first telegram (SND_NKE).

bReadInit: After restarting the PLC, the meter is read out once.

tMinSendTime: Standard t#2s. When this time exceeds, the counter is rereadout. At t#0s the counter is not readout and can be readout with bStart manually.

usiUnit: Unit of the energy values. 0=W/J(h) / 1=KW/KJ(h) / 2=MW/MJ(h) / 3=GW/GJ(h).

bDisabled: TRUE = disable the function block.

VAR_OUTPUT

```
bBusy            : BOOL;
bReady           : BOOL;
bError           : BOOL;
eError           : E_MBUS_ERROR;
dwIdNumber       : DWORD;
byStatus         : BYTE;
byGEN            : BYTE;
byCounter        : BYTE;
usiRecivedAdr    : USINT;
eMedium          : E_MBUS_Medium;
sMan             : STRING(3);
stEnergyPos      : ST_MBus_Info;
stVolumePos      : ST_MBus_Info;
stEnergyNeg      : ST_MBus_Info;
stVolumeNeg      : ST_MBus_Info;
stPower          : ST_MBus_Info;
stFlow           : ST_MBus_Info;
stForwardTemp    : ST_MBus_Info;
stReturnTemp     : ST_MBus_Info;
stDiffTemp       : ST_MBus_Info;
stPulsecounter1  : ST_MBus_Info;
stPulsecounter2  : ST_MBus_Info;
```

bBusy: The *bBusy* output is TRUE while the meter is being read.

bReady: The *bReady* output is TRUE for one cycle, once meter reading is completed.

bError: The *bError* output becomes TRUE as soon as an error occurs. The error is described via the variable *eError*.

eError: The *eError* output issues an error code when an error occurs (see E_MBUS_ERROR [▶ 216]).

dwIdNumber: Serial number of the meter (secondary address).

byStatus: Status of the meter. Please refer to device description for meanings.

byGEN: Meter software version.

byCounter: Transmission counter (number of transmitted RSP_UD).

usiRecivedAdr: Received primary address (0-250).

eMedium: Medium (see E_MBUS_Medium [▶ 219]).

sMan: Manufacturer code.

stEnergyPos: Meter reading, energy consumption (positive) (see ST_MBus_Info [▶ 222]).

stVolumePos: Meter reading, water consumption (positive) (see ST_MBus_Info [▶ 222]).

stEnergyNeg: Meter reading, energy consumption (negative) (see ST_MBus_Info [▶ 222]).

stVolumeNeg: Meter reading, water consumption (negative) (see ST_MBus_Info [▶ 222]).

stPower: Current energy consumption (see ST_MBus_Info [▶ 222]).

stFlow: Current flow (see ST_MBus_Info [▶ 222]).

stForwardTemp: Flow temperature (see ST_MBus_Info [▶ 222]).

stReturnTemp: Return temperature (see ST_MBus_Info [▶ 222]).

stDiffTemp: Temperature difference (see ST_MBus_Info [▶ 222]).

stPulsecounter1: Pulse counter 1 (see ST_MBus_Info [▶ 222]).

stPulsecounter2: Pulse counter 2 (see ST_MBus_Info [▶ 222]).

VAR_IN_OUT

```
stCom          : ST_MBUS_Communication;
```

stCom: About this structure, the block FB_MBUSKL6781() [▶ 29] is connected to the meter function blocks (see ST_MBUS_Communication [▶ 221]).

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055583243/.zip:

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055584651/.zip:

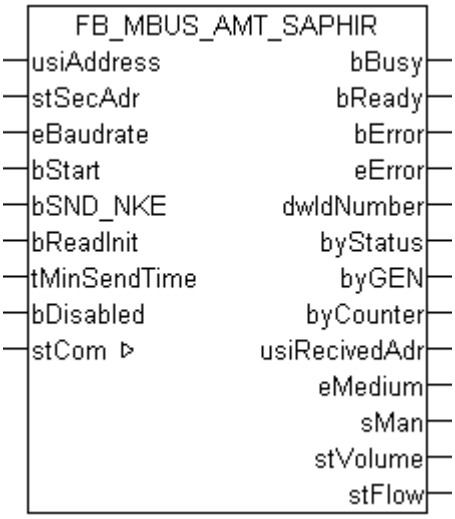
Controller configuration setting: "BCxx50 or BX serial"

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055586059/.zip:

BCxx00 must be set to "Large Model" in the PLC under "Project/Options/Controller Settings".

Controller configuration setting: "BC serial"

6.6.5 FB_MBUS_AMT_SAPHIR



This function block is used to read water meters from Aquametro.

-Saphir

The function block can only be executed together with the function block FB_MBUSKL6781() [▶ 29].

Functionality of the function block [▶ 13]

VAR_INPUT

```
usiAddress     : USINT;
stSecAdr       : ST_MBUS_SecAdr;
eBaudrate      : E_MBUS_Baudrate := eMBUS_Baud2400;
bStart         : BOOL;
bSND_NKE       : BOOL := TRUE;
bReadInit      : BOOL := TRUE;
tMinSendTime   : TIME := t#2s;
bDisabled      : BOOL := FALSE;
```

usiAddress: Primary address [▶ 14] of the counter, that shall be readout with this module.

stSecAdr: Secondary address [▶ 15] of the counter, that shall be readout with this module.

eBaudrate: baud [▶ 216].

bStart: Positive edge on this input, the meter is read out once.

bSND_NKE: TRUE to initialize the meter at each reading, and sets the meter on the first telegram (SND_NKE).

bReadInit: After restarting the PLC, the meter is read out once.

tMinSendTime: Standard t#2s. When this time exceeds, the counter is rereadout. At t#0s the counter is not readout and can be readout with bStart manually.

bDisabled: TRUE = disable the function block.

VAR_OUTPUT

```
bBusy           : BOOL;
bReady          : BOOL;
bError          : BOOL;
eError          : E_MBUS_ERROR;
dwIdNumber      : DWORD;
byStatus        : BYTE;
byGEN           : BYTE;
byCounter       : BYTE;
usiRecivedAdr   : USINT;
eMedium         : E_MBUS_Medium;
sMan            : STRING(3);
stVolume        : ST_MBus_Info;
stFlow          : ST_MBus_Info;
```

bBusy: The *bBusy* output is TRUE while the meter is being read.

bReady: The *bReady* output is TRUE for one cycle, once meter reading is completed.

bError: The *bError* output becomes TRUE as soon as an error occurs. The error is described via the variable *eError*.

eError: The *eError* output issues an error code when an error occurs (see E_MBUS_ERROR [▶ 216]).

dwIdNumber: Serial number of the meter (secondary address).

byStatus: Status of the meter. Please refer to device description for meanings.

byGEN: Meter software version.

byCounter: Transmission counter (number of transmitted RSP_UD).

usiRecivedAdr: Received primary address (0-250).

eMedium: Medium (see E_MBUS_Medium [▶ 219]).

sMan: Manufacturer code.

stVolume: Meter reading, volume (see ST_MBus_Info [▶ 222]).

stFlow: Current flow (see ST_MBus_Info [▶ 222]).

VAR_IN_OUT

```
stCom           : ST_MBUS_Communication;
```

stCom: About this structure, the block FB_MBUSKL6781() [▶ 29] is connected to the meter function blocks (see ST_MBUS_Communication [▶ 221]).

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055583243/.zip:

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055584651/.zip:

Controller configuration setting: "BCxx50 or BX serial"

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055586059/.zip:

BCxx00 must be set to "Large Model" in the PLC under "Project/Options/Controller Settings".

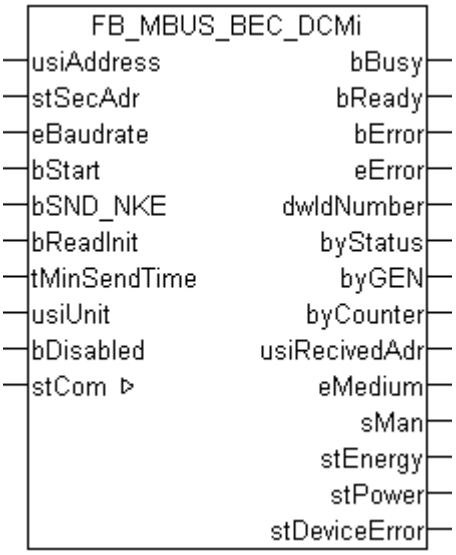
Controller configuration setting: "BC serial"

6.7 Berg

The function blocks only output a selection of the most common data. These data are described on the respective pages under "VAR_OUT". If more or all data are required, the function blocks FB_MBUS_General [▶ 31], FB_MBUS_General_Ext [▶ 35] or FB_MBUS_General_Param [▶ 41] from the General folder should be used. Not that these function blocks do not run on BC or BX systems. The function block FB_MBUS_General_Send [▶ 43] can be used to send data to the device (e.g. setting of the primary address).

Vendor	Type	Device	Function block
Berg	Electricity meter	DZ+	FB_MBUS_BEC_DZ [▶ 70]
	Electricity meter	DCMi	FB_MBUS_BEC_DCMi [▶ 68]

6.7.1 FB_MBUS_BEC_DCMi



This block is used for reading electricity meters from Berg:

-DCMi

The block can only be used in conjunction with the block FB_MBUSKL6781() [▶ 29].

Functionality of the function block [▶ 13]

VAR_INPUT

```
usiAddress      : USINT;
stSecAdr        : ST_MBUS_SecAdr;
eBaudrate       : E_MBUS_Baudrate := eMBUS_Baud2400;
bStart          : BOOL;
bSND_NKE        : BOOL := TRUE;
bReadInit       : BOOL := TRUE;
tMinSendTime    : TIME := t#2s;
usiUnit         : USINT;
bDisabled       : BOOL := FALSE;
```

usiAddress: Primary address [▶ 14] of the counter, that shall be readout with this module.

stSecAdr: Secondary address [▶ 15] of the counter, that shall be readout with this module.

eBaudrate: 300, 2400, 9600 baud [▶ 216].

bStart: Positive edge on this input, the meter is read out once.

bSND_NKE: TRUE to initialize the meter at each reading, and sets the meter on the first telegram (SND_NKE).

bReadInit: After restarting the PLC, the meter is read out once.

tMinSendTime: Standard t#2s. When this time exceeds, the counter is rereadout. At t#0s the counter is not readout and can be readout with bStart manually.

usiUnit: Unit of the energy values. 0=W/J(h) / 1=KW/KJ(h) / 2=MW/MJ(h) / 3=GW/GJ(h).

bDisabled: TRUE = disable the function block.

VAR_OUTPUT

```
bBusy           : BOOL;
bReady          : BOOL;
bError          : BOOL;
eError          : E_MBUS_ERROR;
dwIdNumber      : DWORD;
byStatus        : BYTE;
byGEN           : BYTE;
byCounter       : BYTE;
usiRecivedAdr   : USINT;
eMedium         : E_MBUS_Medium;
sMan            : STRING(3);
stEnergy        : ST_MBus_Info;
stPower         : ST_MBus_Info;
stDeviceError   : ST_MBus_Info;
```

bBusy: The *bBusy* output is TRUE while the meter is being read.

bReady: The *bReady* output is TRUE for one cycle, once meter reading is completed.

bError: The *bError* output becomes TRUE as soon as an error occurs. The error is described via the variable *eError*.

eError: The *eError* output issues an error code when an error occurs (see E_MBUS_ERROR [▶ 216]).

dwIdNumber: Serial number of the meter (secondary address).

byStatus: Status of the meter. Please refer to device description for meanings.

byGEN: Meter software version.

byCounter: Transmission counter (number of transmitted RSP_UD).

usiRecivedAdr: Received primary address (0-250).

eMedium: Medium (see E_MBUS_Medium [▶ 219]).

sMan: Manufacturer code.

stEnergy: Meter reading, energy consumption (see ST_MBus_Info [▶ 222]).

stPower: Current energy consumption (see ST_MBus_Info [▶ 222]).

stDeviceError: Error message from the device (see ST_MBus_Info [▶ 222]).

VAR_IN_OUT

```
stCom           : ST_MBUS_Communication;
```

stCom: About this structure, the block FB_MBUSKL6781() [▶ 29] is connected to the meter function blocks (see ST_MBUS_Communication [▶ 221]).

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055587467/.zip:

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055588875/.zip:

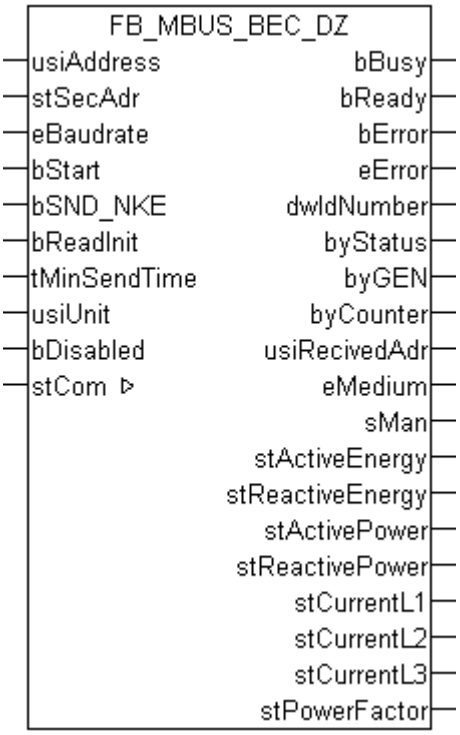
Controller configuration setting: "BCxx50 or BX serial"

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055590283/.zip:

BCxx00 must be set to "Large Model" in the PLC under "Project/Options/Controller Settings".

Controller configuration setting: "BC serial"

6.7.2 FB_MBUS_BEC_DZ



This function block is used to read electricity meters from Berg:

-DZ+

The function block can only be executed together with the function block FB_MBUSKL6781() [▶ 29].

Functionality of the function block [▶ 13]

VAR_INPUT

```
usiAddress          : USINT;
stSecAdr            : ST_MBUS_SecAdr;
eBaudrate           : E_MBUS_Baudrate := eMBUS_Baud2400;
bStart              : BOOL;
bSND_NKE            : BOOL := TRUE;
bReadInit           : BOOL := TRUE;
tMinSendTime        : TIME := t#2s;
usiUnit             : USINT;
bDisabled           : BOOL := FALSE;
```

usiAddress: Primary address [▶ 14] of the counter, that shall be readout with this module.

stSecAdr: Secondary address [▶ 15] of the counter, that shall be readout with this module.

eBaudrate: 300, 600, 1200, 2400, 4800, 9600 baud [▶ 216].

bStart: Positive edge on this input, the meter is read out once.

bSND_NKE: TRUE to initialize the meter at each reading, and sets the meter on the first telegram (SND_NKE).

bReadInit: After restarting the PLC, the meter is read out once.

tMinSendTime: Standard t#2s. When this time exceeds, the counter is rereadout. At t#0s the counter is not readout and can be readout with bStart manually.

usiUnit: Unit of the energy values. 0=W/J(h) / 1=KW/KJ(h) / 2=MW/MJ(h) / 3=GW/GJ(h).

bDisabled: TRUE = disable the function block.

VAR_OUTPUT

```
bBusy             : BOOL;
bReady            : BOOL;
bError            : BOOL;
eError            : E_MBUS_ERROR;
dwIdNumber        : DWORD;
byStatus          : BYTE;
byGEN             : BYTE;
byCounter         : BYTE;
usiRecivedAdr     : USINT;
eMedium           : E_MBUS_Medium;
sMan              : STRING(3);
stActiveEnergy    : ST_MBus_Info;
stReactiveEnergy  : ST_MBus_Info;
stActivePower     : ST_MBus_Info;
stReactivePower   : ST_MBus_Info;
stCurrentL1       : ST_MBus_Info;
stCurrentL2       : ST_MBus_Info;
stCurrentL3       : ST_MBus_Info;
stPowerFactor     : ST_MBus_Info;
```

bBusy: The *bBusy* output is TRUE while the meter is being read.

bReady: The *bReady* output is TRUE for one cycle, once meter reading is completed.

bError: The *bError* output becomes TRUE as soon as an error occurs. The error is described via the variable *eError*.

eError: The *eError* output issues an error code when an error occurs (see E_MBUS_ERROR [▶ 216]).

dwIdNumber: Serial number of the meter (secondary address).

byStatus: Status of the meter. Please refer to device description for meanings.

byGEN: Meter software version.

byCounter: Transmission counter (number of transmitted RSP_UD).

usiRecivedAdr: Received primary address (0-250).

eMedium: Medium (see E_MBUS_Medium [▶ 219]).

sMan: Manufacturer code.

stActiveEnergy: Meter reading, total active energy (see ST_MBus_Info [▶ 222]).

stReactiveEnergy: Meter reading, total reactive energy (see ST_MBus_Info [▶ 222]).

stActivePower: Current consumption, total active power (see ST_MBus_Info [▶ 222]).

stReactivePower: Current consumption, total reactive power (see ST_MBus_Info [▶ 222]).

stCurrentL1: Current L1 (see ST_MBus_Info [▶ 222]).

stCurrentL2: Current L2 (see ST_MBus_Info [▶ 222]).

stCurrentL3: Current L3 (see ST_MBus_Info [▶ 222]).

stPowerFactor: Total power factor (see ST_MBus_Info [▶ 222]).

VAR_IN_OUT

```
stCom          : ST_MBUS_Communication;
```

stCom: About this structure, the block FB_MBUSKL6781() [▶ 29] is connected to the meter function blocks (see ST_MBUS_Communication [▶ 221]).

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055587467/.zip:

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055588875/.zip:

Controller configuration setting: "BCxx50 or BX serial"

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055590283/.zip:

BCxx00 must be set to "Large Model" in the PLC under "Project/Options/Controller Settings".

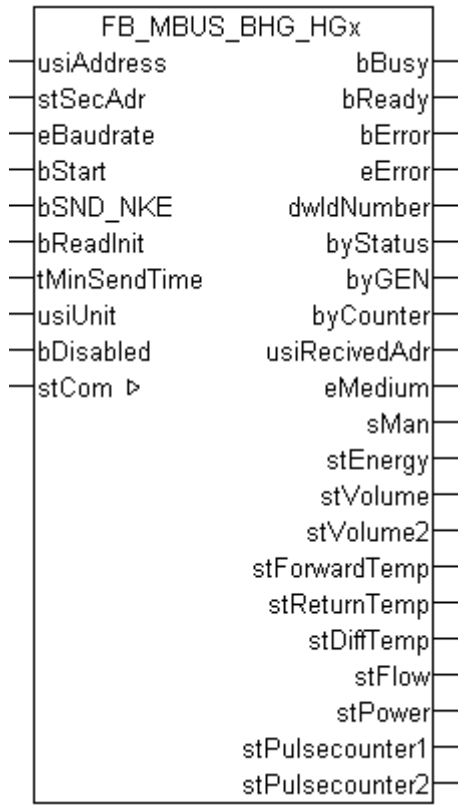
Controller configuration setting: "BC serial"

6.8 Brunata

The function blocks only output a selection of the most common data. These data are described on the respective pages under "VAR_OUT". If more or all data are required, the function blocks FB_MBUS_General [▶ 31], FB_MBUS_General_Ext [▶ 35] or FB_MBUS_General_Param [▶ 41] from the General folder should be used. Not that these function blocks do not run on BC or BX systems. The function block FB_MBUS_General_Send [▶ 43] can be used to send data to the device (e.g. setting of the primary address).

Vendor	Type	Device	Function block
Brunata	Heat meter	HGQ / HGS	FB_MBUS_BHG_HGx [▶ 73]
	Heat meter	Optuna H (775)	FB_MBUS_HYD_Sharky [▶ 117], FB_MBUS_HYD_Sharky_00 [▶ 120]

6.8.1 FB_MBUS_BHG_HGx



This block is used for reading heat meters from Brunata:

-HGQ

-HGS

The block can only be used in conjunction with the block FB_MBUSKL6781() [▶ 29].

Functionality of the function block [▶ 13]

VAR_INPUT

```
usiAddress        : USINT;
stSecAdr          : ST_MBUS_SecAdr;
eBaudrate         : E_MBUS_Baudrate := eMBUS_Baud2400;
bStart            : BOOL;
bSND_NKE          : BOOL := TRUE;
bReadInit         : BOOL := TRUE;
tMinSendTime      : TIME := t#2s;
usiUnit           : USINT;
bDisabled         : BOOL := FALSE;
```

usiAddress: Primary address [▶ 14] of the counter, that shall be readout with this module.

stSecAdr: Secondary address [▶ 15] of the counter, that shall be readout with this module.

eBaudrate: baud [▶ 216].

bStart: Positive edge on this input, the meter is read out once.

bSND_NKE: TRUE to initialize the meter at each reading, and sets the meter on the first telegram (SND_NKE).

bReadInit: After restarting the PLC, the meter is read out once.

tMinSendTime: Standard t#2s. When this time exceeds, the counter is rereadout. At t#0s the counter is not readout and can be readout with bStart manually.

usiUnit: Unit of the energy values. 0=W/J(h) / 1=KW/KJ(h) / 2=MW/MJ(h) / 3=GW/GJ(h).

bDisabled: TRUE = disable the function block.

VAR_OUTPUT

```
bBusy            : BOOL;
bReady           : BOOL;
bError           : BOOL;
eError           : E_MBUS_ERROR;
dwIdNumber       : DWORD;
byStatus         : BYTE;
byGEN            : BYTE;
byCounter        : BYTE;
usiRecivedAdr    : USINT;
eMedium          : E_MBUS_Medium;
sMan             : STRING(3);
stEnergy         : ST_MBus_Info;
stVolume         : ST_MBus_Info;
stVolume2        : ST_MBus_Info;
stForwardTemp    : ST_MBus_Info;
stReturnTemp     : ST_MBus_Info;
stDiffTemp       : ST_MBus_Info;
stFlow           : ST_MBus_Info;
stPower          : ST_MBus_Info;
stPulsecounter1  : ST_MBus_Info;
stPulsecounter2  : ST_MBus_Info;
```

bBusy: The *bBusy* output is TRUE while the meter is being read.

bReady: The *bReady* output is TRUE for one cycle, once meter reading is completed.

bError: The *bError* output becomes TRUE as soon as an error occurs. The error is described via the variable *eError*.

eError: The *eError* output issues an error code when an error occurs (see E_MBUS_ERROR [▶ 216]).

dwIdNumber: Serial number of the meter (secondary address).

byStatus: Status of the meter. Please refer to device description for meanings.

byGEN: Meter software version.

byCounter: Transmission counter (number of transmitted RSP_UD).

usiRecivedAdr: Received primary address (0-250).

eMedium: Medium (see E_MBUS_Medium [▶ 219]).

sMan: Manufacturer code.

stEnergy: Meter reading, energy consumption (see ST_MBus_Info [▶ 222]).

stVolume: Meter reading, volume (see ST_MBus_Info [▶ 222]).

stVolume2: Volume from flow sensor (see ST_MBus_Info [▶ 222]).

stForwardTemp: Flow temperature (see ST_MBus_Info [▶ 222]).

stReturnTemp: Return temperature (see ST_MBus_Info [▶ 222]).

stDiffTemp: Temperature difference (see ST_MBus_Info [▶ 222]).

stFlow: Current flow (see ST_MBus_Info [▶ 222]).

stPower: Current energy consumption (see ST_MBus_Info [▶ 222]).

stPulsecounter1: Pulse counter 1 (aux1) (see ST_MBus_Info [▶ 222]).

stPulsecounter2: Pulse counter 2 (aux2) (see ST_MBus_Info [▶ 222]).

VAR_IN_OUT

```
stCom        : ST_MBUS_Communication;
```

stCom: About this structure, the block FB_MBUSKL6781() [▶ 29] is connected to the meter function blocks (see ST_MBUS_Communication [▶ 221]).

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055591691/.zip:

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055593099/.zip:

Controller configuration setting: "BCxx50 or BX serial"

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055594507/.zip:

BCxx00 must be set to "Large Model" in the PLC under "Project/Options/Controller Settings".

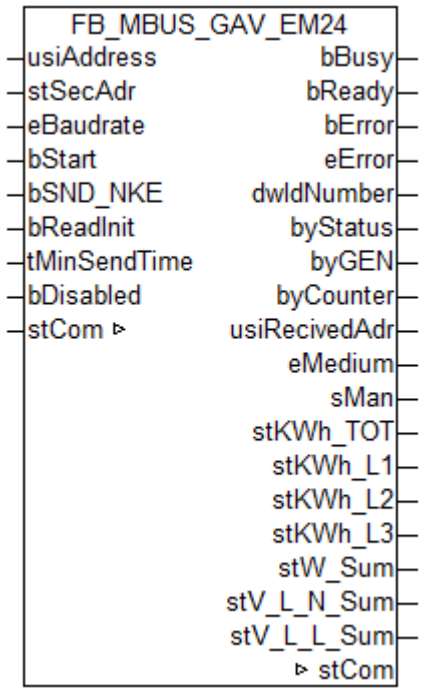
Controller configuration setting: "BC serial"

6.9 Carlo Gavazzi

The function blocks only output a selection of the most common data. These data are described on the respective pages under "VAR_OUT". If more or all data are required, the function blocks FB_MBUS_General [▶ 31], FB_MBUS_General_Ext [▶ 35] or FB_MBUS_General_Param [▶ 41] from the General folder should be used. Not that these function blocks do not run on BC or BX systems. The function block FB_MBUS_General_Send [▶ 43] can be used to send data to the device (e.g. setting of the primary address).

Vendor	Type	Device	Function block
Carlo Gavazzi	Energy meter	EM24	FB_MBUS_GAV_EM24 [▶ 75]

6.9.1 FB_MBUS_GAV_EM24



This module is used to readout energy calculators of the manufacturer Carlo Gavazzi.

It can only be used together with the module FB_MBUSKL6781() [▶ 29]

Functionality of the module [▶ 13]

VAR_INPUT

```
usiAddress          : USINT;
stSecAdr            : ST_MBUS_SecAdr;
eBaudrate           : E_MBUS_Baudrate := eMBUS_Baud2400;
bStart              : BOOL;
bSND_NKE            : BOOL := TRUE;
bReadInit           : BOOL := TRUE;
tMinSendTime        : TIME := t#2s;
bDisabled           : BOOL := FALSE;
```

usiAddress: Primary address [▶ 14] of the counter, that shall be readout with this module.

stSecAdr: Secondary address [▶ 15] of the counter, that shall be readout with this module.

eBaudrate: 300, 2400, 9600 baud [▶ 216].

bStart: Positive edge on this input, the meter is read out once.

bSND_NKE: TRUE to initialize the meter at each reading, and sets the meter on the first telegram (SND_NKE).

bReadInit: After restarting the PLC, the meter is read out once.

tMinSendTime: Standard t#2s. When this time exceeds, the counter is rereadout. At t#0s the counter is not readout and can be readout with bStart manually.

bDisabled: TRUE = disable the function block.

VAR_OUTPUT

```
bBusy           : BOOL;
bReady          : BOOL;
bError          : BOOL;
eError          : E_MBUS_ERROR;
dwIdNumber      : DWORD;
byStatus        : BYTE;
byGEN           : BYTE;
byCounter       : BYTE;
usiRecivedAdr   : USINT;
eMedium         : E_MBUS_Medium;
sMan            : STRING(3);
stKWh_TOT       : ST_MBus_Info;
stKWh_L1        : ST_MBus_Info;
stKWh_L2        : ST_MBus_Info;
stKWh_L3        : ST_MBus_Info;
stW_Sum         : ST_MBus_Info;
stV_L_N_Sum     : ST_MBus_Info;
stV_L_L_Sum     : ST_MBus_Info;
```

bBusy: The *bBusy* output is TRUE while the meter is being read.

bReady: The *bReady* output is TRUE for one cycle, once meter reading is completed.

bError: The *bError* output becomes TRUE as soon as an error occurs. The error is described via the variable *eError*.

eError: The *eError* output issues an error code when an error occurs (see E_MBUS_ERROR [▶ 216]).

dwIdNumber: Serial number of the meter (secondary address).

byStatus: Status of the meter. Please refer to device description for meanings.

byGEN: Meter software version.

byCounter: Transmission counter (number of transmitted RSP_UD).

usiRecivedAdr: Received primary address (0-250).

eMedium: Medium (see E_MBUS_Medium [▶ 219]).

sMan: Manufacturer code.

stKWh_TOT: See manufacturer information (see ST_MBus_Info [▶ 222]).

stKWh_L1: See manufacturer information (see ST_MBus_Info [▶ 222]).

stKWh_L2: See manufacturer information (see ST_MBus_Info [▶ 222]).

stKWh_L3: See manufacturer information (see ST_MBus_Info [▶ 222]).

stW_Sum: See manufacturer information (see ST_MBus_Info [▶ 222]).

stV_L_N_Sum: See manufacturer information (see ST_MBus_Info [▶ 222]).

stV_L_L_Sum: See manufacturer information (see ST_MBus_Info [▶ 222]).

VAR_IN_OUT

```
stCom           : ST_MBUS_Communication;
```

stCom: About this structure, the block FB_MBUSKL6781() [▶ 29] is connected to the meter function blocks (see ST_MBUS_Communication [▶ 221]).

Requirements

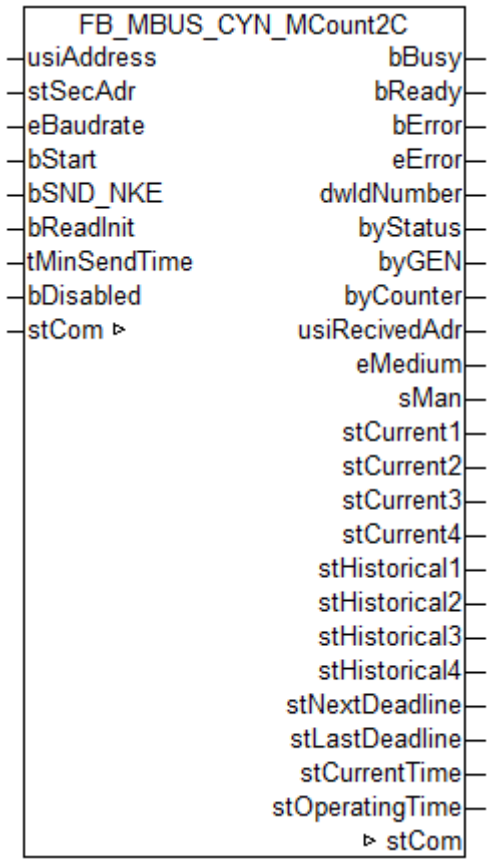
Development environment	Target system	Required libraries
TwinCAT 2.11 R3/x64 from build 2234	PC/CX, BX or BC	TcMBus library from V2.1.0

6.10 Cynox

The function blocks only output a selection of the most common data. These data are described on the respective pages under "VAR_OUT". If more or all data are required, the function blocks FB_MBUS_General [▶ 31], FB_MBUS_General_Ext [▶ 35] or FB_MBUS_General_Param [▶ 41] from the General folder should be used. Not that these function blocks do not run on BC or BX systems. The function block FB_MBUS_General_Send [▶ 43] can be used to send data to the device (e.g. setting of the primary address).

Vendor	Type	Device	Function block
Cynox	Pulse counter	MCount2C	FB_MBUS_CYN_MCount2C [▶ 78]

6.10.1 FB_MBUS_CYN_MCount2C



This module is used to readout pulse counter of the manufacturer Cynox.

It can only be used together with the module FB_MBUSKL6781() [▶ 29]

Functionality of the module [▶ 13]

VAR_INPUT

```
usiAddress          : USINT;
stSecAdr            : ST_MBUS_SecAdr;
eBaudrate           : E_MBUS_Baudrate := eMBUS_Baud2400;
bStart              : BOOL;
bSND_NKE            : BOOL := TRUE;
bReadInit           : BOOL := TRUE;
tMinSendTime        : TIME := t#2s;
bDisabled           : BOOL := FALSE;
```

usiAddress: Primary address [▶ 14] of the counter, that shall be readout with this module.

stSecAdr: Secondary address [▶ 15] of the counter, that shall be readout with this module.

eBaudrate: 2400 baud [▶ 216].

bStart: Positive edge on this input, the meter is read out once.

bSND_NKE: TRUE to initialize the meter at each reading, and sets the meter on the first telegram (SND_NKE).

bReadInit: After restarting the PLC, the meter is read out once.

tMinSendTime: Standard t#2s. When this time exceeds, the counter is rereadout. At t#0s the counter is not readout and can be readout with bStart manually.

bDisabled: TRUE = disable the function block.

VAR_OUTPUT

```
    bBusy           : BOOL;
bReady              : BOOL;
bError              : BOOL;
eError              : E_MBUS_ERROR;
dwIdNumber          : DWORD;
byStatus            : BYTE;
byGEN               : BYTE;
byCounter           : BYTE;
usiRecivedAdr       : USINT;
eMedium             : E_MBUS_Medium;
sMan                : STRING(3);
stCurrent1          : ST_MBus_Info;
stCurrent2          : ST_MBus_Info;
stCurrent3          : ST_MBus_Info;
stCurrent4          : ST_MBus_Info;
stHistorical1       : ST_MBus_Info;
stHistorical2       : ST_MBus_Info;
stHistorical3       : ST_MBus_Info;
stHistorical4       : ST_MBus_Info;
stNextDeadline      : ST_MBus_Info;
stLastDeadline      : ST_MBus_Info;
stCurrentTime       : ST_MBus_Info;
stOperatingTime     : ST_MBus_Info;
```

bBusy: The *bBusy* output is TRUE while the meter is being read.

bReady: The *bReady* output is TRUE for one cycle, once meter reading is completed.

bError: The *bError* output becomes TRUE as soon as an error occurs. The error is described via the variable *eError*.

eError: The *eError* output issues an error code when an error occurs (see E_MBUS_ERROR [▶ 216]).

dwIdNumber: Serial number of the meter (secondary address).

byStatus: Status of the meter. Please refer to device description for meanings.

byGEN: Meter software version.

byCounter: Transmission counter (number of transmitted RSP_UD).

usiRecivedAdr: Received primary address (0-250).

eMedium: Medium (see E_MBUS_Medium [▶ 219]).

sMan: Manufacturer code.

stCurrent1: Current value 1 (see ST_MBus_Info [▶ 222]).

stCurrent2: Current value 2 (see ST_MBus_Info [▶ 222]).

stCurrent3: Current value 3 (see ST_MBus_Info [▶ 222]).

stCurrent4: Current value 4 (see ST_MBus_Info [▶ 222]).

stHistorical1: Historical value 1 (see ST_MBus_Info [▶ 222]).

stHistorical2: Historical value 2 (see ST_MBus_Info [▶ 222]).

stHistorical3: Historical value 3 (see ST_MBus_Info [▶ 222]).

stHistorical4: Historical value 4 (see ST_MBus_Info [▶ 222]).

stNextDeadline: Next deadline (see ST_MBus_Info [▶ 222]).

stLastDeadline: Last deadline (see ST_MBus_Info [▶ 222]).

stCurrentTime: Current time (see ST_MBus_Info [▶ 222]).

stOperatingTime: Operating time (see ST_MBus_Info [▶ 222]).

VAR_IN_OUT

```
stCom           : ST_MBUS_Communication;
```

stCom: About this structure, the block FB_MBUSKL6781() [▶ 29] is connected to the meter function blocks (see ST_MBUS_Communication [▶ 221]).

Requirements

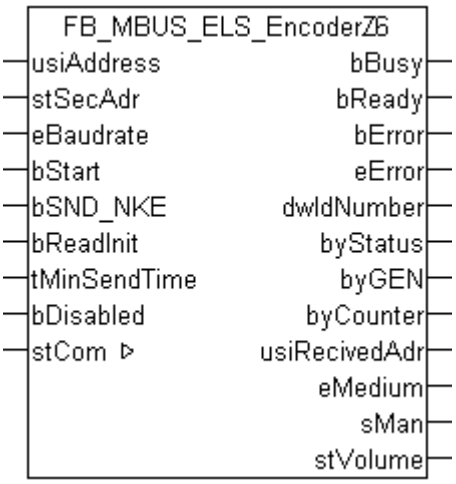
Development environment	Target system	Required libraries
TwinCAT 2.11 R3/x64 from build 2250	PC/CX, BX or BC	TcMBus library from V2.5.0

6.11 Elster

The function blocks only output a selection of the most common data. These data are described on the respective pages under "VAR_OUT". If more or all data are required, the function blocks FB_MBUS_General [▶ 31], FB_MBUS_General_Ext [▶ 35] or FB_MBUS_General_Param [▶ 41] from the General folder should be used. Not that these function blocks do not run on BC or BX systems. The function block FB_MBUS_General_Send [▶ 43] can be used to send data to the device (e.g. setting of the primary address).

Vendor	Type	Device	Function block
Elster	Gas meter	Encoder Z6	FB_MBUS_ELS_EncoderZ6 [▶ 80]

6.11.1 FB_MBUS_ELS_EncoderZ6



This block is used for reading meters from Elster:

-Gas meter Encoder Z6

The block can only be used in conjunction with the block FB_MBUSKL6781() [▶ 29].

Functionality of the function block [▶ 13]

VAR_INPUT

```
usiAddress      : USINT;
stSecAdr        : ST_MBUS_SecAdr;
eBaudrate       : E_MBUS_Baudrate := eMBUS_Baud2400;
bStart          : BOOL;
bSND_NKE        : BOOL := TRUE;
```

```
bReadInit       : BOOL := TRUE;
tMinSendTime    : TIME := t#2s;
bDisabled       : BOOL := FALSE;
```

usiAddress: Primary address [▶ 14] of the counter, that shall be readout with this module.

stSecAdr: Secondary address [▶ 15] of the counter, that shall be readout with this module.

eBaudrate: 300, 2400 baud [▶ 216].

bStart: Positive edge on this input, the meter is read out once.

bSND_NKE: TRUE to initialize the meter at each reading, and sets the meter on the first telegram (SND_NKE).

bReadInit: After restarting the PLC, the meter is read out once.

tMinSendTime: Standard t#2s. When this time exceeds, the counter is rereadout. At t#0s the counter is not readout and can be readout with bStart manually.

bDisabled: TRUE = disable the function block.

VAR_OUTPUT

```
bBusy           : BOOL;
bReady          : BOOL;
bError          : BOOL;
eError          : E_MBUS_ERROR;
dwIdNumber      : DWORD;
byStatus        : BYTE;
byGEN           : BYTE;
byCounter       : BYTE;
usiRecivedAdr   : USINT;
eMedium         : E_MBUS_Medium;
sMan            : STRING(3);
stVolume        : ST_MBus_Info;
```

bBusy: The *bBusy* output is TRUE while the meter is being read.

bReady: The *bReady* output is TRUE for one cycle, once meter reading is completed.

bError: The *bError* output becomes TRUE as soon as an error occurs. The error is described via the variable *eError*.

eError: The *eError* output issues an error code when an error occurs (see E_MBUS_ERROR [▶ 216]).

dwIdNumber: Serial number of the meter (secondary address).

byStatus: Status of the meter. Please refer to device description for meanings.

byGEN: Meter software version.

byCounter: Transmission counter (number of transmitted RSP_UD).

usiRecivedAdr: Received primary address (0-250).

eMedium: Medium (see E_MBUS_Medium [▶ 219]).

sMan: Manufacturer code.

stVolume: Meter reading, volume (see ST_MBus_Info [▶ 222]).

VAR_IN_OUT

```
stCom           : ST_MBUS_Communication;
```

stCom: About this structure, the block FB_MBUSKL6781() [▶ 29] is connected to the meter function blocks (see ST_MBUS_Communication [▶ 221]).

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055595915/.zip: 

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055597323/.zip:

Controller configuration setting: "BCxx50 or BX serial"

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055598731/.zip:

BCxx00 must be set to "Large Model" in the PLC under "Project/Options/Controller Settings".

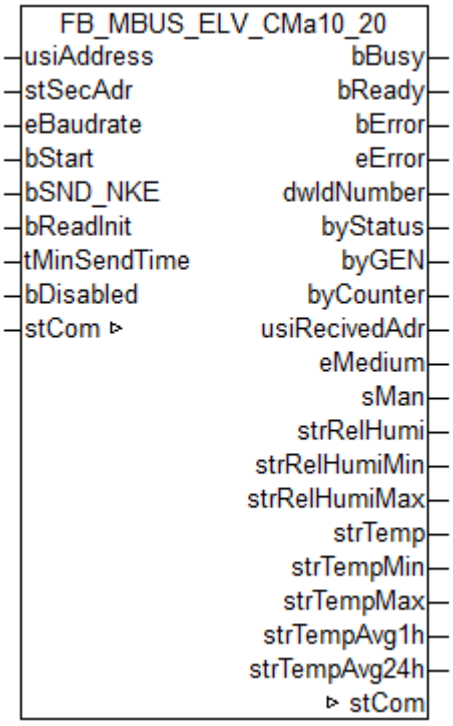
Controller configuration setting: "BC serial"

6.12 elvaco

The function blocks only output a selection of the most common data. These data are described on the respective pages under "VAR_OUT". If more or all data are required, the function blocks FB_MBUS_General [▶ 31], FB_MBUS_General_Ext [▶ 35] or FB_MBUS_General_Param [▶ 41] from the General folder should be used. Not that these function blocks do not run on BC or BX systems. The function block FB_MBUS_General_Send [▶ 43] can be used to send data to the device (e.g. setting of the primary address).

Vendor	Type	Device	Function block
elvaco	Temperature and humidity sensors	CMa10 / CMa20	FB_MBUS_ELV_CMa10_20 [▶ 82]

6.12.1 FB_MBUS_ELV_CMa10_20



This module is used to readout energy calculators of the manufacturer elvaco.

Usable with sensors CMa10 and CMa20.

It can only be used together with the module FB_MBUSKL6781() [▶ 29]

Functionality of the module [▶ 13]

VAR_INPUT

```
usiAddress      : USINT;
stSecAdr        : ST_MBUS_SecAdr;
eBaudrate       : E_MBUS_Baudrate := eMBUS_Baud2400;
bStart          : BOOL;
bSND_NKE        : BOOL := TRUE;
bReadInit       : BOOL := TRUE;
tMinSendTime    : TIME := t#2s;
bDisabled       : BOOL := FALSE;
```

usiAddress: Primary address [▶ 14] of the counter, that shall be readout with this module.

stSecAdr: Secondary address [▶ 15] of the counter, that shall be readout with this module.

eBaudrate: 300, 2400, 9600 baud [▶ 216].

bStart: Positive edge on this input, the meter is read out once.

bSND_NKE: TRUE to initialize the meter at each reading, and sets the meter on the first telegram (SND_NKE).

bReadInit: After restarting the PLC, the meter is read out once.

tMinSendTime: Standard t#2s. When this time exceeds, the counter is rereadout. At t#0s the counter is not readout and can be readout with bStart manually.

bDisabled: TRUE = disable the function block.

VAR_OUTPUT

```
bBusy           : BOOL;
bReady          : BOOL;
bError          : BOOL;
eError          : E_MBUS_ERROR;
dwIdNumber      : DWORD;
byStatus        : BYTE;
byGEN           : BYTE;
byCounter       : BYTE;
usiRecivedAdr   : USINT;
eMedium         : E_MBUS_Medium;
sMan            : STRING(3);
strRelHumi      : ST_MBus_Info;
strRelHumiMin   : ST_MBus_Info;
strRelHumiMax   : ST_MBus_Info;
strTemp         : ST_MBus_Info;
strTempMin      : ST_MBus_Info;
strTempMax      : ST_MBus_Info;
strTempAvg1h    : ST_MBus_Info;
strTempAvg24h   : ST_MBus_Info;
```

bBusy: The *bBusy* output is TRUE while the meter is being read.

bReady: The *bReady* output is TRUE for one cycle, once meter reading is completed.

bError: The *bError* output becomes TRUE as soon as an error occurs. The error is described via the variable *eError*.

eError: The *eError* output issues an error code when an error occurs (see E_MBUS_ERROR [▶ 216]).

dwIdNumber: Serial number of the meter (secondary address).

byStatus: Status of the meter. Please refer to device description for meanings.

byGEN: Meter software version.

byCounter: Transmission counter (number of transmitted RSP_UD).

usiRecivedAdr: Received primary address (0-250).

eMedium: Medium (see E_MBUS_Medium [▶ 219]).

sMan: Manufacturer code.

strRelHumi: Instantaneous relative humidity (see ST_MBus_Info [▶ 222]).

strRelHumiMin: Lowest instantaneous relative humidity since last min/max reset command (see ST_MBus_Info [▶ 222]).

strRelHumiMax: Highest instantaneous relative humidity since last min/max reset command (see ST_MBus_Info [▶ 222]).

strTemp: Instantaneous temperature (see ST_MBus_Info [▶ 222]).

strTempMin: Lowest instantaneous temperature since last min/max reset command (see ST_MBus_Info [▶ 222]).

strTempMax: Highest instantaneous temperature since last min/max reset command (see ST_MBus_Info [▶ 222]).

strTempAvg1h: 1-hour temperature rolling average (see ST_MBus_Info [▶ 222]).

strTempAvg24h: 24-hour temperature rolling average (see ST_MBus_Info [▶ 222]).

VAR_IN_OUT

```
stCom           : ST_MBUS_Communication;
```

stCom: About this structure, the block FB_MBUSKL6781() [▶ 29] is connected to the meter function blocks (see ST_MBUS_Communication [▶ 221]).

Requirements

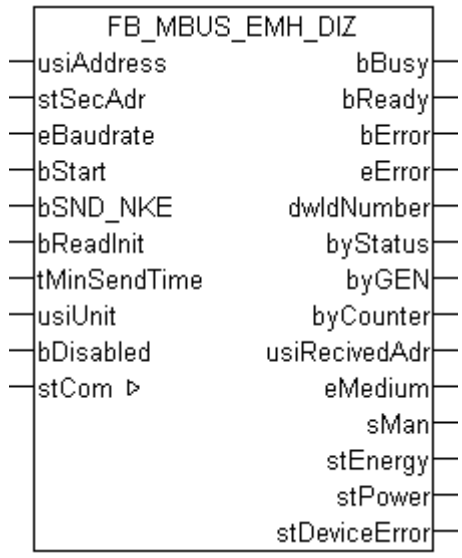
Development environment	Target system	Required libraries
TwinCAT 2.11 R3/x64 from build 2234	PC/CX, BX or BC	TcMBus library from V2.2.0

6.13 EMH

The function blocks only output a selection of the most common data. These data are described on the respective pages under "VAR_OUT". If more or all data are required, the function blocks FB_MBUS_General [▶ 31], FB_MBUS_General_Ext [▶ 35] or FB_MBUS_General_Param [▶ 41] from the General folder should be used. Not that these function blocks do not run on BC or BX systems. The function block FB_MBUS_General_Send [▶ 43] can be used to send data to the device (e.g. setting of the primary address).

Vendor	Type	Device	Function block
EMH	Electricity meter	DIZ	FB_MBUS_EMH_DIZ [▶ 85]
	Electricity meter	EIZ-E	FB_MBUS_EMH_EIZE [▶ 87]
	Electricity meter	EIZ-G	FB_MBUS_EMH_EIZG [▶ 89]
	Electricity meter	MIZ	FB_MBUS_EMH_MIZ [▶ 91]

6.13.1 FB_MBUS_EMH_DIZ



This block is used for reading electricity meters from EMH:

-DIZ

Unidirectional tariff meter only.

The block can only be used in conjunction with the block FB_MBUSKL6781() [▶ 29].

Functionality of the function block [▶ 13]

VAR_INPUT

```
usiAddress      : USINT;
stSecAdr        : ST_MBUS_SecAdr;
eBaudrate       : E_MBUS_Baudrate := eMBUS_Baud2400;
bStart          : BOOL;
bSND_NKE        : BOOL := TRUE;
bReadInit       : BOOL := TRUE;
tMinSendTime    : TIME := t#2s;
usiUnit         : USINT;
bDisabled       : BOOL := FALSE;
```

usiAddress: Primary address [▶ 14] of the counter, that shall be readout with this module.

stSecAdr: Secondary address [▶ 15] of the counter, that shall be readout with this module.

eBaudrate: 300, 2400, 9600 baud [▶ 216].

bStart: Positive edge on this input, the meter is read out once.

bSND_NKE: TRUE to initialize the meter at each reading, and sets the meter on the first telegram (SND_NKE).

bReadInit: After restarting the PLC, the meter is read out once.

tMinSendTime: Standard t#2s. When this time exceeds, the counter is rereadout. At t#0s the counter is not readout and can be readout with bStart manually.

usiUnit: Unit of the energy values. 0=W/J(h) / 1=KW/KJ(h) / 2=MW/MJ(h) / 3=GW/GJ(h).

bDisabled: TRUE = disable the function block.

VAR_OUTPUT

```
bBusy           : BOOL;
bReady          : BOOL;
bError          : BOOL;
```

```
eError          : E_MBUS_ERROR;
dwIdNumber      : DWORD;
byStatus        : BYTE;
byGEN           : BYTE;
byCounter       : BYTE;
usiRecivedAdr   : USINT;
eMedium         : E_MBUS_Medium;
sMan            : STRING(3);
stEnergy        : ST_MBus_Info;
stPower         : ST_MBus_Info;
stDeviceError   : ST_MBus_Info;
```

bBusy: The *bBusy* output is TRUE while the meter is being read.

bReady: The *bReady* output is TRUE for one cycle, once meter reading is completed.

bError: The *bError* output becomes TRUE as soon as an error occurs. The error is described via the variable *eError*.

eError: The *eError* output issues an error code when an error occurs (see E_MBUS_ERROR [▶ 216]).

dwIdNumber: Serial number of the meter (secondary address).

byStatus: Status of the meter. Please refer to device description for meanings.

byGEN: Meter software version.

byCounter: Transmission counter (number of transmitted RSP_UD).

usiRecivedAdr: Received primary address (0-250).

eMedium: Medium (see E_MBUS_Medium [▶ 219]).

sMan: Manufacturer code.

stEnergy: Meter reading, energy consumption (see ST_MBus_Info [▶ 222]).

stPower: Current energy consumption (see ST_MBus_Info [▶ 222]).

stDeviceError: Error message from the device (see ST_MBus_Info [▶ 222]).

VAR_IN_OUT

```
stCom           : ST_MBUS_Communication;
```

stCom: About this structure, the block FB_MBUSKL6781() [▶ 29] is connected to the meter function blocks (see ST_MBUS_Communication [▶ 221]).

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055600139/.zip:

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055601547/.zip:

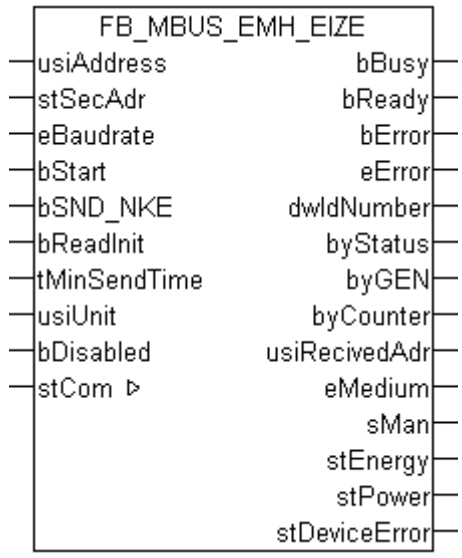
Controller configuration setting: "BCxx50 or BX serial"

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055602955/.zip:

BCxx00 must be set to "Large Model" in the PLC under "Project/Options/Controller Settings".

Controller configuration setting: "BC serial"

6.13.2 FB_MBUS_EMH_EIZE



This function block is used to read electricity meters from EMH:

-EIZ-E

The function block can only be executed together with the function block FB_MBUSKL6781() [▶ 29].

Functionality of the function block [▶ 13]

VAR_INPUT

```
usiAddress      : USINT;
stSecAdr        : ST_MBUS_SecAdr;
eBaudrate       : E_MBUS_Baudrate := eMBUS_Baud2400;
bStart          : BOOL;
bSND_NKE        : BOOL := TRUE;
bReadInit       : BOOL := TRUE;
tMinSendTime    : TIME := t#2s;
usiUnit         : USINT;
bDisabled       : BOOL := FALSE;
```

usiAddress: Primary address [▶ 14] of the counter, that shall be readout with this module.

stSecAdr: Secondary address [▶ 15] of the counter, that shall be readout with this module.

eBaudrate: 300, 2400, 9600 baud [▶ 216].

bStart: Positive edge on this input, the meter is read out once.

bSND_NKE: TRUE to initialize the meter at each reading, and sets the meter on the first telegram (SND_NKE).

bReadInit: After restarting the PLC, the meter is read out once.

tMinSendTime: Standard t#2s. When this time exceeds, the counter is rereadout. At t#0s the counter is not readout and can be readout with bStart manually.

usiUnit: Unit of the energy values. 0=W/J(h) / 1=KW/KJ(h) / 2=MW/MJ(h) / 3=GW/GJ(h).

bDisabled: TRUE = disable the function block.

VAR_OUTPUT

```
bBusy           : BOOL;
bReady          : BOOL;
bError          : BOOL;
eError          : E_MBUS_ERROR;
dwIdNumber      : DWORD;
```

```
byStatus        : BYTE;
byGEN           : BYTE;
byCounter       : BYTE;
usiRecivedAdr   : USINT;
eMedium         : E_MBUS_Medium;
sMan            : STRING(3);
stEnergy        : ST_MBus_Info;
stPower         : ST_MBus_Info;
stDeviceError   : ST_MBus_Info;
```

bBusy: The *bBusy* output is TRUE while the meter is being read.

bReady: The *bReady* output is TRUE for one cycle, once meter reading is completed.

bError: The *bError* output becomes TRUE as soon as an error occurs. The error is described via the variable *eError*.

eError: The *eError* output issues an error code when an error occurs (see E_MBUS_ERROR [▶ 216]).

dwIdNumber: Serial number of the meter (secondary address).

byStatus: Status of the meter. Please refer to device description for meanings.

byGEN: Meter software version.

byCounter: Transmission counter (number of transmitted RSP_UD).

usiRecivedAdr: Received primary address (0-250).

eMedium: Medium (see E_MBUS_Medium [▶ 219]).

sMan: Manufacturer code.

stEnergy: Meter reading, energy consumption (see ST_MBus_Info [▶ 222]).

stPower: Current energy consumption (see ST_MBus_Info [▶ 222]).

stDeviceError: Error message from the device (see ST_MBus_Info [▶ 222]).

VAR_IN_OUT

```
stCom           : ST_MBUS_Communication;
```

stCom: About this structure, the block FB_MBUSKL6781() [▶ 29] is connected to the meter function blocks (see ST_MBUS_Communication [▶ 221]).

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055600139/.zip: 

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055601547/.zip: 

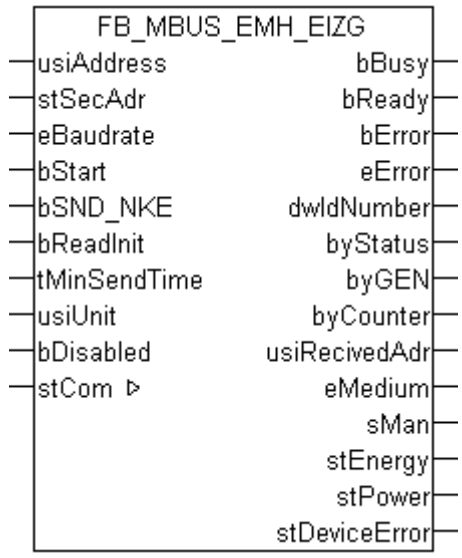
Controller configuration setting: "BCxx50 or BX serial"

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055602955/.zip: 

BCxx00 must be set to "Large Model" in the PLC under "Project/Options/Controller Settings".

Controller configuration setting: "BC serial"

6.13.3 FB_MBUS_EMH_EIZG



This function block is used to read electricity meters from EMH:

-EIZ-G

The function block can only be executed together with the function block FB_MBUSKL6781() [▶ 29].

Functionality of the function block [▶ 13]

VAR_INPUT

```
usiAddress      : USINT;
stSecAdr        : ST_MBUS_SecAdr;
eBaudrate       : E_MBUS_Baudrate := eMBUS_Baud2400;
bStart          : BOOL;
bSND_NKE        : BOOL := TRUE;
bReadInit       : BOOL := TRUE;
tMinSendTime    : TIME := t#2s;
usiUnit         : USINT;
bDisabled       : BOOL := FALSE;
```

usiAddress: Primary address [▶ 14] of the counter, that shall be readout with this module.

stSecAdr: Secondary address [▶ 15] of the counter, that shall be readout with this module.

eBaudrate: 300, 2400, 9600 baud [▶ 216].

bStart: Positive edge on this input, the meter is read out once.

bSND_NKE: TRUE to initialize the meter at each reading, and sets the meter on the first telegram (SND_NKE).

bReadInit: After restarting the PLC, the meter is read out once.

tMinSendTime: Standard t#2s. When this time exceeds, the counter is rereadout. At t#0s the counter is not readout and can be readout with bStart manually.

usiUnit: Unit of the energy values. 0=W/J(h) / 1=KW/KJ(h) / 2=MW/MJ(h) / 3=GW/GJ(h).

bDisabled: TRUE = disable the function block.

VAR_OUTPUT

```
bBusy           : BOOL;
bReady          : BOOL;
bError          : BOOL;
eError          : E_MBUS_ERROR;
dwIdNumber      : DWORD;
```

```
byStatus        : BYTE;
byGEN           : BYTE;
byCounter       : BYTE;
usiRecivedAdr   : USINT;
eMedium         : E_MBUS_Medium;
sMan            : STRING(3);
stEnergy        : ST_MBus_Info;
stPower         : ST_MBus_Info;
stDeviceError   : ST_MBus_Info;
```

bBusy: The *bBusy* output is TRUE while the meter is being read.

bReady: The *bReady* output is TRUE for one cycle, once meter reading is completed.

bError: The *bError* output becomes TRUE as soon as an error occurs. The error is described via the variable *eError*.

eError: The *eError* output issues an error code when an error occurs (see E_MBUS_ERROR [▶ 216]).

dwIdNumber: Serial number of the meter (secondary address).

byStatus: Status of the meter. Please refer to device description for meanings.

byGEN: Meter software version.

byCounter: Transmission counter (number of transmitted RSP_UD).

usiRecivedAdr: Received primary address (0-250).

eMedium: Medium (see E_MBUS_Medium [▶ 219]).

sMan: Manufacturer code.

stEnergy: Meter reading, energy consumption (see ST_MBus_Info [▶ 222]).

stPower: Current energy consumption (see ST_MBus_Info [▶ 222]).

stDeviceError: Error message from the device (see ST_MBus_Info [▶ 222]).

VAR_IN_OUT

```
stCom           : ST_MBUS_Communication;
```

stCom: About this structure, the block FB_MBUSKL6781() [▶ 29] is connected to the meter function blocks (see ST_MBUS_Communication [▶ 221]).

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055600139/.zip: 

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055601547/.zip: 

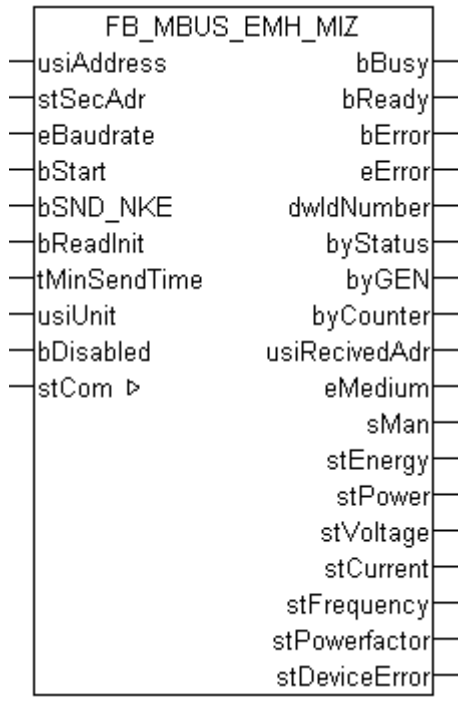
Controller configuration setting: "BCxx50 or BX serial"

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055602955/.zip: 

BCxx00 must be set to "Large Model" in the PLC under "Project/Options/Controller Settings".

Controller configuration setting: "BC serial"

6.13.4 FB_MBUS_EMH_MIZ



This function block is used to read electricity meters from EMH:

-MIZ

The function block can only be executed together with the function block FB_MBUSKL6781() [▶ 29].

Functionality of the function block [▶ 13]

VAR_INPUT

```
usiAddress      : USINT;
stSecAdr        : ST_MBUS_SecAdr;
eBaudrate       : E_MBUS_Baudrate := eMBUS_Baud2400;
bStart          : BOOL;
bSND_NKE        : BOOL := TRUE;
bReadInit       : BOOL := TRUE;
tMinSendTime    : TIME := t#2s;
usiUnit         : USINT;
bDisabled       : BOOL := FALSE;
```

usiAddress: Primary address [▶ 14] of the counter, that shall be readout with this module.

stSecAdr: Secondary address [▶ 15] of the counter, that shall be readout with this module.

eBaudrate: 300, 2400, 9600 baud [▶ 216].

bStart: Positive edge on this input, the meter is read out once.

bSND_NKE: TRUE to initialize the meter at each reading, and sets the meter on the first telegram (SND_NKE).

bReadInit: After restarting the PLC, the meter is read out once.

tMinSendTime: Standard t#2s. When this time exceeds, the counter is rereadout. At t#0s the counter is not readout and can be readout with bStart manually.

usiUnit: Unit of the energy values. 0=W/J(h) / 1=KW/KJ(h) / 2=MW/MJ(h) / 3=GW/GJ(h).

bDisabled: TRUE = disable the function block.

VAR_OUTPUT

```
bBusy           : BOOL;
bReady          : BOOL;
bError          : BOOL;
eError          : E_MBUS_ERROR;
dwIdNumber      : DWORD;
byStatus        : BYTE;
byGEN           : BYTE;
byCounter       : BYTE;
usiRecivedAdr   : USINT;
eMedium         : E_MBUS_Medium;
sMan            : STRING(3);
stEnergy        : ST_MBus_Info;
stPower         : ST_MBus_Info;
stVoltage       : ST_MBus_Info;
stCurrent       : ST_MBus_Info;
stFrequency     : ST_MBus_Info;
stPowerfactor   : ST_MBus_Info;
stDeviceError   : ST_MBus_Info;
```

bBusy: The *bBusy* output is TRUE while the meter is being read.

bReady: The *bReady* output is TRUE for one cycle, once meter reading is completed.

bError: The *bError* output becomes TRUE as soon as an error occurs. The error is described via the variable *eError*.

eError: The *eError* output issues an error code when an error occurs (see E_MBUS_ERROR [▶ 216]).

dwIdNumber: Serial number of the meter (secondary address).

byStatus: Status of the meter. Please refer to device description for meanings.

byGEN: Meter software version.

byCounter: Transmission counter (number of transmitted RSP_UD).

usiRecivedAdr: Received primary address (0-250).

eMedium: Medium (see E_MBUS_Medium [▶ 219]).

sMan: Manufacturer code.

stEnergy: Meter reading, energy consumption (see ST_MBus_Info [▶ 222]).

stPower: Current energy consumption (see ST_MBus_Info [▶ 222]).

stVoltage: Voltage (see ST_MBus_Info [▶ 222]).

stCurrent: Current (see ST_MBus_Info [▶ 222]).

stFrequency: Frequeny (see ST_MBus_Info [▶ 222]).

stPowerfactor: Power factor (see ST_MBus_Info [▶ 222]).

stDeviceError: Error message from the device (see ST_MBus_Info [▶ 222]).

VAR_IN_OUT

```
stCom           : ST_MBUS_Communication;
```

stCom: About this structure, the block FB_MBUSKL6781() [▶ 29] is connected to the meter function blocks (see ST_MBUS_Communication [▶ 221]).

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055600139/.zip:

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055601547/.zip:

Controller configuration setting: "BCxx50 or BX serial"

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055602955/.zip:

BCxx00 must be set to "Large Model" in the PLC under "Project/Options/Controller Settings".

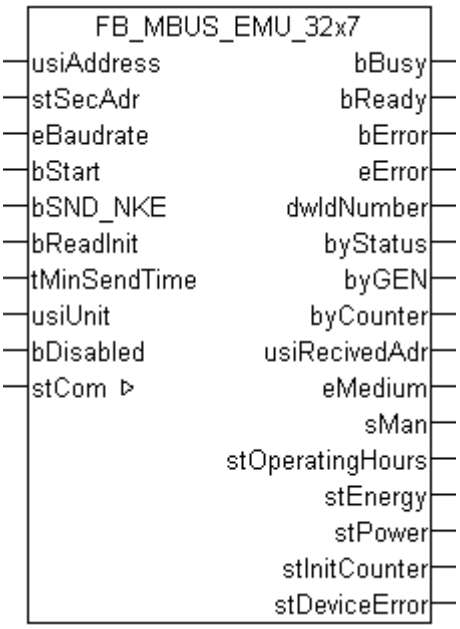
Controller configuration setting: "BC serial"

6.14 EMU

The function blocks only output a selection of the most common data. These data are described on the respective pages under "VAR_OUT". If more or all data are required, the function blocks FB_MBUS_General [▶ 31], FB_MBUS_General_Ext [▶ 35] or FB_MBUS_General_Param [▶ 41] from the General folder should be used. Not that these function blocks do not run on BC or BX systems. The function block FB_MBUS_General_Send [▶ 43] can be used to send data to the device (e.g. setting of the primary address).

Vendor	Type	Device	Function block
EMU	Electricity meter	EMU32x7	FB_MBUS_EMU_32x7 [▶ 93]
	Electricity meter	EMU32x7	FB_MBUS_EMU_32x7_Option8 [▶ 96]
	Electricity meter	Allrounder 3/5	FB_MBUS_EMU_3_5_Allrounder [▶ 99]
	Electricity meter	DHZ 5/63	FB_MBUS_EMU_DHZ_5_63 [▶ 102]

6.14.1 FB_MBUS_EMU_32x7



This function block is used to read electricity meters from EMU:

-EMU32.x7

Only the standard meter data will be read. The meter transmits this data in the standard EMU parameterization, or if the parameter set is set in the device to 00000 hexadecimal. Please refer to the meter documentation for further information regarding this.

In the normal version, the current consumption of the M-Bus interface is equivalent to 3 standard loads. If an M-Bus master interface is used that is designed, for example, for up to 120 standard loads, a maximum of 40 EMU M-Bus meters can be connected. The meter can optionally be supplied with 230 V. The current consumption of the M-Bus interface is then equivalent to one standard load.

The transmission of data from the EMU meter to the M-Bus protocol computer only works if the EMU meter is connected to at least two phases of the mains voltage network.

The EMU meter transmits current data to the device's M-Bus interface every 40 seconds, so that the readout data is approx. 40 - 45 seconds old.

The function block can only be executed together with the function block FB_MBUSKL6781() [▶ 29].

Functionality of the function block [▶ 13]

VAR_INPUT

```
usiAddress          : USINT;
stSecAdr            : ST_MBUS_SecAdr;
eBaudrate           : E_MBUS_Baudrate := eMBUS_Baud2400;
bStart              : BOOL;
bSND_NKE            : BOOL := TRUE;
bReadInit           : BOOL := TRUE;
tMinSendTime        : TIME := t#2s;
usiUnit             : USINT;
bDisabled           : BOOL := FALSE;
```

usiAddress: Primary address [▶ 14] of the counter, that shall be readout with this module.

stSecAdr: Secondary address [▶ 15] of the counter, that shall be readout with this module.

eBaudrate: 300, 600, 1200, 2400, 4800, 9600 baud [▶ 216].

bStart: Positive edge on this input, the meter is read out once.

bSND_NKE: TRUE to initialize the meter at each reading, and sets the meter on the first telegram (SND_NKE).

bReadInit: After restarting the PLC, the meter is read out once.

tMinSendTime: Standard t#2s. When this time exceeds, the counter is rereadout. At t#0s the counter is not readout and can be readout with bStart manually.

usiUnit: Unit of the energy values. 0=W/J(h) / 1=KW/KJ(h) / 2=MW/MJ(h) / 3=GW/GJ(h).

bDisabled: TRUE = disable the function block.

VAR_OUTPUT

```
bBusy               : BOOL;
bReady              : BOOL;
bError              : BOOL;
eError              : E_MBUS_ERROR;
dwIdNumber          : DWORD;
byStatus            : BYTE;
byGEN               : BYTE;
byCounter           : BYTE;
usiRecivedAdr       : USINT;
eMedium             : E_MBUS_Medium;
sMan                : STRING(3);
stOperatingHours    : ST_MBus_Info;
stEnergy            : ST_MBus_Info;
stPower             : ST_MBus_Info;
stInitCounter       : ST_MBus_Info;
stDeviceError       : ST_MBus_Info;
```

bBusy: The *bBusy* output is TRUE while the meter is being read.

bReady: The *bReady* output is TRUE for one cycle, once meter reading is completed.

bError: The *bError* output becomes TRUE as soon as an error occurs. The error is described via the variable *eError*.

eError: The *eError* output issues an error code when an error occurs (see E_MBUS_ERROR [▶ 216]).

dwIdNumber: Serial number of the meter (secondary address).

byStatus: Status of the meter. Please refer to device description for meanings.

byGEN: Meter software version.

byCounter: Transmission counter (number of transmitted RSP_UD).

usiRecivedAdr: Received primary address (0-250).

eMedium: Medium (see E_MBUS_Medium [▶ 219]).

sMan: Manufacturer code.

stOperatingHours: Operating hours of EMU meter (see ST_MBus_Info [▶ 222]).

stEnergy: Meter reading, energy consumption (see ST_MBus_Info [▶ 222]).

stPower: Current energy consumption (see ST_MBus_Info [▶ 222]).

stInitCounter: Number of voltage outages on EMU meter (see ST_MBus_Info [▶ 222]).

stDeviceError: Error message from the device (see ST_MBus_Info [▶ 222]).

VAR_IN_OUT

```
stCom          : ST_MBUS_Communication;
```

stCom: About this structure, the block FB_MBUSKL6781() [▶ 29] is connected to the meter function blocks (see ST_MBUS_Communication [▶ 221]).

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055604363/.zip:

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055605771/.zip:

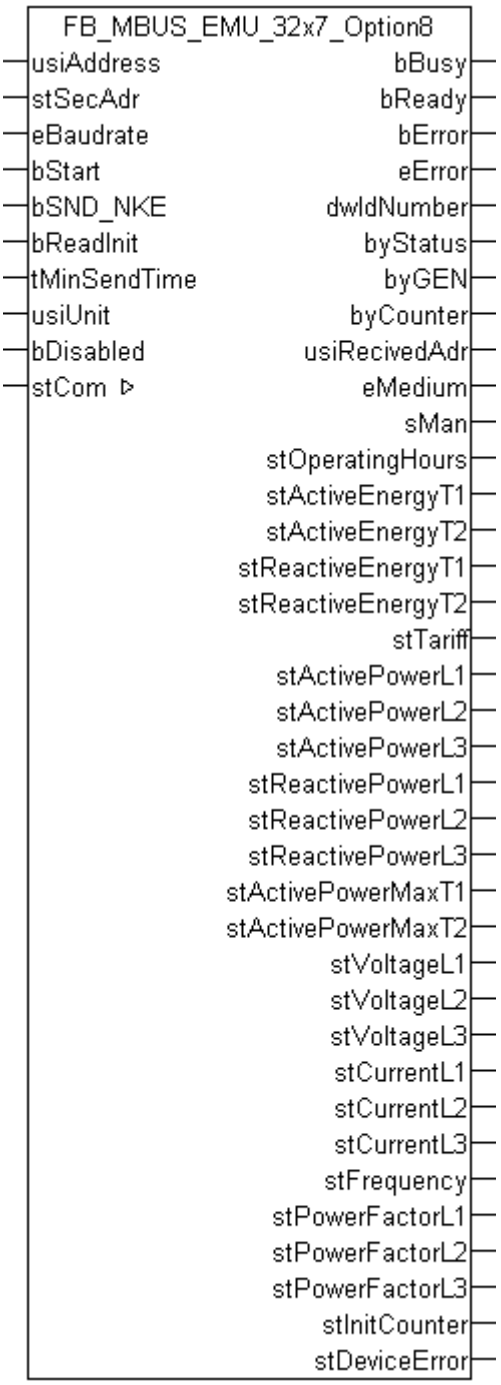
Controller configuration setting: "BCxx50 or BX serial"

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055607179/.zip:

BCxx00 must be set to "Large Model" in the PLC under "Project/Options/Controller Settings".

Controller configuration setting: "BC serial"

6.14.2 FB_MBUS_EMU_32x7_Option8



This function block is used to read electricity meters from EMU:

-EMU32.x7

The parameter set must be set in the device to 70000 hexadecimal (variant 8) in order to read out this data. Please refer to the meter documentation for further information regarding this.

In the normal version, the current consumption of the M-Bus interface is equivalent to 3 standard loads. If an M-Bus master interface is used that is designed, for example, for up to 120 standard loads, a maximum of 40 EMU M-Bus meters can be connected. The meter can optionally be supplied with 230 V. The current consumption of the M-Bus interface is then equivalent to one standard load.

The transmission of data from the EMU meter to the M-Bus protocol computer only works if the EMU meter is connected to at least two phases of the mains voltage network.

The EMU meter transmits current data to the device's M-Bus interface every 40 seconds, so that the readout data is approx. 40 - 45 seconds old.

The function block can only be executed together with the function block FB_MBUSKL6781() [▶ 29].

Functionality of the function block [▶ 13]

VAR_INPUT

```
usiAddress      : USINT;
stSecAdr        : ST_MBUS_SecAdr;
eBaudrate       : E_MBUS_Baudrate := eMBUS_Baud2400;
bStart          : BOOL;
bSND_NKE        : BOOL := TRUE;
bReadInit       : BOOL := TRUE;
tMinSendTime    : TIME := t#2s;
usiUnit         : USINT;
bDisabled       : BOOL := FALSE;
```

usiAddress: Primary address [▶ 14] of the counter, that shall be readout with this module.

stSecAdr: Secondary address [▶ 15] of the counter, that shall be readout with this module.

eBaudrate: 300, 600, 1200, 2400, 4800, 9600 baud [▶ 216].

bStart: Positive edge on this input, the meter is read out once.

bSND_NKE: TRUE to initialize the meter at each reading, and sets the meter on the first telegram (SND_NKE).

bReadInit: After restarting the PLC, the meter is read out once.

tMinSendTime: Standard t#2s. When this time exceeds, the counter is rereadout. At t#0s the counter is not readout and can be readout with bStart manually.

usiUnit: Unit of the energy values. 0=W/J(h) / 1=KW/KJ(h) / 2=MW/MJ(h) / 3=GW/GJ(h).

bDisabled: TRUE = disable the function block.

VAR_OUTPUT

```
bBusy               : BOOL;
bReady              : BOOL;
bError              : BOOL;
eError              : E_MBUS_ERROR;
dwIdNumber          : DWORD;
byStatus            : BYTE;
byGEN               : BYTE;
byCounter           : BYTE;
usiRecivedAdr       : USINT;
eMedium             : E_MBUS_Medium;
sMan                : STRING(3);
stOperatingHours    : ST_MBus_Info;
stActiveEnergyT1    : ST_MBus_Info;
stActiveEnergyT2    : ST_MBus_Info;
stReactiveEnergyT1  : ST_MBus_Info;
stReactiveEnergyT2  : ST_MBus_Info;
stTariff            : ST_MBus_Info;
stActivePowerL1     : ST_MBus_Info;
stActivePowerL2     : ST_MBus_Info;
stActivePowerL3     : ST_MBus_Info;
stReactivePowerL1   : ST_MBus_Info;
stReactivePowerL2   : ST_MBus_Info;
stReactivePowerL3   : ST_MBus_Info;
stActivePowerMaxT1  : ST_MBus_Info;
stActivePowerMaxT2  : ST_MBus_Info;
stVoltageL1         : ST_MBus_Info;
stVoltageL2         : ST_MBus_Info;
stVoltageL3         : ST_MBus_Info;
stCurrentL1         : ST_MBus_Info;
stCurrentL2         : ST_MBus_Info;
stCurrentL3         : ST_MBus_Info;
stFrequency         : ST_MBus_Info;
stPowerFactorL1     : ST_MBus_Info;
stPowerFactorL2     : ST_MBus_Info;
```

```
stPowerFactorL3     : ST_MBus_Info;
stInitCounter       : ST_MBus_Info;
stDeviceError       : ST_MBus_Info;
```

bBusy: The *bBusy* output is TRUE while the meter is being read.

bReady: The *bReady* output is TRUE for one cycle, once meter reading is completed.

bError: The *bError* output becomes TRUE as soon as an error occurs. The error is described via the variable *eError*.

eError: The *eError* output issues an error code when an error occurs (see E_MBUS_ERROR [▶ 216]).

dwIdNumber: Serial number of the meter (secondary address).

byStatus: Status of the meter. Please refer to device description for meanings.

byGEN: Meter software version.

byCounter: Transmission counter (number of transmitted RSP_UD).

usiRecivedAdr: Received primary address (0-250).

eMedium: Medium (see E_MBUS_Medium [▶ 219]).

sMan: Manufacturer code.

stOperatingHours: Operating hours of EMU meter (see ST_MBus_Info [▶ 222]).

stActiveEnergyT1: Meter reading, active energy tariff 1 (see ST_MBus_Info [▶ 222]).

stActiveEnergyT2: Meter reading, active energy tariff 2 (see ST_MBus_Info [▶ 222]).

stReactiveEnergyT1: Meter reading, reactive energy tariff 1 (see ST_MBus_Info [▶ 222]).

stReactiveEnergyT2: Meter reading, reactive energy tariff 2 (see ST_MBus_Info [▶ 222]).

stTariff: Tariff presently operating (see ST_MBus_Info [▶ 222]).

stActivePowerL1: Instantaneous active power L1 (see ST_MBus_Info [▶ 222]).

stActivePowerL2: Instantaneous active power L2 (see ST_MBus_Info [▶ 222]).

stActivePowerL3: Instantaneous active power L3 (see ST_MBus_Info [▶ 222]).

stReactivePowerL1: Instantaneous reactive power L1 (see ST_MBus_Info [▶ 222]).

stReactivePowerL2: Instantaneous reactive power L2 (see ST_MBus_Info [▶ 222]).

stReactivePowerL3: Instantaneous reactive power L3 (see ST_MBus_Info [▶ 222]).

stActivePowerMaxT1: Maximum demand active power tariff 1 (see ST_MBus_Info [▶ 222]).

stActivePowerMaxT2: Maximum demand active power tariff 2 (see ST_MBus_Info [▶ 222]).

stVoltageL1: Instantaneous voltage L1 (see ST_MBus_Info [▶ 222]).

stVoltageL2: Instantaneous voltage L2 (see ST_MBus_Info [▶ 222]).

stVoltageL3: Instantaneous voltage L3 (see ST_MBus_Info [▶ 222]).

stCurrentL1: Instantaneous current L1 (see ST_MBus_Info [▶ 222]).

stCurrentL2: Instantaneous current L2 (see ST_MBus_Info [▶ 222]).

stCurrentL3: Instantaneous current L3 (see ST_MBus_Info [▶ 222]).

stFrequency: Instantaneous network frequency (see ST_MBus_Info [▶ 222]).

stPowerFactorL1: Instantaneous power factor L1 (cos phi) (see ST_MBus_Info [▶ 222]).

stPowerFactorL2: Instantaneous power factor L2 (cos phi) (see ST_MBus_Info [▶ 222]).

stPowerFactorL3: Instantaneous power factor L3 (cos phi) (see ST_MBus_Info [▶ 222]).

stInitCounter: Number of voltage outages on EMU meter (see ST_MBus_Info [▶ 222]).

stDeviceError: Error message from the device (see ST_MBus_Info [▶ 222]).

VAR_IN_OUT

```
stCom          : ST_MBUS_Communication;
```

stCom: About this structure, the block FB_MBUSKL6781() [▶ 29] is connected to the meter function blocks (see ST_MBUS_Communication [▶ 221]).

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055604363/.zip:

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055605771/.zip:

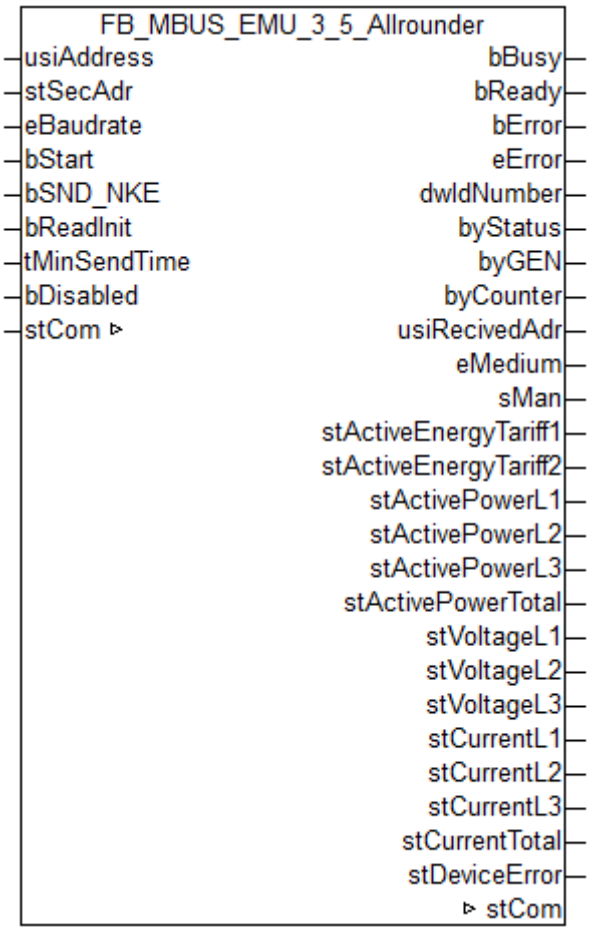
Controller configuration setting: "BCxx50 or BX serial"

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055607179/.zip:

BCxx00 must be set to "Large Model" in the PLC under "Project/Options/Controller Settings".

Controller configuration setting: "BC serial"

6.14.3 FB_MBUS_EMU_3_5_Allrounder



This module is used to readout electricity meters of the manufacturer EMU.

It can only be used together with the module FB_MBUSKL6781() [▶ 29]

Functionality of the module [▶ 13]

VAR_INPUT

```
usiAddress      : USINT;
stSecAdr        : ST_MBUS_SecAdr;
eBaudrate       : E_MBUS_Baudrate := eMBUS_Baud2400;
bStart          : BOOL;
bSND_NKE        : BOOL := TRUE;
bReadInit       : BOOL := TRUE;
tMinSendTime    : TIME := t#2s;
bDisabled       : BOOL := FALSE;
```

usiAddress: Primary address [▶ 14] of the counter, that shall be readout with this module.

stSecAdr: Secondary address [▶ 15] of the counter, that shall be readout with this module.

eBaudrate: 300..9600 baud [▶ 216].

bStart: Positive edge on this input, the meter is read out once.

bSND_NKE: TRUE to initialize the meter at each reading, and sets the meter on the first telegram (SND_NKE).

bReadInit: After restarting the PLC, the meter is read out once.

tMinSendTime: Standard t#2s. When this time exceeds, the counter is rereadout. At t#0s the counter is not readout and can be readout with bStart manually.

bDisabled: TRUE = disable the function block.

VAR_OUTPUT

```
bBusy                   : BOOL;
bReady                  : BOOL;
bError                  : BOOL;
eError                  : E_MBUS_ERROR;
dwIdNumber              : DWORD;
byStatus                : BYTE;
byGEN                   : BYTE;
byCounter               : BYTE;
usiRecivedAdr           : USINT;
eMedium                 : E_MBUS_Medium;
sMan                    : STRING(3);
stActiveEnergyTariff1   : ST_MBus_Info;
stActiveEnergyTariff2   : ST_MBus_Info;
stActivePowerL1         : ST_MBus_Info;
stActivePowerL2         : ST_MBus_Info;
stActivePowerL3         : ST_MBus_Info;
stActivePowerTotal      : ST_MBus_Info;
stVoltageL1             : ST_MBus_Info;
stVoltageL2             : ST_MBus_Info;
stVoltageL3             : ST_MBus_Info;
stCurrentL1             : ST_MBus_Info;
stCurrentL2             : ST_MBus_Info;
stCurrentL3             : ST_MBus_Info;
stCurrentTotal          : ST_MBus_Info;
stDeviceError           : ST_MBus_Info;
```

bBusy: The bBusy output is TRUE as long as the meter readout is running.

bReady: The bReady output is TRUE for one cycle when the meter readout has been completed.

bError: The output becomes TRUE as soon as an error occurs. This error is described via the variable eError.

eError: The output issues an error code in the event of an error (see E_MBUS_ERROR [▶ 216]). At the same time bError becomes TRUE.

dwIdNumber: Serial number of counter (secondary address).

byStatus: Status of device.

byGEN: Software version dof device.

byCounter: Number of accesses of the master to data of the respective slave.

usiRecivedAdr: Recieved primary address (0-250).

eMedium: Medium (siehe E_MBUS_Medium [▶ 219]).

sMan: Manufacturer's abbreviation.

stActiveEnergyTariff1: Active energy tariff 1 (see ST_MBus_Info [▶ 222]).

stActiveEnergyTariff2: Active energy tariff 2 (see ST_MBus_Info [▶ 222]).

stActivePowerL1: Actual active power L1 (see ST_MBus_Info [▶ 222]).

stActivePowerL2: Actual active power L2 (see ST_MBus_Info [▶ 222]).

stActivePowerL3: Actual active power L3 (see ST_MBus_Info [▶ 222]).

stActivePowerTotal: Total active power (see ST_MBus_Info [▶ 222]).

stVoltageL1: Voltage L1 (see ST_MBus_Info [▶ 222]).

stVoltageL2: Voltage L2 (see ST_MBus_Info [▶ 222]).

stVoltageL3: Voltage L3 (see ST_MBus_Info [▶ 222]).

stCurrentL1: Current L1 (see ST_MBus_Info [▶ 222]).

stCurrentL2: Current L2 (see ST_MBus_Info [▶ 222]).

stCurrentL3: Current L3 (see ST_MBus_Info [▶ 222]).

stCurrentTotal: Total current (see ST_MBus_Info [▶ 222]).

stDeviceError: Error message from the device (see ST_MBus_Info [▶ 222]).

VAR_IN_OUT

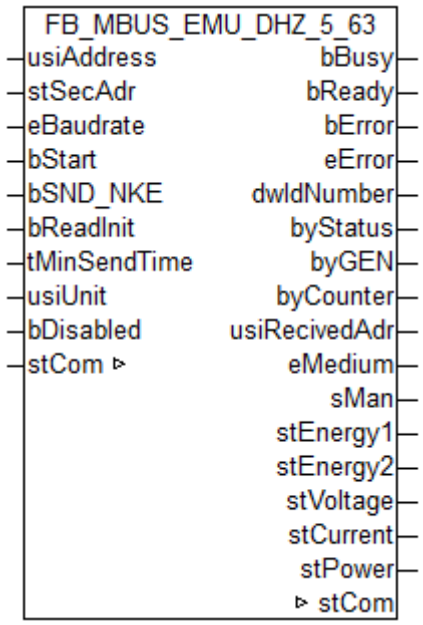
```
stCom           : ST_MBUS_Communication;
```

stCom: About this structure, the block FB_MBUSKL6781() [▶ 29] is connected to the meter function blocks (see ST_MBUS_Communication [▶ 221]).

Requirements

Development environment	Target system type	PLC libraries to include
TwinCAT 2.11 R3/x64 higher than Build 2234	PC/CX, BX or BC	TcMBus-Library higher than V2.3.0

6.14.4 FB_MBUS_EMU_DHZ_5_63



This module is used to readout electricity meters of the manufacturer EMU.

It can only be used together with the module FB_MBUSKL6781() [▶ 29]

Functionality of the module [▶ 13]

VAR_INPUT

```
usiAddress      : USINT;
stSecAdr        : ST_MBUS_SecAdr;
eBaudrate       : E_MBUS_Baudrate := eMBUS_Baud2400;
bStart          : BOOL;
bSND_NKE        : BOOL := TRUE;
bReadInit       : BOOL := TRUE;
tMinSendTime    : TIME := t#2s;
bDisabled       : BOOL := FALSE;
```

usiAddress: Primary address [▶ 14] of the counter, that shall be readout with this module.

stSecAdr: Secondary address [▶ 15] of the counter, that shall be readout with this module.

eBaudrate: 300..9600 baud [▶ 216].

bStart: Positive edge on this input, the meter is read out once.

bSND_NKE: TRUE to initialize the meter at each reading, and sets the meter on the first telegram (SND_NKE).

bReadInit: After restarting the PLC, the meter is read out once.

tMinSendTime: Standard t#2s. When this time exceeds, the counter is rereadout. At t#0s the counter is not readout and can be readout with bStart manually.

bDisabled: TRUE = disable the function block.

VAR_OUTPUT

```
bBusy           : BOOL;
bReady          : BOOL;
bError          : BOOL;
eError          : E_MBUS_ERROR;
dwIdNumber      : DWORD;
byStatus        : BYTE;
byGEN           : BYTE;
byCounter       : BYTE;
```

```
usiRecivedAdr   : USINT;
eMedium         : E_MBUS_Medium;
sMan            : STRING(3);
stEnergy1       : ST_MBus_Info;
stEnergy2       : ST_MBus_Info;
stVoltage       : ST_MBus_Info;
stCurrent       : ST_MBus_Info;
stPower         : ST_MBus_Info;
```

bBusy: The *bBusy* output is TRUE while the meter is being read.

bReady: The *bReady* output is TRUE for one cycle, once meter reading is completed.

bError: The *bError* output becomes TRUE as soon as an error occurs. The error is described via the variable *eError*.

eError: The *eError* output issues an error code when an error occurs (see E_MBUS_ERROR [▶ 216]).

dwIdNumber: Serial number of the meter (secondary address).

byStatus: Status of the meter. Please refer to device description for meanings.

byGEN: Meter software version.

byCounter: Transmission counter (number of transmitted RSP_UD).

usiRecivedAdr: Received primary address (0-250).

eMedium: Medium (see E_MBUS_Medium [▶ 219]).

sMan: Manufacturer code.

stEnergy1: Energy 1 (see ST_MBus_Info [▶ 222]).

stEnergy2: Energy 2 (see ST_MBus_Info [▶ 222]).

stVoltage: Mains voltage (see ST_MBus_Info [▶ 222]).

stCurrent: Current (see ST_MBus_Info [▶ 222]).

stPower: Active power (see ST_MBus_Info [▶ 222]).

VAR_IN_OUT

```
stCom           : ST_MBUS_Communication;
```

stCom: About this structure, the block FB_MBUSKL6781() [▶ 29] is connected to the meter function blocks (see ST_MBUS_Communication [▶ 221]).

Requirements

Development environment	Target system type	PLC libraries to include
TwinCAT 2.11 R3/x64 higher than Build 2234	PC/CX, BX or BC	TcMBus-Library higher than V2.3.0

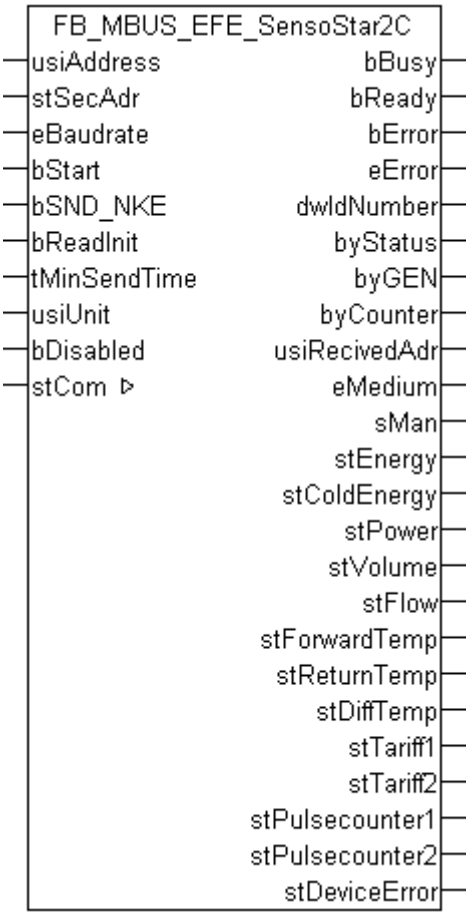
6.15 Engelmann

The function blocks only output a selection of the most common data. These data are described on the respective pages under "VAR_OUT". If more or all data are required, the function blocks FB_MBUS_General [▶ 31], FB_MBUS_General_Ext [▶ 35] or FB_MBUS_General_Param [▶ 41] from the General folder should be used. Not that these function blocks do not run on BC or BX systems. The function block FB_MBUS_General_Send [▶ 43] can be used to send data to the device (e.g. setting of the primary address).

Vendor	Type	Device	Function block
Engelmann	Heat meter	Sensostar 2C	FB_MBUS_EFF_SensoStar2C [▶ 104]

Vendor	Type	Device	Function block

6.15.1 FB_MBUS_EFE_SensoStar2C



This block is used for reading heat meters from Engelmann:

-SENSOSTAR 2C

The block can only be used in conjunction with the block FB_MBUSKL6781() [▶ 29].

Functionality of the function block [▶ 13]

VAR_INPUT

```
usiAddress        : USINT;
stSecAdr          : ST_MBUS_SecAdr;
eBaudrate         : E_MBUS_Baudrate := eMBUS_Baud2400;
bStart            : BOOL;
bSND_NKE          : BOOL := TRUE;
bReadInit         : BOOL := TRUE;
tMinSendTime      : TIME := t#2s;
usiUnit           : USINT;
bDisabled         : BOOL := FALSE;
```

usiAddress: Primary address [▶ 14] of the counter, that shall be readout with this module.

stSecAdr: Secondary address [▶ 15] of the counter, that shall be readout with this module.

eBaudrate: baud [▶ 216].

bStart: Positive edge on this input, the meter is read out once.

bSND_NKE: TRUE to initialize the meter at each reading and sets the meter on the first telegram (SND_NKE).

bReadInit: After restarting the PLC, the meter is read out once.

tMinSendTime: Standard t#2s. When this time exceeds, the counter is rereadout. At t#0s the counter is not readout and can be readout with bStart manually.

usiUnit: Unit of the energy values. 0=W/J(h) / 1=KW/KJ(h) / 2=MW/MJ(h) / 3=GW/GJ(h).

bDisabled: TRUE = disable the function block.

VAR_OUTPUT

```
bBusy            : BOOL;
bReady           : BOOL;
bError           : BOOL;
eError           : E_MBUS_ERROR;
dwIdNumber       : DWORD;
byStatus         : BYTE;
byGEN            : BYTE;
byCounter        : BYTE;
usiRecivedAdr    : USINT;
eMedium          : E_MBUS_Medium;
sMan             : STRING(3);
stEnergy         : ST_MBus_Info;
stColdEnergy     : ST_MBus_Info;
stPower          : ST_MBus_Info;
stVolume         : ST_MBus_Info;
stFlow           : ST_MBus_Info;
stForwardTemp    : ST_MBus_Info;
stReturnTemp     : ST_MBus_Info;
stDiffTemp       : ST_MBus_Info;
stTariff1        : ST_MBus_Info;
stTariff2        : ST_MBus_Info;
stPulsecounter1  : ST_MBus_Info;
stPulsecounter2  : ST_MBus_Info;
stDeviceError    : ST_MBus_Info;
```

bBusy: The *bBusy* output is TRUE while the meter is being read.

bReady: The *bReady* output is TRUE for one cycle, once meter reading is completed.

bError: The *bError* output becomes TRUE as soon as an error occurs. The error is described via the variable *eError*.

eError: The *eError* output issues an error code when an error occurs (see E_MBUS_ERROR [▶ 216]).

dwIdNumber: Serial number of the meter (secondary address).

byStatus: Status of the meter. Please refer to device description for meanings.

byGEN: Meter software version.

byCounter: Transmission counter (number of transmitted RSP_UD).

usiRecivedAdr: Received primary address (0-250).

eMedium: Medium (see E_MBUS_Medium [▶ 219]).

sMan: Manufacturer code.

stEnergy: Meter reading, energy consumption (see ST_MBus_Info [▶ 222]).

stColdEnergy: Meter reading, cold energy consumption (see ST_MBus_Info [▶ 222]).

stPower: Current energy consumption (see ST_MBus_Info [▶ 222]).

stVolume: Meter reading, volume (see ST_MBus_Info [▶ 222]).

stFlow: Current flow (see ST_MBus_Info [▶ 222]).

stForwardTemp: Flow temperature (see ST_MBus_Info [▶ 222]).

stReturnTemp: Return temperature (see ST_MBus_Info [▶ 222]).

stDiffTemp: Temperature difference (see ST_MBus_Info [▶ 222]).

stTariff1: Tariff register 1 (see ST_MBus_Info [▶ 222]).

stTariff2: Tariff register 2 (see ST_MBus_Info [▶ 222]).

stPulsecounter1: Pulse counter 1 (see ST_MBus_Info [▶ 222]).

stPulsecounter2: Pulse counter 2 (see ST_MBus_Info [▶ 222]).

stDeviceError: Error message from the device (see ST_MBus_Info [▶ 222]).

VAR_IN_OUT

```
stCom        : ST_MBUS_Communication;
```

stCom: About this structure, the block FB_MBUSKL6781() [▶ 29] is connected to the meter function blocks (see ST_MBUS_Communication [▶ 221]).

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055608587/.zip:

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055609995/.zip:

Controller configuration setting: "BCxx50 or BX serial"

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055611403/.zip:

BCxx00 must be set to "Large Model" in the PLC under "Project/Options/Controller Settings".

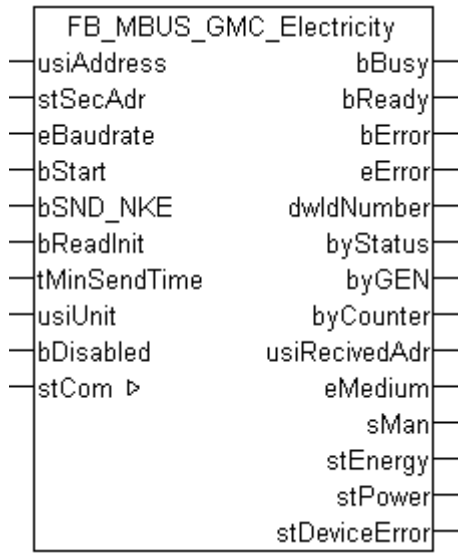
Controller configuration setting: "BC serial"

6.16 Gossen Metrawatt

The function blocks only output a selection of the most common data. These data are described on the respective pages under "VAR_OUT". If more or all data are required, the function blocks FB_MBUS_General [▶ 31], FB_MBUS_General_Ext [▶ 35] or FB_MBUS_General_Param [▶ 41] from the General folder should be used. Not that these function blocks do not run on BC or BX systems. The function block FB_MBUS_General_Send [▶ 43] can be used to send data to the device (e.g. setting of the primary address).

Vendor	Type	Device	Function block
Gossen Metrawatt	Electricity meter	U128x	FB_MBUS_GMC_Electricity [▶ 107]
	Electricity meter	U138x	

6.16.1 FB_MBUS_GMC_Electricity



This block is used for reading electricity meters from Gossen Metrawatt:

-U128x

-U138x

The block can only be used in conjunction with the block FB_MBUSKL6781() [▶ 29].

Functionality of the function block [▶ 13]

VAR_INPUT

```
usiAddress      : USINT;
stSecAdr        : ST_MBUS_SecAdr;
eBaudrate       : E_MBUS_Baudrate := eMBUS_Baud2400;
bStart          : BOOL;
bSND_NKE        : BOOL := TRUE;
bReadInit       : BOOL := TRUE;
tMinSendTime    : TIME := t#2s;
usiUnit         : USINT;
bDisabled       : BOOL := FALSE;
```

usiAddress: Primary address [▶ 14] of the counter, that shall be readout with this module.

stSecAdr: Secondary address [▶ 15] of the counter, that shall be readout with this module.

eBaudrate: 300, 600, 1200, 2400, 4800, 9600 baud [▶ 216].

bStart: Positive edge on this input, the meter is read out once.

bSND_NKE: TRUE to initialize the meter at each reading, and sets the meter on the first telegram (SND_NKE).

bReadInit: After restarting the PLC, the meter is read out once.

tMinSendTime: Standard t#2s. When this time exceeds, the counter is rereadout. At t#0s the counter is not readout and can be readout with bStart manually.

usiUnit: Unit of the energy values. 0=W/J(h) / 1=KW/KJ(h) / 2=MW/MJ(h) / 3=GW/GJ(h).

bDisabled: TRUE = disable the function block.

VAR_OUTPUT

```
bBusy           : BOOL;
bReady          : BOOL;
bError          : BOOL;
```

```
eError          : E_MBUS_ERROR;
dwIdNumber      : DWORD;
byStatus        : BYTE;
byGEN           : BYTE;
byCounter       : BYTE;
usiRecivedAdr   : USINT;
eMedium         : E_MBUS_Medium;
sMan            : STRING(3);
stEnergy        : ST_MBus_Info;
stPower         : ST_MBus_Info;
stDeviceError   : ST_MBus_Info;
```

bBusy: The *bBusy* output is TRUE while the meter is being read.

bReady: The *bReady* output is TRUE for one cycle, once meter reading is completed.

bError: The *bError* output becomes TRUE as soon as an error occurs. The error is described via the variable *eError*.

eError: The *eError* output issues an error code when an error occurs (see E_MBUS_ERROR [▶ 216]).

dwIdNumber: Serial number of the meter (secondary address).

byStatus: Status of the meter. Please refer to device description for meanings.

byGEN: Meter software version.

byCounter: Transmission counter (number of transmitted RSP_UD).

usiRecivedAdr: Received primary address (0-250).

eMedium: Medium (see E_MBUS_Medium [▶ 219]).

sMan: Manufacturer code.

stEnergy: Meter reading, energy consumption (see ST_MBus_Info [▶ 222]).

stPower: Current energy consumption (see ST_MBus_Info [▶ 222]).

stDeviceError: Error message from the device (see ST_MBus_Info [▶ 222]).

VAR_IN_OUT

```
stCom           : ST_MBUS_Communication;
```

stCom: About this structure, the block FB_MBUSKL6781() [▶ 29] is connected to the meter function blocks (see ST_MBUS_Communication [▶ 221]).

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055612811/.zip: 

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055614219/.zip: 

Controller configuration setting: "BCxx50 or BX serial"

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055615627/.zip: 

BCxx00 must be set to "Large Model" in the PLC under "Project/Options/Controller Settings".

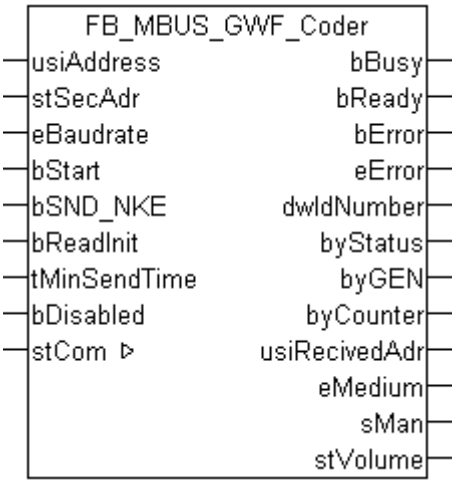
Controller configuration setting: "BC serial"

6.17 GWF

The function blocks only output a selection of the most common data. These data are described on the respective pages under "VAR_OUT". If more or all data are required, the function blocks FB_MBUS_General [▶ 31], FB_MBUS_General_Ext [▶ 35] or FB_MBUS_General_Param [▶ 41] from the General folder should be used. Not that these function blocks do not run on BC or BX systems. The function block FB_MBUS_General_Send [▶ 43] can be used to send data to the device (e.g. setting of the primary address).

Vendor	Type	Device	Function block
GWF	Water meter		FB_MBUS_GWF_Coder [▶ 109]
	Gas meter	S1	
	Gas meter	Z1	

6.17.1 FB_MBUS_GWF_Coder



This block is used for reading meters from GWF:

-Water meter

-Gas meter S1

-Gas meter Z1

The block can only be used in conjunction with the block FB_MBUSKL6781() [▶ 29].

Functionality of the function block [▶ 13]

VAR_INPUT

```
usiAddress      : USINT;
stSecAdr        : ST_MBUS_SecAdr;
eBaudrate       : E_MBUS_Baudrate := eMBUS_Baud2400;
bStart          : BOOL;
bSND_NKE        : BOOL := TRUE;
bReadInit       : BOOL := TRUE;
tMinSendTime    : TIME := t#2s;
bDisabled       : BOOL := FALSE;
```

usiAddress: Primary address [▶ 14] of the counter, that shall be readout with this module.

stSecAdr: Secondary address [▶ 15] of the counter, that shall be readout with this module.

eBaudrate: 300, 2400, 9600 baud [▶ 216].

bStart: Positive edge on this input, the meter is read out once.

bSND_NKE: TRUE to initialize the meter at each reading, and sets the meter on the first telegram (SND_NKE).

bReadInit: After restarting the PLC, the meter is read out once.

tMinSendTime: Standard t#2s. When this time exceeds, the counter is rereadout. At t#0s the counter is not readout and can be readout with bStart manually.

bDisabled: TRUE = disable the function block.

VAR_OUTPUT

```
bBusy           : BOOL;
bReady          : BOOL;
bError          : BOOL;
eError          : E_MBUS_ERROR;
dwIdNumber      : DWORD;
byStatus        : BYTE;
byGEN           : BYTE;
byCounter       : BYTE;
usiRecivedAdr   : USINT;
eMedium         : E_MBUS_Medium;
sMan            : STRING(3);
stVolume        : ST_MBus_Info;
```

bBusy: The *bBusy* output is TRUE while the meter is being read.

bReady: The *bReady* output is TRUE for one cycle, once meter reading is completed.

bError: The *bError* output becomes TRUE as soon as an error occurs. The error is described via the variable *eError*.

eError: The *eError* output issues an error code when an error occurs (see E_MBUS_ERROR [▶ 216]).

dwIdNumber: Serial number of the meter (secondary address).

byStatus: Status of the meter. Please refer to device description for meanings.

byGEN: Meter software version.

byCounter: Transmission counter (number of transmitted RSP_UD).

usiRecivedAdr: Received primary address (0-250).

eMedium: Medium (see E_MBUS_Medium [▶ 219]).

sMan: Manufacturer code.

stVolume: Meter reading, volume (see ST_MBus_Info [▶ 222]).

VAR_IN_OUT

```
stCom           : ST_MBUS_Communication;
```

stCom: About this structure, the block FB_MBUSKL6781() [▶ 29] is connected to the meter function blocks (see ST_MBUS_Communication [▶ 221]).

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055617035/.zip:

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055618443/.zip:

Controller configuration setting: "BCxx50 or BX serial"

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055619851/.zip:

BCxx00 must be set to "Large Model" in the PLC under "Project/Options/Controller Settings".

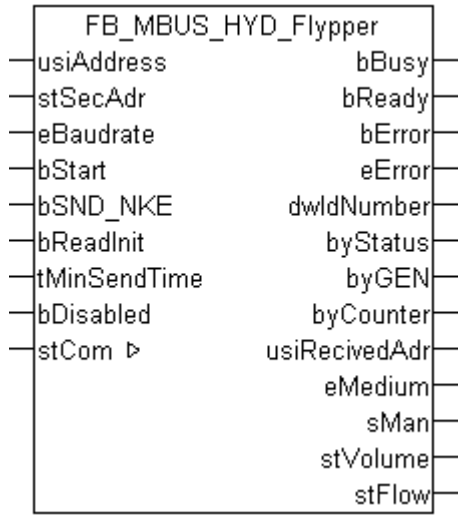
Controller configuration setting: "BC serial"

6.18 Hydrometer

The function blocks only output a selection of the most common data. These data are described on the respective pages under "VAR_OUT". If more or all data are required, the function blocks FB_MBUS_General [▶ 31], FB_MBUS_General_Ext [▶ 35] or FB_MBUS_General_Param [▶ 41] from the General folder should be used. Not that these function blocks do not run on BC or BX systems. The function block FB_MBUS_General_Send [▶ 43] can be used to send data to the device (e.g. setting of the primary address).

Vendor	Type	Device	Function block
Hydrometer	2 pulse inputs	HYDRO-PORT Pulse	FB_MBUS_HYD_PortPulse [▶ 115]
	2 analog inputs+1 temperature sensor	HYDRO-PORT Analog	FB_MBUS_HYD_PortAnalog [▶ 113]
	Water meter	Flypper	FB_MBUS_HYD_Flypper [▶ 111]
	Heat meter	Sharky 773	FB_MBUS_HYD_Sharky [▶ 117], FB_MBUS_HYD_Sharky_00 [▶ 120]
	Heat meter	Sharky 775	

6.18.1 FB_MBUS_HYD_Flypper



This block is used for reading water meters from Hydrometer:

-Flypper

The block can only be used in conjunction with the block FB_MBUSKL6781() [▶ 29].

Functionality of the function block [▶ 13]

VAR_INPUT

```
usiAddress      : USINT;
stSecAdr        : ST_MBUS_SecAdr;
eBaudrate       : E_MBUS_Baudrate := eMBUS_Baud2400;
bStart          : BOOL;
bSND_NKE        : BOOL := TRUE;
bReadInit       : BOOL := TRUE;
tMinSendTime    : TIME := t#2s;
bDisabled       : BOOL := FALSE;
```

usiAddress: Primary address [▶ 14] of the counter, that shall be readout with this module.

stSecAdr: Secondary address [▶ 15] of the counter, that shall be readout with this module.

eBaudrate: 300, 2400 baud [▶ 216].

bStart: Positive edge on this input, the meter is read out once.

bSND_NKE: TRUE to initialize the meter at each reading, and sets the meter on the first telegram (SND_NKE).

bReadInit: After restarting the PLC, the meter is read out once.

tMinSendTime: Standard t#2s. When this time exceeds, the counter is rereadout. At t#0s the counter is not readout and can be readout with bStart manually.

bDisabled: TRUE = disable the function block.

VAR_OUTPUT

```
bBusy           : BOOL;
bReady          : BOOL;
bError          : BOOL;
eError          : E_MBUS_ERROR;
dwIdNumber      : DWORD;
byStatus        : BYTE;
byGEN           : BYTE;
byCounter       : BYTE;
usiRecivedAdr   : USINT;
eMedium         : E_MBUS_Medium;
sMan            : STRING(3);
stVolume        : ST_MBus_Info;
stFlow          : ST_MBus_Info;
```

bBusy: The *bBusy* output is TRUE while the meter is being read.

bReady: The *bReady* output is TRUE for one cycle, once meter reading is completed.

bError: The *bError* output becomes TRUE as soon as an error occurs. The error is described via the variable *eError*.

eError: The *eError* output issues an error code when an error occurs (see E_MBUS_ERROR [▶ 216]).

dwIdNumber: Serial number of the meter (secondary address).

byStatus: Status of the meter. Please refer to device description for meanings.

byGEN: Meter software version.

byCounter: Transmission counter (number of transmitted RSP_UD).

usiRecivedAdr: Received primary address (0-250).

eMedium: Medium (see E_MBUS_Medium [▶ 219]).

sMan: Manufacturer code.

stVolume: Meter reading, volume (see ST_MBus_Info [▶ 222]).

stFlow: Current flow (see ST_MBus_Info [▶ 222]).

VAR_IN_OUT

```
stCom           : ST_MBUS_Communication;
```

stCom: About this structure, the block FB_MBUSKL6781() [▶ 29] is connected to the meter function blocks (see ST_MBUS_Communication [▶ 221]).

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055621259/.zip:

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055622667/.zip:

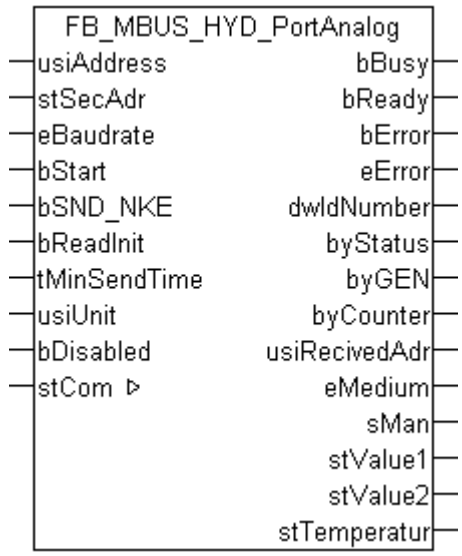
Controller configuration setting: "BCxx50 or BX serial"

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055624075/.zip:

BCxx00 must be set to "Large Model" in the PLC under "Project/Options/Controller Settings".

Controller configuration setting: "BC serial"

6.18.2 FB_MBUS_HYD_PortAnalog



This function block is used for reading energy meters with analog output from Hydrometer:

-HYDRO-PORT analog (2x0/4-20 mA / 1xPT temperature sensor)

The function block can only be executed together with the function block FB_MBUSKL6781() [▶ 29].

Functionality of the function block [▶ 13]

VAR_INPUT

```
usiAddress      : USINT;
stSecAdr        : ST_MBUS_SecAdr;
eBaudrate       : E_MBUS_Baudrate := eMBUS_Baud2400;
bStart          : BOOL;
bSND_NKE        : BOOL := TRUE;
bReadInit       : BOOL := TRUE;
tMinSendTime    : TIME := t#2s;
usiUnit         : USINT;
bDisabled       : BOOL := FALSE;
```

usiAddress: Primary address [▶ 14] of the counter, that shall be readout with this module.

stSecAdr: Secondary address [▶ 15] of the counter, that shall be readout with this module.

eBaudrate: 300, 2400, 9600 baud [▶ 216].

bStart: Positive edge on this input, the meter is read out once.

bSND_NKE: TRUE to initialize the meter at each reading, and sets the meter on the first telegram (SND_NKE).

bReadInit: After restarting the PLC, the meter is read out once.

tMinSendTime: Standard t#2s. When this time exceeds, the counter is rereadout. At t#0s the counter is not readout and can be readout with bStart manually.

usiUnit: Unit of the energy values. 0=W/J(h) / 1=KW/KJ(h) / 2=MW/MJ(h) / 3=GW/GJ(h).

bDisabled: TRUE = disable the function block.

VAR_OUTPUT

```
bBusy           : BOOL;
bReady          : BOOL;
bError          : BOOL;
eError          : E_MBUS_ERROR;
dwIdNumber      : DWORD;
byStatus        : BYTE;
byGEN           : BYTE;
byCounter       : BYTE;
usiRecivedAdr   : USINT;
eMedium         : E_MBUS_Medium;
sMan            : STRING(3);
stValue1        : ST_MBus_Info;
stValue2        : ST_MBus_Info;
stTemperatur    : ST_MBus_Info;
```

bBusy: The *bBusy* output is TRUE while the meter is being read.

bReady: The *bReady* output is TRUE for one cycle, once meter reading is completed.

bError: The *bError* output becomes TRUE as soon as an error occurs. The error is described via the variable *eError*.

eError: The *eError* output issues an error code when an error occurs (see E_MBUS_ERROR [▶ 216]).

dwIdNumber: Serial number of the meter (secondary address).

byStatus: Status of the meter. Please refer to device description for meanings.

byGEN: Meter software version.

byCounter: Transmission counter (number of transmitted RSP_UD).

usiRecivedAdr: Received primary address (0-250).

eMedium: Medium (see E_MBUS_Medium [▶ 219]).

sMan: Manufacturer code.

stValue1: Meter reading 1 (see ST_MBus_Info [▶ 222]).

stValue2: Meter reading 2 (see ST_MBus_Info [▶ 222]).

stTemperatur: Temperature (see ST_MBus_Info [▶ 222]).

VAR_IN_OUT

```
stCom           : ST_MBUS_Communication;
```

stCom: About this structure, the block FB_MBUSKL6781() [▶ 29] is connected to the meter function blocks (see ST_MBUS_Communication [▶ 221]).

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055621259/.zip:

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055622667/.zip:

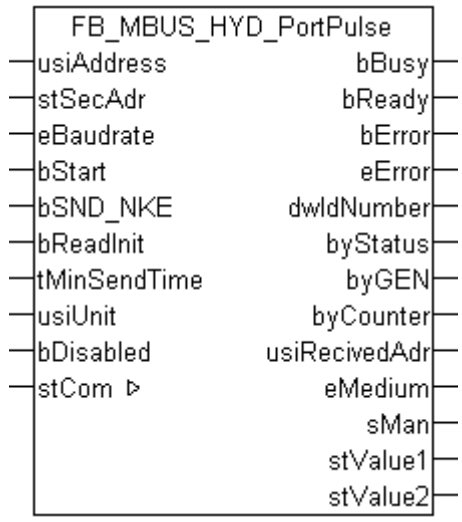
Controller configuration setting: "BCxx50 or BX serial"

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055624075/.zip:

BCxx00 must be set to "Large Model" in the PLC under "Project/Options/Controller Settings".

Controller configuration setting: "BC serial"

6.18.3 FB_MBUS_HYD_PortPulse



This function block is used for reading energy meters with pulse output from Hydrometer:

-HYDRO-PORT Pulse

The function block can only be executed together with the function block FB_MBUSKL6781() [▶ 29].

Functionality of the function block [▶ 13]

VAR_INPUT

```
usiAddress      : USINT;
stSecAdr        : ST_MBUS_SecAdr;
eBaudrate       : E_MBUS_Baudrate := eMBUS_Baud2400;
bStart          : BOOL;
bSND_NKE        : BOOL := TRUE;
bReadInit       : BOOL := TRUE;
tMinSendTime    : TIME := t#2s;
usiUnit         : USINT;
bDisabled       : BOOL := FALSE;
```

usiAddress: Primary address [▶ 14] of the counter, that shall be readout with this module.

stSecAdr: Secondary address [▶ 15] of the counter, that shall be readout with this module.

eBaudrate: 300, 2400, 9600 baud [▶ 216].

bStart: Positive edge on this input, the meter is read out once.

bSND_NKE: TRUE to initialize the meter at each reading, and sets the meter on the first telegram (SND_NKE).

bReadInit: After restarting the PLC, the meter is read out once.

tMinSendTime: Standard t#2s. When this time exceeds, the counter is rereadout. At t#0s the counter is not readout and can be readout with bStart manually.

usiUnit: Unit of the energy values. 0=W/J(h) / 1=KW/KJ(h) / 2=MW/MJ(h) / 3=GW/GJ(h).

bDisabled: TRUE = disable the function block.

VAR_OUTPUT

```
bBusy           : BOOL;
bReady          : BOOL;
bError          : BOOL;
eError          : E_MBUS_ERROR;
dwIdNumber      : DWORD;
byStatus        : BYTE;
byGEN           : BYTE;
```

```
byCounter       : BYTE;
usiRecivedAdr   : USINT;
eMedium         : E_MBUS_Medium;
sMan            : STRING(3);
stValue1        : ST_MBus_Info;
stValue2        : ST_MBus_Info;
```

bBusy: The *bBusy* output is TRUE while the meter is being read.

bReady: The *bReady* output is TRUE for one cycle, once meter reading is completed.

bError: The *bError* output becomes TRUE as soon as an error occurs. The error is described via the variable *eError*.

eError: The *eError* output issues an error code when an error occurs (see E_MBUS_ERROR [▶ 216]).

dwIdNumber: Serial number of the meter (secondary address).

byStatus: Status of the meter. Please refer to device description for meanings.

byGEN: Meter software version.

byCounter: Transmission counter (number of transmitted RSP_UD).

usiRecivedAdr: Received primary address (0-250).

eMedium: Medium (see E_MBUS_Medium [▶ 219]).

sMan: Manufacturer code.

stValue1: Meter reading 1 (see ST_MBus_Info [▶ 222]).

stValue2: Meter reading 2 (see ST_MBus_Info [▶ 222]).

VAR_IN_OUT

```
stCom           : ST_MBUS_Communication;
```

stCom: About this structure, the block FB_MBUSKL6781() [▶ 29] is connected to the meter function blocks (see ST_MBUS_Communication [▶ 221]).

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055621259/.zip:

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055622667/.zip:

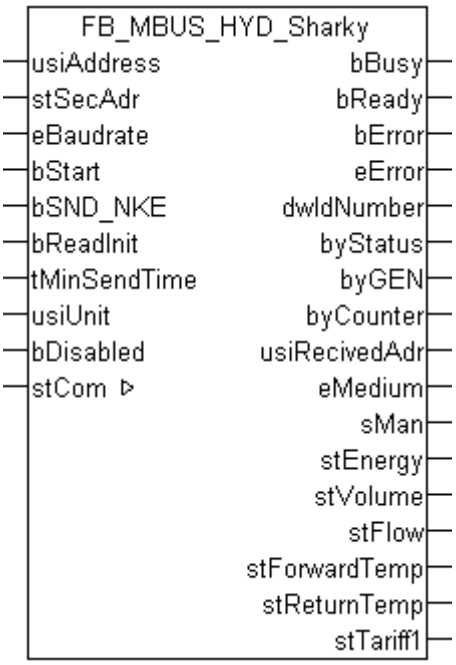
Controller configuration setting: "BCxx50 or BX serial"

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055624075/.zip:

BCxx00 must be set to "Large Model" in the PLC under "Project/Options/Controller Settings".

Controller configuration setting: "BC serial"

6.18.4 FB_MBUS_HYD_Sharky



This function block is used to read energy meters from:

Hydrometer:

-Sharky 773

-Sharky 775

-ENERGY INT 6

Brunata:

-Brunata Optuna H (775)

Aquametro:

-AMNTRONIC SONIC D

Only the most common values (see *VAR_OUTPUT*) of the telegrams:

00 (Application Reset-Subcode 00 / All)

10 (Application Reset-Subcode 10 / User data)

20 (Application Reset-Subcode 20 / Simple billing)

30 (Application Reset-Subcode 30 / Enhanced billing)

40 (Application Reset-Subcode 40 / Multi tariff billing)

or 50 (Application Reset-Subcode 50 / Instant values)

are read. The device is not switched to these telegrams; it must be set to one of these telegrams.

The function block FB_MBUS_HYD_Sharky_00() [▶ 120] can be used if further data are required, or the function block FB_MBUS_General_Send() [▶ 43] can be used to select the required telegram, and the function block FB_MBUS_General() [▶ 31] can be used to read all data of the respective telegram.

The function block can only be executed together with the function block FB_MBUSKL6781() [▶ 29].

Functionality of the function block [▶ 13]

VAR_INPUT

```
usiAddress      : USINT;
stSecAdr        : ST_MBUS_SecAdr;
eBaudrate       : E_MBUS_Baudrate := eMBUS_Baud2400;
bStart          : BOOL;
bSND_NKE        : BOOL := TRUE;
bReadInit       : BOOL := TRUE;
tMinSendTime    : TIME := t#2s;
usiUnit         : USINT;
bDisabled       : BOOL := FALSE;
```

usiAddress: Primary address [▶ 14] of the counter, that shall be readout with this module.

stSecAdr: Secondary address [▶ 15] of the counter, that shall be readout with this module.

eBaudrate: 300, 2400 baud [▶ 216].

bStart: Positive edge on this input, the meter is read out once.

bSND_NKE: TRUE to initialize the meter at each reading, and sets the meter on the first telegram (SND_NKE).

bReadInit: After restarting the PLC, the meter is read out once.

tMinSendTime: Standard t#2s. When this time exceeds, the counter is rereadout. At t#0s the counter is not readout and can be readout with bStart manually.

usiUnit: Unit of the energy values. 0=W/J(h) / 1=KW/KJ(h) / 2=MW/MJ(h) / 3=GW/GJ(h).

bDisabled: TRUE = disable the function block.

VAR_OUTPUT

```
bBusy           : BOOL;
bReady          : BOOL;
bError          : BOOL;
eError          : E_MBUS_ERROR;
dwIdNumber      : DWORD;
byStatus        : BYTE;
byGEN           : BYTE;
byCounter       : BYTE;
usiRecivedAdr   : USINT;
eMedium         : E_MBUS_Medium;
sMan            : STRING(3);
stEnergy        : ST_MBus_Info;
stVolume        : ST_MBus_Info;
stFlow          : ST_MBus_Info;
stForwardTemp   : ST_MBus_Info;
stReturnTemp    : ST_MBus_Info;
stTariff1       : ST_MBus_Info;
```

bBusy: The *bBusy* output is TRUE while the meter is being read.

bReady: The *bReady* output is TRUE for one cycle, once meter reading is completed.

bError: The *bError* output becomes TRUE as soon as an error occurs. The error is described via the variable *eError*.

eError: The *eError* output issues an error code when an error occurs (see E_MBUS_ERROR [▶ 216]).

dwIdNumber: Serial number of the meter (secondary address).

byStatus: Status of the meter. Please refer to device description for meanings.

byGEN: Meter software version.

byCounter: Transmission counter (number of transmitted RSP_UD).

usiRecivedAdr: Received primary address (0-250).

eMedium: Medium (see E_MBUS_Medium [▶ 219]).

sMan: Manufacturer code.

stEnergy: Meter reading, energy consumption (see ST_MBus_Info [▶ 222]).

stVolume: Meter reading, volume (see ST_MBus_Info [▶ 222]).

stFlow: Current flow (see ST_MBus_Info [▶ 222]).

stForwardTemp: Flow temperature (see ST_MBus_Info [▶ 222]).

stReturnTemp: Return temperature (see ST_MBus_Info [▶ 222]).

stTariff1: Energy tariff 1 (see ST_MBus_Info [▶ 222]).

VAR_IN_OUT

```
stCom          : ST_MBUS_Communication;
```

stCom: About this structure, the block FB_MBUSKL6781() [▶ 29] is connected to the meter function blocks (see ST_MBUS_Communication [▶ 221]).

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055621259/.zip:

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055622667/.zip:

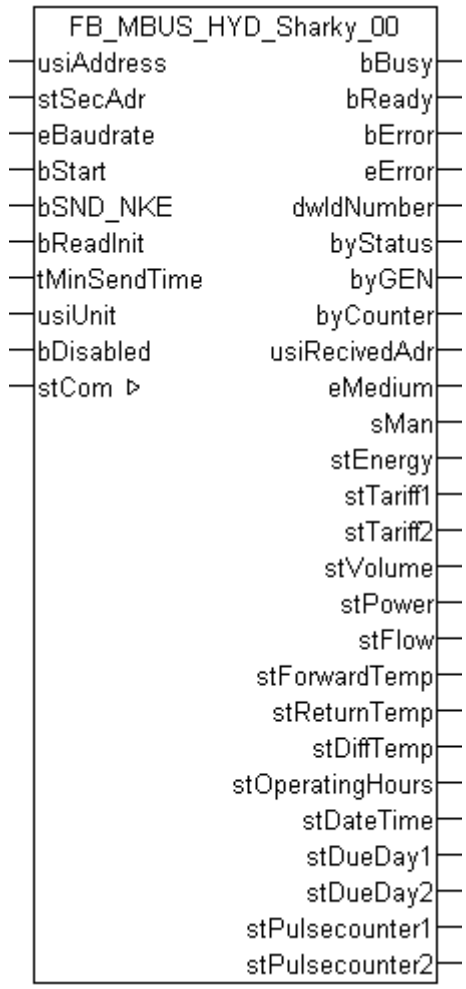
Controller configuration setting: "BCxx50 or BX serial"

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055624075/.zip:

BCxx00 must be set to "Large Model" in the PLC under "Project/Options/Controller Settings".

Controller configuration setting: "BC serial"

6.18.5 FB_MBUS_HYD_Sharky_00



This function block is used to read energy meters from:

Hydrometer:

-Sharky 773

-Sharky 775

-ENERGY INT 6

Brunata:

-Brunata Optuna H (775)

Aquametro:

-AMNTRONIC SONIC D

All values of telegram 00 (application reset subcode 00 / All) are read. The device automatically switches to the corresponding mode.

stPulsecounter1 and stPulsecounter2 are only output if the pulse module is connected.

If further telegrams are required, the function block FB_MBUS_General_Send() [▶ 43] can be used to select the required telegram, and the function block FB_MBUS_General() [▶ 31] can be used to read all data of the respective telegram.

The function block can only be executed together with the function block FB_MBUSKL6781() [▶ 29].

Functionality of the function block [▶ 13]

VAR_INPUT

```
usiAddress          : USINT;
stSecAdr            : ST_MBUS_SecAdr;
eBaudrate           : E_MBUS_Baudrate := eMBUS_Baud2400;
bStart              : BOOL;
bSND_NKE            : BOOL := TRUE;
bReadInit           : BOOL := TRUE;
tMinSendTime        : TIME := t#2s;
usiUnit             : USINT;
bDisabled           : BOOL := FALSE;
```

usiAddress: Primary address [▶ 14] of the counter, that shall be readout with this module.

stSecAdr: Secondary address [▶ 15] of the counter, that shall be readout with this module.

eBaudrate: 300, 2400 baud [▶ 216].

bStart: Positive edge on this input, the meter is read out once.

bSND_NKE: TRUE to initialize the meter at each reading, and sets the meter on the first telegram (SND_NKE).

bReadInit: After restarting the PLC, the meter is read out once.

tMinSendTime: Standard t#2s. When this time exceeds, the counter is rereadout. At t#0s the counter is not readout and can be readout with bStart manually.

usiUnit: Unit of the energy values. 0=W/J(h) / 1=KW/KJ(h) / 2=MW/MJ(h) / 3=GW/GJ(h).

bDisabled: TRUE = disable the function block.

VAR_OUTPUT

```
bBusy               : BOOL;
bReady              : BOOL;
bError              : BOOL;
eError              : E_MBUS_ERROR;
dwIdNumber          : DWORD;
byStatus            : BYTE;
byGEN               : BYTE;
byCounter           : BYTE;
usiRecivedAdr       : USINT;
eMedium             : E_MBUS_Medium;
sMan                : STRING(3);
stEnergy            : ST_MBus_Info;
stTariff1           : ST_MBus_Info;
stTariff2           : ST_MBus_Info;
stVolume            : ST_MBus_Info;
stPower             : ST_MBus_Info;
stFlow              : ST_MBus_Info;
stForwardTemp       : ST_MBus_Info;
stReturnTemp        : ST_MBus_Info;
stDiffTemp          : ST_MBus_Info;
stOperatingHours    : ST_MBus_Info;
stDateTime          : ST_MBus_Info;
stDueDay1           : ST_MBUS_DueDayHYD1;
stDueDay2           : ST_MBUS_DueDayHYD1;
stPulsecounter1     : ST_MBus_Info;
stPulsecounter2     : ST_MBus_Info;
```

bBusy: The *bBusy* output is TRUE while the meter is being read.

bReady: The *bReady* output is TRUE for one cycle, once meter reading is completed.

bError: The *bError* output becomes TRUE as soon as an error occurs. The error is described via the variable *eError*.

eError: The *eError* output issues an error code when an error occurs (see E_MBUS_ERROR [▶ 216]).

dwIdNumber: Serial number of the meter (secondary address).

byStatus: Status of the meter. Please refer to device description for meanings.

byGEN: Meter software version.

byCounter: Transmission counter (number of transmitted RSP_UD).

usiRecivedAdr: Received primary address (0-250).

eMedium: Medium (see E_MBUS_Medium [▶ 219]).

sMan: Manufacturer code.

stEnergy: Meter reading, energy consumption (see ST_MBus_Info [▶ 222]).

stTariff1: Meter reading, tariff 1 (see ST_MBus_Info [▶ 222]).

stTariff2: Meter reading, tariff 2 (see ST_MBus_Info [▶ 222]).

stVolume: Meter reading, volume (see ST_MBus_Info [▶ 222]).

stPower: Current energy consumption (see ST_MBus_Info [▶ 222]).

stFlow: Current flow (see ST_MBus_Info [▶ 222]).

stForwardTemp: Flow temperature (see ST_MBus_Info [▶ 222]).

stReturnTemp: Return temperature (see ST_MBus_Info [▶ 222]).

stDiffTemp: Temperature difference (see ST_MBus_Info [▶ 222]).

stOperatingHours: Operating hours (see ST_MBus_Info [▶ 222]).

stDateTime: Date / Tiime (see ST_MBus_Info [▶ 222]).

stDueDay1: Values due day 1 (see ST_MBUS_DueDayHYD1 [▶ 224]).

stDueDay2: Values due day 2 (see ST_MBUS_DueDayHYD1 [▶ 224]).

stPulsecounter1: Meter reading, pulse counter 1 (see ST_MBus_Info [▶ 222]).

stPulsecounter2: Meter reading, pulse counter 2 (see ST_MBus_Info [▶ 222]).

VAR_IN_OUT

```
stCom          : ST_MBUS_Communication;
```

stCom: About this structure, the block FB_MBUSKL6781() [▶ 29] is connected to the meter function blocks (see ST_MBUS_Communication [▶ 221]).

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055621259/.zip:

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055622667/.zip:

Controller configuration setting: "BCxx50 or BX serial"

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055624075/.zip:

BCxx00 must be set to "Large Model" in the PLC under "Project/Options/Controller Settings".

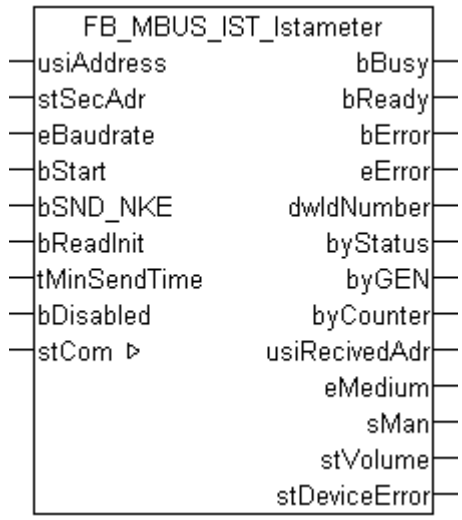
Controller configuration setting: "BC serial"

6.19 ista

The function blocks only output a selection of the most common data. These data are described on the respective pages under "VAR_OUT". If more or all data are required, the function blocks FB_MBUS_General [▶ 31], FB_MBUS_General_Ext [▶ 35] or FB_MBUS_General_Param [▶ 41] from the General folder should be used. Not that these function blocks do not run on BC or BX systems. The function block FB_MBUS_General_Send [▶ 43] can be used to send data to the device (e.g. setting of the primary address).

Vendor	Type	Device	Function block
ista	Water meter	domaqua® m	FB_MBUS_IST_Istameter [▶ 123]
	Water meter	istameter® m	
	Water meter	istameter III	FB_MBUS_IST_IstameterIII [▶ 125]
	Pulse counter	pulsonic II	FB_MBUS_IST_PulsonicII [▶ 127]
	Heat meter	sensonic II	FB_MBUS_IST_SensonicII [▶ 129]

6.19.1 FB_MBUS_IST_Istameter



This function block is used to read water meters from Ista:

-istameter® m

-domaqua® m

The function block can only be executed together with the function block FB_MBUSKL6781() [▶ 29].

The devices are supplied from a battery. The number of read operations is therefore limited. An internal meter prevents communication exceeding 96 times per day on average. The user must make sure that excessive queries are prevented.

Functionality of the function block [▶ 13]

VAR_INPUT

```
usiAddress      : USINT;
stSecAdr        : ST_MBUS_SecAdr;
eBaudrate       : E_MBUS_Baudrate := eMBUS_Baud2400;
bStart          : BOOL;
bSND_NKE        : BOOL := TRUE;
bReadInit       : BOOL := TRUE;
tMinSendTime    : TIME := t#2s;
bDisabled       : BOOL := FALSE;
```

usiAddress: Primary address [▶ 14] of the counter, that shall be readout with this module.

stSecAdr: Secondary address [▶ 15] of the counter, that shall be readout with this module.

eBaudrate: 300, 2400 baud [▶ 216].

bStart: Positive edge on this input, the meter is read out once.

bSND_NKE: TRUE to initialize the meter at each reading, and sets the meter on the first telegram (SND_NKE).

bReadInit: After restarting the PLC, the meter is read out once.

tMinSendTime: Standard t#2s. When this time exceeds, the counter is rereadout. At t#0s the counter is not readout and can be readout with bStart manually.

bDisabled: TRUE = disable the function block.

VAR_OUTPUT

```
bBusy           : BOOL;
bReady          : BOOL;
bError          : BOOL;
eError          : E_MBUS_ERROR;
dwIdNumber      : DWORD;
byStatus        : BYTE;
byGEN           : BYTE;
byCounter       : BYTE;
usiRecivedAdr   : USINT;
eMedium         : E_MBUS_Medium;
sMan            : STRING(3);
stVolume        : ST_MBus_Info;
stDeviceError   : ST_MBus_Info;
```

bBusy: The *bBusy* output is TRUE while the meter is being read.

bReady: The *bReady* output is TRUE for one cycle, once meter reading is completed.

bError: The *bError* output becomes TRUE as soon as an error occurs. The error is described via the variable *eError*.

eError: The *eError* output issues an error code when an error occurs (see E_MBUS_ERROR [▶ 216]).

dwIdNumber: Serial number of the meter (secondary address).

byStatus: Status of the meter. Please refer to device description for meanings.

byGEN: Meter software version.

byCounter: Transmission counter (number of transmitted RSP_UD).

usiRecivedAdr: Received primary address (0-250).

eMedium: Medium (see E_MBUS_Medium [▶ 219]).

sMan: Manufacturer code.

stVolume: Meter reading, volume (see ST_MBus_Info [▶ 222]).

stDeviceError: Error message from the device (see ST_MBus_Info [▶ 222]).

VAR_IN_OUT

```
stCom           : ST_MBUS_Communication;
```

stCom: About this structure, the block FB_MBUSKL6781() [▶ 29] is connected to the meter function blocks (see ST_MBUS_Communication [▶ 221]).

Download sample program for PC/CX systems: https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055625483/.zip

Download sample program for BX systems: https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055626891/.zip

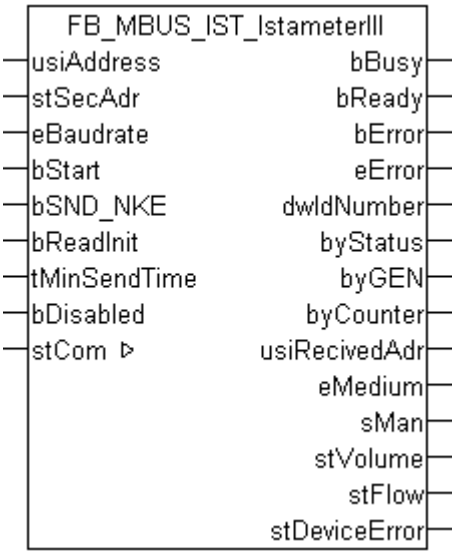
Controller configuration setting: "BCxx50 or BX serial"

Download sample program for BC systems: https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055628299/.zip

BCxx00 must be set to "Large Model" in the PLC under "Project/Options/Controller Settings".

Controller configuration setting: "BC serial"

6.19.2 FB_MBUS_IST_IstameterIII



This function block is used to read water meters from Ista:

-istameter III

The function block can only be executed together with the function block FB_MBUSKL6781() [▶ 29].

The devices are supplied from a battery. The number of read operations is therefore limited. An internal meter prevents communication exceeding 96 times per day on average. The user must make sure that excessive queries are prevented.

Functionality of the function block [▶ 13]

VAR_INPUT

```
usiAddress      : USINT;
stSecAdr        : ST_MBUS_SecAdr;
eBaudrate       : E_MBUS_Baudrate := eMBUS_Baud2400;
bStart          : BOOL;
bSND_NKE        : BOOL := TRUE;
bReadInit       : BOOL := TRUE;
tMinSendTime    : TIME := t#2s;
bDisabled       : BOOL := FALSE;
```

usiAddress: Primary address [▶ 14] of the counter, that shall be readout with this module.

stSecAdr: Secondary address [▶ 15] of the counter, that shall be readout with this module.

eBaudrate: 300, 2400 baud [▶ 216].

bStart: Positive edge on this input, the meter is read out once.

bSND_NKE: TRUE to initialize the meter at each reading, and sets the meter on the first telegram (SND_NKE).

bReadInit: After restarting the PLC, the meter is read out once.

tMinSendTime: Standard t#2s. When this time exceeds, the counter is rereadout. At t#0s the counter is not readout and can be readout with bStart manually.

bDisabled: TRUE = disable the function block.

VAR_OUTPUT

```
bBusy          : BOOL;
bReady         : BOOL;
bError         : BOOL;
eError         : E_MBUS_ERROR;
dwIdNumber     : DWORD;
byStatus       : BYTE;
byGEN          : BYTE;
byCounter      : BYTE;
usiRecivedAdr  : USINT;
eMedium        : E_MBUS_Medium;
sMan           : STRING(3);
stVolume       : ST_MBus_Info;
stFlow         : ST_MBus_Info;
stDeviceError  : ST_MBus_Info;
```

bBusy: The *bBusy* output is TRUE while the meter is being read.

bReady: The *bReady* output is TRUE for one cycle, once meter reading is completed.

bError: The *bError* output becomes TRUE as soon as an error occurs. The error is described via the variable *eError*.

eError: The *eError* output issues an error code when an error occurs (see E_MBUS_ERROR [▶ 216]).

dwIdNumber: Serial number of the meter (secondary address).

byStatus: Status of the meter. Please refer to device description for meanings.

byGEN: Meter software version.

byCounter: Transmission counter (number of transmitted RSP_UD).

usiRecivedAdr: Received primary address (0-250).

eMedium: Medium (see E_MBUS_Medium [▶ 219]).

sMan: Manufacturer code.

stVolume: Meter reading, volume (see ST_MBus_Info [▶ 222]).

stFlow: Current flow (see ST_MBus_Info [▶ 222]).

stDeviceError: Error message from the device (see ST_MBus_Info [▶ 222]).

VAR_IN_OUT

```
stCom          : ST_MBUS_Communication;
```

stCom: About this structure, the block FB_MBUSKL6781() [▶ 29] is connected to the meter function blocks (see ST_MBUS_Communication [▶ 221]).

Download sample program for PC/CX systems: https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055625483/.zip

Download sample program for BX systems: https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055626891/.zip

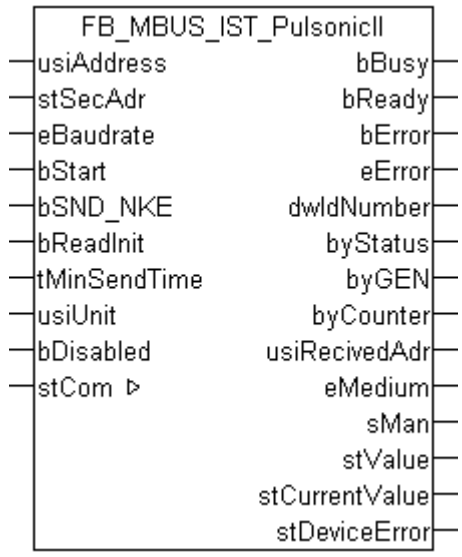
Controller configuration setting: "BCxx50 or BX serial"

Download sample program for BC systems: https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055628299/.zip

BCxx00 must be set to "Large Model" in the PLC under "Project/Options/Controller Settings".

Controller configuration setting: "BC serial"

6.19.3 FB_MBUS_IST_PulsonicII



This function block is used to read energy meters with pulse output from Ista:

-Pulsonic II

The function block can only be executed together with the function block FB_MBUSKL6781() [▶ 29].

The devices are supplied from a battery. The number of read operations is therefore limited. An internal meter prevents communication exceeding 96 times per day on average. The user must make sure that excessive queries are prevented.

Functionality of the function block [▶ 13]

VAR_INPUT

```
usiAddress      : USINT;
stSecAdr        : ST_MBUS_SecAdr;
eBaudrate       : E_MBUS_Baudrate := eMBUS_Baud2400;
bStart          : BOOL;
bSND_NKE        : BOOL := TRUE;
bReadInit       : BOOL := TRUE;
tMinSendTime    : TIME := t#2s;
usiUnit         : USINT;
bDisabled       : BOOL := FALSE;
```

usiAddress: Primary address [▶ 14] of the counter, that shall be readout with this module.

stSecAdr: Secondary address [▶ 15] of the counter, that shall be readout with this module.

eBaudrate: 300, 2400 baud [▶ 216].

bStart: Positive edge on this input, the meter is read out once.

bSND_NKE: TRUE to initialize the meter at each reading, and sets the meter on the first telegram (SND_NKE).

bReadInit: After restarting the PLC, the meter is read out once.

tMinSendTime: Standard t#2s. When this time exceeds, the counter is rereadout. At t#0s the counter is not readout and can be readout with bStart manually.

usiUnit: Unit of the energy values. 0=W/J(h) / 1=KW/KJ(h) / 2=MW/MJ(h) / 3=GW/GJ(h).

bDisabled: TRUE = disable the function block.

VAR_OUTPUT

```
bBusy           : BOOL;
bReady          : BOOL;
bError          : BOOL;
eError          : E_MBUS_ERROR;
dwIdNumber      : DWORD;
byStatus        : BYTE;
byGEN           : BYTE;
byCounter       : BYTE;
usiRecivedAdr   : USINT;
eMedium         : E_MBUS_Medium;
sMan            : STRING(3);
stValue         : ST_MBus_Info;
stCurrentValue  : ST_MBus_Info;
stDeviceError   : ST_MBus_Info;
```

bBusy: The *bBusy* output is TRUE while the meter is being read.

bReady: The *bReady* output is TRUE for one cycle, once meter reading is completed.

bError: The *bError* output becomes TRUE as soon as an error occurs. The error is described via the variable *eError*.

eError: The *eError* output issues an error code when an error occurs (see E_MBUS_ERROR [▶ 216]).

dwIdNumber: Serial number of the meter (secondary address).

byStatus: Status of the meter. Please refer to device description for meanings.

byGEN: Meter software version.

byCounter: Transmission counter (number of transmitted RSP_UD).

usiRecivedAdr: Received primary address (0-250).

eMedium: Medium (see E_MBUS_Medium [▶ 219]).

sMan: Manufacturer code.

stValue: Meter reading (see ST_MBus_Info [▶ 222]).

stCurrentValue: Current flow / power (see ST_MBus_Info [▶ 222]).

stDeviceError: Error message from the device (see ST_MBus_Info [▶ 222]).

VAR_IN_OUT

```
stCom           : ST_MBUS_Communication;
```

stCom: About this structure, the block FB_MBUSKL6781() [▶ 29] is connected to the meter function blocks (see ST_MBUS_Communication [▶ 221]).

Download sample program for PC/CX systems: https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055625483/.zip

Download sample program for BX systems: https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055626891/.zip

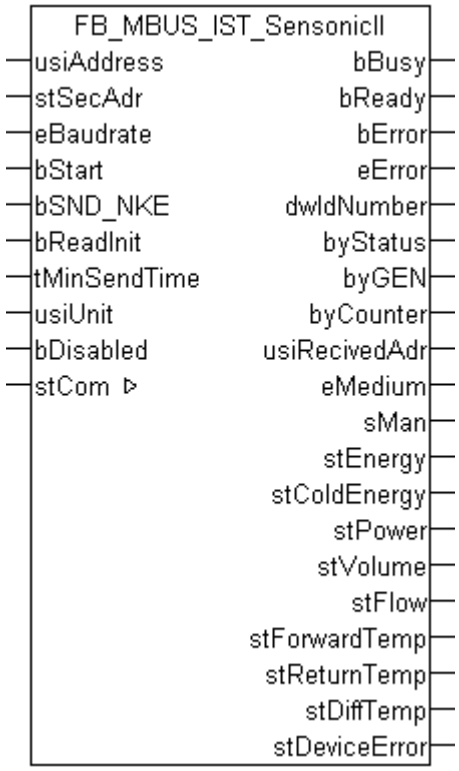
Controller configuration setting: "BCxx50 or BX serial"

Download sample program for BC systems: https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055628299/.zip

BCxx00 must be set to "Large Model" in the PLC under "Project/Options/Controller Settings".

Controller configuration setting: "BC serial"

6.19.4 FB_MBUS_IST_SensonicII



This function block is used to read heat meters from Ista:

-Sensonic II

The function block can only be executed together with the function block FB_MBUSKL6781() [▶ 29].

The devices are supplied from a battery. The number of read operations is therefore limited. An internal meter prevents communication exceeding 96 times per day on average. The user must make sure that excessive queries are prevented.

Functionality of the function block [▶ 13]

VAR_INPUT

```
usiAddress      : USINT;
stSecAdr        : ST_MBUS_SecAdr;
eBaudrate       : E_MBUS_Baudrate := eMBUS_Baud2400;
bStart          : BOOL;
bSND_NKE        : BOOL := TRUE;
bReadInit       : BOOL := TRUE;
tMinSendTime    : TIME := t#2s;
usiUnit         : USINT;
bDisabled       : BOOL := FALSE;
```

usiAddress: Primary address [▶ 14] of the counter, that shall be readout with this module.

stSecAdr: Secondary address [▶ 15] of the counter, that shall be readout with this module.

eBaudrate: 300, 2400 baud [▶ 216].

bStart: Positive edge on this input, the meter is read out once.

bSND_NKE: TRUE to initialize the meter at each reading, and sets the meter on the first telegram (SND_NKE).

bReadInit: After restarting the PLC, the meter is read out once.

tMinSendTime: Standard t#2s. When this time exceeds, the counter is rereadout. At t#0s the counter is not readout and can be readout with bStart manually.

usiUnit: Unit of the energy values. 0=W/J(h) / 1=KW/KJ(h) / 2=MW/MJ(h) / 3=GW/GJ(h).

bDisabled: TRUE = disable the function block.

VAR_OUTPUT

```
bBusy          : BOOL;
bReady         : BOOL;
bError         : BOOL;
eError         : E_MBUS_ERROR;
dwIdNumber     : DWORD;
byStatus       : BYTE;
byGEN          : BYTE;
byCounter      : BYTE;
usiRecivedAdr  : USINT;
eMedium        : E_MBUS_Medium;
sMan           : STRING(3);
stEnergy       : ST_MBus_Info;
stColdEnergy   : ST_MBus_Info;
stPower        : ST_MBus_Info;
stVolume       : ST_MBus_Info;
stFlow         : ST_MBus_Info;
stForwardTemp  : ST_MBus_Info;
stReturnTemp   : ST_MBus_Info;
stDiffTemp     : ST_MBus_Info;
stDeviceError  : ST_MBus_Info;
```

bBusy: The *bBusy* output is TRUE while the meter is being read.

bReady: The *bReady* output is TRUE for one cycle, once meter reading is completed.

bError: The *bError* output becomes TRUE as soon as an error occurs. The error is described via the variable *eError*.

eError: The *eError* output issues an error code when an error occurs (see E_MBUS_ERROR [▶ 216]).

dwIdNumber: Serial number of the meter (secondary address).

byStatus: Status of the meter. Please refer to device description for meanings.

byGEN: Meter software version.

byCounter: Transmission counter (number of transmitted RSP_UD).

usiRecivedAdr: Received primary address (0-250).

eMedium: Medium (see E_MBUS_Medium [▶ 219]).

sMan: Manufacturer code.

stEnergy: Meter reading, energy consumption (see ST_MBus_Info [▶ 222]).

stColdEnergy: Meter reading, cold energy consumption (see ST_MBus_Info [▶ 222]).

stPower: Current energy consumption (see ST_MBus_Info [▶ 222]).

stVolume: Meter reading, volume (see ST_MBus_Info [▶ 222]).

stFlow: Current flow (see ST_MBus_Info [▶ 222]).

stForwardTemp: Flow temperature (see ST_MBus_Info [▶ 222]).

stReturnTemp: Return temperature (see ST_MBus_Info [▶ 222]).

stDiffTemp: Temperature difference (see ST_MBus_Info [▶ 222]).

stDeviceError: Error message from the device (see ST_MBus_Info [▶ 222]).

VAR_IN_OUT

```
stCom          : ST_MBUS_Communication;
```

stCom: About this structure, the block FB_MBUSKL6781() [▶ 29] is connected to the meter function blocks (see ST_MBUS_Communication [▶ 221]).

Download sample program for PC/CX systems: https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055625483/.zip

Download sample program for BX systems: https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055626891/.zip

Controller configuration setting: "BCxx50 or BX serial"

Download sample program for BC systems: https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055628299/.zip

BCxx00 must be set to "Large Model" in the PLC under "Project/Options/Controller Settings".

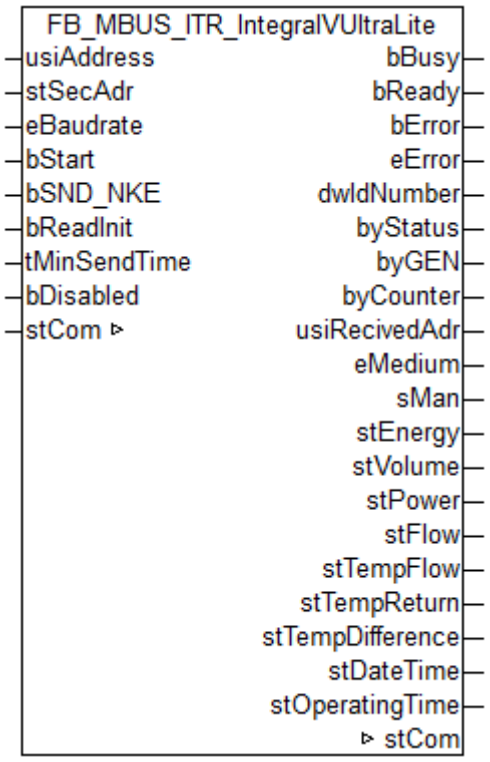
Controller configuration setting: "BC serial"

6.20 Itron

The function blocks only output a selection of the most common data. These data are described on the respective pages under "VAR_OUT". If more or all data are required, the function blocks FB_MBUS_General [▶ 31], FB_MBUS_General_Ext [▶ 35] or FB_MBUS_General_Param [▶ 41] from the General folder should be used. Not that these function blocks do not run on BC or BX systems. The function block FB_MBUS_General_Send [▶ 43] can be used to send data to the device (e.g. setting of the primary address).

Vendor	Type	Device	Function block
Itron	Energy meter	Integral-V UltraLite	FB_MBUS_ITR_IntegralVUltraLite [▶ 132]

6.20.1 FB_MBUS_ITR_IntegralVUltraLite



This module is used to readout energy meters of the manufacturer Itron.

It can only be used together with the module FB_MBUSKL6781() [▶ 29].

Functionality of the module [▶ 13]

VAR_INPUT

```
usiAddress      : USINT;
stSecAdr        : ST_MBUS_SecAdr;
eBaudrate       : E_MBUS_Baudrate := eMBUS_Baud2400;
bStart          : BOOL;
bSND_NKE        : BOOL := TRUE;
bReadInit       : BOOL := TRUE;
tMinSendTime    : TIME := t#2s;
bDisabled       : BOOL := FALSE;
```

usiAddress: Primary address [▶ 14] of the counter, that shall be readout with this module.

stSecAdr: Secondary address [▶ 15] of the counter, that shall be readout with this module.

eBaudrate: 300, 2400 baud [▶ 216].

bStart: Positive edge on this input, the meter is read out once.

bSND_NKE: TRUE to initialize the meter at each reading, and sets the meter on the first telegram (SND_NKE).

bReadInit: After restarting the PLC, the meter is read out once.

tMinSendTime: Standard t#2s. When this time exceeds, the counter is rereadout. At t#0s the counter is not readout and can be readout with bStart manually.

bDisabled: TRUE = disable the function block.

VAR_OUTPUT

```
bBusy           : BOOL;
bReady          : BOOL;
bError          : BOOL;
```

```
eError          : E_MBUS_ERROR;
dwIdNumber      : DWORD;
byStatus        : BYTE;
byGEN           : BYTE;
byCounter       : BYTE;
usiRecivedAdr   : USINT;
eMedium         : E_MBUS_Medium;
sMan            : STRING(3);
stEnergy        : ST_MBus_Info;
stVolume        : ST_MBus_Info;
stPower         : ST_MBus_Info;
stFlow          : ST_MBus_Info;
stTempFlow      : ST_MBus_Info;
stTempReturn    : ST_MBus_Info;
stTempDifference: ST_MBus_Info;
stDateTime      : ST_MBus_Info;
stOperatingTime : ST_MBus_Info;
```

bBusy: The *bBusy* output is TRUE while the meter is being read.

bReady: The *bReady* output is TRUE for one cycle, once meter reading is completed.

bError: The *bError* output becomes TRUE as soon as an error occurs. The error is described via the variable *eError*.

eError: The *eError* output issues an error code when an error occurs (see E_MBUS_ERROR [▶ 216]).

dwIdNumber: Serial number of the counter (secondary address).

byStatus: Status of the counter. Please refer to device description for meanings.

byGEN: Counter software version.

byCounter: Transmission counter (number of transmitted RSP_UD).

usiRecivedAdr: Received primary address (0-250).

eMedium: Medium (see E_MBUS_Medium [▶ 219]).

sMan: Manufacturer code.

stEnergy: Actual energy (see ST_MBus_Info [▶ 222]).

stVolume: Actual volume (see ST_MBus_Info [▶ 222]).

stPower: Actual power (see ST_MBus_Info [▶ 222]).

stFlow: Actual flow rate (see ST_MBus_Info [▶ 222]).

stTempFlow: Actual flow temperature (see ST_MBus_Info [▶ 222]).

stTempReturn: Actual return temperature (see ST_MBus_Info [▶ 222]).

stTempDifference: Actual temperature difference (see ST_MBus_Info [▶ 222]).

stDateTime: Date and time (see ST_MBus_Info [▶ 222]).

stOperatingTime: Operating time (see ST_MBus_Info [▶ 222]).

VAR_IN_OUT

```
stCom           : ST_MBUS_Communication;
```

stCom: About this structure, the block FB_MBUSKL6781() [▶ 29] is connected to the meter function blocks (see ST_MBUS_Communication [▶ 221]).

Requirements

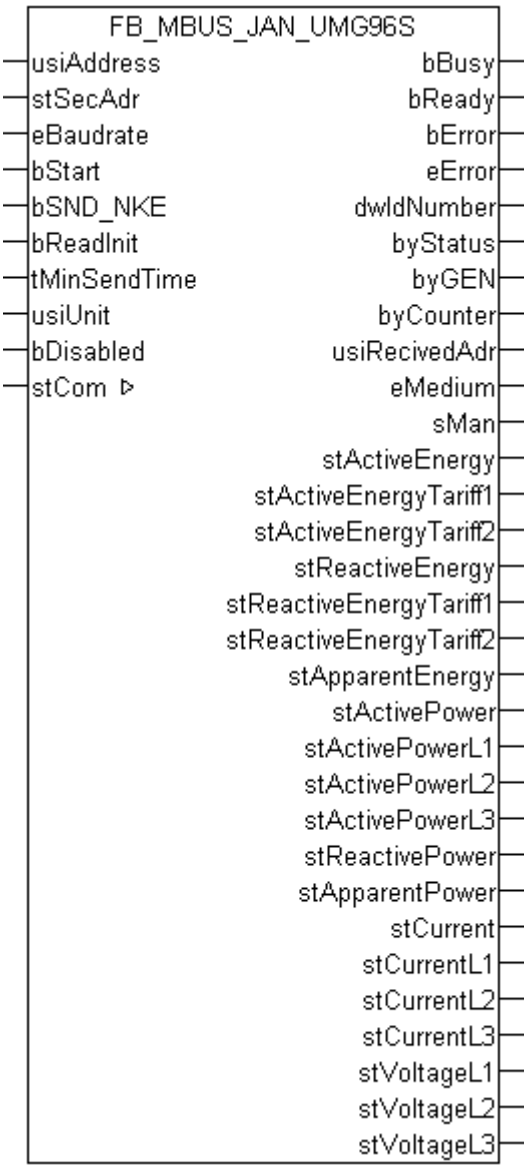
Development environment	Target system	Required libraries
TwinCAT 2.11 R3/x64 from build 2256	PC/CX, BX or BC	TcMBus library from V2.8.0

6.21 Janitza

The function blocks only output a selection of the most common data. These data are described on the respective pages under "VAR_OUT". If more or all data are required, the function blocks FB_MBUS_General [▶ 31], FB_MBUS_General_Ext [▶ 35] or FB_MBUS_General_Param [▶ 41] from the General folder should be used. Not that these function blocks do not run on BC or BX systems. The function block FB_MBUS_General_Send [▶ 43] can be used to send data to the device (e.g. setting of the primary address).

Vendor	Type	Device	Function block
Janitza	Electricity meter	UMG96S	FB_MBUS_JAN_UMG96S [▶ 134]

6.21.1 FB_MBUS_JAN_UMG96S



This block is used for reading electricity meters from Janitza:

-UMG96S

The block can only be used in conjunction with the block FB_MBUSKL6781() [▶ 29].

Functionality of the function block [▶ 13]

VAR_INPUT

```
usiAddress          : USINT;
stSecAdr            : ST_MBUS_SecAdr;
eBaudrate           : E_MBUS_Baudrate := eMBUS_Baud2400;
bStart              : BOOL;
bSND_NKE            : BOOL := TRUE;
bReadInit           : BOOL := TRUE;
tMinSendTime        : TIME := t#2s;
usiUnit             : USINT;
bDisabled           : BOOL := FALSE;
```

usiAddress: Primary address [▶ 14] of the counter, that shall be readout with this module.

stSecAdr: Secondary address [▶ 15] of the counter, that shall be readout with this module.

eBaudrate: 300, 2400, 9600 baud [▶ 216].

bStart: Positive edge on this input, the meter is read out once.

bSND_NKE: TRUE to initialize the meter at each reading, and sets the meter on the first telegram (SND_NKE).

bReadInit: After restarting the PLC, the meter is read out once.

tMinSendTime: Standard t#2s. When this time exceeds, the counter is rereadout. At t#0s the counter is not readout and can be readout with bStart manually.

usiUnit: Unit of the energy values. 0=W/J(h) / 1=KW/KJ(h) / 2=MW/MJ(h) / 3=GW/GJ(h).

bDisabled: TRUE = disable the function block.

VAR_OUTPUT

```
bBusy                   : BOOL;
bReady                  : BOOL;
bError                  : BOOL;
eError                  : E_MBUS_ERROR;
dwIdNumber              : DWORD;
byStatus                : BYTE;
byGEN                   : BYTE;
byCounter               : BYTE;
usiRecivedAdr           : USINT;
eMedium                 : E_MBUS_Medium;
sMan                    : STRING(3);
stActiveEnergy          : ST_MBus_Info;
stActiveEnergyTariff1   : ST_MBus_Info;
stActiveEnergyTariff2   : ST_MBus_Info;
stReactiveEnergy        : ST_MBus_Info;
stReactiveEnergyTariff1 : ST_MBus_Info;
stReactiveEnergyTariff2 : ST_MBus_Info;
stApparentEnergy        : ST_MBus_Info;
stActivePower           : ST_MBus_Info;
stActivePowerL1         : ST_MBus_Info;
stActivePowerL2         : ST_MBus_Info;
stActivePowerL3         : ST_MBus_Info;
stReactivePower         : ST_MBus_Info;
stApparentPower         : ST_MBus_Info;
stCurrent               : ST_MBus_Info;
stCurrentL1             : ST_MBus_Info;
stCurrentL2             : ST_MBus_Info;
stCurrentL3             : ST_MBus_Info;
stVoltageL1             : ST_MBus_Info;
stVoltageL2             : ST_MBus_Info;
stVoltageL3             : ST_MBus_Info;
```

bBusy: The *bBusy* output is TRUE while the meter is being read.

bReady: The *bReady* output is TRUE for one cycle, once meter reading is completed.

bError: The *bError* output becomes TRUE as soon as an error occurs. The error is described via the variable *eError*.

eError: The *eError* output issues an error code when an error occurs (see E_MBUS_ERROR [▶ 216]).

dwIdNumber: Serial number of the meter (secondary address).

byStatus: Status of the meter. Please refer to device description for meanings.

byGEN: Meter software version.

byCounter: Transmission counter (number of transmitted RSP_UD).

usiRecivedAdr: Received primary address (0-250).

eMedium: Medium (see E_MBUS_Medium [▶ 219]).

sMan: Manufacturer code.

stActiveEnergy: Meter reading, active energy (telegram 2, data point 14) (see ST_MBus_Info [▶ 222]).

stActiveEnergyTariff1: Meter reading, active energy, tariff 1(telegram 2, data point 15) (see ST_MBus_Info [▶ 222]).

stActiveEnergyTariff2: Meter reading, active energy, tariff 2(telegram 2, data point 16) (see ST_MBus_Info [▶ 222]).

stReactiveEnergy: Meter reading, reactive energy (telegram 2, data point 17) (see ST_MBus_Info [▶ 222]).

stReactiveEnergyTariff1: Meter reading, reactive energy, tariff 1(telegram 2, data point 18) (see ST_MBus_Info [▶ 222]).

stReactiveEnergyTariff2: Meter reading, reactive energy, tariff 2(telegram 2, data point 19) (see ST_MBus_Info [▶ 222]).

stApparentEnergy: Meter reading, apparent energy (telegram 2, data point 20) (see ST_MBus_Info [▶ 222]).

stActivePower: Active power (telegram 2, data point 29) (see ST_MBus_Info [▶ 222]).

stActivePowerL1: Active power phase L1 (telegram 2, data point 38) (see ST_MBus_Info [▶ 222]).

stActivePowerL2: Active power phase L2 (telegram 2, data point 39) (see ST_MBus_Info [▶ 222]).

stActivePowerL3: Active power phase L3 (telegram 2, data point 40) (see ST_MBus_Info [▶ 222]).

stReactivePower: Reactive power (telegram 2, data point 30) (see ST_MBus_Info [▶ 222]).

stApparentPower: Apparent power (telegram 2, data point 31) (see ST_MBus_Info [▶ 222]).

stCurrent: Current (telegram 2, data point 28) (see ST_MBus_Info [▶ 222]).

stCurrentL1: Current phase L1 (telegram 2, data point 35) (see ST_MBus_Info [▶ 222]).

stCurrentL2: Current phase L2 (telegram 2, data point 36) (see ST_MBus_Info [▶ 222]).

stCurrentL3: Current phase L3 (telegram 2, data point 37) (see ST_MBus_Info [▶ 222]).

stVoltageL1: Voltage phase L1 (telegram 2, data point 32) (see ST_MBus_Info [▶ 222]).

stVoltageL2: Voltage phase L2 (telegram 2, data point 33) (see ST_MBus_Info [▶ 222]).

stVoltageL3: Voltage phase L3 (telegram 2, data point 34) (see ST_MBus_Info [▶ 222]).

VAR_IN_OUT

```
stCom            : ST_MBUS_Communication;
```

stCom: About this structure, the block FB_MBUSKL6781() [▶ 29] is connected to the meter function blocks (see ST_MBUS_Communication [▶ 221]).

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055629707/.zip:

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055631115/.zip:

Controller configuration setting: "BCxx50 or BX serial"

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055632523/.zip:

BCxx00 must be set to "Large Model" in the PLC under "Project/Options/Controller Settings".

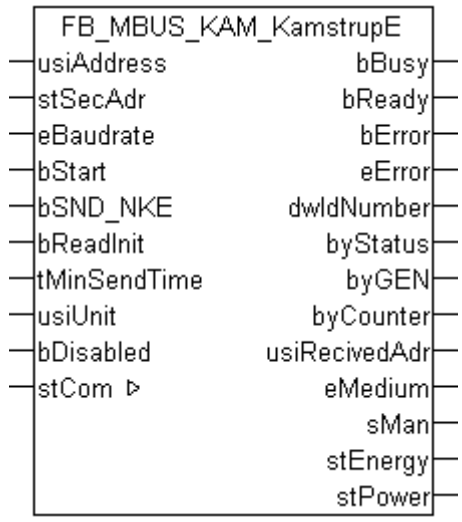
Controller configuration setting: "BC serial"

6.22 Kamstrup

The function blocks only output a selection of the most common data. These data are described on the respective pages under "VAR_OUT". If more or all data are required, the function blocks FB_MBUS_General [▶ 31], FB_MBUS_General_Ext [▶ 35] or FB_MBUS_General_Param [▶ 41] from the General folder should be used. Not that these function blocks do not run on BC or BX systems. The function block FB_MBUS_General_Send [▶ 43] can be used to send data to the device (e.g. setting of the primary address).

Vendor	Type	Device	Function block
Kamstrup [▶ 137]	Electricity meter	Kamstrup 162	FB_MBUS_KAM_KamstrupE [▶ 138]
	Electricity meter	Kamstrup 351	
	Electricity meter	Kamstrup 382	
	Heat/cold meter	Maxical III	FB_MBUS_KAM_Maxical_III [▶ 140]
	Heat/cold meter	Multical 401	FB_MBUS_KAM_Multical [▶ 142]
	Heat/cold meter	Multical 402	FB_MBUS_KAM_Multical402 [▶ 144]
	Water meter	Multical 41	FB_MBUS_KAM_Multical41 [▶ 147]
	Heat/cold meter	Multical 601	FB_MBUS_KAM_Multical601 [▶ 149]

6.22.1 FB_MBUS_KAM_KamstrupE



This block is used for reading electricity meters from Kamstrup:

-Kamstrup 162

-Kamstrup 351

-Kamstrup 382

The block can only be used in conjunction with the block FB_MBUSKL6781() [▶ 29].

Functionality of the function block [▶ 13]

VAR_INPUT

```
usiAddress      : USINT;
stSecAdr        : ST_MBUS_SecAdr;
eBaudrate       : E_MBUS_Baudrate := eMBUS_Baud2400;
bStart          : BOOL;
bSND_NKE        : BOOL := TRUE;
bReadInit       : BOOL := TRUE;
tMinSendTime    : TIME := t#2s;
usiUnit         : USINT;
bDisabled       : BOOL := FALSE;
```

usiAddress: Primary address [▶ 14] of the counter, that shall be readout with this module.

stSecAdr: Secondary address [▶ 15] of the counter, that shall be readout with this module.

eBaudrate: 300, 2400 baud [▶ 216].

bStart: Positive edge on this input, the meter is read out once.

bSND_NKE: TRUE to initialize the meter at each reading, and sets the meter on the first telegram (SND_NKE).

bReadInit: After restarting the PLC, the meter is read out once.

tMinSendTime: Standard t#2s. When this time exceeds, the counter is rereadout. At t#0s the counter is not readout and can be readout with bStart manually.

usiUnit: Unit of the energy values. 0=W/J(h) / 1=KW/KJ(h) / 2=MW/MJ(h) / 3=GW/GJ(h).

bDisabled: TRUE = disable the function block.

VAR_OUTPUT

```
bBusy           : BOOL;
bReady          : BOOL;
bError          : BOOL;
eError          : E_MBUS_ERROR;
dwIdNumber      : DWORD;
byStatus        : BYTE;
byGEN           : BYTE;
byCounter       : BYTE;
usiRecivedAdr   : USINT;
eMedium         : E_MBUS_Medium;
sMan            : STRING(3);
stEnergy        : ST_MBus_Info;
stPower         : ST_MBus_Info;
```

bBusy: The *bBusy* output is TRUE while the meter is being read.

bReady: The *bReady* output is TRUE for one cycle, once meter reading is completed.

bError: The *bError* output becomes TRUE as soon as an error occurs. The error is described via the variable *eError*.

eError: The *eError* output issues an error code when an error occurs (see E_MBUS_ERROR [▶ 216]).

dwIdNumber: Serial number of the meter (secondary address).

byStatus: Status of the meter. Please refer to device description for meanings.

byGEN: Meter software version.

byCounter: Transmission counter (number of transmitted RSP_UD).

usiRecivedAdr: Received primary address (0-250).

eMedium: Medium (see E_MBUS_Medium [▶ 219]).

sMan: Manufacturer code.

stEnergy: Meter reading, energy consumption (see ST_MBus_Info [▶ 222]).

stPower: Current energy consumption (see ST_MBus_Info [▶ 222]).

VAR_IN_OUT

```
stCom           : ST_MBUS_Communication;
```

stCom: About this structure, the block FB_MBUSKL6781() [▶ 29] is connected to the meter function blocks (see ST_MBUS_Communication [▶ 221]).

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055633931/.zip:

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055635339/.zip:

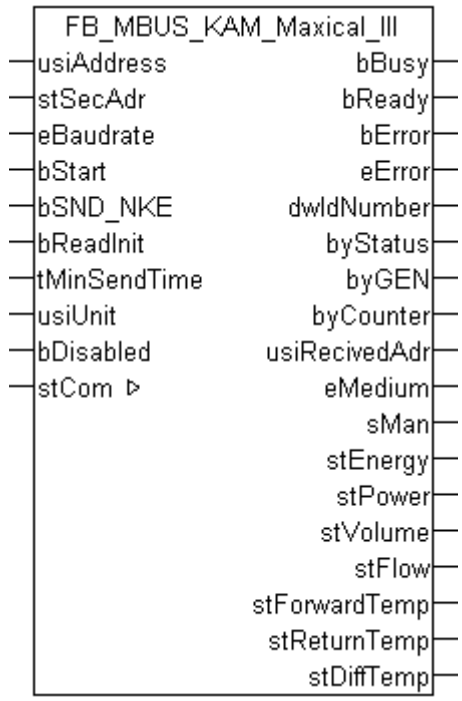
Controller configuration setting: "BCxx50 or BX serial"

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055636747/.zip:

BCxx00 must be set to "Large Model" in the PLC under "Project/Options/Controller Settings".

Controller configuration setting: "BC serial"

6.22.2 FB_MBUS_KAM_Maxical_III



This function block is used for reading heat/cold meters from Kamstrup:

-Maxical III

The function block can only be executed together with the function block FB_MBUSKL6781() [▶ 29].

Functionality of the function block [▶ 13]

VAR_INPUT

```
usiAddress      : USINT;
stSecAdr        : ST_MBUS_SecAdr;
eBaudrate       : E_MBUS_Baudrate := eMBUS_Baud2400;
bStart          : BOOL;
bSND_NKE        : BOOL := TRUE;
bReadInit       : BOOL := TRUE;
tMinSendTime    : TIME := t#2s;
usiUnit         : USINT;
bDisabled       : BOOL := FALSE;
```

usiAddress: Primary address [▶ 14] of the counter, that shall be readout with this module.

stSecAdr: Secondary address [▶ 15] of the counter, that shall be readout with this module.

eBaudrate: 300, 2400 baud [▶ 216].

bStart: Positive edge on this input, the meter is read out once.

bSND_NKE: TRUE to initialize the meter at each reading, and sets the meter on the first telegram (SND_NKE).

bReadInit: After restarting the PLC, the meter is read out once.

tMinSendTime: Standard t#2s. When this time exceeds, the counter is rereadout. At t#0s the counter is not readout and can be readout with bStart manually.

usiUnit: Unit of the energy values. 0=W/J(h) / 1=KW/KJ(h) / 2=MW/MJ(h) / 3=GW/GJ(h).

bDisabled: TRUE = disable the function block.

VAR_OUTPUT

```
bBusy           : BOOL;
bReady          : BOOL;
bError          : BOOL;
eError          : E_MBUS_ERROR;
dwIdNumber      : DWORD;
byStatus        : BYTE;
byGEN           : BYTE;
byCounter       : BYTE;
usiRecivedAdr   : USINT;
eMedium         : E_MBUS_Medium;
sMan            : STRING(3);
stEnergy        : ST_MBus_Info;
stPower         : ST_MBus_Info;
stVolume        : ST_MBus_Info;
stFlow          : ST_MBus_Info;
stForwardTemp   : ST_MBus_Info;
stReturnTemp    : ST_MBus_Info;
stDiffTemp      : ST_MBus_Info;
```

bBusy: The *bBusy* output is TRUE while the meter is being read.

bReady: The *bReady* output is TRUE for one cycle, once meter reading is completed.

bError: The *bError* output becomes TRUE as soon as an error occurs. The error is described via the variable *eError*.

eError: The *eError* output issues an error code when an error occurs (see E_MBUS_ERROR [▶ 216]).

dwIdNumber: Serial number of the meter (secondary address).

byStatus: Status of the meter. Please refer to device description for meanings.

byGEN: Meter software version.

byCounter: Transmission counter (number of transmitted RSP_UD).

usiRecivedAdr: Received primary address (0-250).

eMedium: Medium (see E_MBUS_Medium [▶ 219]).

sMan: Manufacturer code.

stEnergy: Meter reading, energy consumption (see ST_MBus_Info [▶ 222]).

stPower: Current energy consumption (see ST_MBus_Info [▶ 222]).

stVolume: Meter reading, volume (see ST_MBus_Info [▶ 222]).

stFlow: Current flow (see ST_MBus_Info [▶ 222]).

stForwardTemp: Flow temperature (see ST_MBus_Info [▶ 222]).

stReturnTemp: Return temperature (see ST_MBus_Info [▶ 222]).

stDiffTemp: Temperature difference (see ST_MBus_Info [▶ 222]).

VAR_IN_OUT

```
stCom           : ST_MBUS_Communication;
```

stCom: About this structure, the block FB_MBUSKL6781() [▶ 29] is connected to the meter function blocks (see ST_MBUS_Communication [▶ 221]).

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055633931/.zip:

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055635339/.zip:

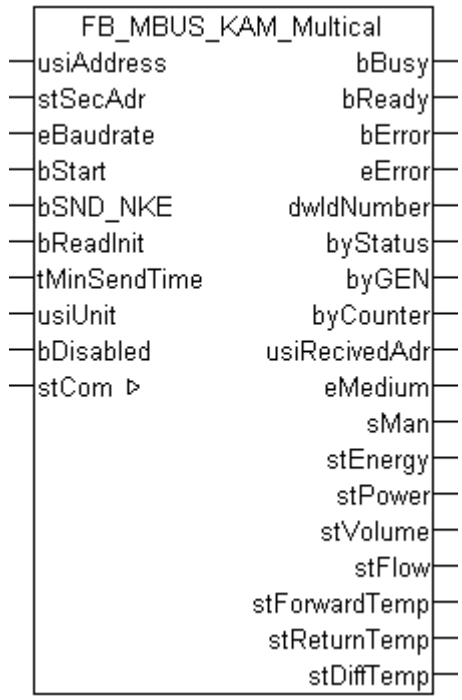
Controller configuration setting: "BCxx50 or BX serial"

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055636747/.zip:

BCxx00 must be set to "Large Model" in the PLC under "Project/Options/Controller Settings".

Controller configuration setting: "BC serial"

6.22.3 FB_MBUS_KAM_Multical



This function block is used for reading heat/cold meters from Kamstrup:

-Multical 401

-Multical 601

The function block can only be executed together with the function block FB_MBUSKL6781() [▶ 29].

Functionality of the function block [▶ 13]

VAR_INPUT

```
usiAddress      : USINT;
stSecAdr        : ST_MBUS_SecAdr;
eBaudrate       : E_MBUS_Baudrate := eMBUS_Baud2400;
bStart          : BOOL;
bSND_NKE        : BOOL := TRUE;
bReadInit       : BOOL := TRUE;
tMinSendTime    : TIME := t#2s;
usiUnit         : USINT;
bDisabled       : BOOL := FALSE;
```

usiAddress: Primary address [▶ 14] of the counter, that shall be readout with this module.

stSecAdr: Secondary address [▶ 15] of the counter, that shall be readout with this module.

eBaudrate: 300, 2400 baud [▶ 216].

bStart: Positive edge on this input, the meter is read out once.

bSND_NKE: TRUE to initialize the meter at each reading, and sets the meter on the first telegram (SND_NKE).

bReadInit: After restarting the PLC, the meter is read out once.

tMinSendTime: Standard t#2s. When this time exceeds, the counter is rereadout. At t#0s the counter is not readout and can be readout with bStart manually.

usiUnit: Unit of the energy values. 0=W/J(h) / 1=KW/KJ(h) / 2=MW/MJ(h) / 3=GW/GJ(h).

bDisabled: TRUE = disable the function block.

VAR_OUTPUT

```
bBusy           : BOOL;
bReady          : BOOL;
bError          : BOOL;
eError          : E_MBUS_ERROR;
dwIdNumber      : DWORD;
byStatus        : BYTE;
byGEN           : BYTE;
byCounter       : BYTE;
usiRecivedAdr   : USINT;
eMedium         : E_MBUS_Medium;
sMan            : STRING(3);
stEnergy        : ST_MBus_Info;
stPower         : ST_MBus_Info;
stVolume        : ST_MBus_Info;
stFlow          : ST_MBus_Info;
stForwardTemp   : ST_MBus_Info;
stReturnTemp    : ST_MBus_Info;
stDiffTemp      : ST_MBus_Info;
```

bBusy: The *bBusy* output is TRUE while the meter is being read.

bReady: The *bReady* output is TRUE for one cycle, once meter reading is completed.

bError: The *bError* output becomes TRUE as soon as an error occurs. The error is described via the variable *eError*.

eError: The *eError* output issues an error code when an error occurs (see E_MBUS_ERROR [▶ 216]).

dwIdNumber: Serial number of the meter (secondary address).

byStatus: Status of the meter. Please refer to device description for meanings.

byGEN: Meter software version.

byCounter: Transmission counter (number of transmitted RSP_UD).

usiRecivedAdr: Received primary address (0-250).

eMedium: Medium (see E_MBUS_Medium [▶ 219]).

sMan: Manufacturer code.

stEnergy: Meter reading, energy consumption (see ST_MBus_Info [▶ 222]).

stPower: Current energy consumption (see ST_MBus_Info [▶ 222]).

stVolume: Meter reading, volume (see ST_MBus_Info [▶ 222]).

stFlow: Current flow (see ST_MBus_Info [▶ 222]).

stForwardTemp: Flow temperature (see ST_MBus_Info [▶ 222]).

stReturnTemp: Return temperature (see ST_MBus_Info [▶ 222]).

stDiffTemp: Temperature difference (see ST_MBus_Info [▶ 222]).

VAR_IN_OUT

```
stCom           : ST_MBUS_Communication;
```

stCom: About this structure, the block FB_MBUSKL6781() [▶ 29] is connected to the meter function blocks (see ST_MBUS_Communication [▶ 221]).

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055633931/.zip:

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055635339/.zip:

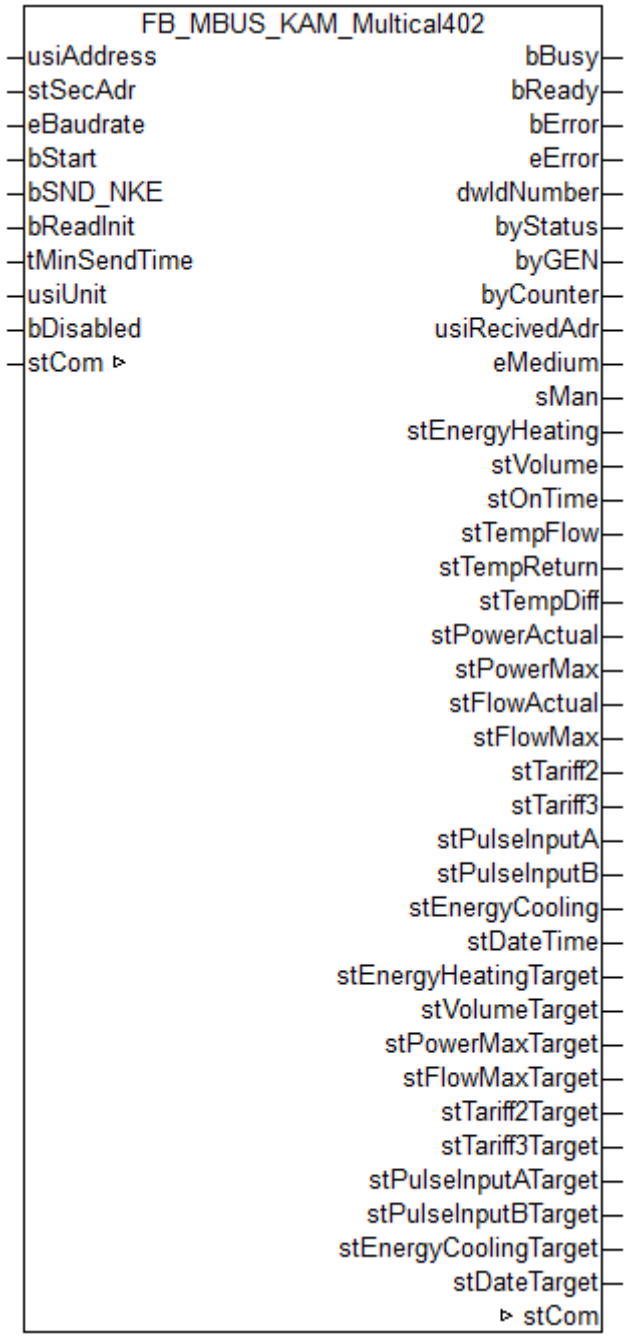
Controller configuration setting: "BCxx50 or BX serial"

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055636747/.zip:

BCxx00 must be set to "Large Model" in the PLC under "Project/Options/Controller Settings".

Controller configuration setting: "BC serial"

6.22.4 FB_MBUS_KAM_Multical402



This function block is used to read energy meters from Kamstrup.

The function block can only be executed together with the function block FB_MBUSKL6781() [▶ 29].

Functionality of the function block [▶ 13]

VAR_INPUT

```
usiAddress      : USINT;
stSecAdr        : ST_MBUS_SecAdr;
eBaudrate       : E_MBUS_Baudrate := eMBUS_Baud2400;
bStart          : BOOL;
bSND_NKE        : BOOL := TRUE;
bReadInit       : BOOL := TRUE;
tMinSendTime    : TIME := t#2s;
bDisabled       : BOOL := FALSE;
```

usiAddress: Primary address [▶ 14] of the counter, that shall be readout with this module.

stSecAdr: Secondary address [▶ 15] of the counter, that shall be readout with this module.

eBaudrate: 300, 2400, 9600 baud [▶ 216].

bStart: Positive edge on this input, the meter is read out once.

bSND_NKE: TRUE to initialize the meter at each reading, and sets the meter on the first telegram (SND_NKE).

bReadInit: After restarting the PLC, the meter is read out once.

tMinSendTime: Standard t#2s. When this time exceeds, the counter is rereadout. At t#0s the counter is not readout and can be readout with bStart manually.

bDisabled: TRUE = disable the function block.

VAR_OUTPUT

```
bBusy                   : BOOL;
bReady                  : BOOL;
bError                  : BOOL;
eError                  : E_MBUS_ERROR;
dwIdNumber              : DWORD;
byStatus                : BYTE;
byGEN                   : BYTE;
byCounter               : BYTE;
usiRecivedAdr           : USINT;
eMedium                 : E_MBUS_Medium;
sMan                    : STRING(3);
stEnergyHeating         : ST_MBus_Info;
stVolume                : ST_MBus_Info;
stOnTime                : ST_MBus_Info;
stTempFlow              : ST_MBus_Info;
stTempReturn            : ST_MBus_Info;
stTempDiff              : ST_MBus_Info;
stPowerActual           : ST_MBus_Info;
stPowerMax              : ST_MBus_Info;
stFlowActual            : ST_MBus_Info;
stFlowMax               : ST_MBus_Info;
stTariff2               : ST_MBus_Info;
stTariff3               : ST_MBus_Info;
stPulseInputA           : ST_MBus_Info;
stPulseInputB           : ST_MBus_Info;
stEnergyCooling         : ST_MBus_Info;
stDateTime              : ST_MBus_Info;
stEnergyHeatingTarget   : ST_MBus_Info;
stVolumeTarget          : ST_MBus_Info;
stPowerMaxTarget        : ST_MBus_Info;
stFlowMaxTarget         : ST_MBus_Info;
stTariff2Target         : ST_MBus_Info;
stTariff3Target         : ST_MBus_Info;
stPulseInputATarget     : ST_MBus_Info;
stPulseInputBTarget     : ST_MBus_Info;
stEnergyCoolingTarget   : ST_MBus_Info;
stDateTarget            : ST_MBus_Info;
```

bBusy: The *bBusy* output is TRUE while the meter is being read.

bReady: The *bReady* output is TRUE for one cycle, once meter reading is completed.

bError: The *bError* output becomes TRUE as soon as an error occurs. The error is described via the variable *eError*.

eError: The *eError* output issues an error code when an error occurs (see E_MBUS_ERROR [▶ 216]).

dwIdNumber: Serial number of the meter (secondary address).

byStatus: Status of the meter. Please refer to device description for meanings.

byGEN: Meter software version.

byCounter: Transmission counter (number of transmitted RSP_UD).

usiRecivedAdr: Received primary address (0-250).

eMedium: Medium (see E_MBUS_Medium [▶ 219]).

sMan: Manufacturer code.

stEnergyHeating: Heat energy (see ST_MBus_Info [▶ 222]).

stVolume: Volume (see ST_MBus_Info [▶ 222]).

stOnTime: Operating hours (see ST_MBus_Info [▶ 222]).

stTempFlow: Flow temperature (see ST_MBus_Info [▶ 222]).

stTempReturn: Return temperature (see ST_MBus_Info [▶ 222]).

stTempDiff: Temperature difference (see ST_MBus_Info [▶ 222]).

stPowerActual: Actual active power (see ST_MBus_Info [▶ 222]).

stPowerMax: Active power (maximum) (see ST_MBus_Info [▶ 222]).

stFlowActual: Actual flow (see ST_MBus_Info [▶ 222]).

stFlowMax: Flow (maximum) (see ST_MBus_Info [▶ 222]).

stTariff2: Tariff 2 (see ST_MBus_Info [▶ 222]).

stTariff3: Tariff 3 (see ST_MBus_Info [▶ 222]).

stPulseInputA: Pulse input A (see ST_MBus_Info [▶ 222]).

stPulseInputB: Pulse input B (see ST_MBus_Info [▶ 222]).

stEnergyCooling: Cooling energy (see ST_MBus_Info [▶ 222]).

stDateTime: Date and time (see ST_MBus_Info [▶ 222]).

stEnergyHeatingTarget: See manufacturer information (see ST_MBus_Info [▶ 222]).

stVolumeTarget: See manufacturer information (see ST_MBus_Info [▶ 222]).

stPowerMaxTarget: See manufacturer information (see ST_MBus_Info [▶ 222]).

stFlowMaxTarget: See manufacturer information (see ST_MBus_Info [▶ 222]).

stTariff2Target: See manufacturer information (see ST_MBus_Info [▶ 222]).

stTariff3Target: See manufacturer information (see ST_MBus_Info [▶ 222]).

stPulseInputATarget: See manufacturer information (see ST_MBus_Info [▶ 222]).

stPulseInputBTarget: See manufacturer information (see ST_MBus_Info [▶ 222]).

stEnergyCoolingTarget: See manufacturer information (see ST_MBus_Info [▶ 222]).

stDateTarget: See manufacturer information (see ST_MBus_Info [▶ 222]).

Requirements

Development environment	Target system type	PLC libraries to include
TwinCAT 2.11 R3/x64 higher than Build 2234	PC/CX, BX or BC	TcMBus-Library higher than V2.3.0

VAR_IN_OUT

```
stCom          : ST_MBUS_Communication;
```

stCom: About this structure, the block FB_MBUSKL6781() [▶ 29] is connected to the meter function blocks (see ST_MBUS_Communication [▶ 221]).

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055633931/.zip:

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055635339/.zip:

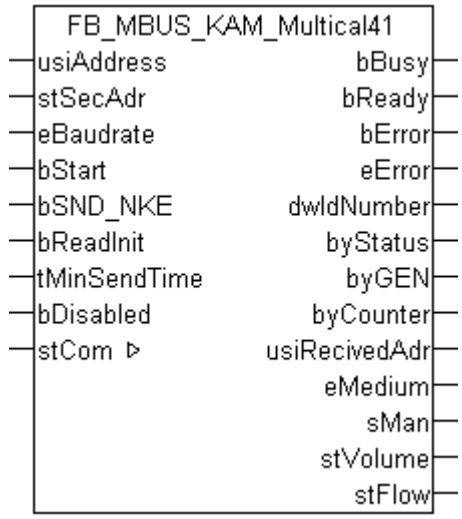
Controller configuration setting: "BCxx50 or BX serial"

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055636747/.zip:

BCxx00 must be set to "Large Model" in the PLC under "Project/Options/Controller Settings".

Controller configuration setting: "BC serial"

6.22.5 FB_MBUS_KAM_Multical41



This function block is used to read water meters from Kamstrup:

-Multical 41

The function block can only be executed together with the function block FB_MBUSKL6781() [▶ 29].

Functionality of the function block [▶ 13]

VAR_INPUT

```
usiAddress      : USINT;
stSecAdr        : ST_MBUS_SecAdr;
eBaudrate       : E_MBUS_Baudrate := eMBUS_Baud2400;
bStart          : BOOL;
bSND_NKE        : BOOL := TRUE;
bReadInit       : BOOL := TRUE;
tMinSendTime    : TIME := t#2s;
bDisabled       : BOOL := FALSE;
```

usiAddress: Primary address [▶ 14] of the counter, that shall be readout with this module.

stSecAdr: Secondary address [▶ 15] of the counter, that shall be readout with this module.

eBaudrate: 300, 2400 baud [▶ 216].

bStart: Positive edge on this input, the meter is read out once.

bSND_NKE: TRUE to initialize the meter at each reading and sets the meter on the first telegram (SND_NKE).

bReadInit: After restarting the PLC, the meter is read out once.

tMinSendTime: Standard t#2s. When this time exceeds, the counter is rereadout. At t#0s the counter is not readout and can be readout with bStart manually.

bDisabled: TRUE = disable the function block.

VAR_OUTPUT

```
bBusy           : BOOL;
bReady          : BOOL;
bError          : BOOL;
eError          : E_MBUS_ERROR;
dwIdNumber      : DWORD;
byStatus        : BYTE;
byGEN           : BYTE;
byCounter       : BYTE;
usiRecivedAdr   : USINT;
eMedium         : E_MBUS_Medium;
sMan            : STRING(3);
stVolume        : ST_MBus_Info;
stFlow          : ST_MBus_Info;
```

bBusy: The *bBusy* output is TRUE while the meter is being read.

bReady: The *bReady* output is TRUE for one cycle, once meter reading is completed.

bError: The *bError* output becomes TRUE as soon as an error occurs. The error is described via the variable *eError*.

eError: The *eError* output issues an error code when an error occurs (see E_MBUS_ERROR [▶ 216]).

dwIdNumber: Serial number of the meter (secondary address).

byStatus: Status of the meter. Please refer to device description for meanings.

byGEN: Meter software version.

byCounter: Transmission counter (number of transmitted RSP_UD).

usiRecivedAdr: Received primary address (0-250).

eMedium: Medium (see E_MBUS_Medium [▶ 219]).

sMan: Manufacturer code.

stVolume: Meter reading, volume (see ST_MBus_Info [▶ 222]).

stFlow: Current flow (see ST_MBus_Info [▶ 222]).

VAR_IN_OUT

```
stCom          : ST_MBUS_Communication;
```

stCom: About this structure, the block FB_MBUSKL6781() [▶ 29] is connected to the meter function blocks (see ST_MBUS_Communication [▶ 221]).

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055633931/.zip:

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055635339/.zip:

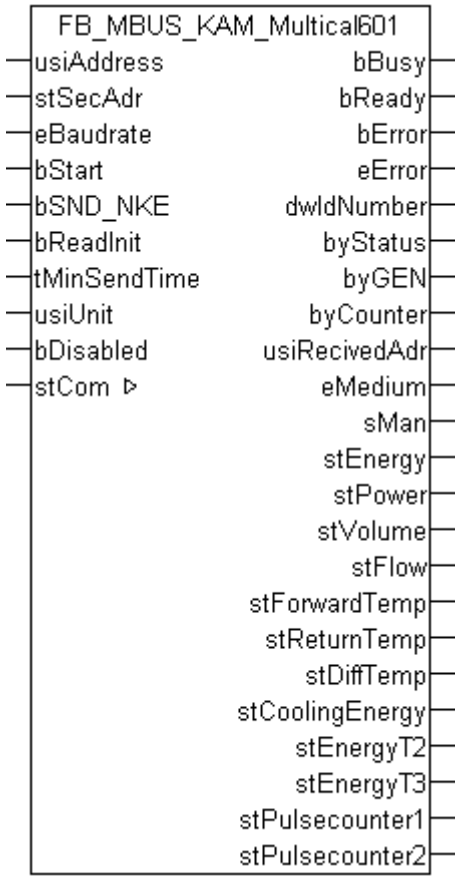
Controller configuration setting: "BCxx50 or BX serial"

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055636747/.zip:

BCxx00 must be set to "Large Model" in the PLC under "Project/Options/Controller Settings".

Controller configuration setting: "BC serial"

6.22.6 FB_MBUS_KAM_Multical601



This function block is used for reading heat/cold meters from Kamstrup:

-Multical 601

The function block can only be executed together with the function block FB_MBUSKL6781() [▶ 29].

Functionality of the function block [▶ 13]

VAR_INPUT

```
usiAddress        : USINT;
stSecAdr          : ST_MBUS_SecAdr;
eBaudrate         : E_MBUS_Baudrate := eMBUS_Baud2400;
bStart            : BOOL;
bSND_NKE          : BOOL := TRUE;
bReadInit         : BOOL := TRUE;
tMinSendTime      : TIME := t#2s;
usiUnit           : USINT;
bDisabled         : BOOL := FALSE;
```

usiAddress: Primary address [▶ 14] of the counter, that shall be readout with this module.

stSecAdr: Secondary address [▶ 15] of the counter, that shall be readout with this module.

eBaudrate: 300, 2400, 9600 baud [▶ 216].

bStart: Positive edge on this input, the meter is read out once.

bSND_NKE: TRUE to initialize the meter at each reading, and sets the meter on the first telegram (SND_NKE).

bReadInit: After restarting the PLC, the meter is read out once.

tMinSendTime: Standard t#2s. When this time exceeds, the counter is rereadout. At t#0s the counter is not readout and can be readout with bStart manually.

usiUnit: Unit of the energy values. 0=W/J(h) / 1=KW/KJ(h) / 2=MW/MJ(h) / 3=GW/GJ(h).

bDisabled: TRUE = disable the function block.

VAR_OUTPUT

```
bBusy             : BOOL;
bReady            : BOOL;
bError            : BOOL;
eError            : E_MBUS_ERROR;
dwIdNumber        : DWORD;
byStatus          : BYTE;
byGEN             : BYTE;
byCounter         : BYTE;
usiRecivedAdr     : USINT;
eMedium           : E_MBUS_Medium;
sMan              : STRING(3);
stEnergy          : ST_MBus_Info;
stPower           : ST_MBus_Info;
stVolume          : ST_MBus_Info;
stFlow            : ST_MBus_Info;
stForwardTemp     : ST_MBus_Info;
stReturnTemp      : ST_MBus_Info;
stDiffTemp        : ST_MBus_Info;
stCoolingEnergy   : ST_MBus_Info;
stEnergyT2        : ST_MBus_Info;
stEnergyT3        : ST_MBus_Info;
stPulsecounter1   : ST_MBus_Info;
stPulsecounter2   : ST_MBus_Info;
```

bBusy: The *bBusy* output is TRUE while the meter is being read.

bReady: The *bReady* output is TRUE for one cycle, once meter reading is completed.

bError: The *bError* output becomes TRUE as soon as an error occurs. The error is described via the variable *eError*.

eError: The *eError* output issues an error code when an error occurs (see E_MBUS_ERROR [▶ 216]).

dwIdNumber: Serial number of the meter (secondary address).

byStatus: Status of the meter. Please refer to device description for meanings.

byGEN: Meter software version.

byCounter: Transmission counter (number of transmitted RSP_UD).

usiRecivedAdr: Received primary address (0-250).

eMedium: Medium (see E_MBUS_Medium [▶ 219]).

sMan: Manufacturer code.

stEnergy: Meter reading, energy consumption (see ST_MBus_Info [▶ 222]).

stPower: Current energy consumption (see ST_MBus_Info [▶ 222]).

stVolume: Meter reading, volume (see ST_MBus_Info [▶ 222]).

stFlow: Current flow (see ST_MBus_Info [▶ 222]).

stForwardTemp: Flow temperature (see ST_MBus_Info [▶ 222]).

stReturnTemp: Return temperature (see ST_MBus_Info [▶ 222]).

stDiffTemp: Temperature difference (see ST_MBus_Info [▶ 222]).

stCoolingEnergy: Meter reading, cooling energy consumption (see ST_MBus_Info [▶ 222]).

stEnergyT2: Meter reading, energy consumption tariff 2 (see ST_MBus_Info [▶ 222]).

stEnergyT3: Meter reading, energy consumption tariff 3 (see ST_MBus_Info [▶ 222]).

stPulsecounter1: Pulse counter 1 (see ST_MBus_Info [▶ 222]).

stPulsecounter2: Pulse counter 2 (see ST_MBus_Info [▶ 222]).

VAR_IN_OUT

```
stCom          : ST_MBUS_Communication;
```

stCom: About this structure, the block FB_MBUSKL6781() [▶ 29] is connected to the meter function blocks (see ST_MBUS_Communication [▶ 221]).

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055633931/.zip:

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055635339/.zip:

Controller configuration setting: "BCxx50 or BX serial"

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055636747/.zip:

BCxx00 must be set to "Large Model" in the PLC under "Project/Options/Controller Settings".

Controller configuration setting: "BC serial"

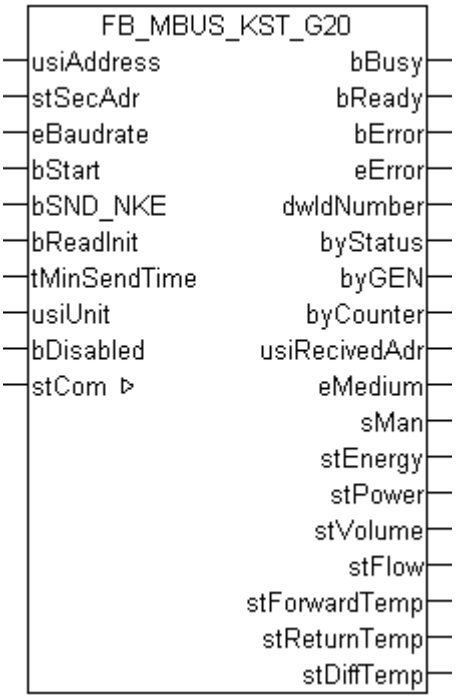
6.23 Kundo

The function blocks only output a selection of the most common data. These data are described on the respective pages under "VAR_OUT". If more or all data are required, the function blocks FB_MBUS_General [▶ 31], FB_MBUS_General_Ext [▶ 35] or FB_MBUS_General_Param [▶ 41] from the General folder should be used. Not that these function blocks do not run on BC or BX systems. The function block FB_MBUS_General_Send [▶ 43] can be used to send data to the device (e.g. setting of the primary address).

Vendor	Type	Device	Function block
KUNDO	Heat/cold meter	Compact WMZ G20	FB_MBUS_KST_G20 [▶ 152]
	Heat/cold meter	Compact WMZ G21	
	External M-Bus module	him1s	FB_MBUS_KST_him1 [▶ 154]

Vendor	Type	Device	Function block
	External M-Bus module	him1plus	
	Pulse input	him1plus	FB_MBUS_KST_him1Puls [▶ 156]

6.23.1 FB_MBUS_KST_G20



This block is used for reading heat/cold meters from KUNDO System Technik:

-Kompakt WZM G20 (with internal M-Bus module)

-Kompakt WZM G21 (with internal M-Bus module)

The block can only be used in conjunction with the block FB_MBUSKL6781() [▶ 29].

Functionality of the function block [▶ 13]

VAR_INPUT

```
usiAddress      : USINT;
stSecAdr        : ST_MBUS_SecAdr;
eBaudrate       : E_MBUS_Baudrate := eMBUS_Baud2400;
bStart          : BOOL;
bSND_NKE        : BOOL := TRUE;
bReadInit       : BOOL := TRUE;
tMinSendTime    : TIME := t#2s;
usiUnit         : USINT;
bDisabled       : BOOL := FALSE;
```

usiAddress: Primary address [▶ 14] of the counter, that shall be readout with this module.

stSecAdr: Secondary address [▶ 15] of the counter, that shall be readout with this module.

eBaudrate: 300, 2400, 9600 baud [▶ 216].

bStart: Positive edge on this input, the meter is read out once.

bSND_NKE: TRUE to initialize the meter at each reading, and sets the meter on the first telegram (SND_NKE).

bReadInit: After restarting the PLC, the meter is read out once.

tMinSendTime: Standard t#2s. When this time exceeds, the counter is rereadout. At t#0s the counter is not readout and can be readout with bStart manually.

usiUnit: Unit of the energy values. 0=W/J(h) / 1=KW/KJ(h) / 2=MW/MJ(h) / 3=GW/GJ(h).

bDisabled: TRUE = disable the function block.

VAR_OUTPUT

```
bBusy          : BOOL;
bReady         : BOOL;
bError         : BOOL;
eError         : E_MBUS_ERROR;
dwIdNumber     : DWORD;
byStatus       : BYTE;
byGEN          : BYTE;
byCounter      : BYTE;
usiRecivedAdr  : USINT;
eMedium        : E_MBUS_Medium;
sMan           : STRING(3);
stEnergy       : ST_MBus_Info;
stPower        : ST_MBus_Info;
stVolume       : ST_MBus_Info;
stFlow         : ST_MBus_Info;
stForwardTemp  : ST_MBus_Info;
stReturnTemp   : ST_MBus_Info;
stDiffTemp     : ST_MBus_Info;
```

bBusy: The *bBusy* output is TRUE while the meter is being read.

bReady: The *bReady* output is TRUE for one cycle, once meter reading is completed.

bError: The *bError* output becomes TRUE as soon as an error occurs. The error is described via the variable *eError*.

eError: The *eError* output issues an error code when an error occurs (see E_MBUS_ERROR [▶ 216]).

dwIdNumber: Serial number of the meter (secondary address).

byStatus: Status of the meter. Please refer to device description for meanings.

byGEN: Meter software version.

byCounter: Transmission counter (number of transmitted RSP_UD).

usiRecivedAdr: Received primary address (0-250).

eMedium: Medium (see E_MBUS_Medium [▶ 219]).

sMan: Manufacturer code.

stEnergy: Meter reading, energy consumption (see ST_MBus_Info [▶ 222]).

stPower: Current energy consumption (see ST_MBus_Info [▶ 222]).

stVolume: Meter reading, volume (see ST_MBus_Info [▶ 222]).

stFlow: Current flow (see ST_MBus_Info [▶ 222]).

stForwardTemp: Flow temperature (see ST_MBus_Info [▶ 222]).

stReturnTemp: Return temperature (see ST_MBus_Info [▶ 222]).

stDiffTemp: Temperature difference (see ST_MBus_Info [▶ 222]).

VAR_IN_OUT

```
stCom          : ST_MBUS_Communication;
```

stCom: About this structure, the block FB_MBUSKL6781() [▶ 29] is connected to the meter function blocks (see ST_MBUS_Communication [▶ 221]).

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055638155/.zip:

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055639563/.zip:

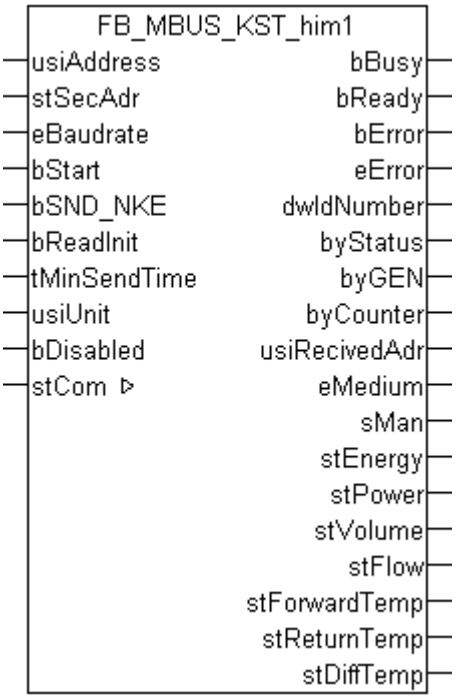
Controller configuration setting: "BCxx50 or BX serial"

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055640971/.zip:

BCxx00 must be set to "Large Model" in the PLC under "Project/Options/Controller Settings".

Controller configuration setting: "BC serial"

6.23.2 FB_MBUS_KST_him1



This function block is used for reading M-Bus modules from KUNDO System Technik:

-him1s

-him1plus

These modules can be used for reading consumption data from a KUNDO arithmetic unit.

The function block can only be executed together with the function block FB_MBUSKL6781() [▶ 29].

Functionality of the function block [▶ 13]

VAR_INPUT

```
usiAddress      : USINT;
stSecAdr        : ST_MBUS_SecAdr;
eBaudrate       : E_MBUS_Baudrate := eMBUS_Baud2400;
bStart          : BOOL;
bSND_NKE        : BOOL := TRUE;
bReadInit       : BOOL := TRUE;
tMinSendTime    : TIME := t#2s;
usiUnit         : USINT;
bDisabled       : BOOL := FALSE;
```

usiAddress: Primary address [▶ 14] of the counter, that shall be readout with this module.

stSecAdr: Secondary address [▶ 15] of the counter, that shall be readout with this module.

eBaudrate: 300, 2400 baud [▶ 216].

bStart: Positive edge on this input, the meter is read out once.

bSND_NKE: TRUE to initialize the meter at each reading, and sets the meter on the first telegram (SND_NKE).

bReadInit: After restarting the PLC, the meter is read out once.

tMinSendTime: Standard t#2s. When this time exceeds, the counter is rereadout. At t#0s the counter is not readout and can be readout with bStart manually.

usiUnit: Unit of the energy values. 0=W/J(h) / 1=KW/KJ(h) / 2=MW/MJ(h) / 3=GW/GJ(h).

bDisabled: TRUE = disable the function block.

VAR_OUTPUT

```
bBusy           : BOOL;
bReady          : BOOL;
bError          : BOOL;
eError          : E_MBUS_ERROR;
dwIdNumber      : DWORD;
byStatus        : BYTE;
byGEN           : BYTE;
byCounter       : BYTE;
usiRecivedAdr   : USINT;
eMedium         : E_MBUS_Medium;
sMan            : STRING(3);
stEnergy        : ST_MBus_Info;
stPower         : ST_MBus_Info;
stVolume        : ST_MBus_Info;
stFlow          : ST_MBus_Info;
stForwardTemp   : ST_MBus_Info;
stReturnTemp    : ST_MBus_Info;
stDiffTemp      : ST_MBus_Info;
```

bBusy: The *bBusy* output is TRUE while the meter is being read.

bReady: The *bReady* output is TRUE for one cycle, once meter reading is completed.

bError: The *bError* output becomes TRUE as soon as an error occurs. The error is described via the variable *eError*.

eError: The *eError* output issues an error code when an error occurs (see E_MBUS_ERROR [▶ 216]).

dwIdNumber: Serial number of the meter (secondary address).

byStatus: Status of the meter. Please refer to device description for meanings.

byGEN: Meter software version.

byCounter: Transmission counter (number of transmitted RSP_UD).

usiRecivedAdr: Received primary address (0-250).

eMedium: Medium (see E_MBUS_Medium [▶ 219]).

sMan: Manufacturer code.

stEnergy: Meter reading, energy consumption (see ST_MBus_Info [▶ 222]).

stPower: Current energy consumption (see ST_MBus_Info [▶ 222]).

stVolume: Meter reading, volume (see ST_MBus_Info [▶ 222]).

stFlow: Current flow (see ST_MBus_Info [▶ 222]).

stForwardTemp: Flow temperature (see ST_MBus_Info [▶ 222]).

stReturnTemp: Return temperature (see ST_MBus_Info [▶ 222]).

stDiffTemp: Temperature difference (see ST_MBus_Info [▶ 222]).

VAR_IN_OUT

```
stCom          : ST_MBUS_Communication;
```

stCom: About this structure, the block FB_MBUSKL6781() [▶ 29] is connected to the meter function blocks (see ST_MBUS_Communication [▶ 221]).

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055638155/.zip:

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055639563/.zip:

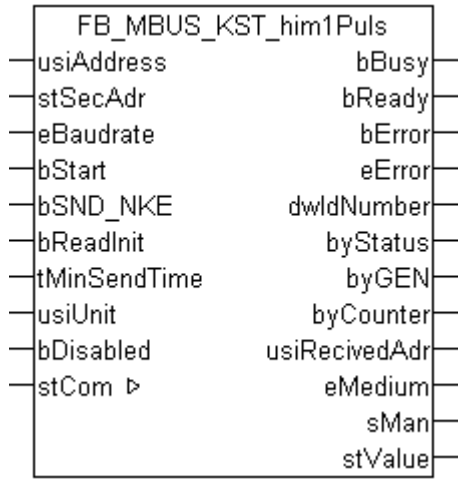
Controller configuration setting: "BCxx50 or BX serial"

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055640971/.zip:

BCxx00 must be set to "Large Model" in the PLC under "Project/Options/Controller Settings".

Controller configuration setting: "BC serial"

6.23.3 FB_MBUS_KST_him1Puls



This function block is used for reading M-Bus modules from KUNDO System Technik:

-him1plus (pulse input)

The function block can only be executed together with the function block FB_MBUSKL6781() [▶ 29].

Functionality of the function block [▶ 13]

VAR_INPUT

```
usiAddress     : USINT;
stSecAdr       : ST_MBUS_SecAdr;
eBaudrate      : E_MBUS_Baudrate := eMBUS_Baud2400;
bStart         : BOOL;
bSND_NKE       : BOOL := TRUE;
bReadInit      : BOOL := TRUE;
tMinSendTime   : TIME := t#2s;
usiUnit        : USINT;
bDisabled      : BOOL := FALSE;
```

usiAddress: Primary address [▶ 14] of the counter, that shall be readout with this module.

stSecAdr: Secondary address [▶ 15] of the counter, that shall be readout with this module.

eBaudrate: 300, 2400 baud [▶ 216].

bStart: Positive edge on this input, the meter is read out once.

bSND_NKE: TRUE to initialize the meter at each reading, and sets the meter on the first telegram (SND_NKE).

bReadInit: After restarting the PLC, the meter is read out once.

tMinSendTime: Standard t#2s. When this time exceeds, the counter is rereadout. At t#0s the counter is not readout and can be readout with bStart manually.

usiUnit: Unit of the energy values. 0=W/J(h) / 1=KW/KJ(h) / 2=MW/MJ(h) / 3=GW/GJ(h).

bDisabled: TRUE = disable the function block.

VAR_OUTPUT

```
bBusy           : BOOL;
bReady          : BOOL;
bError          : BOOL;
eError          : E_MBUS_ERROR;
dwIdNumber      : DWORD;
byStatus        : BYTE;
byGEN           : BYTE;
byCounter       : BYTE;
usiRecivedAdr   : USINT;
eMedium         : E_MBUS_Medium;
sMan            : STRING(3);
stValue         : ST_MBus_Info;
```

bBusy: The *bBusy* output is TRUE while the meter is being read.

bReady: The *bReady* output is TRUE for one cycle, once meter reading is completed.

bError: The *bError* output becomes TRUE as soon as an error occurs. The error is described via the variable *eError*.

eError: The *eError* output issues an error code when an error occurs (see E_MBUS_ERROR [▶ 216]).

dwIdNumber: Serial number of the meter (secondary address).

byStatus: Status of the meter. Please refer to device description for meanings.

byGEN: Meter software version.

byCounter: Transmission counter (number of transmitted RSP_UD).

usiRecivedAdr: Received primary address (0-250).

eMedium: Medium (see E_MBUS_Medium [▶ 219]).

sMan: Manufacturer code.

stValue: Meter reading (see ST_MBus_Info [▶ 222]).

VAR_IN_OUT

```
stCom           : ST_MBUS_Communication;
```

stCom: About this structure, the block FB_MBUSKL6781() [▶ 29] is connected to the meter function blocks (see ST_MBUS_Communication [▶ 221]).

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055638155/.zip:

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055639563/.zip:

Controller configuration setting: "BCxx50 or BX serial"

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055640971/.zip:

BCxx00 must be set to "Large Model" in the PLC under "Project/Options/Controller Settings".

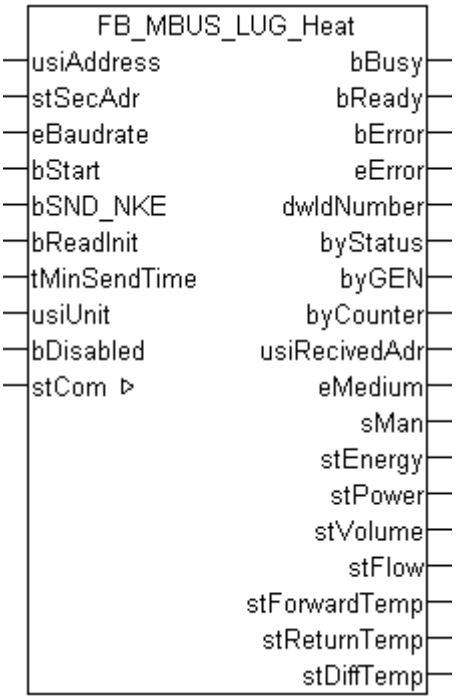
Controller configuration setting: "BC serial"

6.24 Landis & Gyr

The function blocks only output a selection of the most common data. These data are described on the respective pages under "VAR_OUT". If more or all data are required, the function blocks FB_MBUS_General [▶ 31], FB_MBUS_General_Ext [▶ 35] or FB_MBUS_General_Param [▶ 41] from the General folder should be used. Not that these function blocks do not run on BC or BX systems. The function block FB_MBUS_General_Send [▶ 43] can be used to send data to the device (e.g. setting of the primary address).

Vendor	Type	Device	Function block
Landis & Gyr	Heat/cold meter	ULTRAHEAT 2WR5	FB_MBUS_LUG_Heat [▶ 158]
	Heat/cold meter	ULTRAHEAT 2WR6	
	Heat/cold meter	ULTRAHEAT UH50	

6.24.1 FB_MBUS_LUG_Heat



This block is used for reading heat/cold meters from Landis & Gyr:

-2WR5

-2WR6

-UH50

The block can only be used in conjunction with the block FB_MBUSKL6781() [▶ 29].

Functionality of the function block [▶ 13]

VAR_INPUT

```
usiAddress      : USINT;
stSecAdr        : ST_MBUS_SecAdr;
eBaudrate       : E_MBUS_Baudrate := eMBUS_Baud2400;
bStart          : BOOL;
bSND_NKE        : BOOL := TRUE;
bReadInit       : BOOL := TRUE;
tMinSendTime    : TIME := t#2s;
usiUnit         : USINT;
bDisabled       : BOOL := FALSE;
```

usiAddress: Primary address [▶ 14] of the counter, that shall be readout with this module.

stSecAdr: Secondary address [▶ 15] of the counter, that shall be readout with this module.

eBaudrate: 300, 1200, 2400, 4800 baud [▶ 216].

bStart: Positive edge on this input, the meter is read out once.

bSND_NKE: TRUE to initialize the meter at each reading, and sets the meter on the first telegram (SND_NKE).

bReadInit: After restarting the PLC, the meter is read out once.

tMinSendTime: Standard t#2s. When this time exceeds, the counter is rereadout. At t#0s the counter is not readout and can be readout with bStart manually.

usiUnit: Unit of the energy values. 0=W/J(h) / 1=KW/KJ(h) / 2=MW/MJ(h) / 3=GW/GJ(h).

bDisabled: TRUE = disable the function block.

VAR_OUTPUT

```
bBusy           : BOOL;
bReady          : BOOL;
bError          : BOOL;
eError          : E_MBUS_ERROR;
dwIdNumber      : DWORD;
byStatus        : BYTE;
byGEN           : BYTE;
byCounter       : BYTE;
usiRecivedAdr   : USINT;
eMedium         : E_MBUS_Medium;
sMan            : STRING(3);
stEnergy        : ST_MBus_Info;
stPower         : ST_MBus_Info;
stVolume        : ST_MBus_Info;
stFlow          : ST_MBus_Info;
stForwardTemp   : ST_MBus_Info;
stReturnTemp    : ST_MBus_Info;
stDiffTemp      : ST_MBus_Info;
```

bBusy: The *bBusy* output is TRUE while the meter is being read.

bReady: The *bReady* output is TRUE for one cycle, once meter reading is completed.

bError: The *bError* output becomes TRUE as soon as an error occurs. The error is described via the variable *eError*.

eError: The *eError* output issues an error code when an error occurs (see E_MBUS_ERROR [▶ 216]).

dwIdNumber: Serial number of the meter (secondary address).

byStatus: Status of the meter. Please refer to device description for meanings.

byGEN: Meter software version.

byCounter: Transmission counter (number of transmitted RSP_UD).

usiRecivedAdr: Received primary address (0-250).

eMedium: Medium (see E_MBUS_Medium [▶ 219]).

sMan: Manufacturer code.

stEnergy: Meter reading, energy consumption (see ST_MBus_Info [▶ 222]).

stPower: Current energy consumption (see ST_MBus_Info [▶ 222]).

stVolume: Meter reading, volume (see ST_MBus_Info [▶ 222]).

stFlow: Current flow (see ST_MBus_Info [▶ 222]).

stForwardTemp: Flow temperature (see ST_MBus_Info [▶ 222]).

stReturnTemp: Return temperature (see ST_MBus_Info [▶ 222]).

stDiffTemp: Temperature difference (see ST_MBus_Info [▶ 222]).

VAR_IN_OUT

```
stCom          : ST_MBUS_Communication;
```

stCom: About this structure, the block FB_MBUSKL6781() [▶ 29] is connected to the meter function blocks (see ST_MBUS_Communication [▶ 221]).

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055642379/.zip:

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055645195/.zip:

Controller configuration setting: "BCxx50 or BX serial"

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055643787/.zip:

BCxx00 must be set to "Large Model" in the PLC under "Project/Options/Controller Settings".

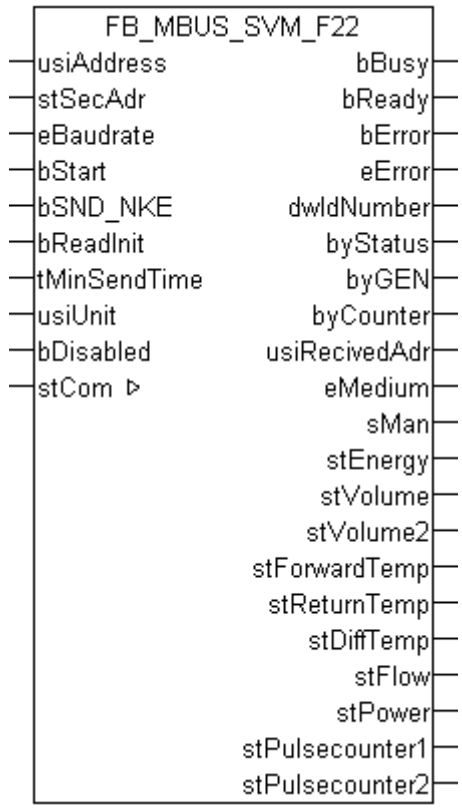
Controller configuration setting: "BC serial"

6.25 Metrima

The function blocks only output a selection of the most common data. These data are described on the respective pages under "VAR_OUT". If more or all data are required, the function blocks FB_MBUS_General [▶ 31], FB_MBUS_General_Ext [▶ 35] or FB_MBUS_General_Param [▶ 41] from the General folder should be used. Not that these function blocks do not run on BC or BX systems. The function block FB_MBUS_General_Send [▶ 43] can be used to send data to the device (e.g. setting of the primary address).

Vendor	Type	Device	Function block
Metrima	Heat meter	F22 (default values)	FB_MBUS_SVM_F22 [▶ 161]
	Heat meter	F22 (with additional output values)	FB_MBUS_SVM_F22_Ext [▶ 163]

6.25.1 FB_MBUS_SVM_F22



This block is used for reading heat meters from Metrima:

-F22

The block can only be used in conjunction with the block FB_MBUSKL6781() [▶ 29].

Functionality of the function block [▶ 13]

VAR_INPUT

```
usiAddress        : USINT;
stSecAdr          : ST_MBUS_SecAdr;
eBaudrate         : E_MBUS_Baudrate := eMBUS_Baud2400;
bStart            : BOOL;
bSND_NKE          : BOOL := TRUE;
bReadInit         : BOOL := TRUE;
tMinSendTime      : TIME := t#2s;
usiUnit           : USINT;
bDisabled         : BOOL := FALSE;
```

usiAddress: Primary address [▶ 14] of the counter, that shall be readout with this module.

stSecAdr: Secondary address [▶ 15] of the counter, that shall be readout with this module.

eBaudrate: 300, 2400, 9600 baud [▶ 216].

bStart: Positive edge on this input, the meter is read out once.

bSND_NKE: TRUE to initialize the meter at each reading, and sets the meter on the first telegram (SND_NKE).

bReadInit: After restarting the PLC, the meter is read out once.

tMinSendTime: Standard t#2s. When this time exceeds, the counter is rereadout. At t#0s the counter is not readout and can be readout with bStart manually.

usiUnit: Unit of the energy values. 0=W/J(h) / 1=KW/KJ(h) / 2=MW/MJ(h) / 3=GW/GJ(h).

bDisabled: TRUE = disable the function block.

VAR_OUTPUT

```
bBusy            : BOOL;
bReady           : BOOL;
bError           : BOOL;
eError           : E_MBUS_ERROR;
dwIdNumber       : DWORD;
byStatus         : BYTE;
byGEN            : BYTE;
byCounter        : BYTE;
usiRecivedAdr    : USINT;
eMedium          : E_MBUS_Medium;
sMan             : STRING(3);
stEnergy         : ST_MBus_Info;
stVolume         : ST_MBus_Info;
stVolume2        : ST_MBus_Info;
stForwardTemp    : ST_MBus_Info;
stReturnTemp     : ST_MBus_Info;
stDiffTemp       : ST_MBus_Info;
stFlow           : ST_MBus_Info;
stPower          : ST_MBus_Info;
stPulsecounter1  : ST_MBus_Info;
stPulsecounter2  : ST_MBus_Info;
```

bBusy: The *bBusy* output is TRUE while the meter is being read.

bReady: The *bReady* output is TRUE for one cycle, once meter reading is completed.

bError: The *bError* output becomes TRUE as soon as an error occurs. The error is described via the variable *eError*.

eError: The *eError* output issues an error code when an error occurs (see E_MBUS_ERROR [▶ 216]).

dwIdNumber: Serial number of the meter (secondary address).

byStatus: Status of the meter. Please refer to device description for meanings.

byGEN: Meter software version.

byCounter: Transmission counter (number of transmitted RSP_UD).

usiRecivedAdr: Received primary address (0-250).

eMedium: Medium (see E_MBUS_Medium [▶ 219]).

sMan: Manufacturer code.

stEnergy: Meter reading, energy consumption (see ST_MBus_Info [▶ 222]).

stVolume: Meter reading, volume (see ST_MBus_Info [▶ 222]).

stVolume2: Volume according to energy (see ST_MBus_Info [▶ 222]).

stForwardTemp: Flow temperature (see ST_MBus_Info [▶ 222]).

stReturnTemp: Return temperature (see ST_MBus_Info [▶ 222]).

stDiffTemp: Temperature difference (see ST_MBus_Info [▶ 222]).

stFlow: Current flow (see ST_MBus_Info [▶ 222]).

stPower: Current energy consumption (see ST_MBus_Info [▶ 222]).

stPulsecounter1: Pulse counter 1 (see ST_MBus_Info [▶ 222]).

stPulsecounter2: Pulse counter 2 (see ST_MBus_Info [▶ 222]).

VAR_IN_OUT

```
stCom        : ST_MBUS_Communication;
```

stCom: About this structure, the block FB_MBUSKL6781() [▶ 29] is connected to the meter function blocks (see ST_MBUS_Communication [▶ 221]).

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055646603/.zip:

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055648011/.zip:

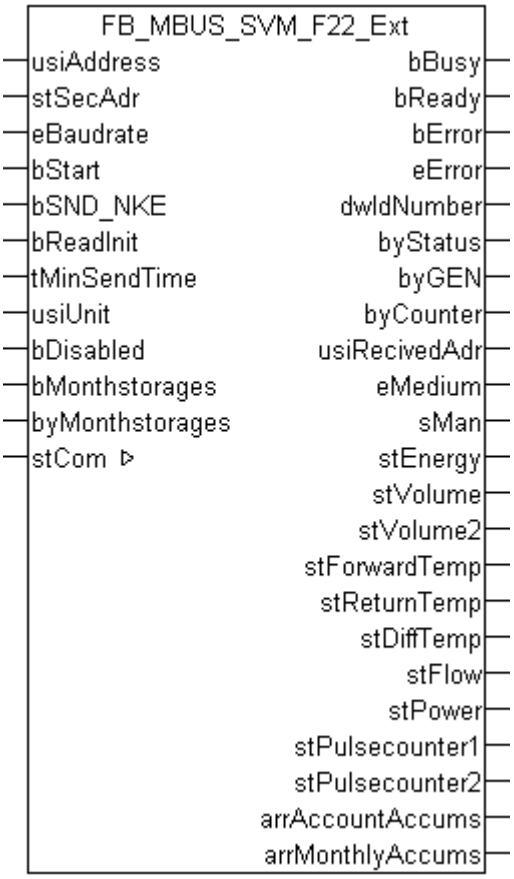
Controller configuration setting: "BCxx50 or BX serial"

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055649419/.zip:

BCxx00 must be set to "Large Model" in the PLC under "Project/Options/Controller Settings".

Controller configuration setting: "BC serial"

6.25.2 FB_MBUS_SVM_F22_Ext



This function block is used to read heat meters from Metrima:

-F22

The function block can only be executed together with the function block FB_MBUSKL6781() [▶ 29].

This function block is not suitable for BC/BX.

Functionality of the function block [▶ 13]

VAR_INPUT

```
usiAddress          : USINT;
stSecAdr            : ST_MBUS_SecAdr;
eBaudrate           : E_MBUS_Baudrate := eMBUS_Baud2400;
bStart              : BOOL;
bSND_NKE            : BOOL := TRUE;
bReadInit           : BOOL := TRUE;
tMinSendTime        : TIME := t#2s;
usiUnit             : USINT;
bDisabled           : BOOL := FALSE;
bMonthstorages      : BOOL;
byMonthstorages     : BYTE;
```

usiAddress: Primary address [▶ 14] of the counter, that shall be readout with this module.

stSecAdr: Secondary address [▶ 15] of the counter, that shall be readout with this module.

eBaudrate: 300, 2400, 9600 baud [▶ 216].

bStart: Positive edge on this input, the meter is read out once.

bSND_NKE: TRUE to initialize the meter at each reading, and sets the meter on the first telegram (SND_NKE).

bReadInit: After restarting the PLC, the meter is read out once.

tMinSendTime: Standard t#2s. When this time exceeds, the counter is rereadout. At t#0s the counter is not readout and can be readout with bStart manually.

usiUnit: Unit of the energy values. 0=W/J(h) / 1=KW/KJ(h) / 2=MW/MJ(h) / 3=GW/GJ(h).

bDisabled: TRUE = disable the function block.

bMonthstorages:

byMonthstorages:

VAR_OUTPUT

```
bBusy               : BOOL;
bReady              : BOOL;
bError              : BOOL;
eError              : E_MBUS_ERROR;
dwIdNumber          : DWORD;
byStatus            : BYTE;
byGEN               : BYTE;
byCounter           : BYTE;
usiRecivedAdr       : USINT;
eMedium             : E_MBUS_Medium;
sMan                : STRING(3);
stEnergy            : ST_MBus_Info;
stVolume            : ST_MBus_Info;
stVolume2           : ST_MBus_Info;
stForwardTemp       : ST_MBus_Info;
stReturnTemp        : ST_MBus_Info;
stDiffTemp          : ST_MBus_Info;
stFlow              : ST_MBus_Info;
stPower             : ST_MBus_Info;
stPulsecounter1     : ST_MBus_Info;
stPulsecounter2     : ST_MBus_Info;
arrAccountAccums    : ARRAY [1..2] OF ST_MBus_F22;
arrMonthlyAccums    : ARRAY [1..37] OF ST_MBus_F22;
```

bBusy: The *bBusy* output is TRUE while the meter is being read.

bReady: The *bReady* output is TRUE for one cycle, once meter reading is completed.

bError: The *bError* output becomes TRUE as soon as an error occurs. The error is described via the variable *eError*.

eError: The *eError* output issues an error code when an error occurs (see E_MBUS_ERROR [▶ 216]).

dwIdNumber: Serial number of the meter (secondary address).

byStatus: Status of the meter. Please refer to device description for meanings.

byGEN: Meter software version.

byCounter: Transmission counter (number of transmitted RSP_UD).

usiRecivedAdr: Received primary address (0-250).

eMedium: Medium (see E_MBUS_Medium [▶ 219]).

sMan: Manufacturer code.

stEnergy: Meter reading, energy consumption (see ST_MBus_Info [▶ 222]).

stVolume: Meter reading, volume (see ST_MBus_Info [▶ 222]).

stVolume2: Meter reading, volume (Corresponding to energy registers) (see ST_MBus_Info [▶ 222]).

stForwardTemp: Flow temperature (see ST_MBus_Info [▶ 222]).

stReturnTemp: Return temperature (see ST_MBus_Info [▶ 222]).

stDiffTemp: Temperature difference (see ST_MBus_Info [▶ 222]).

stFlow: Current flow (see ST_MBus_Info [▶ 222]).

stPower: Current energy consumption (see ST_MBus_Info [▶ 222]).

stPulsecounter1: Pulse counter 1 (see ST_MBus_Info [▶ 222]).

stPulsecounter2: Pulse counter 2 (see ST_MBus_Info [▶ 222]).

arrAccountAccums: 2 Account (Energy, Volume from watermeter, Volume according to energy, Pulse counter 1 H.C.A coded, Pulse counter 2 H.C.A coded, Date). Values are read only if *bMonthstorages* = true.

arrMonthlyAccums: Max. 37 monthstorages (Energy, Volume from watermeter, Volume according to energy, Pulse counter 1 H.C.A coded, Pulse counter 2 H.C.A coded, Date). Values are read only if *bMonthstorages* = true. The number of values depends on the variable *byMonthstorages*.

VAR_IN_OUT

```
stCom          : ST_MBUS_Communication;
```

stCom: About this structure, the block FB_MBUSKL6781() [▶ 29] is connected to the meter function blocks (see ST_MBUS_Communication [▶ 221]).

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055646603/.zip:

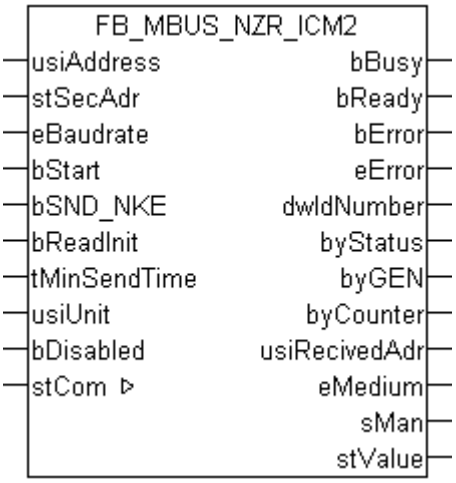
6.26 NZR

The function blocks only output a selection of the most common data. These data are described on the respective pages under "VAR_OUT". If more or all data are required, the function blocks FB_MBUS_General [▶ 31], FB_MBUS_General_Ext [▶ 35] or FB_MBUS_General_Param [▶ 41] from the General folder should be used. Not that these function blocks do not run on BC or BX systems. The function block FB_MBUS_General_Send [▶ 43] can be used to send data to the device (e.g. setting of the primary address).

Vendor	Type	Device	Function block
NZR	Electricity meter	EcoCount "S"	FB_MBUS_TIP_SINUS85M [▶ 207]
	2-way pulse adapter	IC-M2	FB_MBUS_NZR_ICM2 [▶ 166]
	2-way pulse adapter	IC-M2C	
	Water meter	Modularis 2	FB_MBUS_NZR_Modularis2 [▶ 168]

Vendor	Type	Device	Function block

6.26.1 FB_MBUS_NZR_ICM2



This block is used for reading energy meters with pulse output from NZR:

-IC-M2

-IC-M2C

The block can only be used in conjunction with the block FB_MBUSKL6781() [▶ 29].

Up to 2 encoder can be connected simultaneously to an IC-M2 / IC-M2C. The IC-M2 / IC-M2C behaves like 2 independent slaves.

Functionality of the function block [▶ 13]

VAR_INPUT

```
usiAddress      : USINT;
stSecAdr        : ST_MBUS_SecAdr;
eBaudrate       : E_MBUS_Baudrate := eMBUS_Baud2400;
bStart          : BOOL;
bSND_NKE        : BOOL := TRUE;
bReadInit       : BOOL := TRUE;
tMinSendTime    : TIME := t#2s;
usiUnit         : USINT;
bDisabled       : BOOL := FALSE;
```

usiAddress: Primary address [▶ 14] of the counter, that shall be readout with this module.

stSecAdr: Secondary address [▶ 15] of the counter, that shall be readout with this module.

eBaudrate: 300, 2400 baud [▶ 216].

bStart: Positive edge on this input, the meter is read out once.

bSND_NKE: TRUE to initialize the meter at each reading, and sets the meter on the first telegram (SND_NKE).

bReadInit: After restarting the PLC, the meter is read out once.

tMinSendTime: Standard t#2s. When this time exceeds, the counter is rereadout. At t#0s the counter is not readout and can be readout with bStart manually.

usiUnit: Unit of the energy values. 0=W/J(h) / 1=KW/KJ(h) / 2=MW/MJ(h) / 3=GW/GJ(h).

bDisabled: TRUE = disable the function block.

VAR_OUTPUT

```
bBusy           : BOOL;
bReady          : BOOL;
bError          : BOOL;
eError          : E_MBUS_ERROR;
dwIdNumber      : DWORD;
byStatus        : BYTE;
byGEN           : BYTE;
byCounter       : BYTE;
usiRecivedAdr   : USINT;
eMedium         : E_MBUS_Medium;
sMan            : STRING(3);
stValue         : ST_MBus_Info;
```

bBusy: The *bBusy* output is TRUE while the meter is being read.

bReady: The *bReady* output is TRUE for one cycle, once meter reading is completed.

bError: The *bError* output becomes TRUE as soon as an error occurs. The error is described via the variable *eError*.

eError: The *eError* output issues an error code when an error occurs (see E_MBUS_ERROR [▶ 216]).

dwIdNumber: Serial number of the meter (secondary address).

byStatus: Status of the meter. Please refer to device description for meanings.

byGEN: Meter software version.

byCounter: Transmission counter (number of transmitted RSP_UD).

usiRecivedAdr: Received primary address (0-250).

eMedium: Medium (see E_MBUS_Medium [▶ 219]).

sMan: Manufacturer code.

stValue: Meter reading (see ST_MBus_Info [▶ 222]).

VAR_IN_OUT

```
stCom           : ST_MBUS_Communication;
```

stCom: About this structure, the block FB_MBUSKL6781() [▶ 29] is connected to the meter function blocks (see ST_MBUS_Communication [▶ 221]).

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055650827/.zip:

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055652235/.zip: https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055652235/.zip

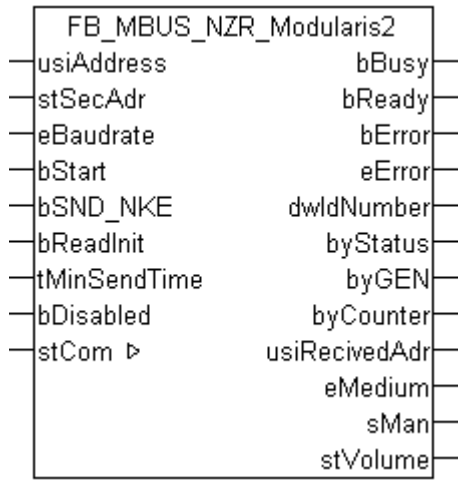
Controller configuration setting: "BCxx50 or BX serial"

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055653643/.zip:

BCxx00 must be set to "Large Model" in the PLC under "Project/Options/Controller Settings".

Controller configuration setting: "BC serial"

6.26.2 FB_MBUS_NZR_Modularis2



This function block is used to read water meters from NZR:

-Modularis 2

The function block can only be executed together with the function block FB_MBUSKL6781() [▶ 29].

Functionality of the function block [▶ 13]

VAR_INPUT

```
usiAddress      : USINT;
stSecAdr        : ST_MBUS_SecAdr;
eBaudrate       : E_MBUS_Baudrate := eMBUS_Baud2400;
bStart          : BOOL;
bSND_NKE        : BOOL := TRUE;
bReadInit       : BOOL := TRUE;
tMinSendTime    : TIME := t#2s;
bDisabled       : BOOL := FALSE;
```

usiAddress: Primary address [▶ 14] of the counter, that shall be readout with this module.

stSecAdr: Secondary address [▶ 15] of the counter, that shall be readout with this module.

eBaudrate: 300, 2400 baud [▶ 216].

bStart: Positive edge on this input, the meter is read out once.

bSND_NKE: TRUE to initialize the meter at each reading, and sets the meter on the first telegram (SND_NKE).

bReadInit: After restarting the PLC, the meter is read out once.

tMinSendTime: Standard t#2s. When this time exceeds, the counter is rereadout. At t#0s the counter is not readout and can be readout with bStart manually.

bDisabled: TRUE = disable the function block.

VAR_OUTPUT

```
bBusy           : BOOL;
bReady          : BOOL;
bError          : BOOL;
eError          : E_MBUS_ERROR;
dwIdNumber      : DWORD;
byStatus        : BYTE;
byGEN           : BYTE;
byCounter       : BYTE;
usiRecivedAdr   : USINT;
eMedium         : E_MBUS_Medium;
sMan            : STRING(3);
stVolume        : ST_MBus_Info;
```

bBusy: The *bBusy* output is TRUE while the meter is being read.

bReady: The *bReady* output is TRUE for one cycle, once meter reading is completed.

bError: The *bError* output becomes TRUE as soon as an error occurs. The error is described via the variable *eError*.

eError: The *eError* output issues an error code when an error occurs (see E_MBUS_ERROR [▶ 216]).

dwIdNumber: Serial number of the meter (secondary address).

byStatus: Status of the meter. Please refer to device description for meanings.

byGEN: Meter software version.

byCounter: Transmission counter (number of transmitted RSP_UD).

usiRecivedAdr: Received primary address (0-250).

eMedium: Medium (see E_MBUS_Medium [▶ 219]).

sMan: Manufacturer code.

stVolume: Meter reading, volume (see ST_MBus_Info [▶ 222]).

VAR_IN_OUT

```
stCom          : ST_MBUS_Communication;
```

stCom: About this structure, the block FB_MBUSKL6781() [▶ 29] is connected to the meter function blocks (see ST_MBUS_Communication [▶ 221]).

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055650827/.zip:

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055652235/.zip: https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055652235/.zip

Controller configuration setting: "BCxx50 or BX serial"

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055653643/.zip:

BCxx00 must be set to "Large Model" in the PLC under "Project/Options/Controller Settings".

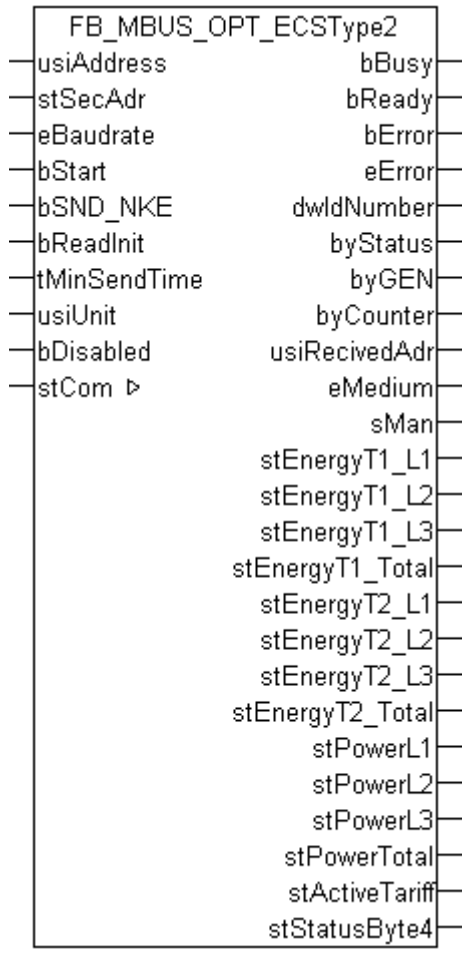
Controller configuration setting: "BC serial"

6.27 OPTEC

The function blocks only output a selection of the most common data. These data are described on the respective pages under "VAR_OUT". If more or all data are required, the function blocks FB_MBUS_General [▶ 31], FB_MBUS_General_Ext [▶ 35] or FB_MBUS_General_Param [▶ 41] from the General folder should be used. Not that these function blocks do not run on BC or BX systems. The function block FB_MBUS_General_Send [▶ 43] can be used to send data to the device (e.g. setting of the primary address).

Vendor	Type	Device	Function block
OPTEC	Electricity meter	ECS Type 2	FB_MBUS_OPT_ECSType2 [▶ 170]

6.27.1 FB_MBUS_OPT_ECSType2



This block is used for reading electricity meters from OPTEC:

-ECS (Default values type 2)

The block can only be used in conjunction with the block FB_MBUSKL6781() [▶ 29].

Functionality of the function block [▶ 13]

VAR_INPUT

```
usiAddress        : USINT;
stSecAdr          : ST_MBUS_SecAdr;
eBaudrate         : E_MBUS_Baudrate := eMBUS_Baud2400;
bStart            : BOOL;
bSND_NKE          : BOOL := TRUE;
bReadInit         : BOOL := TRUE;
tMinSendTime      : TIME := t#2s;
usiUnit           : USINT;
bDisabled         : BOOL := FALSE;
```

usiAddress: Primary address [▶ 14] of the counter, that shall be readout with this module.

stSecAdr: Secondary address [▶ 15] of the counter, that shall be readout with this module.

eBaudrate: 300, 600, 1200, 2400, 4800, 9600 baud [▶ 216].

bStart: Positive edge on this input, the meter is read out once.

bSND_NKE: TRUE to initialize the meter at each reading, and sets the meter on the first telegram (SND_NKE).

bReadInit: After restarting the PLC, the meter is read out once.

tMinSendTime: Standard t#2s. When this time exceeds, the counter is reread out. At t#0s the counter is not readout and can be readout with bStart manually.

usiUnit: Unit of the energy values. 0=W/J(h) / 1=KW/KJ(h) / 2=MW/MJ(h) / 3=GW/GJ(h).

bDisabled: TRUE = disable the function block.

VAR_OUTPUT

```
bBusy             : BOOL;
bReady            : BOOL;
bError            : BOOL;
eError            : E_MBUS_ERROR;
dwIdNumber        : DWORD;
byStatus          : BYTE;
byGEN             : BYTE;
byCounter         : BYTE;
usiRecivedAdr     : USINT;
eMedium           : E_MBUS_Medium;
sMan              : STRING(3);
stEnergyT1_L1     : ST_MBus_Info;
stEnergyT1_L2     : ST_MBus_Info;
stEnergyT1_L3     : ST_MBus_Info;
stEnergyT1_Total  : ST_MBus_Info;
stEnergyT2_L1     : ST_MBus_Info;
stEnergyT2_L2     : ST_MBus_Info;
stEnergyT2_L3     : ST_MBus_Info;
stEnergyT2_Total  : ST_MBus_Info;
stPowerL1         : ST_MBus_Info;
stPowerL2         : ST_MBus_Info;
stPowerL3         : ST_MBus_Info;
stPowerTotal      : ST_MBus_Info;
stActiveTariff    : ST_MBus_Info;
stStatusByte4     : ST_MBus_Info;
```

bBusy: The *bBusy* output is TRUE while the meter is being read.

bReady: The *bReady* output is TRUE for one cycle, once meter reading is completed.

bError: The *bError* output becomes TRUE as soon as an error occurs. The error is described via the variable *eError*.

eError: The *eError* output issues an error code when an error occurs (see E_MBUS_ERROR [▶ 216]).

dwIdNumber: Serial number of the meter (secondary address).

byStatus: Status of the meter. Please refer to device description for meanings.

byGEN: Meter software version.

byCounter: Transmission counter (number of transmitted RSP_UD).

usiRecivedAdr: Received primary address (0-250).

eMedium: Medium (see E_MBUS_Medium [▶ 219]).

sMan: Manufacturer code.

stEnergyT1_L1: Meter reading, active energy, tariff 1, phase L1 (see ST_MBus_Info [▶ 222]).

stEnergyT1_L2: Meter reading, active energy, tariff 1, phase L2 (see ST_MBus_Info [▶ 222]).

stEnergyT1_L3: Meter reading, active energy, tariff 1, phase L3 (see ST_MBus_Info [▶ 222]).

stEnergyT1_Total: Meter reading, active energy, tariff 1, total (see ST_MBus_Info [▶ 222]).

stEnergyT2_L1: Meter reading, active energy, tariff 2, phase L1 (see ST_MBus_Info [▶ 222]).

stEnergyT2_L2: Meter reading, active energy, tariff 2, phase L2 (see ST_MBus_Info [▶ 222]).

stEnergyT2_L3: Meter reading, active energy, tariff 2, phase L3 (see ST_MBus_Info [▶ 222]).

stEnergyT2_Total: Meter reading, active energy, tariff 2, total (see ST_MBus_Info [▶ 222]).

stPowerL1: Active power, phase L1 (see ST_MBus_Info [▶ 222]).

stPowerL2: Active power, phase L2 (see ST_MBus_Info [▶ 222]).

stPowerL3: Active power, phase L3 (see ST_MBus_Info [▶ 222]).

stPowerTotal: Active power, total (see ST_MBus_Info [▶ 222]).

stActiveTariff: Current tariff (see ST_MBus_Info [▶ 222]).

stStatusByte4: Range overflow alarms (see ST_MBus_Info [▶ 222]).

VAR_IN_OUT

```
stCom          : ST_MBUS_Communication;
```

stCom: About this structure, the block FB_MBUSKL6781() [▶ 29] is connected to the meter function blocks (see ST_MBUS_Communication [▶ 221]).

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055655051/.zip:

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055656459/.zip:

Controller configuration setting: "BCxx50 or BX serial"

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055657867/.zip:

BCxx00 must be set to "Large Model" in the PLC under "Project/Options/Controller Settings".

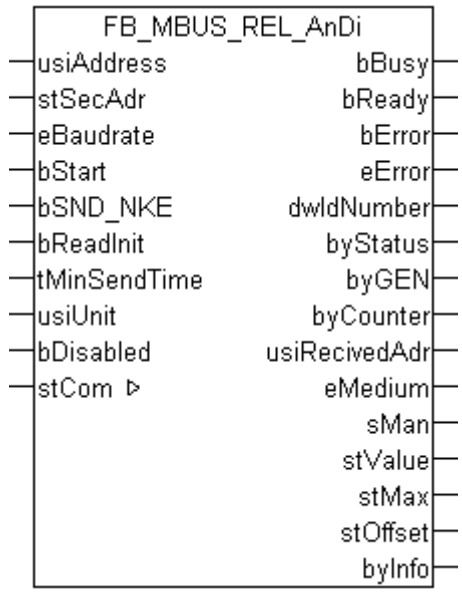
Controller configuration setting: "BC serial"

6.28 Relay

The function blocks only output a selection of the most common data. These data are described on the respective pages under "VAR_OUT". If more or all data are required, the function blocks FB_MBUS_General [▶ 31], FB_MBUS_General_Ext [▶ 35] or FB_MBUS_General_Param [▶ 41] from the General folder should be used. Not that these function blocks do not run on BC or BX systems. The function block FB_MBUS_General_Send [▶ 43] can be used to send data to the device (e.g. setting of the primary address).

Vendor	Type	Device	Function block
Relay	1-4 analog inputs	AnDi 1-4	FB_MBUS_REL_AnDi [▶ 173]
	4 digital inputs	PadIn 4	FB_MBUS_REL_PadIn4 [▶ 175]
	1-way pulse adapter	PadPuls M1	FB_MBUS_REL_PadPulsM1 [▶ 177]
	1-way pulse adapter	PadPuls M1C	
	2-way pulse adapter	PadPuls M2	FB_MBUS_REL_PadPulsM2 [▶ 179]
	2-way pulse adapter	PadPuls M2C	
	4-way pulse adapter	PadPuls M4	FB_MBUS_REL_PadPulsM4 [▶ 181]
	4-way pulse adapter	PadPuls M4L	

6.28.1 FB_MBUS_REL_AnDi



This block is used for reading analog converters from Relay:

-AnDi 1 (1x 0/4-20 mA or 0-10 V)

-AnDi 2 (2x 0/4-20 mA or 0-10 V)

-AnDi 3 (3x 0/4-20 mA or 0-10 V)

-AnDi 4 (4x 0/4-20 mA or 0-10 V)

The block can only be used in conjunction with the block FB_MBUSKL6781() [▶ 29].

Up to 4 sensors can be connected simultaneously to an AnDi 1/2/3/4. The AnDi 1/2/3/4 behaves like 4 independent slaves.

Functionality of the function block [▶ 13]

VAR_INPUT

```
usiAddress      : USINT;
stSecAdr        : ST_MBUS_SecAdr;
eBaudrate       : E_MBUS_Baudrate := eMBUS_Baud2400;
bStart          : BOOL;
bSND_NKE        : BOOL := TRUE;
bReadInit       : BOOL := TRUE;
tMinSendTime    : TIME := t#2s;
usiUnit         : USINT;
bDisabled       : BOOL := FALSE;
```

usiAddress: Primary address [▶ 14] of the counter, that shall be readout with this module.

stSecAdr: Secondary address [▶ 15] of the counter, that shall be readout with this module.

eBaudrate: 300, 2400 baud [▶ 216].

bStart: Positive edge on this input, the meter is read out once.

bSND_NKE: TRUE to initialize the meter at each reading, and sets the meter on the first telegram (SND_NKE).

bReadInit: After restarting the PLC, the meter is read out once.

tMinSendTime: Standard t#2s. When this time exceeds, the counter is rereadout. At t#0s the counter is not readout and can be readout with bStart manually.

usiUnit: Unit of the energy values. 0=W/J(h) / 1=KW/KJ(h) / 2=MW/MJ(h) / 3=GW/GJ(h).

bDisabled: TRUE = disable the function block.

VAR_OUTPUT

```
bBusy           : BOOL;
bReady          : BOOL;
bError          : BOOL;
eError          : E_MBUS_ERROR;
dwIdNumber      : DWORD;
byStatus        : BYTE;
byGEN           : BYTE;
byCounter       : BYTE;
usiRecivedAdr   : USINT;
eMedium         : E_MBUS_Medium;
sMan            : STRING(3);
stValue         : ST_MBus_Info;
stMax           : ST_MBus_Info;
stOffset        : ST_MBus_Info;
byInfo          : BYTE;
```

bBusy: The *bBusy* output is TRUE while the meter is being read.

bReady: The *bReady* output is TRUE for one cycle, once meter reading is completed.

bError: The *bError* output becomes TRUE as soon as an error occurs. The error is described via the variable *eError*.

eError: The *eError* output issues an error code when an error occurs (see E_MBUS_ERROR [▶ 216]).

dwIdNumber: Serial number of the meter (secondary address).

byStatus: Status of the meter. Please refer to device description for meanings.

byGEN: Meter software version.

byCounter: Transmission counter (number of transmitted RSP_UD).

usiRecivedAdr: Received primary address (0-250).

eMedium: Medium (see E_MBUS_Medium [▶ 219]).

sMan: Manufacturer code.

stValue: Meter reading (see ST_MBus_Info [▶ 222]).

stMax: Maximum value (see ST_MBus_Info [▶ 222]).

stOffset: Offset (see ST_MBus_Info [▶ 222]).

byInfo: Byte with the following information:
nBit7-4: Information about the available modules in AnDi4
nBit3: protection bit (1: protected)
nBit2-1: no. of selected channel (0: Port1 ... 3: Port4)
nBit0: operating mode (1: current, 2: voltage)

VAR_IN_OUT

```
stCom           : ST_MBUS_Communication;
```

stCom: About this structure, the block FB_MBUSKL6781() [▶ 29] is connected to the meter function blocks (see ST_MBUS_Communication [▶ 221]).

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055659275/.zip:

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055660683/.zip:

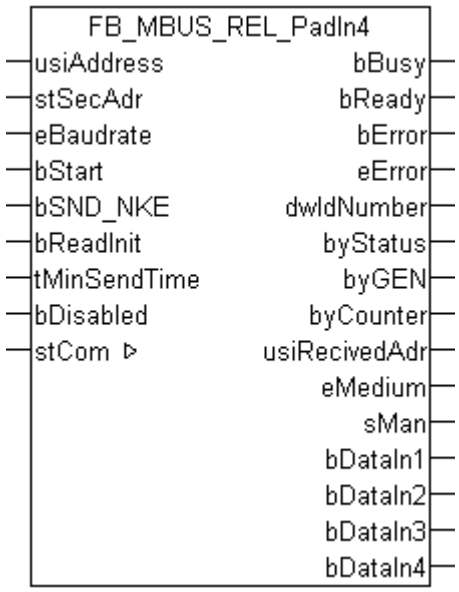
Controller configuration setting: "BCxx50 or BX serial"

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055662091/.zip:

BCxx00 must be set to "Large Model" in the PLC under "Project/Options/Controller Settings".

Controller configuration setting: "BC serial"

6.28.2 FB_MBUS_REL_PadIn4



This function block is used for reading digital inputs from Relay:

-PadIn 4 (4 digital inputs)

The function block can only be executed together with the function block FB_MBUSKL6781() [▶ 29].

Functionality of the function block [▶ 13]

VAR_INPUT

```
usiAddress      : USINT;
stSecAdr        : ST_MBUS_SecAdr;
eBaudrate       : E_MBUS_Baudrate := eMBUS_Baud2400;
bStart          : BOOL;
bSND_NKE        : BOOL := TRUE;
bReadInit       : BOOL := TRUE;
tMinSendTime    : TIME := t#2s;
bDisabled       : BOOL := FALSE;
```

usiAddress: Primary address [▶ 14] of the counter, that shall be readout with this module.

stSecAdr: Secondary address [▶ 15] of the counter, that shall be readout with this module.

eBaudrate: 300, 2400, 9600 baud [▶ 216].

bStart: Positive edge on this input, the meter is read out once.

bSND_NKE: TRUE to initialize the meter at each reading, and sets the meter on the first telegram (SND_NKE).

bReadInit: After restarting the PLC, the meter is read out once.

tMinSendTime: Standard t#2s. When this time exceeds, the counter is rereadout. At t#0s the counter is not readout and can be readout with bStart manually.

bDisabled: TRUE = disable the function block.

VAR_OUTPUT

```
bBusy           : BOOL;
bReady          : BOOL;
bError          : BOOL;
eError          : E_MBUS_ERROR;
dwIdNumber      : DWORD;
byStatus        : BYTE;
byGEN           : BYTE;
byCounter       : BYTE;
usiRecivedAdr   : USINT;
eMedium         : E_MBUS_Medium;
sMan            : STRING(3);
bDataIn1        : BOOL;
bDataIn2        : BOOL;
bDataIn3        : BOOL;
bDataIn4        : BOOL;
```

bBusy: The *bBusy* output is TRUE while the meter is being read.

bReady: The *bReady* output is TRUE for one cycle, once meter reading is completed.

bError: The *bError* output becomes TRUE as soon as an error occurs. The error is described via the variable *eError*.

eError: The *eError* output issues an error code when an error occurs (see E_MBUS_ERROR [▶ 216]).

dwIdNumber: Serial number of the meter (secondary address).

byStatus: Status of the meter. Please refer to device description for meanings.

byGEN: Meter software version.

byCounter: Transmission counter (number of transmitted RSP_UD).

usiRecivedAdr: Received primary address (0-250).

eMedium: Medium (see E_MBUS_Medium [▶ 219]).

sMan: Manufacturer code.

bDataIn1: Input 1.

bDataIn2: Input 2.

bDataIn3: Input 3.

bDataIn4: Input 4.

VAR_IN_OUT

```
stCom           : ST_MBUS_Communication;
```

stCom: About this structure, the block FB_MBUSKL6781() [▶ 29] is connected to the meter function blocks (see ST_MBUS_Communication [▶ 221]).

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055659275/.zip:

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055660683/.zip:

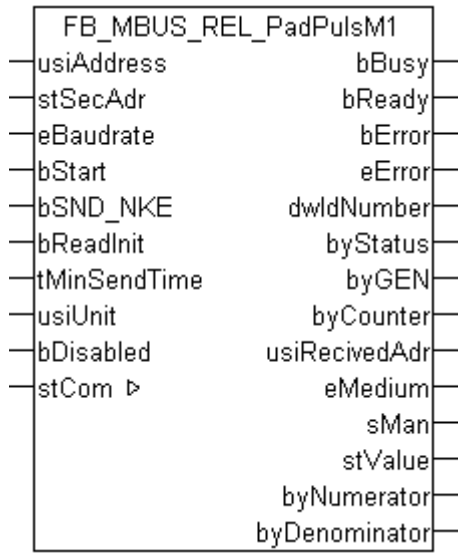
Controller configuration setting: "BCxx50 or BX serial"

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055662091/.zip:

BCxx00 must be set to "Large Model" in the PLC under "Project/Options/Controller Settings".

Controller configuration setting: "BC serial"

6.28.3 FB_MBUS_REL_PadPulsM1



This function block is used for reading energy meters with pulse output from Relay:

-PadPuls M1

-PadPuls M1C

The function block can only be executed together with the function block FB_MBUSKL6781() [▶ 29].

Functionality of the function block [▶ 13]

VAR_INPUT

```
usiAddress      : USINT;
stSecAdr        : ST_MBUS_SecAdr;
eBaudrate       : E_MBUS_Baudrate := eMBUS_Baud2400;
bStart          : BOOL;
bSND_NKE        : BOOL := TRUE;
bReadInit       : BOOL := TRUE;
tMinSendTime    : TIME := t#2s;
usiUnit         : USINT;
bDisabled       : BOOL := FALSE;
```

usiAddress: Primary address [▶ 14] of the counter, that shall be readout with this module.

stSecAdr: Secondary address [▶ 15] of the counter, that shall be readout with this module.

eBaudrate: 300, 2400, 9600 baud [▶ 216].

bStart: Positive edge on this input, the meter is read out once.

bSND_NKE: TRUE to initialize the meter at each reading, and sets the meter on the first telegram (SND_NKE).

bReadInit: After restarting the PLC, the meter is read out once.

tMinSendTime: Standard t#2s. When this time exceeds, the counter is rereadout. At t#0s the counter is not readout and can be readout with bStart manually.

usiUnit: Unit of the energy values. 0=W/J(h) / 1=KW/KJ(h) / 2=MW/MJ(h) / 3=GW/GJ(h).

bDisabled: TRUE = disable the function block.

VAR_OUTPUT

```
bBusy           : BOOL;
bReady          : BOOL;
bError          : BOOL;
```

```
eError          : E_MBUS_ERROR;
dwIdNumber      : DWORD;
byStatus        : BYTE;
byGEN           : BYTE;
byCounter       : BYTE;
usiRecivedAdr   : USINT;
eMedium         : E_MBUS_Medium;
sMan            : STRING(3);
stValue         : ST_MBus_Info;
byNumerator     : BYTE;
byDenominator   : BYTE;
```

bBusy: The *bBusy* output is TRUE while the meter is being read.

bReady: The *bReady* output is TRUE for one cycle, once meter reading is completed.

bError: The *bError* output becomes TRUE as soon as an error occurs. The error is described via the variable *eError*.

eError: The *eError* output issues an error code when an error occurs (see E_MBUS_ERROR [▶ 216]).

dwIdNumber: Serial number of the meter (secondary address).

byStatus: Status of the meter. Please refer to device description for meanings.

byGEN: Meter software version.

byCounter: Transmission counter (number of transmitted RSP_UD).

usiRecivedAdr: Received primary address (0-250).

eMedium: Medium (see E_MBUS_Medium [▶ 219]).

sMan: Manufacturer code.

stValue: Meter reading (see ST_MBus_Info [▶ 222]).

byNumerator: Numerator Impulse value.

byDenominator: Denominator Impulse value.

VAR_IN_OUT

```
stCom           : ST_MBUS_Communication;
```

stCom: About this structure, the block FB_MBUSKL6781() [▶ 29] is connected to the meter function blocks (see ST_MBUS_Communication [▶ 221]).

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055659275/.zip:

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055660683/.zip:

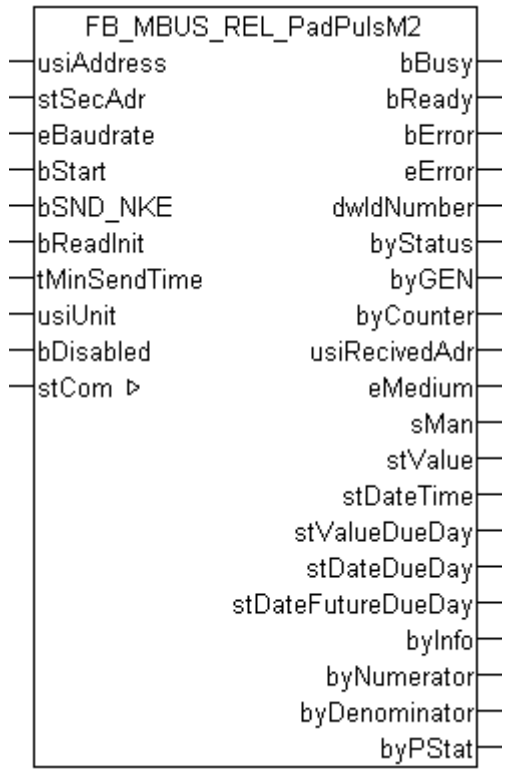
Controller configuration setting: "BCxx50 or BX serial"

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055662091/.zip:

BCxx00 must be set to "Large Model" in the PLC under "Project/Options/Controller Settings".

Controller configuration setting: "BC serial"

6.28.4 FB_MBUS_REL_PadPulsM2



This function block is used for reading energy meters with pulse output from Relay:

-PadPuls M2

-PadPuls M2C

The function block can only be executed together with the function block FB_MBUSKL6781() [▶ 29].

Up to 2 pulse generators can be connected to a PadPuls M2 / PadPuls M2C at the same time. The PadPuls M2 / PadPuls M2C behaves like 2 independent slaves.

Functionality of the function block [▶ 13]

VAR_INPUT

```
usiAddress      : USINT;
stSecAdr        : ST_MBUS_SecAdr;
eBaudrate       : E_MBUS_Baudrate := eMBUS_Baud2400;
bStart          : BOOL;
bSND_NKE        : BOOL := TRUE;
bReadInit       : BOOL := TRUE;
tMinSendTime    : TIME := t#2s;
usiUnit         : USINT;
bDisabled       : BOOL := FALSE;
```

usiAddress: Primary address [▶ 14] of the counter, that shall be readout with this module.

stSecAdr: Secondary address [▶ 15] of the counter, that shall be readout with this module.

eBaudrate: 300, 2400 baud [▶ 216].

bStart: Positive edge on this input, the meter is read out once.

bSND_NKE: TRUE to initialize the meter at each reading, and sets the meter on the first telegram (SND_NKE).

bReadInit: After restarting the PLC, the meter is read out once.

tMinSendTime: Standard t#2s. When this time exceeds, the counter is rereadout. At t#0s the counter is not readout and can be readout with bStart manually.

usiUnit: Unit of the energy values. 0=W/J(h) / 1=KW/KJ(h) / 2=MW/MJ(h) / 3=GW/GJ(h).

bDisabled: TRUE = disable the function block.

VAR_OUTPUT

```
bBusy               : BOOL;
bReady              : BOOL;
bError              : BOOL;
eError              : E_MBUS_ERROR;
dwIdNumber          : DWORD;
byStatus            : BYTE;
byGEN               : BYTE;
byCounter           : BYTE;
usiRecivedAdr       : USINT;
eMedium             : E_MBUS_Medium;
sMan                : STRING(3);
stValue             : ST_MBus_Info;
stDateTime          : ST_MBus_Info;
stValueDueDay       : ST_MBus_Info;
stDateDueDay        : ST_MBus_Info;
stDateFutureDueDay  : ST_MBus_Info;
byInfo              : BYTE;
byNumerator         : BYTE;
byDenominator       : BYTE;
byPStat             : BYTE;
```

bBusy: The *bBusy* output is TRUE while the meter is being read.

bReady: The *bReady* output is TRUE for one cycle, once meter reading is completed.

bError: The *bError* output becomes TRUE as soon as an error occurs. The error is described via the variable *eError*.

eError: The *eError* output issues an error code when an error occurs (see E_MBUS_ERROR [▶ 216]).

dwIdNumber: Serial number of the meter (secondary address).

byStatus: Status of the meter. Please refer to device description for meanings.

byGEN: Meter software version.

byCounter: Transmission counter (number of transmitted RSP_UD).

usiRecivedAdr: Received primary address (0-250).

eMedium: Medium (see E_MBUS_Medium [▶ 219]).

sMan: Manufacturer code.

stValue: Meter reading (see ST_MBus_Info [▶ 222]).

stDateTime: Current date (see ST_MBus_Info [▶ 222]).

stValueDueDay: Due-date counter (see ST_MBus_Info [▶ 222]).

stDateDueDay: Last due-date, date of the due-date counter (see ST_MBus_Info [▶ 222]).

stDateFutureDueDay: Next (future) due-date (see ST_MBus_Info [▶ 222]).

byInfo: 1 byte with information about tariff and sampling method.

byNumerator: Numerator of pulse increment (1..99).

byDenominator: denominator of pulse increment (1..255, 0 -> 256).

byPStat: State of inputs (current input state of the ports).

VAR_IN_OUT

```
stCom          : ST_MBUS_Communication;
```

stCom: About this structure, the block FB_MBUSKL6781() [▶ 29] is connected to the meter function blocks (see ST_MBUS_Communication [▶ 221]).

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055659275/.zip:

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055660683/.zip:

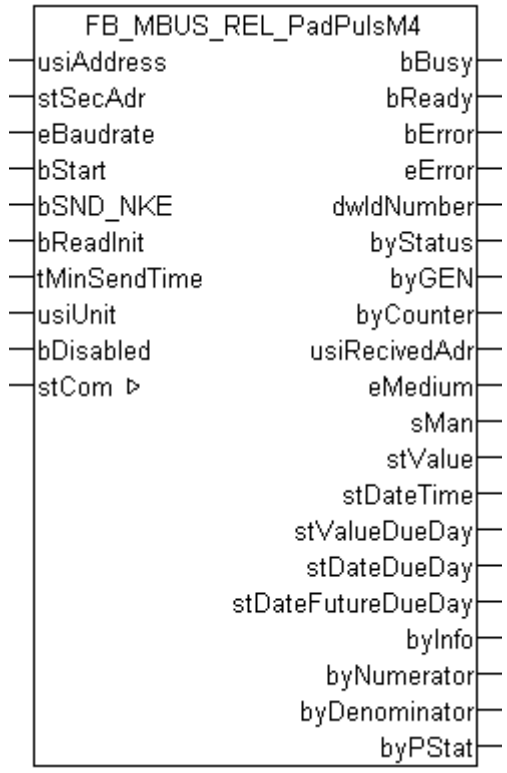
Controller configuration setting: "BCxx50 or BX serial"

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055662091/.zip:

BCxx00 must be set to "Large Model" in the PLC under "Project/Options/Controller Settings".

Controller configuration setting: "BC serial"

6.28.5 FB_MBUS_REL_PadPulsM4



This function block is used for reading energy meters with pulse output from Relay:

-PadPuls M4

-PadPuls M4L

The function block can only be executed together with the function block FB_MBUSKL6781() [▶ 29].

Up to 4 pulse generators can be connected to a PadPuls M4 / PadPuls M4L at the same time. The PadPuls M4 / PadPuls M4L behaves like 4 independent slaves.

Functionality of the function block [▶ 13]

VAR_INPUT

```
usiAddress      : USINT;
stSecAdr        : ST_MBUS_SecAdr;
eBaudrate       : E_MBUS_Baudrate := eMBUS_Baud2400;
bStart          : BOOL;
bSND_NKE        : BOOL := TRUE;
bReadInit       : BOOL := TRUE;
tMinSendTime    : TIME := t#2s;
usiUnit         : USINT;
bDisabled       : BOOL := FALSE;
```

usiAddress: Primary address [▶ 14] of the counter, that shall be readout with this module.

stSecAdr: Secondary address [▶ 15] of the counter, that shall be readout with this module.

eBaudrate: 300, 2400 baud [▶ 216].

bStart: Positive edge on this input, the meter is read out once.

bSND_NKE: TRUE to initialize the meter at each reading, and sets the meter on the first telegram (SND_NKE).

bReadInit: After restarting the PLC, the meter is read out once.

tMinSendTime: Standard t#2s. When this time exceeds, the counter is rereadout. At t#0s the counter is not readout and can be readout with bStart manually.

usiUnit: Unit of the energy values. 0=W/J(h) / 1=KW/KJ(h) / 2=MW/MJ(h) / 3=GW/GJ(h).

bDisabled: TRUE = disable the function block.

VAR_OUTPUT

```
bBusy               : BOOL;
bReady              : BOOL;
bError              : BOOL;
eError              : E_MBUS_ERROR;
dwIdNumber          : DWORD;
byStatus            : BYTE;
byGEN               : BYTE;
byCounter           : BYTE;
usiRecivedAdr       : USINT;
eMedium             : E_MBUS_Medium;
sMan                : STRING(3);
stValue             : ST_MBus_Info;
stDateTime          : ST_MBus_Info;
stValueDueDay       : ST_MBus_Info;
stDateDueDay        : ST_MBus_Info;
stDateFutureDueDay  : ST_MBus_Info;
byInfo              : BYTE;
byNumerator         : BYTE;
byDenominator       : BYTE;
byPStat             : BYTE;
```

bBusy: The *bBusy* output is TRUE while the meter is being read.

bReady: The *bReady* output is TRUE for one cycle, once meter reading is completed.

bError: The *bError* output becomes TRUE as soon as an error occurs. The error is described via the variable *eError*.

eError: The *eError* output issues an error code when an error occurs (see E_MBUS_ERROR [▶ 216]).

dwIdNumber: Serial number of the meter (secondary address).

byStatus: Status of the meter. Please refer to device description for meanings.

byGEN: Meter software version.

byCounter: Transmission counter (number of transmitted RSP_UD).

usiRecivedAdr: Received primary address (0-250).

eMedium: Medium (see E_MBUS_Medium [▶ 219]).

sMan: Manufacturer code.

stValue: Meter reading (see ST_MBus_Info [▶ 222]).

stDateTime: Current date (see ST_MBus_Info [▶ 222]).

stValueDueDay: Due-date counter (see ST_MBus_Info [▶ 222]).

stDateDueDay: Last due-date, date of the due-date counter (see ST_MBus_Info [▶ 222]).

stDateFutureDueDay: Next (future) due-date (see ST_MBus_Info [▶ 222]).

byInfo: 1 byte with information about tariff and sampling method.

byNumerator: Numerator of pulse increment (1..99).

byDenominator: Denominator of pulse increment (1..255, 0 -> 256).

byPStat: State of inputs (current input state of the ports).

VAR_IN_OUT

```
stCom          : ST_MBUS_Communication;
```

stCom: About this structure, the block FB_MBUSKL6781() [▶ 29] is connected to the meter function blocks (see ST_MBUS_Communication [▶ 221]).

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055659275/.zip:

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055660683/.zip:

Controller configuration setting: "BCxx50 or BX serial"

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055662091/.zip:

BCxx00 must be set to "Large Model" in the PLC under "Project/Options/Controller Settings".

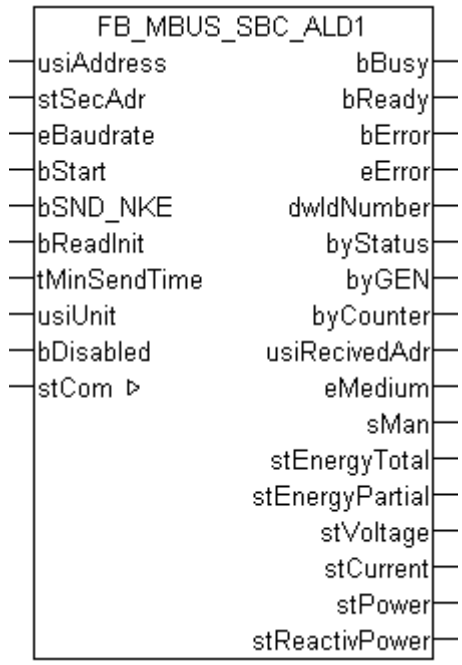
Controller configuration setting: "BC serial"

6.29 Saia-Burgess

The function blocks only output a selection of the most common data. These data are described on the respective pages under "VAR_OUT". If more or all data are required, the function blocks FB_MBUS_General [▶ 31], FB_MBUS_General_Ext [▶ 35] or FB_MBUS_General_Param [▶ 41] from the General folder should be used. Not that these function blocks do not run on BC or BX systems. The function block FB_MBUS_General_Send [▶ 43] can be used to send data to the device (e.g. setting of the primary address).

Vendor	Type	Device	Function block
Saia-Burgess	Electricity meter	ALD1	FB_MBUS_SBC_ALD1 [▶ 184]
	Electricity meter	ALE3	FB_MBUS_SBC_ALE3 [▶ 186]
	Electricity meter	AWD3	

6.29.1 FB_MBUS_SBC_ALD1



This block is used for reading electricity meters from Saia-Burgess:

-ALD1

The block can only be used in conjunction with the block FB_MBUSKL6781() [▶ 29].

Functionality of the function block [▶ 13]

VAR_INPUT

```
usiAddress        : USINT;
stSecAdr          : ST_MBUS_SecAdr;
eBaudrate         : E_MBUS_Baudrate := eMBUS_Baud2400;
bStart            : BOOL;
bSND_NKE          : BOOL := TRUE;
bReadInit         : BOOL := TRUE;
tMinSendTime      : TIME := t#2s;
usiUnit           : USINT;
bDisabled         : BOOL := FALSE;
```

usiAddress: Primary address [▶ 14] of the counter, that shall be readout with this module.

stSecAdr: Secondary address [▶ 15] of the counter, that shall be readout with this module.

eBaudrate: 300, 2400, 9600 baud [▶ 216].

bStart: Positive edge on this input, the meter is read out once.

bSND_NKE: TRUE to initialize the meter at each reading, and sets the meter on the first telegram (SND_NKE).

bReadInit: After restarting the PLC, the meter is read out once.

tMinSendTime: Standard t#2s. When this time exceeds, the counter is rereadout. At t#0s the counter is not readout and can be readout with bStart manually.

usiUnit: Unit of the energy values. 0=W/J(h) / 1=KW/KJ(h) / 2=MW/MJ(h) / 3=GW/GJ(h).

bDisabled: TRUE = disable the function block.

VAR_OUTPUT

```
bBusy            : BOOL;
bReady           : BOOL;
bError           : BOOL;
eError           : E_MBUS_ERROR;
dwIdNumber       : DWORD;
byStatus         : BYTE;
byGEN            : BYTE;
byCounter        : BYTE;
usiRecivedAdr    : USINT;
eMedium          : E_MBUS_Medium;
sMan             : STRING(3);
stEnergyTotal    : ST_MBus_Info;
stEnergyPartial  : ST_MBus_Info;
stVoltage        : ST_MBus_Info;
stCurrent        : ST_MBus_Info;
stPower          : ST_MBus_Info;
stReactivPower   : ST_MBus_Info;
```

bBusy: The *bBusy* output is TRUE while the meter is being read.

bReady: The *bReady* output is TRUE for one cycle, once meter reading is completed.

bError: The *bError* output becomes TRUE as soon as an error occurs. The error is described via the variable *eError*.

eError: The *eError* output issues an error code when an error occurs (see E_MBUS_ERROR [▶ 216]).

dwIdNumber: Serial number of the meter (secondary address).

byStatus: Status of the meter. Please refer to device description for meanings.

byGEN: Meter software version.

byCounter: Transmission counter (number of transmitted RSP_UD).

usiRecivedAdr: Received primary address (0-250).

eMedium: Medium (see E_MBUS_Medium [▶ 219]).

sMan: Manufacturer code.

stEnergyTotal: Meter reading, total energy (see ST_MBus_Info [▶ 222]).

stEnergyPartial: Meter reading, energy partial (see ST_MBus_Info [▶ 222]).

stVoltage: Voltage (see ST_MBus_Info [▶ 222]).

stCurrent: Electric current (see ST_MBus_Info [▶ 222]).

stPower: Power (see ST_MBus_Info [▶ 222]).

stReactivPower: Reactive power (see ST_MBus_Info [▶ 222]).

VAR_IN_OUT

```
stCom        : ST_MBUS_Communication;
```

stCom: About this structure, the block FB_MBUSKL6781() [▶ 29] is connected to the meter function blocks (see ST_MBUS_Communication [▶ 221]).

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055663499/.zip:

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055664907/.zip:

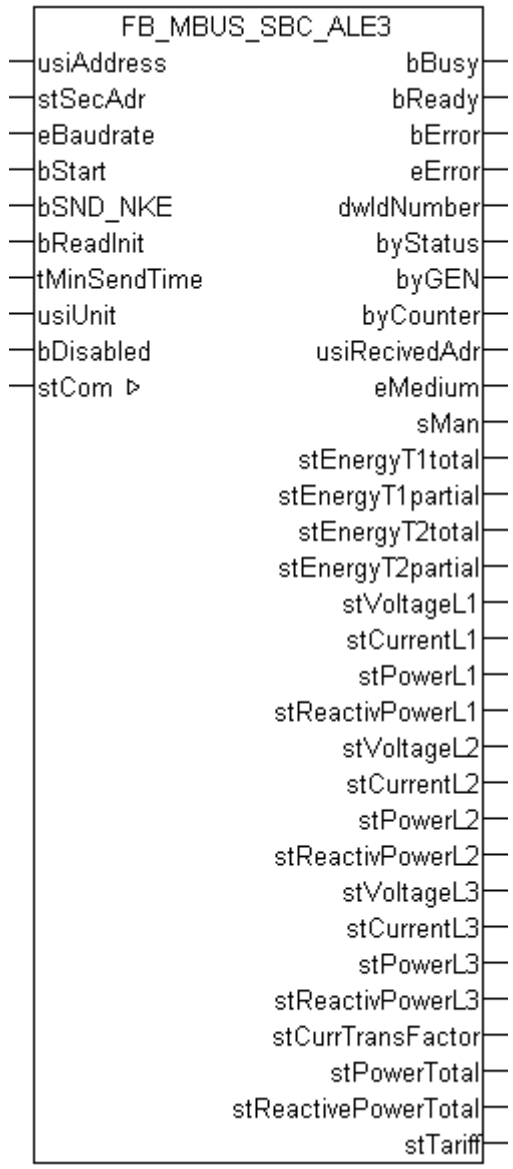
Controller configuration setting: "BCxx50 or BX serial"

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055666315/.zip:

BCxx00 must be set to "Large Model" in the PLC under "Project/Options/Controller Settings".

Controller configuration setting: "BC serial"

6.29.2 FB_MBUS_SBC_ALE3



This function block is used to read electricity meters from Saia-Burgess:

-ALE3

-AWD3

The function block can only be executed together with the function block FB_MBUSKL6781() [▶ 29].

Functionality of the function block [▶ 13]

VAR_INPUT

```
usiAddress          : USINT;
stSecAdr            : ST_MBUS_SecAdr;
eBaudrate           : E_MBUS_Baudrate := eMBUS_Baud2400;
bStart              : BOOL;
bSND_NKE            : BOOL := TRUE;
bReadInit           : BOOL := TRUE;
```

```
tMinSendTime      : TIME := t#2s;
usiUnit           : USINT;
bDisabled         : BOOL := FALSE;
```

usiAddress: Primary address [▶ 14] of the counter, that shall be readout with this module.

stSecAdr: Secondary address [▶ 15] of the counter, that shall be readout with this module.

eBaudrate: 300, 2400, 9600 baud [▶ 216].

bStart: Positive edge on this input, the meter is read out once.

bSND_NKE: TRUE to initialize the meter at each reading, and sets the meter on the first telegram (SND_NKE).

bReadInit: After restarting the PLC, the meter is read out once.

tMinSendTime: Standard t#2s. When this time exceeds, the counter is rereadout. At t#0s the counter is not readout and can be readout with bStart manually.

usiUnit: Unit of the energy values. 0=W/J(h) / 1=KW/KJ(h) / 2=MW/MJ(h) / 3=GW/GJ(h).

bDisabled: TRUE = disable the function block.

VAR_OUTPUT

```
bBusy                 : BOOL;
bReady                : BOOL;
bError                : BOOL;
eError                : E_MBUS_ERROR;
dwIdNumber            : DWORD;
byStatus              : BYTE;
byGEN                 : BYTE;
byCounter             : BYTE;
usiRecivedAdr         : USINT;
eMedium               : E_MBUS_Medium;
sMan                  : STRING(3);
stEnergyT1total       : ST_MBus_Info;
stEnergyT1partial     : ST_MBus_Info;
stEnergyT2total       : ST_MBus_Info;
stEnergyT2partial     : ST_MBus_Info;
stVoltageL1           : ST_MBus_Info;
stCurrentL1           : ST_MBus_Info;
stPowerL1             : ST_MBus_Info;
stReactivPowerL1      : ST_MBus_Info;
stVoltageL2           : ST_MBus_Info;
stCurrentL2           : ST_MBus_Info;
stPowerL2             : ST_MBus_Info;
stReactivPowerL2      : ST_MBus_Info;
stVoltageL3           : ST_MBus_Info;
stCurrentL3           : ST_MBus_Info;
stPowerL3             : ST_MBus_Info;
stReactivPowerL3      : ST_MBus_Info;
stCurrTransFactor     : ST_MBus_Info;
stPowerTotal          : ST_MBus_Info;
stReactivePowerTotal  : ST_MBus_Info;
stTariff              : ST_MBus_Info;
```

bBusy: The *bBusy* output is TRUE while the meter is being read.

bReady: The *bReady* output is TRUE for one cycle, once meter reading is completed.

bError: The *bError* output becomes TRUE as soon as an error occurs. The error is described via the variable *eError*.

eError: The *eError* output issues an error code when an error occurs (see E_MBUS_ERROR [▶ 216]).

dwIdNumber: Serial number of the meter (secondary address).

byStatus: Status of the meter. Please refer to device description for meanings.

byGEN: Meter software version.

byCounter: Transmission counter (number of transmitted RSP_UD).

usiRecivedAdr: Received primary address (0-250).

eMedium: Medium (see E_MBUS_Medium [▶ 219]).

sMan: Manufacturer code.

stEnergyT1total: Meter reading, total energy, tariff 1 (see ST_MBus_Info [▶ 222]).

stEnergyT1partial: Meter reading, energy partial, tariff 1 (see ST_MBus_Info [▶ 222]).

stEnergyT2total: Meter reading, total energy, tariff 1 (see ST_MBus_Info [▶ 222]).

stEnergyT2partial: Meter reading, energy partial, tariff 2 (see ST_MBus_Info [▶ 222]).

stVoltageL1: Voltage phase 1 (see ST_MBus_Info [▶ 222]).

stCurrentL1: Electric Current phase 1 (see ST_MBus_Info [▶ 222]).

stPowerL1: Power phase 1 (see ST_MBus_Info [▶ 222]).

stReactivPowerL1: Reactive power phase 1 (see ST_MBus_Info [▶ 222]).

stVoltageL2: Voltage phase 2 (see ST_MBus_Info [▶ 222]).

stCurrentL2: Electric Current phase 2 (see ST_MBus_Info [▶ 222]).

stPowerL2: Power phase 2 (see ST_MBus_Info [▶ 222]).

stReactivPowerL2: Reactive power phase 2 (see ST_MBus_Info [▶ 222]).

stVoltageL3: Voltage phase 3 (see ST_MBus_Info [▶ 222]).

stCurrentL3: Electric Current phase 3 (see ST_MBus_Info [▶ 222]).

stPowerL3: Power phase 3 (see ST_MBus_Info [▶ 222]).

stReactivPowerL3: Reactive power phase 3 (see ST_MBus_Info [▶ 222]).

stCurrTransFactor: Transformation factor of the current transformer (=0 for ALE3 device) (see ST_MBus_Info [▶ 222]).

stPowerTotal: Power total (see ST_MBus_Info [▶ 222]).

stReactivePowerTotal: Reactive Power total (see ST_MBus_Info [▶ 222]).

stTariff: Current tariff (=0 for AWD3 device) (see ST_MBus_Info [▶ 222]).

VAR_IN_OUT

```
stCom        : ST_MBUS_Communication;
```

stCom: About this structure, the block FB_MBUSKL6781() [▶ 29] is connected to the meter function blocks (see ST_MBUS_Communication [▶ 221]).

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055663499/.zip:

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055664907/.zip:

Controller configuration setting: "BCxx50 or BX serial"

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055666315/.zip:

BCxx00 must be set to "Large Model" in the PLC under "Project/Options/Controller Settings".

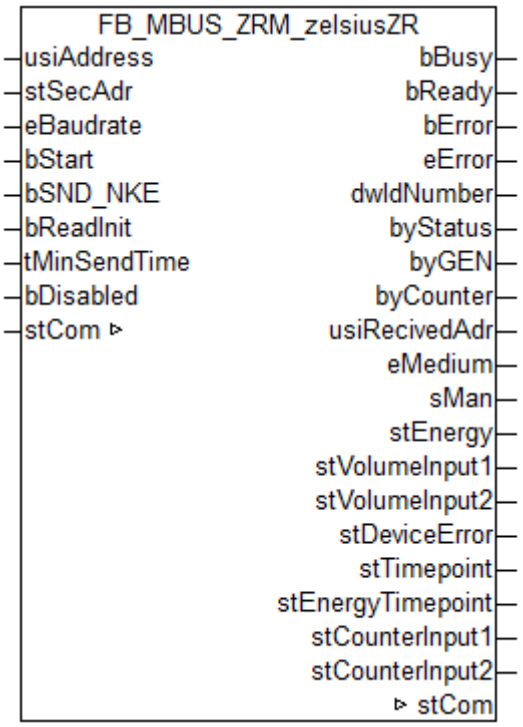
Controller configuration setting: "BC serial"

6.30 SANEXT

The function blocks only output a selection of the most common data. These data are described on the respective pages under "VAR_OUT". If more or all data are required, the function blocks FB_MBUS_General [▶ 31], FB_MBUS_General_Ext [▶ 35] or FB_MBUS_General_Param [▶ 41] from the General folder should be used. Not that these function blocks do not run on BC or BX systems. The function block FB_MBUS_General_Send [▶ 43] can be used to send data to the device (e.g. setting of the primary address).

Vendor	Type	Device	Function block
SANEXT	Heat meter	Sanext Combi	FB_MBUS_ZRM_zelsiusZR [▶ 189]

6.30.1 FB_MBUS_ZRM_zelsiusZR



This module is used to readout heat/cold meter of the manufacturer Zenner.

Also usable with:

- Sanext Combi of SANEXT (starting with V2.7.0)

It can only be used together with the module FB_MBUSKL6781() [▶ 29].

Functionality of the module [▶ 13]

VAR_INPUT

```
usiAddress      : USINT;
stSecAdr        : ST_MBUS_SecAdr;
eBaudrate       : E_MBUS_Baudrate := eMBUS_Baud2400;
bStart          : BOOL;
bSND_NKE        : BOOL := TRUE;
bReadInit       : BOOL := TRUE;
tMinSendTime    : TIME := t#2s;
bDisabled       : BOOL := FALSE;
```

usiAddress: Primary address [▶ 14] of the counter, that shall be readout with this module.

stSecAdr: Secondary address [▶ 15] of the counter, that shall be readout with this module.

eBaudrate: 300, 2400, 9600 baud [▶ 216].

bStart: Positive edge on this input, the meter is read out once.

bSND_NKE: TRUE to initialize the meter at each reading, and sets the meter on the first telegram (SND_NKE).

bReadInit: After restarting the PLC, the meter is read out once.

tMinSendTime: Standard t#2s. When this time exceeds, the counter is rereadout. At t#0s the counter is not readout and can be readout with bStart manually.

bDisabled: TRUE = disable the function block.

VAR_OUTPUT

```
bBusy               : BOOL;
bReady              : BOOL;
bError              : BOOL;
eError              : E_MBUS_ERROR;
dwIdNumber          : DWORD;
byStatus            : BYTE;
byGEN               : BYTE;
byCounter           : BYTE;
usiRecivedAdr       : USINT;
eMedium             : E_MBUS_Medium;
sMan                : STRING(3);
stEnergy            : ST_MBus_Info;
stVolumeInput1      : ST_MBus_Info;
stVolumeInput2      : ST_MBus_Info;
stDeviceError       : ST_MBus_Info;
stTimepoint         : ST_MBus_Info;
stEnergyTimepoint   : ST_MBus_Info;
stCounterInput1     : ST_MBus_Info;
stCounterInput2     : ST_MBus_Info;
```

bBusy: The *bBusy* output is TRUE while the meter is being read.

bReady: The *bReady* output is TRUE for one cycle, once meter reading is completed.

bError: The *bError* output becomes TRUE as soon as an error occurs. The error is described via the variable *eError*.

eError: The *eError* output issues an error code when an error occurs (see E_MBUS_ERROR [▶ 216]).

dwIdNumber: Serial number of the meter (secondary address).

byStatus: Status of the meter. Please refer to device description for meanings.

byGEN: Meter software version.

byCounter: Transmission counter (number of transmitted RSP_UD).

usiRecivedAdr: Received primary address (0-250).

eMedium: Medium (see E_MBUS_Medium [▶ 219]).

sMan: Manufacturer code.

stEnergy: Meter reading, energy consumption (see ST_MBus_Info [▶ 222]).

stVolumeInput1: Volume Input 1 (see ST_MBus_Info [▶ 222]).

stVolumeInput2: Volume Input 2 (see ST_MBus_Info [▶ 222]).

stDeviceError: Error state M-Bus output (see ST_MBus_Info [▶ 222]).

stTimepoint: Deadline (date and time of the next deadline) (see ST_MBus_Info [▶ 222]).

stEnergyTimepoint: Heat energy at the deadline (see ST_MBus_Info [▶ 222]).

stCounterInput1: Counter value input 1 at the deadline (see ST_MBus_Info [▶ 222]).

stCounterInput2: Counter value input 2 at the deadline (see ST_MBus_Info [▶ 222]).

VAR_IN_OUT

```
stCom           : ST_MBUS_Communication;
```

stCom: About this structure, the block FB_MBUSKL6781() [▶ 29] is connected to the meter function blocks (see ST_MBUS_Communication [▶ 221]).

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055667723/.zip:

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055669131/.zip:

Controller configuration setting: "BCxx50 or BX serial"

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055670539/.zip:

BCxx00 must be set to "Large Model" in the PLC under "Project/Options/Controller Settings".

Controller configuration setting: "BC serial"

Requirements

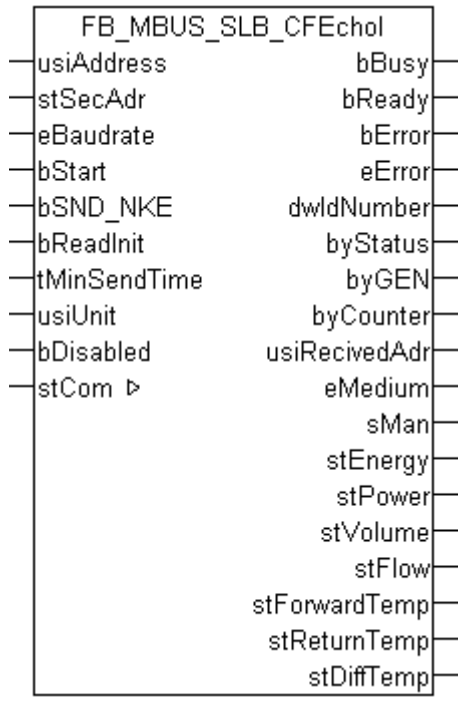
Development environment	Target system	Required libraries
TwinCAT 2.11 R3/x64 from build 2234	PC/CX, BX or BC	TcMBus library from V2.1.0

6.31 Schlumberger

The function blocks only output a selection of the most common data. These data are described on the respective pages under "VAR_OUT". If more or all data are required, the function blocks FB_MBUS_General [▶ 31], FB_MBUS_General_Ext [▶ 35] or FB_MBUS_General_Param [▶ 41] from the General folder should be used. Not that these function blocks do not run on BC or BX systems. The function block FB_MBUS_General_Send [▶ 43] can be used to send data to the device (e.g. setting of the primary address).

Vendor	Type	Device	Function block
Schlumberger	Heat meter	Integral-Mk MaXX	FB_MBUS_SLB_MK_MaXX [▶ 194]
	Heat meter	CF Echo I	FB_MBUS_SLB_CFEchoI [▶ 192]

6.31.1 FB_MBUS_SLB_CFEchoI



This block is used for reading heat meters from Schlumberger:

-CF Echo I

The block can only be used in conjunction with the block FB_MBUSKL6781() [▶ 29].

Functionality of the function block [▶ 13]

VAR_INPUT

```
usiAddress      : USINT;
stSecAdr        : ST_MBUS_SecAdr;
eBaudrate       : E_MBUS_Baudrate := eMBUS_Baud2400;
bStart          : BOOL;
bSND_NKE        : BOOL := TRUE;
bReadInit       : BOOL := TRUE;
tMinSendTime    : TIME := t#2s;
usiUnit         : USINT;
bDisabled       : BOOL := FALSE;
```

usiAddress: Primary address [▶ 14] of the counter, that shall be readout with this module.

stSecAdr: Secondary address [▶ 15] of the counter, that shall be readout with this module.

eBaudrate: 300, 2400 baud [▶ 216].

bStart: Positive edge on this input, the meter is read out once.

bSND_NKE: TRUE to initialize the meter at each reading, and sets the meter on the first telegram (SND_NKE).

bReadInit: After restarting the PLC, the meter is read out once.

tMinSendTime: Standard t#2s. When this time exceeds, the counter is rereadout. At t#0s the counter is not readout and can be readout with bStart manually.

usiUnit: Unit of the energy values. 0=W/J(h) / 1=KW/KJ(h) / 2=MW/MJ(h) / 3=GW/GJ(h).

bDisabled: TRUE = disable the function block.

VAR_OUTPUT

```
bBusy           : BOOL;
bReady          : BOOL;
bError          : BOOL;
eError          : E_MBUS_ERROR;
dwIdNumber      : DWORD;
byStatus        : BYTE;
byGEN           : BYTE;
byCounter       : BYTE;
usiRecivedAdr   : USINT;
eMedium         : E_MBUS_Medium;
sMan            : STRING(3);
stEnergy        : ST_MBus_Info;
stPower         : ST_MBus_Info;
stVolume        : ST_MBus_Info;
stFlow          : ST_MBus_Info;
stForwardTemp   : ST_MBus_Info;
stReturnTemp    : ST_MBus_Info;
stDiffTemp      : ST_MBus_Info;
```

bBusy: The *bBusy* output is TRUE while the meter is being read.

bReady: The *bReady* output is TRUE for one cycle, once meter reading is completed.

bError: The *bError* output becomes TRUE as soon as an error occurs. The error is described via the variable *eError*.

eError: The *eError* output issues an error code when an error occurs (see E_MBUS_ERROR [▶ 216]).

dwIdNumber: Serial number of the meter (secondary address).

byStatus: Status of the meter. Please refer to device description for meanings.

byGEN: Meter software version.

byCounter: Transmission counter (number of transmitted RSP_UD).

usiRecivedAdr: Received primary address (0-250).

eMedium: Medium (see E_MBUS_Medium [▶ 219]).

sMan: Manufacturer code.

stEnergy: Meter reading, energy consumption (see ST_MBus_Info [▶ 222]).

stPower: Current energy consumption (see ST_MBus_Info [▶ 222]).

stVolume: Meter reading, volume (see ST_MBus_Info [▶ 222]).

stFlow: Current flow (see ST_MBus_Info [▶ 222]).

stForwardTemp: Flow temperature (see ST_MBus_Info [▶ 222]).

stReturnTemp: Return temperature (see ST_MBus_Info [▶ 222]).

stDiffTemp: Temperature difference (see ST_MBus_Info [▶ 222]).

VAR_IN_OUT

```
stCom           : ST_MBUS_Communication;
```

stCom: About this structure, the block FB_MBUSKL6781() [▶ 29] is connected to the meter function blocks (see ST_MBUS_Communication [▶ 221]).

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055671947/.zip:

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055673355/.zip:

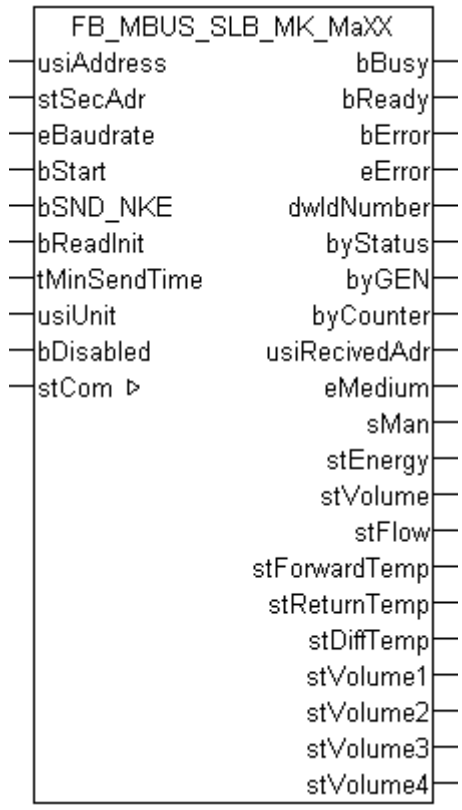
Controller configuration setting: "BCxx50 or BX serial"

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055674763/.zip:

BCxx00 must be set to "Large Model" in the PLC under "Project/Options/Controller Settings".

Controller configuration setting: "BC serial"

6.31.2 FB_MBUS_SLB_MK_MaXX



This function block is used to read heat meters from Schlumberger:

-Integral-MK Maxx / Up to 4 additional water meters can be connected to this device.

The function block can only be executed together with the function block FB_MBUSKL6781() [▶ 29].

Functionality of the function block [▶ 13]

VAR_INPUT

```
usiAddress      : USINT;
stSecAdr        : ST_MBUS_SecAdr;
eBaudrate       : E_MBUS_Baudrate := eMBUS_Baud2400;
bStart          : BOOL;
bSND_NKE        : BOOL := TRUE;
bReadInit       : BOOL := TRUE;
tMinSendTime    : TIME := t#2s;
usiUnit         : USINT;
bDisabled       : BOOL := FALSE;
```

usiAddress: Primary address [▶ 14] of the counter, that shall be readout with this module.

stSecAdr: Secondary address [▶ 15] of the counter, that shall be readout with this module.

eBaudrate: 300, 2400 baud [▶ 216].

bStart: Positive edge on this input, the meter is read out once.

bSND_NKE: TRUE to initialize the meter at each reading, and sets the meter on the first telegram (SND_NKE).

bReadInit: After restarting the PLC, the meter is read out once.

tMinSendTime: Standard t#2s. When this time exceeds, the counter is rereadout. At t#0s the counter is not readout and can be readout with bStart manually.

usiUnit: Unit of the energy values. 0=W/J(h) / 1=KW/KJ(h) / 2=MW/MJ(h) / 3=GW/GJ(h).

bDisabled: TRUE = disable the function block.

VAR_OUTPUT

```
bBusy           : BOOL;
bReady          : BOOL;
bError          : BOOL;
eError          : E_MBUS_ERROR;
dwIdNumber      : DWORD;
byStatus        : BYTE;
byGEN           : BYTE;
byCounter       : BYTE;
usiRecivedAdr   : USINT;
eMedium         : E_MBUS_Medium;
sMan            : STRING(3);
stEnergy        : ST_MBus_Info;
stVolume        : ST_MBus_Info;
stFlow          : ST_MBus_Info;
stForwardTemp   : ST_MBus_Info;
stReturnTemp    : ST_MBus_Info;
stDiffTemp      : ST_MBus_Info;
stVolume1       : ST_MBus_Info;
stVolume2       : ST_MBus_Info;
stVolume3       : ST_MBus_Info;
stVolume4       : ST_MBus_Info;
```

bBusy: The *bBusy* output is TRUE while the meter is being read.

bReady: The *bReady* output is TRUE for one cycle, once meter reading is completed.

bError: The *bError* output becomes TRUE as soon as an error occurs. The error is described via the variable *eError*.

eError: The *eError* output issues an error code when an error occurs (see E_MBUS_ERROR [▶ 216]).

dwIdNumber: Serial number of the meter (secondary address).

byStatus: Status of the meter. Please refer to device description for meanings.

byGEN: Meter software version.

byCounter: Transmission counter (number of transmitted RSP_UD).

usiRecivedAdr: Received primary address (0-250).

eMedium: Medium (see E_MBUS_Medium [▶ 219]).

sMan: Manufacturer code.

stEnergy: Meter reading, energy consumption (see ST_MBus_Info [▶ 222]).

stVolume: Meter reading, volume (see ST_MBus_Info [▶ 222]).

stFlow: Current flow (see ST_MBus_Info [▶ 222]).

stForwardTemp: Flow temperature (see ST_MBus_Info [▶ 222]).

stReturnTemp: Return temperature (see ST_MBus_Info [▶ 222]).

stDiffTemp: Temperature difference (see ST_MBus_Info [▶ 222]).

stVolume1: Meter reading of additional water meter 1 (see ST_MBus_Info [▶ 222]).

stVolume2: Meter reading of additional water meter 2 (see ST_MBus_Info [▶ 222]).

stVolume3: Meter reading of additional water meter 3 (see ST_MBus_Info [▶ 222]).

stVolume4: Meter reading of additional water meter 4 (see ST_MBus_Info [▶ 222]).

VAR_IN_OUT

```
stCom          : ST_MBUS_Communication;
```

stCom: About this structure, the block FB_MBUSKL6781() [▶ 29] is connected to the meter function blocks (see ST_MBUS_Communication [▶ 221]).

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055671947/.zip: 

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055673355/.zip: 

Controller configuration setting: "BCxx50 or BX serial"

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055674763/.zip: 

BCxx00 must be set to "Large Model" in the PLC under "Project/Options/Controller Settings".

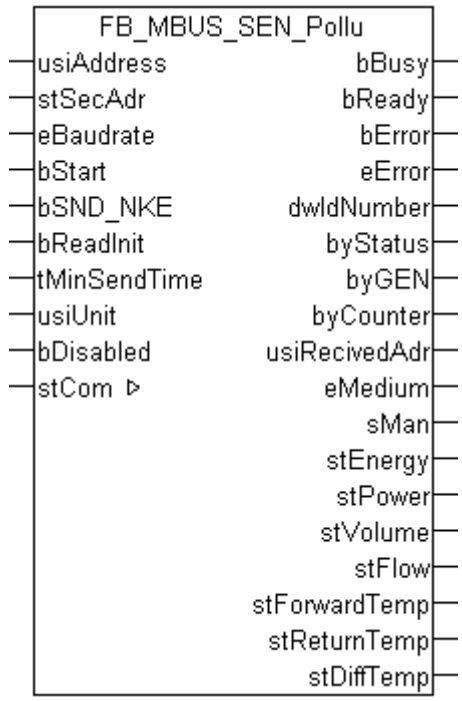
Controller configuration setting: "BC serial"

6.32 Sensus

The function blocks only output a selection of the most common data. These data are described on the respective pages under "VAR_OUT". If more or all data are required, the function blocks FB_MBUS_General [▶ 31], FB_MBUS_General_Ext [▶ 35] or FB_MBUS_General_Param [▶ 41] from the General folder should be used. Not that these function blocks do not run on BC or BX systems. The function block FB_MBUS_General_Send [▶ 43] can be used to send data to the device (e.g. setting of the primary address).

Vendor	Type	Device	Function block
Sensus	Heat/cold meter	PolluStat E	FB_MBUS_SEN_Pollu [▶ 197]
	Heat/cold meter	PolluTherm	
	Heat/cold meter	PolluCom E	
	Water meter		FB_MBUS_SEN_Water [▶ 199]

6.32.1 FB_MBUS_SEN_Pollu



This block is used for reading heat/cold meters from Sensus:

-PolluStat E

-PolluCom E

-PolluTherm

The block can only be used in conjunction with the block FB_MBUSKL6781() [▶ 29].

Functionality of the function block [▶ 13]

VAR_INPUT

```
usiAddress      : USINT;
stSecAdr        : ST_MBUS_SecAdr;
eBaudrate       : E_MBUS_Baudrate := eMBUS_Baud2400;
bStart          : BOOL;
bSND_NKE        : BOOL := TRUE;
bReadInit       : BOOL := TRUE;
tMinSendTime    : TIME := t#2s;
usiUnit         : USINT;
bDisabled       : BOOL := FALSE;
```

usiAddress: Primary address [▶ 14] of the counter, that shall be readout with this module.

stSecAdr: Secondary address [▶ 15] of the counter, that shall be readout with this module.

eBaudrate: 300, 2400, 9600 baud [▶ 216].

bStart: Positive edge on this input, the meter is read out once.

bSND_NKE: TRUE to initialize the meter at each reading, and sets the meter on the first telegram (SND_NKE).

bReadInit: After restarting the PLC, the meter is read out once.

tMinSendTime: Standard t#2s. When this time exceeds, the counter is rereadout. At t#0s the counter is not readout and can be readout with bStart manually.

usiUnit: Unit of the energy values. 0=W/J(h) / 1=KW/KJ(h) / 2=MW/MJ(h) / 3=GW/GJ(h).

bDisabled: TRUE = disable the function block.

VAR_OUTPUT

```
bBusy           : BOOL;
bReady          : BOOL;
bError          : BOOL;
eError          : E_MBUS_ERROR;
dwIdNumber      : DWORD;
byStatus        : BYTE;
byGEN           : BYTE;
byCounter       : BYTE;
usiRecivedAdr   : USINT;
eMedium         : E_MBUS_Medium;
sMan            : STRING(3);
stEnergy        : ST_MBus_Info;
stPower         : ST_MBus_Info;
stVolume        : ST_MBus_Info;
stFlow          : ST_MBus_Info;
stForwardTemp   : ST_MBus_Info;
stReturnTemp    : ST_MBus_Info;
stDiffTemp      : ST_MBus_Info;
```

bBusy: The *bBusy* output is TRUE while the meter is being read.

bReady: The *bReady* output is TRUE for one cycle, once meter reading is completed.

bError: The *bError* output becomes TRUE as soon as an error occurs. The error is described via the variable *eError*.

eError: The *eError* output issues an error code when an error occurs (see E_MBUS_ERROR [▶ 216]).

dwIdNumber: Serial number of the meter (secondary address).

byStatus: Status of the meter. Please refer to device description for meanings.

byGEN: Meter software version.

byCounter: Transmission counter (number of transmitted RSP_UD).

usiRecivedAdr: Received primary address (0-250).

eMedium: Medium (see E_MBUS_Medium [▶ 219]).

sMan: Manufacturer code.

stEnergy: Meter reading, energy consumption (see ST_MBus_Info [▶ 222]).

stPower: Current energy consumption (see ST_MBus_Info [▶ 222]).

stVolume: Meter reading, volume (see ST_MBus_Info [▶ 222]).

stFlow: Current flow (see ST_MBus_Info [▶ 222]).

stForwardTemp: Flow temperature (see ST_MBus_Info [▶ 222]).

stReturnTemp: Return temperature (see ST_MBus_Info [▶ 222]).

stDiffTemp: Temperature difference (see ST_MBus_Info [▶ 222]).

VAR_IN_OUT

```
stCom           : ST_MBUS_Communication;
```

stCom: About this structure, the block FB_MBUSKL6781() [▶ 29] is connected to the meter function blocks (see ST_MBUS_Communication [▶ 221]).

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055676171/.zip:

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055677579/.zip:

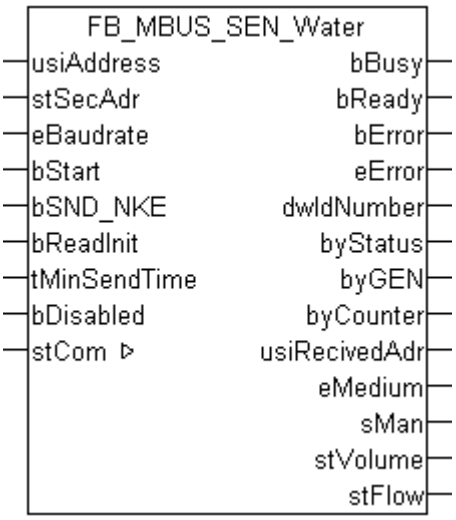
Controller configuration setting: "BCxx50 or BX serial"

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055678987/.zip:

BCxx00 must be set to "Large Model" in the PLC under "Project/Options/Controller Settings".

Controller configuration setting: "BC serial"

6.32.2 FB_MBUS_SEN_Water



This function block is used to read water meters from Sensus.

The function block can only be executed together with the function block FB_MBUSKL6781() [▶ 29].

Functionality of the function block [▶ 13]

VAR_INPUT

```
usiAddress      : USINT;
stSecAdr        : ST_MBUS_SecAdr;
eBaudrate       : E_MBUS_Baudrate := eMBUS_Baud2400;
bStart          : BOOL;
bSND_NKE        : BOOL := TRUE;
bReadInit       : BOOL := TRUE;
tMinSendTime    : TIME := t#2s;
bDisabled       : BOOL := FALSE;
```

usiAddress: Primary address [▶ 14] of the counter, that shall be readout with this module.

stSecAdr: Secondary address [▶ 15] of the counter, that shall be readout with this module.

eBaudrate: 300, 2400, 9600 baud [▶ 216].

bStart: Positive edge on this input, the meter is read out once.

bSND_NKE: TRUE to initialize the meter at each reading, and sets the meter on the first telegram (SND_NKE).

bReadInit: After restarting the PLC, the meter is read out once.

tMinSendTime: Standard t#2s. When this time exceeds, the counter is rereadout. At t#0s the counter is not readout and can be readout with bStart manually.

bDisabled: TRUE = disable the function block.

VAR_OUTPUT

```
bBusy           : BOOL;
bReady          : BOOL;
bError          : BOOL;
eError          : E_MBUS_ERROR;
dwIdNumber      : DWORD;
byStatus        : BYTE;
byGEN           : BYTE;
byCounter       : BYTE;
usiRecivedAdr   : USINT;
eMedium         : E_MBUS_Medium;
sMan            : STRING(3);
stVolume        : ST_MBus_Info;
stFlow          : ST_MBus_Info;
```

bBusy: The *bBusy* output is TRUE while the meter is being read.

bReady: The *bReady* output is TRUE for one cycle, once meter reading is completed.

bError: The *bError* output becomes TRUE as soon as an error occurs. The error is described via the variable *eError*.

eError: The *eError* output issues an error code when an error occurs (see E_MBUS_ERROR [▶ 216]).

dwIdNumber: Serial number of the meter (secondary address).

byStatus: Status of the meter. Please refer to device description for meanings.

byGEN: Meter software version.

byCounter: Transmission counter (number of transmitted RSP_UD).

usiRecivedAdr: Received primary address (0-250).

eMedium: Medium (see E_MBUS_Medium [▶ 219]).

sMan: Manufacturer code.

stVolume: Meter reading, volume (see ST_MBus_Info [▶ 222]).

stFlow: Current flow (see ST_MBus_Info [▶ 222]).

VAR_IN_OUT

```
stCom           : ST_MBUS_Communication;
```

stCom: About this structure, the block FB_MBUSKL6781() [▶ 29] is connected to the meter function blocks (see ST_MBUS_Communication [▶ 221]).

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055676171/.zip:

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055677579/.zip:

Controller configuration setting: "BCxx50 or BX serial"

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055678987/.zip:

BCxx00 must be set to "Large Model" in the PLC under "Project/Options/Controller Settings".

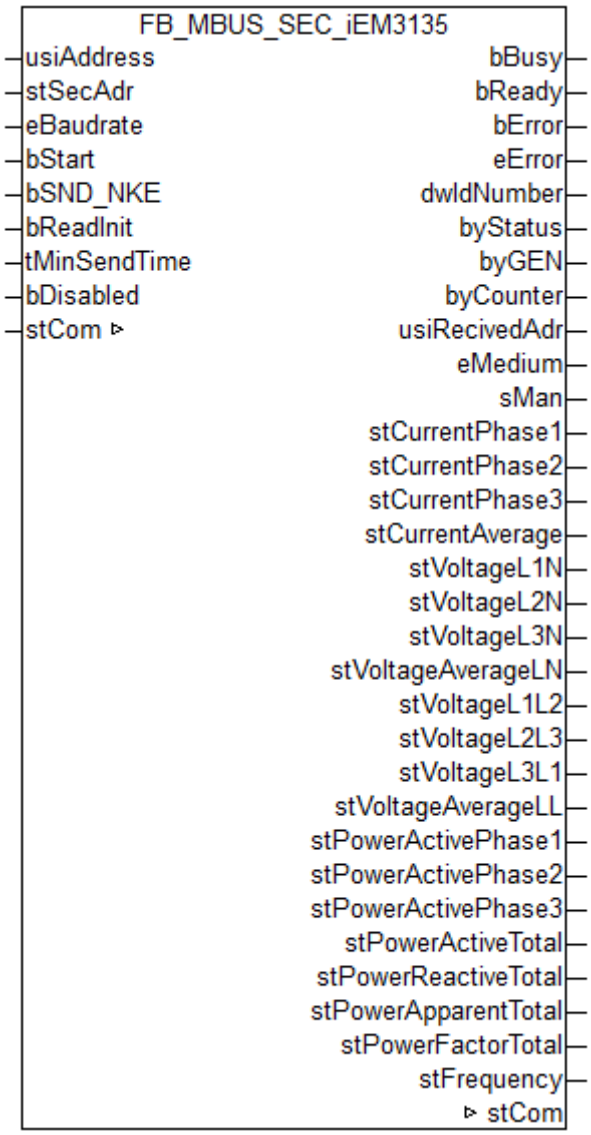
Controller configuration setting: "BC serial"

6.33 Schneider Electric

The function blocks only output a selection of the most common data. These data are described on the respective pages under "VAR_OUT". If more or all data are required, the function blocks FB_MBUS_General [▶ 31], FB_MBUS_General_Ext [▶ 35] or FB_MBUS_General_Param [▶ 41] from the General folder should be used. Not that these function blocks do not run on BC or BX systems. The function block FB_MBUS_General_Send [▶ 43] can be used to send data to the device (e.g. setting of the primary address).

Vendor	Type	Device	Function block
Schneider Electric	Electricity meter	iEM3135	FB_MBUS_SEC_iEM3135 [▶ 201]

6.33.1 FB_MBUS_SEC_iEM3135



This module is used to readout electricity meters of the manufacturer Schneider Electric.

It can only be used together with the module FB_MBUSKL6781() [▶ 29].

Functionality of the module [▶ 13]

VAR_INPUT

```
usiAddress      : USINT;
stSecAdr        : ST_MBUS_SecAdr;
eBaudrate       : E_MBUS_Baudrate := eMBUS_Baud2400;
bStart          : BOOL;
bSND_NKE        : BOOL := TRUE;
bReadInit       : BOOL := TRUE;
tMinSendTime    : TIME := t#2s;
bDisabled       : BOOL := FALSE;
```

usiAddress: Primary address [▶ 14] of the counter, that shall be readout with this module.

stSecAdr: Secondary address [▶ 15] of the counter, that shall be readout with this module.

eBaudrate: 300..9600 baud [▶ 216].

bStart: Positive edge on this input, the meter is read out once.

bSND_NKE: TRUE to initialize the meter at each reading, and sets the meter on the first telegram (SND_NKE).

bReadInit: After restarting the PLC, the meter is read out once.

tMinSendTime: Standard t#2s. When this time exceeds, the counter is rereadout. At t#0s the counter is not readout and can be readout with bStart manually.

bDisabled: TRUE = disable the function block.

VAR_OUTPUT

```
bBusy               : BOOL;
bReady              : BOOL;
bError              : BOOL;
eError              : E_MBUS_ERROR;
dwIdNumber          : DWORD;
byStatus            : BYTE;
byGEN               : BYTE;
byCounter           : BYTE;
usiRecivedAdr       : USINT;
eMedium             : E_MBUS_Medium;
sMan                : STRING(3);
stCurrentPhase1     : ST_MBus_Info;
stCurrentPhase2     : ST_MBus_Info;
stCurrentPhase3     : ST_MBus_Info;
stCurrentAverage    : ST_MBus_Info;
stVoltageL1N        : ST_MBus_Info;
stVoltageL2N        : ST_MBus_Info;
stVoltageL3N        : ST_MBus_Info;
stVoltageAverageLN  : ST_MBus_Info;
stVoltageL1L2       : ST_MBus_Info;
stVoltageL2L3       : ST_MBus_Info;
stVoltageL3L1       : ST_MBus_Info;
stVoltageAverageLL  : ST_MBus_Info;
stPowerActivePhase1 : ST_MBus_Info;
stPowerActivePhase2 : ST_MBus_Info;
stPowerActivePhase3 : ST_MBus_Info;
stPowerActiveTotal  : ST_MBus_Info;
stPowerReactiveTotal: ST_MBus_Info;
stPowerApparentTotal: ST_MBus_Info;
stPowerFactorTotal  : ST_MBus_Info;
stFrequency         : ST_MBus_Info;
```

bBusy: The *bBusy* output is TRUE while the meter is being read.

bReady: The *bReady* output is TRUE for one cycle, once meter reading is completed.

bError: The *bError* output becomes TRUE as soon as an error occurs. The error is described via the variable *eError*.

eError: The *eError* output issues an error code when an error occurs (see E_MBUS_ERROR [▶ 216]).

dwIdNumber: Serial number of the counter (secondary address).

byStatus: Status of the counter. Please refer to device description for meanings.

byGEN: Counter software version.

byCounter: Transmission counter (number of transmitted RSP_UD).

usiRecivedAdr: Received primary address (0-250).

eMedium: Medium (see E_MBUS_Medium [▶ 219]).

sMan: Manufacturer code.

stCurrentPhase1: Phase 1 current (see ST_MBus_Info [▶ 222]).

stCurrentPhase2: Phase 2 current (see ST_MBus_Info [▶ 222]).

stCurrentPhase3: Phase 3 current (see ST_MBus_Info [▶ 222]).

stCurrentAverage: Average current (see ST_MBus_Info [▶ 222]).

stVoltageL1N: Voltage L1-N (see ST_MBus_Info [▶ 222]).

stVoltageL2N: Voltage L2-N (see ST_MBus_Info [▶ 222]).

stVoltageL3N: Voltage L3-N (see ST_MBus_Info [▶ 222]).

stVoltageAverageLN: Average voltage line-to-neutral (see ST_MBus_Info [▶ 222]).

stVoltageL1L2: Voltage L1-L2 (see ST_MBus_Info [▶ 222]).

stVoltageL2L3: Voltage L2-L3 (see ST_MBus_Info [▶ 222]).

stVoltageL3L1: Voltage L3-L1 (see ST_MBus_Info [▶ 222]).

stVoltageAverageLL: Average voltage line-to-line (see ST_MBus_Info [▶ 222]).

stPowerActivePhase1: Active power phase 1 (see ST_MBus_Info [▶ 222]).

stPowerActivePhase2: Active power phase 2 (see ST_MBus_Info [▶ 222]).

stPowerActivePhase3: Active power phase 3 (see ST_MBus_Info [▶ 222]).

stPowerActiveTotal: Total active power (see ST_MBus_Info [▶ 222]).

stPowerReactiveTotal: Total reactive power (see ST_MBus_Info [▶ 222]).

stPowerApparentTotal: Total apparent power (see ST_MBus_Info [▶ 222]).

stPowerFactorTotal: Total power factor (see ST_MBus_Info [▶ 222]).

stFrequency: Frequency (see ST_MBus_Info [▶ 222]).

VAR_IN_OUT

```
stCom           : ST_MBUS_Communication;
```

stCom: About this structure, the block FB_MBUSKL6781() [▶ 29] is connected to the meter function blocks (see ST_MBUS_Communication [▶ 221]).

Requirements

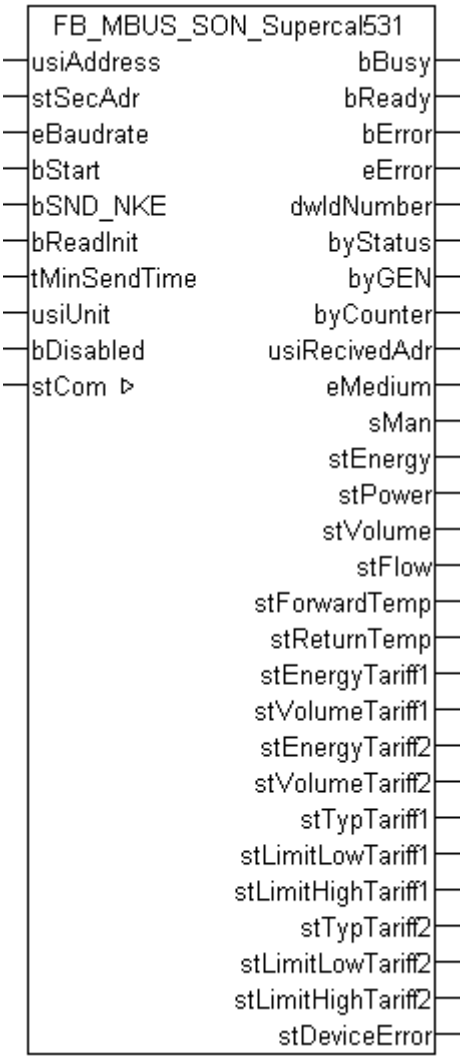
Development environment	Target system	Required libraries
TwinCAT 2.11 R3/x64 from build 2254	PC/CX, BX or BC	TcMBus library from V2.6.0

6.34 Sontex

The function blocks only output a selection of the most common data. These data are described on the respective pages under "VAR_OUT". If more or all data are required, the function blocks FB_MBUS_General [▶ 31], FB_MBUS_General_Ext [▶ 35] or FB_MBUS_General_Param [▶ 41] from the General folder should be used. Not that these function blocks do not run on BC or BX systems. The function block FB_MBUS_General_Send [▶ 43] can be used to send data to the device (e.g. setting of the primary address).

Vendor	Type	Device	Function block
Sontex	Heat/cold meter	Supercal 531 (default values)	FB_MBUS_SON_Supercal531 [▶ 204]

6.34.1 FB_MBUS_SON_Supercal531



This block is used for reading heat/cold meters from Sontex:

-Supercal 531

The block can only be used in conjunction with the block FB_MBUSKL6781() [▶ 29].

Functionality of the function block [▶ 13]

VAR_INPUT

```
usiAddress      : USINT;
stSecAdr        : ST_MBUS_SecAdr;
eBaudrate       : E_MBUS_Baudrate := eMBUS_Baud2400;
bStart          : BOOL;
bSND_NKE        : BOOL := TRUE;
bReadInit       : BOOL := TRUE;
tMinSendTime    : TIME := t#2s;
usiUnit         : USINT;
bDisabled       : BOOL := FALSE;
```

usiAddress: Primary address [▶ 14] of the counter, that shall be readout with this module.

stSecAdr: Secondary address [▶ 15] of the counter, that shall be readout with this module.

eBaudrate: 300, 2400, 9600 baud [▶ 216].

bStart: Positive edge on this input, the meter is read out once.

bSND_NKE: TRUE to initialize the meter at each reading, and sets the meter on the first telegram (SND_NKE).

bReadInit: After restarting the PLC, the meter is read out once.

tMinSendTime: Standard t#2s. When this time exceeds, the counter is rereadout. At t#0s the counter is not readout and can be readout with bStart manually.

usiUnit: Unit of the energy values. 0=W/J(h) / 1=KW/KJ(h) / 2=MW/MJ(h) / 3=GW/GJ(h).

bDisabled: TRUE = disable the function block.

VAR_OUTPUT

```
bBusy               : BOOL;
bReady              : BOOL;
bError              : BOOL;
eError              : E_MBUS_ERROR;
dwIdNumber          : DWORD;
byStatus            : BYTE;
byGEN               : BYTE;
byCounter           : BYTE;
usiRecivedAdr       : USINT;
eMedium             : E_MBUS_Medium;
sMan                : STRING(3);
stEnergy            : ST_MBus_Info;
stPower             : ST_MBus_Info;
stVolume            : ST_MBus_Info;
stFlow              : ST_MBus_Info;
stForwardTemp       : ST_MBus_Info;
stReturnTemp        : ST_MBus_Info;
stEnergyTariff1     : ST_MBus_Info;
stVolumeTariff1     : ST_MBus_Info;
stEnergyTariff2     : ST_MBus_Info;
stVolumeTariff2     : ST_MBus_Info;
stTypTariff1        : ST_MBus_Info;
stLimitLowTariff1   : ST_MBus_Info;
stLimitHighTariff1  : ST_MBus_Info;
stTypTariff2        : ST_MBus_Info;
stLimitLowTariff2   : ST_MBus_Info;
stLimitHighTariff2  : ST_MBus_Info;
stDeviceError       : ST_MBus_Info;
```

bBusy: The *bBusy* output is TRUE while the meter is being read.

bReady: The *bReady* output is TRUE for one cycle, once meter reading is completed.

bError: The *bError* output becomes TRUE as soon as an error occurs. The error is described via the variable *eError*.

eError: The *eError* output issues an error code when an error occurs (see E_MBUS_ERROR [▶ 216]).

dwIdNumber: Serial number of the meter (secondary address).

byStatus: Status of the meter. Please refer to device description for meanings.

byGEN: Meter software version.

byCounter: Transmission counter (number of transmitted RSP_UD).

usiRecivedAdr: Received primary address (0-250).

eMedium: Medium (see E_MBUS_Medium [▶ 219]).

sMan: Manufacturer code.

stEnergy: Meter reading, energy consumption (see ST_MBus_Info [▶ 222]).

stPower: Current energy consumption (see ST_MBus_Info [▶ 222]).

stVolume: Meter reading, volume (see ST_MBus_Info [▶ 222]).

stFlow: Current flow (see ST_MBus_Info [▶ 222]).

stForwardTemp: Flow temperature (see ST_MBus_Info [▶ 222]).

stReturnTemp: Return temperature (see ST_MBus_Info [▶ 222]).

stEnergyTariff1: Meter reading, energy consumption tariff 1 (see ST_MBus_Info [▶ 222]).

stVolumeTariff1: Meter reading, water consumption tariff 1 (see ST_MBus_Info [▶ 222]).

stEnergyTariff2: Meter reading, energy consumption tariff 2 (see ST_MBus_Info [▶ 222]).

stVolumeTariff2: Meter reading, water consumption tariff 2 (see ST_MBus_Info [▶ 222]).

stTypTariff1: Type tariff 1 (see ST_MBus_Info [▶ 222]).

stLimitLowTariff1: Lower limit value tariff 1 (see ST_MBus_Info [▶ 222]).

stLimitHighTariff1: Upper limit value tariff 1 (see ST_MBus_Info [▶ 222]).

stTypTariff2: Type tariff 2 (see ST_MBus_Info [▶ 222]).

stLimitLowTariff2: Lower limit value tariff 2 (see ST_MBus_Info [▶ 222]).

stLimitHighTariff2: Upper limit value tariff 2 (see ST_MBus_Info [▶ 222]).

stDeviceError: Error message from the device (see ST_MBus_Info [▶ 222]).

VAR_IN_OUT

```
stCom           : ST_MBUS_Communication;
```

stCom: About this structure, the block FB_MBUSKL6781() [▶ 29] is connected to the meter function blocks (see ST_MBUS_Communication [▶ 221]).

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055680395/.zip:

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055681803/.zip:

Controller configuration setting: "BCxx50 or BX serial"

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055683211/.zip:

BCxx00 must be set to "Large Model" in the PLC under "Project/Options/Controller Settings".

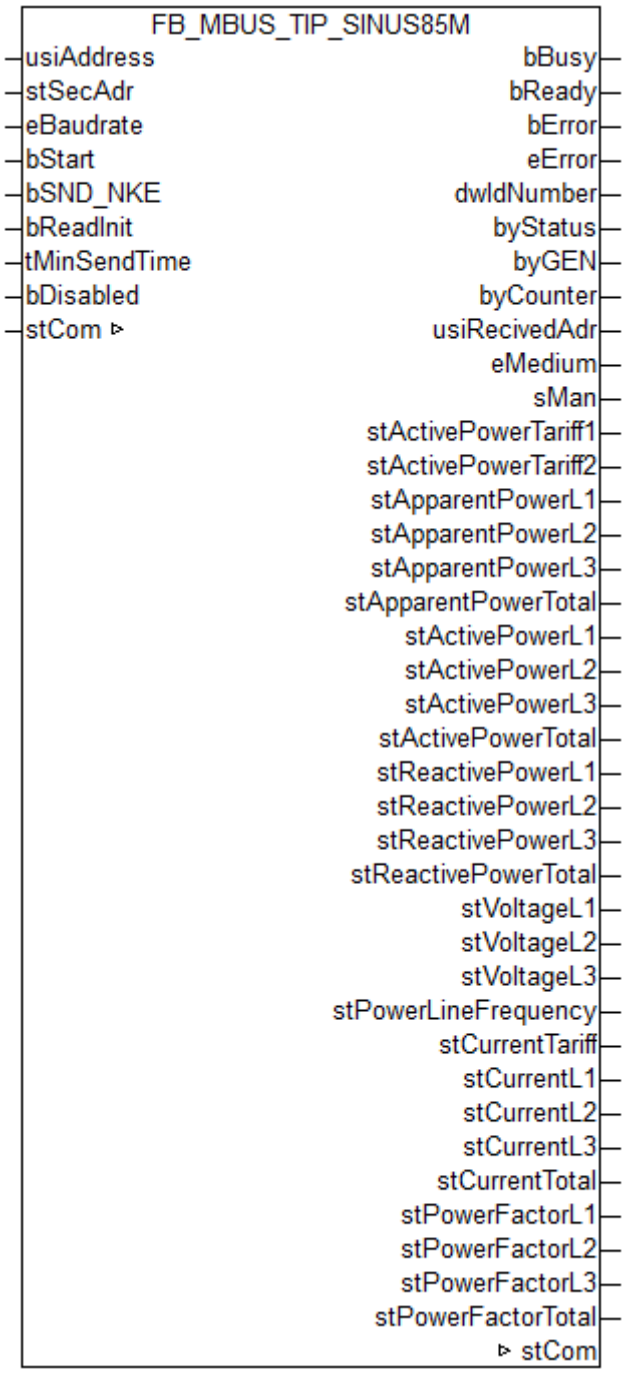
Controller configuration setting: "BC serial"

6.35 TIP

The function blocks only output a selection of the most common data. These data are described on the respective pages under "VAR_OUT". If more or all data are required, the function blocks FB_MBUS_General [▶ 31], FB_MBUS_General_Ext [▶ 35] or FB_MBUS_General_Param [▶ 41] from the General folder should be used. Not that these function blocks do not run on BC or BX systems. The function block FB_MBUS_General_Send [▶ 43] can be used to send data to the device (e.g. setting of the primary address).

Vendor	Type	Device	Function block
TIP	Electricity meter	SINUS 85 M	FB_MBUS_TIP_SINUS85M [▶ 207]

6.35.1 FB_MBUS_TIP_SINUS85M



This module is used to readout electricity meters of the manufacturer Thüringer Industrie Produkte GmbH.

Compatible to EcoCount „S“ of manufacturer NZR.

It can only be used together with the module FB_MBUSKL6781() [▶ 29].

Functionality of the module [▶ 13]

VAR_INPUT

```
usiAddress      : USINT;
stSecAdr        : ST_MBUS_SecAdr;
eBaudrate       : E_MBUS_Baudrate := eMBUS_Baud2400;
bStart          : BOOL;
bSND_NKE        : BOOL := TRUE;
bReadInit       : BOOL := TRUE;
tMinSendTime    : TIME := t#2s;
bDisabled       : BOOL := FALSE;
```

usiAddress: Primary address [▶ 14] of the counter, that shall be readout with this module.

stSecAdr: Secondary address [▶ 15] of the counter, that shall be readout with this module.

eBaudrate: 300, 2400, 9600 baud [▶ 216].

bStart: Positive edge on this input, the meter is read out once.

bSND_NKE: TRUE to initialize the meter at each reading, and sets the meter on the first telegram (SND_NKE).

bReadInit: After restarting the PLC, the meter is read out once.

tMinSendTime: Standard t#2s. When this time exceeds, the counter is rereadout. At t#0s the counter is not readout and can be readout with bStart manually.

bDisabled: TRUE = disable the function block.

VAR_OUTPUT

```
bBusy               : BOOL;
bReady              : BOOL;
bError              : BOOL;
eError              : E_MBUS_ERROR;
dwIdNumber          : DWORD;
byStatus            : BYTE;
byGEN               : BYTE;
byCounter           : BYTE;
usiRecivedAdr       : USINT;
eMedium             : E_MBUS_Medium;
sMan                : STRING(3);
stActivePowerTariff1: ST_MBus_Info;
stActivePowerTariff2: ST_MBus_Info;
stApparentPowerL1   : ST_MBus_Info;
stApparentPowerL2   : ST_MBus_Info;
stApparentPowerL3   : ST_MBus_Info;
stApparentPowerTotal: ST_MBus_Info;
stActivePowerL1     : ST_MBus_Info;
stActivePowerL2     : ST_MBus_Info;
stActivePowerL3     : ST_MBus_Info;
stActivePowerTotal  : ST_MBus_Info;
stReactivePowerL1   : ST_MBus_Info;
stReactivePowerL2   : ST_MBus_Info;
stReactivePowerL3   : ST_MBus_Info;
stReactivePowerTotal: ST_MBus_Info;
stVoltageL1         : ST_MBus_Info;
stVoltageL2         : ST_MBus_Info;
stVoltageL3         : ST_MBus_Info;
stPowerLineFrequency: ST_MBus_Info;
stCurrentTariff     : ST_MBus_Info;
stCurrentL1         : ST_MBus_Info;
stCurrentL2         : ST_MBus_Info;
stCurrentL3         : ST_MBus_Info;
stCurrentTotal      : ST_MBus_Info;
stPowerFactorL1     : ST_MBus_Info;
```

```
stPowerFactorL2     : ST_MBus_Info;
stPowerFactorL3     : ST_MBus_Info;
stPowerFactorTotal  : ST_MBus_Info;
```

bBusy: The *bBusy* output is TRUE while the meter is being read.

bReady: The *bReady* output is TRUE for one cycle, once meter reading is completed.

bError: The *bError* output becomes TRUE as soon as an error occurs. The error is described via the variable *eError*.

eError: The *eError* output issues an error code when an error occurs (see E_MBUS_ERROR [▶ 216]).

dwIdNumber: Serial number of the counter (secondary address).

byStatus: Status of the counter.

byGEN: Counter software version.

byCounter: Transmission counter (number of transmitted RSP_UD).

usiRecivedAdr: Received primary address (0-250).

eMedium: Medium (see E_MBUS_Medium [▶ 219]).

sMan: Manufacturer code.

stActivePowerTariff1: Active power import tariff 1 (see ST_MBus_Info [▶ 222]).

stActivePowerTariff2: Active power import tariff 2 (see ST_MBus_Info [▶ 222]).

stApparentPowerL1: Actual apparent power L1 (see ST_MBus_Info [▶ 222]).

stApparentPowerL2: Actual apparent power L2 (see ST_MBus_Info [▶ 222]).

stApparentPowerL3: Actual apparent power L3 (see ST_MBus_Info [▶ 222]).

stApparentPowerTotal: Actual apparent power total (see ST_MBus_Info [▶ 222]).

stActivePowerL1: Actual active power phase L1 (see ST_MBus_Info [▶ 222]).

stActivePowerL2: Actual active power phase L2 (see ST_MBus_Info [▶ 222]).

stActivePowerL3: Actual active power phase L3 (see ST_MBus_Info [▶ 222]).

stActivePowerTotal: Actual active power total (see ST_MBus_Info [▶ 222]).

stReactivePowerL1: Actual reactive power phase L1 (see ST_MBus_Info [▶ 222]).

stReactivePowerL2: Actual reactive power phase L2 (see ST_MBus_Info [▶ 222]).

stReactivePowerL3: Actual reactive power phase L3 (see ST_MBus_Info [▶ 222]).

stReactivePowerTotal: Actual reactive power total (see ST_MBus_Info [▶ 222]).

stVoltageL1: Actual voltage phase L1 (see ST_MBus_Info [▶ 222]).

stVoltageL2: Actual voltage phase L2 (see ST_MBus_Info [▶ 222]).

stVoltageL3: Actual voltage phase L3 (see ST_MBus_Info [▶ 222]).

stPowerLineFrequency: Actual power line frequency (see ST_MBus_Info [▶ 222]).

stCurrentTariff: Actual tariff (see ST_MBus_Info [▶ 222]).

stCurrentL1: Actual current phase L1 (see ST_MBus_Info [▶ 222]).

stCurrentL2: Actual current phase L2 (see ST_MBus_Info [▶ 222]).

stCurrentL3: Actual current phase L3 (see ST_MBus_Info [▶ 222]).

stCurrentTotal: Actual current total (see ST_MBus_Info [▶ 222]).

stPowerFactorL1: Actual power factor phase L1 (cos Phi) (see ST_MBus_Info [▶ 222]).

stPowerFactorL2: Actual power factor phase L2 (cos Phi) (see ST_MBus_Info [▶ 222]).

stPowerFactorL3: Actual power factor phase L3 (cos Phi) (see ST_MBus_Info [▶ 222]).

stPowerFactorTotal: Actual power factor total (cos Phi) (see ST_MBus_Info [▶ 222]).

VAR_IN_OUT

```
stCom           : ST_MBUS_Communication;
```

stCom: About this structure, the block FB_MBUSKL6781() [▶ 29] is connected to the meter function blocks (see ST_MBUS_Communication [▶ 221]).

Requirements

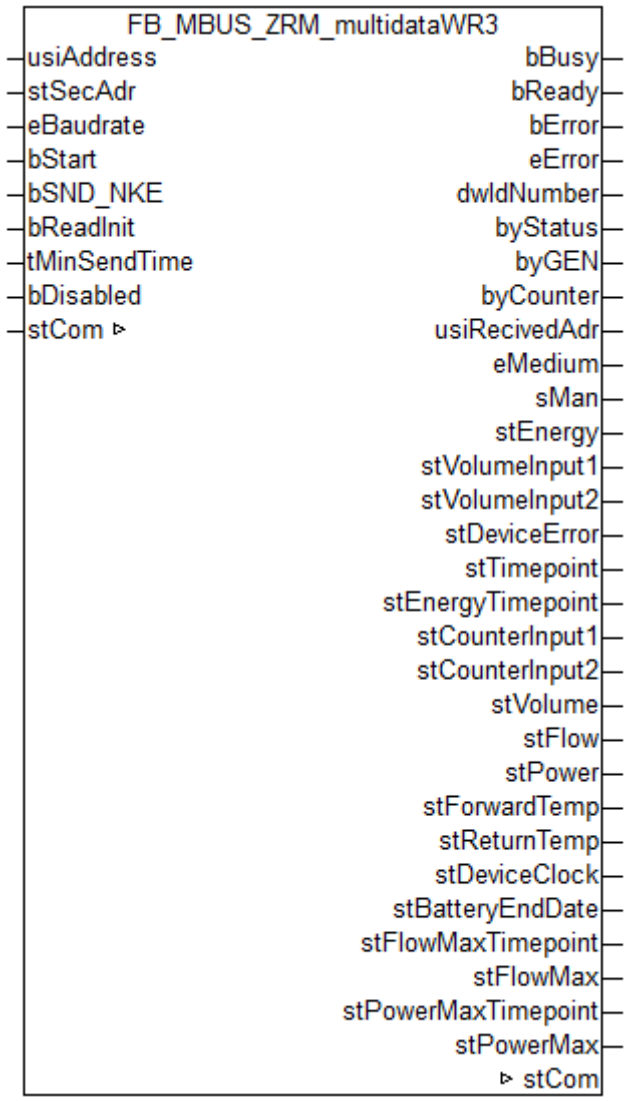
Development environment	Target system	Required libraries
TwinCAT 2.11 R3/x64 from build 2242	PC/CX, BX or BC	TcMBus library from V2.4.0

6.36 Zenner

The function blocks only output a selection of the most common data. These data are described on the respective pages under "VAR_OUT". If more or all data are required, the function blocks FB_MBUS_General [▶ 31], FB_MBUS_General_Ext [▶ 35] or FB_MBUS_General_Param [▶ 41] from the General folder should be used. Not that these function blocks do not run on BC or BX systems. The function block FB_MBUS_General_Send [▶ 43] can be used to send data to the device (e.g. setting of the primary address).

Vendor	Type	Device	Function block
Zenner	Arithmetic unit	multidataWR3	FB_MBUS_ZRM_multidataWR3 [▶ 211]
	Heat meter	zelsiusZR	FB_MBUS_ZRM_zelsiusZR [▶ 189]

6.36.1 FB_MBUS_ZRM_multidataWR3



This module is used to readout energy calculators of the manufacturer Zenner.

It can only be used together with the module FB_MBUSKL6781() [▶ 29].

Functionality of the module [▶ 13]

VAR_INPUT

```
usiAddress      : USINT;
stSecAdr        : ST_MBUS_SecAdr;
eBaudrate       : E_MBUS_Baudrate := eMBUS_Baud2400;
bStart          : BOOL;
bSND_NKE        : BOOL := TRUE;
bReadInit       : BOOL := TRUE;
tMinSendTime    : TIME := t#2s;
bDisabled       : BOOL := FALSE;
```

usiAddress: Primary address [▶ 14] of the counter, that shall be readout with this module.

stSecAdr: Secondary address [▶ 15] of the counter, that shall be readout with this module.

eBaudrate: 300, 2400, 9600 baud [▶ 216].

bStart: Positive edge on this input, the meter is read out once.

bSND_NKE: TRUE to initialize the meter at each reading and sets the meter on the first telegram (SND_NKE).

bReadInit: After restarting the PLC, the meter is read out once.

tMinSendTime: Standard t#2s. When this time exceeds, the counter is rereadout. At t#0s the counter is not readout and can be readout with bStart manually.

bDisabled: TRUE = disable the function block.

VAR_OUTPUT

```
bBusy               : BOOL;
bReady              : BOOL;
bError              : BOOL;
eError              : E_MBUS_ERROR;
dwIdNumber          : DWORD;
byStatus            : BYTE;
byGEN               : BYTE;
byCounter           : BYTE;
usiRecivedAdr       : USINT;
eMedium             : E_MBUS_Medium;
sMan                : STRING(3);
stEnergy            : ST_MBus_Info;
stVolumeInput1      : ST_MBus_Info;
stVolumeInput2      : ST_MBus_Info;
stDeviceError       : ST_MBus_Info;
stTimepoint         : ST_MBus_Info;
stEnergyTimepoint   : ST_MBus_Info;
stCounterInput1     : ST_MBus_Info;
stCounterInput2     : ST_MBus_Info;
stVolume            : ST_MBus_Info;
stFlow              : ST_MBus_Info;
stPower             : ST_MBus_Info;
stForwardTemp       : ST_MBus_Info;
stReturnTemp        : ST_MBus_Info;
stDeviceClock       : ST_MBus_Info;
stBatteryEndDate    : ST_MBus_Info;
stFlowMaxTimepoint  : ST_MBus_Info;
stFlowMax           : ST_MBus_Info;
stPowerMaxTimepoint : ST_MBus_Info;
stPowerMax          : ST_MBus_Info;
```

bBusy: The *bBusy* output is TRUE while the meter is being read.

bReady: The *bReady* output is TRUE for one cycle, once meter reading is completed.

bError: The *bError* output becomes TRUE as soon as an error occurs. The error is described via the variable *eError*.

eError: The *eError* output issues an error code when an error occurs (see E_MBUS_ERROR [▶ 216]).

dwIdNumber: Serial number of the counter (secondary address).

byStatus: Status of the counter. Please refer to device description for meanings.

byGEN: Counter software version.

byCounter: Transmission counter (number of transmitted RSP_UD).

usiRecivedAdr: Received primary address (0-250).

eMedium: Medium (see E_MBUS_Medium [▶ 219]).

sMan: Manufacturer code.

stEnergy: Counter reading, energy consumption (see ST_MBus_Info [▶ 222]).

stVolumeInput1: Volume input 1 (see ST_MBus_Info [▶ 222]).

stVolumeInput2: Volume input 2 (see ST_MBus_Info [▶ 222]).

stDeviceError: Error state M-Bus output (see ST_MBus_Info [▶ 222]).

stTimepoint: Deadline (date and time of the next deadline) (see ST_MBus_Info [▶ 222]).

stEnergyTimepoint: Heat energy at the deadline (see ST_MBus_Info [▶ 222]).

stCounterInput1: Counter value input 1 at the deadline (see ST_MBus_Info [▶ 222]).

stCounterInput2: Counter value input 2 at the deadline (see ST_MBus_Info [▶ 222]).

stVolume: Volume (see ST_MBus_Info [▶ 222]).

stFlow: Actual flow rate (see ST_MBus_Info [▶ 222]).

stPower: Current energy consumption (see ST_MBus_Info [▶ 222]).

stForwardTemp: Flow temperature (see ST_MBus_Info [▶ 222]).

stReturnTemp: Return temperature (see ST_MBus_Info [▶ 222]).

stDeviceClock: Actual counter time (see ST_MBus_Info [▶ 222]).

stBatteryEndDate: Probable durability of the battery (see ST_MBus_Info [▶ 222]).

stFlowMaxTimepoint: Storage time maximum value flow rate (absolute) (see ST_MBus_Info [▶ 222]).

stFlowMax: Maximum value flow rate (absolute) (see ST_MBus_Info [▶ 222]).

stPowerMaxTimepoint: Storage time maximum value power (absolute) (see ST_MBus_Info [▶ 222]).

stPowerMax: Maximum value power (absolute) (see ST_MBus_Info [▶ 222]).

VAR_IN_OUT

```
stCom           : ST_MBUS_Communication;
```

stCom: About this structure, the block FB_MBUSKL6781() [▶ 29] is connected to the meter function blocks (see ST_MBUS_Communication [▶ 221]).

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055667723/.zip:

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055669131/.zip:

Controller configuration setting: "BCxx50 or BX serial"

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055670539/.zip:

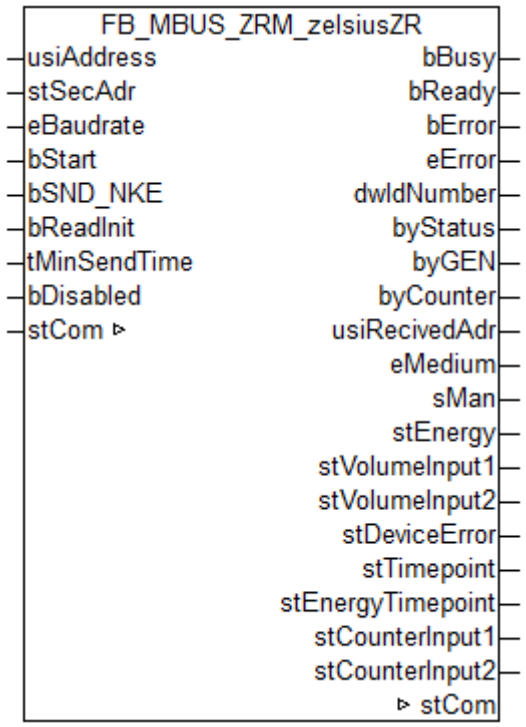
BCxx00 must be set to "Large Model" in the PLC under "Project/Options/Controller Settings".

Controller configuration setting: "BC serial"

Requirements

Development environment	Target system type	PLC libraries to include
TwinCAT 2.11 R3/x64 higher than build 2234	PC/CX, BX or BC	TcMBus-library higher than V2.1.0

6.36.2 FB_MBUS_ZRM_zelsiusZR



This function block is used to read heat meters from Zenner.

Also usable with:

- Sanext Combi from SANEXT (from V2.7.0)

The function block can only be executed together with the function block FB_MBUSKL6781() [▶ 29].

Functionality of the function block [▶ 13]

VAR_INPUT

```
usiAddress      : USINT;
stSecAdr        : ST_MBUS_SecAdr;
eBaudrate       : E_MBUS_Baudrate := eMBUS_Baud2400;
bStart          : BOOL;
bSND_NKE        : BOOL := TRUE;
bReadInit       : BOOL := TRUE;
tMinSendTime    : TIME := t#2s;
bDisabled       : BOOL := FALSE;
```

usiAddress: Primary address [▶ 14] of the counter, that shall be readout with this module.

stSecAdr: Secondary address [▶ 15] of the counter, that shall be readout with this module.

eBaudrate: 300, 2400, 9600 baud [▶ 216].

bStart: Positive edge on this input, the meter is read out once.

bSND_NKE: TRUE to initialize the meter at each reading, and sets the meter on the first telegram (SND_NKE).

bReadInit: After restarting the PLC, the meter is read out once.

tMinSendTime: Standard t#2s. When this time exceeds, the counter is rereadout. At t#0s the counter is not readout and can be readout with bStart manually.

bDisabled: TRUE = disable the function block.

VAR_OUTPUT

```
bBusy               : BOOL;
bReady              : BOOL;
bError              : BOOL;
eError              : E_MBUS_ERROR;
dwIdNumber          : DWORD;
byStatus            : BYTE;
byGEN               : BYTE;
byCounter           : BYTE;
usiRecivedAdr       : USINT;
eMedium             : E_MBUS_Medium;
sMan                : STRING(3);
stEnergy            : ST_MBus_Info;
stVolumeInput1      : ST_MBus_Info;
stVolumeInput2      : ST_MBus_Info;
stDeviceError       : ST_MBus_Info;
stTimepoint         : ST_MBus_Info;
stEnergyTimepoint   : ST_MBus_Info;
stCounterInput1     : ST_MBus_Info;
stCounterInput2     : ST_MBus_Info;
```

bBusy: The *bBusy* output is TRUE while the meter is being read.

bReady: The *bReady* output is TRUE for one cycle, once meter reading is completed.

bError: The *bError* output becomes TRUE as soon as an error occurs. The error is described via the variable *eError*.

eError: The *eError* output issues an error code when an error occurs (see E_MBUS_ERROR [▶ 216]).

dwIdNumber: Serial number of the meter (secondary address).

byStatus: Status of the meter. Please refer to device description for meanings.

byGEN: Meter software version.

byCounter: Transmission counter (number of transmitted RSP_UD).

usiRecivedAdr: Received primary address (0-250).

eMedium: Medium (see E_MBUS_Medium [▶ 219]).

sMan: Manufacturer code.

stEnergy: Meter reading, energy consumption (see ST_MBus_Info [▶ 222]).

stVolumeInput1: Volume Input 1 (see ST_MBus_Info [▶ 222]).

stVolumeInput2: Volume Input 2 (see ST_MBus_Info [▶ 222]).

stDeviceError: Error state M-Bus output (see ST_MBus_Info [▶ 222]).

stTimepoint: Deadline (date and time of the next deadline) (see ST_MBus_Info [▶ 222]).

stEnergyTimepoint: Heat energy at the deadline (see ST_MBus_Info [▶ 222]).

stCounterInput1: Counter value input 1 at the deadline (see ST_MBus_Info [▶ 222]).

stCounterInput2: Counter value input 2 at the deadline (see ST_MBus_Info [▶ 222]).

Requirements

Development environment	Target system type	PLC libraries to include
TwinCAT 2.11 R3/x64 higher than Build 2234	PC/CX, BX or BC	TcMBus-library higher than V2.1.0

VAR_IN_OUT

```
stCom           : ST_MBUS_Communication;
```

stCom: About this structure, the block FB_MBUSKL6781() [▶ 29] is connected to the meter function blocks (see ST_MBUS_Communication [▶ 221]).

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055667723/.zip:

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055669131/.zip:

Controller configuration setting: "BCxx50 or BX serial"

https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055670539/.zip:

BCxx00 must be set to "Large Model" in the PLC under "Project/Options/Controller Settings".

Controller configuration setting: "BC serial"

6.37 Data types

6.37.1 E_MBus_Baudrate

Baudrate.

```
TYPE E_MBus_Baudrate :
(
  eMBUS_NoBaudrate  := 0,
  eMBUS_Baud300     := 30,
  eMBUS_Baud600     := 60,
  eMBUS_Baud1200    := 120,
  eMBUS_Baud2400    := 240,
  eMBUS_Baud4800    := 480,
  eMBUS_Baud9600    := 960
)
END_TYPE
```

eMBUS_NoBaudrate: =Standard =2400 baud.

eMBUS_Baud300: 300 baud.

eMBUS_Baud600: 600 baud.

eMBUS_Baud1200: 1200 baud.

eMBUS_Baud2400: 2400 baud.

eMBUS_Baud4800: 4800 baud.

eMBUS_Baud9600: 9600 baud.

M-Bus counters are generally supplied with 2400 baud.

The M-Bus master terminal KL6781 supports 300, 600, 1200, 2400, 4800, 9600 baud.

If the input *eBaudrate* is not assigned or set to 0, the Bus Terminal is set to 2400 baud.

These values are transferred to the terminal when the PLC is started or when a change takes place at the *eBaudrate* input.

Not all M-Bus devices support baud rates above 2400. Devices that are set to a different baud rate than *eBaudrate* are not read.

6.37.2 E_MBus_Error

Error massage.

```
TYPE E_MBus_Error :
(
  eMBUS_no_error                    := 0,
```

```
  eMBUS_busy                                := 1,
  eMBUS_Disabled                            := 3,
  eMBUS_FBKL6781_Disabled                   := 4,

  eMBUSERROR_CIField_wrong_72hex_expected  := 101,
  eMBUSERROR_no_data_received               := 102,
  eMBUSERROR_error_checksum                 := 103,
  eMBUSERROR_error_in_head_data             := 104,
  eMBUSERROR_usiAddress_over_250            := 105,
  eMBUSERROR_send_error                     := 106,
  eMBUSERROR_received_address_wrong         := 108,
  eMBUSERROR_cMBUS_MaxCom_below_1           := 109,
  eMBUSERROR_iComId_over_cMBUS_MaxCom       := 110,
  eMBUSERROR_manufacturer_sign_wrong        := 111,
  eMBUSERROR_baudrate_wrong                 := 112,
  eMBUSERROR_ReceiveBufferFull              := 113,
  eMBUSERROR_E5hex_no_received              := 114,
  eMBUSERROR_no_stop_character              := 115,
  eMBUSERROR_length_wrong                   := 116,
  eMBUSERROR_wrong_terminal                 := 117,
  eMBUSERROR_Terminal_is_not_initialized    := 118,
  eMBUSERROR_stSecAdr_udiIdNumber_wrong     := 119,
  eMBUSERROR_missing_parts_telegram         := 120,
  eMBUSERROR_no_stop_character_received     := 121,
  eMBUSERROR_too_many_characters            := 122,
  eMBUSERROR_TimeOut_FB_KL6781              := 123,
  eMBUSERROR_TimeOut_MeterFB                := 124,

  eMBUSERROR_COM_PARAMETERCHANGED           := 201,
  eMBUSERROR_COM_TXBUFFOVERRUN              := 202,
  eMBUSERROR_COM_STRINGOVERRUN              := 210,
  eMBUSERROR_COM_ZEROCHARINVALID            := 211,
  eMBUSERROR_COM_INVALIDPOINTER             := 220,
  eMBUSERROR_COM_INVALIDRXPOINTER           := 221,
  eMBUSERROR_COM_INVALIDRXLENGTH            := 222,
  eMBUSERROR_COM_DATASIZEOVERRUN            := 223,
  eMBUSERROR_COM_INVALIDBAUDRATE            := 16#1001,
  eMBUSERROR_COM_INVALIDNUMDATABITS         := 16#1002,
  eMBUSERROR_COM_INVALIDNUMSTOPBITS         := 16#1003,
  eMBUSERROR_COM_INVALIDPARITY              := 16#1004,
  eMBUSERROR_COM_INVALIDHANDSHAKE           := 16#1005,
  eMBUSERROR_COM_INVALIDNUMREGISTERS        := 16#1006,
  eMBUSERROR_COM_INVALIDREGISTER            := 16#1007,
  eMBUSERROR_COM_TIMEOUT                    := 16#1008
)
END_TYPE
```

eMBUS_no_error: No error is present at the block. The block is currently not querying a counter.

eMBUS_busy: The block is querying a meter.

eMBUS_Disabled: FB is disabled.

eMBUS_FBKL6781_Disabled: Function block FB_MBUSKL6781() [▶ 29] is disabled.

eMBUSERROR_CIField_wrong_72hex_expected: The 7th byte in the response telegram contains the CI field. In this byte the hexadecimal number 72 is expected. It stands for variable data structure, low byte is sent first. Only this data structure is supported.

eMBUSERROR_no_data_received: No data was received. This can different have causes, e.g. invalid address, invalid baud rate, incorrect wiring.

eMBUSERROR_error_checksum: The response telegram includes a checksum (sum of all bytes from byte 5). The received checksum does not match the calculated checksum. This happens if the protocol was not received cleanly (e.g. in the event of interference on the cable or if the cable is too long).

eMBUSERROR_error_in_head_data: The first 4 bytes are not included in the checksum. These 4 bytes are monitored separately.

eMBUSERROR_usiAddress_over_250: Addresses higher than 250 are not permitted. The input *usiAddress* of the meter block was assigned a higher value than 250.

eMBUSERROR_send_error: Error message for error during sending.

eMBUSERROR_received_address_wrong: Received address does not match the sent address.

eMBUSERROR_cMBUS_MaxCom_below_1: Reserve.

eMBUSERROR_iComId_over_cMBUS_MaxCom: Reserve.

eMBUSERROR_manufacturer_sign_wrong: The response telegram includes a manufacturer code. This code is allocated to the meter blocks. This message appears if the received manufacturer code does not match the block used.

eMBUSERROR_baudrate_wrong: Input *eBaudrate* of the block was assigned invalid values. Only E_MBUS_Baudrate [▶ 216] are allowed.

eMBUSERROR_ReceiveBufferFull: The receive buffer of the serial interface is full. This may happen with long telegrams and/or long cycle times. The PLC is unable to read the data quick enough from the receive buffer, resulting in data loss. The situation may be resolved by reducing the cycle time.

eMBUSERROR_E5hex_no_received: E5 no received.

eMBUSERROR_no_stop_character: No stop character.

eMBUSERROR_length_wrong: Data lenght wrong.

eMBUSERROR_wrong_terminal: Wrong terminal.

eMBUSERROR_Terminal_is_not_initialized: Terminal is not initilalized.

eMBUSERROR_stSecAdr_udiIdNumber_wrong: The input variable *stSecAdr.udiIdNumber* is not used.

eMBUSERROR_missing_parts_telegram: Values (bytes) are missing.

eMBUSERROR_no_stop_character_received: Stop sign was not received (16hex).

eMBUSERROR_too_many_characters: Too many characters have been received.

eMBUSERROR_TimeOut_FB_KL6781: Timeout *FB_KL6781*.

eMBUSERROR_TimeOut_MeterFB: Timeout meter function block.

eMBUSERROR_COM_PARAMETERCHANGED: Input parameters changed during reception.

eMBUSERROR_COM_TXBUFFOVERRUN: String > transmit buffer.

eMBUSERROR_COM_STRINGOVERRUN: End of string.

eMBUSERROR_COM_ZEROCHARINVALID: String cannot receive zero characters.

eMBUSERROR_COM_INVALIDPOINTER: Invalid data pointer, e. g. zero.

eMBUSERROR_COM_INVALIDRXPOINTER: Invalid data pointer for *ReceiveData*.

eMBUSERROR_COM_INVALIDRXLENGTH: Invalid length for *ReceiveData*, e. g. zero

eMBUSERROR_COM_DATASIZEOVERRUN: End of data block.

eMBUSERROR_COM_INVALIDBAUDRATE: Invalid baudrate.

eMBUSERROR_COM_INVALIDNUMDATABITS: Invalid data bits.

eMBUSERROR_COM_INVALIDNUMSTOPBITS: Invalid stop bits.

eMBUSERROR_COM_INVALIDPARITY: Invalid parity.

eMBUSERROR_COM_INVALIDHANDSHAKE: Invalid handshake.

eMBUSERROR_COM_INVALIDNUMREGISTERS: Invalid numregister.

eMBUSERROR_COM_INVALIDREGISTER: Invalid register.

eMBUSERROR_COM_TIMEOUT: COM timeout.

6.37.3 E_MBus_Fct

Function of the value.

```
TYPE E_MBus_Fct :
(
  eMBUS_ValueNull              := -1,
  eMBUS_InstantaneousValue     := 0,
  eMBUS_Max                    := 1,
  eMBUS_Min                    := 2,
  eMBUS_ValueDuringErrorState  := 3,

  eMBUS_ManufacturerSpecific   := 256
)
END_TYPE
```

eMBUS_ValueNull: Unassigned.

eMBUS_InstantaneousValue: Instantaneous value.

eMBUS_Max: Maximum value.

eMBUS_Min: Minimum value.

eMBUS_ValueDuringErrorState: Value during error state.

eMBUS_ManufacturerSpecific: Manufacturer specific.

6.37.4 E_MBus_Medium

Medium.

```
TYPE E_MBus_Medium :
(
  eMBUS_MediumNull                      := -1,
  eMBUS_MediumOther                     := 0,
  eMBUS_MediumOil                       := 1,
  eMBUS_MediumElectricity               := 2,
  eMBUS_MediumGas                       := 3,
  eMBUS_MediumHeat_Outlet               := 4,
  eMBUS_MediumSteam                     := 5,
  eMBUS_MediumHot_Water                 := 6,
  eMBUS_MediumWater                     := 7,
  eMBUS_MediumHeat_Cost_Allocator       := 8,
  eMBUS_MediumCompressed_Air            := 9,
  eMBUS_MediumCooling_load_meter_outlet := 10,
  eMBUS_MediumCooling_load_meter_intlet := 11,
  eMBUS_MediumHeat_inlet                := 12,
  eMBUS_MediumHeat_cooling_load_Meter   := 13,
  eMBUS_MediumBusSystem                 := 14,
  eMBUS_MediumUnknownMedium             := 15,
  eMBUS_MediumReserved16                := 16,
  eMBUS_MediumReserved17                := 17,
  eMBUS_MediumReserved18                := 18,
  eMBUS_MediumReserved19                := 19,
  eMBUS_MediumReserved20                := 20,
  eMBUS_MediumReserved21                := 21,
  eMBUS_MediumColdWater                 := 22,
  eMBUS_MediumDualWater                 := 23,
  eMBUS_MediumPressure                  := 24,
  eMBUS_MediumA_D_Converter             := 25,
  eMBUS_MediumReserved26                := 26,
  eMBUS_MediumReserved27                := 27,
  eMBUS_MediumReserved28                := 28,
  eMBUS_MediumReserved29                := 29,
  eMBUS_MediumReserved30                := 30
)
END_TYPE
```

eMBUS_MediumNull: Unassigned.

eMBUS_MediumOther: Other.

eMBUS_MediumOil: Oil.

eMBUS_MediumElectricity: Electricity.

eMBUS_MediumGas: Gas.

eMBUS_MediumHeat_Outlet: Heat outlet.

eMBUS_MediumSteam: Steam.

eMBUS_MediumHot_Water: Hot water.

eMBUS_MediumWater: Water.

eMBUS_MediumHeat_Cost_Allocator: Heat cost allocator.

eMBUS_MediumCompressed_Air: Compressed air.

eMBUS_MediumCooling_load_meter_outlet: Cooling load meter (outlet).

eMBUS_MediumCooling_load_meter_intlet: Cooling load meter (intlet).

eMBUS_MediumHeat_inlet: Heat (inlet).

eMBUS_MediumHeat_cooling_load_Meter: Heat / cooling load meter.

eMBUS_MediumBusSystem: Bus / system.

eMBUS_MediumUnknownMedium: Unknown medium.

eMBUS_MediumReserved16: Reserved.

eMBUS_MediumReserved17: Reserved.

eMBUS_MediumReserved18: Reserved.

eMBUS_MediumReserved19: Reserved.

eMBUS_MediumReserved20: Reserved.

eMBUS_MediumReserved21: Reserved.

eMBUS_MediumColdWater: Cold water.

eMBUS_MediumDualWater: Dual water.

eMBUS_MediumPressure: Pressure.

eMBUS_MediumA_D_Converter: A/D converter.

eMBUS_MediumReserved26: Reserved.

eMBUS_MediumReserved27: Reserved.

eMBUS_MediumReserved28: Reserved.

eMBUS_MediumReserved29: Reserved.

eMBUS_MediumReserved30: Reserved.

6.37.5 ST_KL6781inData22B

Process image of the inputs

Is linked to the terminal in the System Manager.

```
TYPE ST_KL6781inData22B :
STRUCT
  Status  : WORD;
  D       : ARRAY[0..21] OF BYTE;
END_STRUCT
END_TYPE
```

Status: Status word

D: 22 bytes for input data of the M-Bus

6.37.6 ST_KL6781outData22B

Process image of the outputs.

Is linked to the terminal in the System Manager.

```
TYPE ST_KL6781outData22B :
STRUCT
  Ctrl  : WORD;
  D     : ARRAY[0..21] OF BYTE;
END_STRUCT
END_TYPE
```

Ctrl: Control word

D: 22 bytes for output data of the M-Bus

6.37.7 ST_MBus_Communication

Internal structure.

About this structure, the block FB_MBUSKL6781() [▶ 29] is connected to the meter function blocks.

```
TYPE ST_MBus_Communication :
STRUCT
  bStart            : BOOL;
  bBusy             : BOOL;
  bSND_NKE          : BOOL;
  bSend             : BOOL;
  bStartManuell     : BOOL;
  bBlockadeSecAdr   : BOOL;
  usiAddress        : USINT;
  byCField          : BYTE;
  stSecAdr          : ST_MBUS_SecAdr;
  eError            : E_MBUS_ERROR;
  eBaudrate         : E_MBUS_Baudrate := eMBUS_Baud2400;
  arrMBusLongFrame  : ARRAY[1..260] OF BYTE;
  bySendByte        : BYTE;
  uiMaxCount        : UINT;
  uiCount           : UINT;
  stKomRxBuffer     : ST_KL6781ComBuffer;
  stKomTxBuffer     : ST_KL6781ComBuffer;
END_STRUCT
END_TYPE
```

bStart: Start.

bBusy: This bit is set for as long as the block is active.

bSND_NKE: Send SNDK_NKE.

bSend: Send datas.

bStartManuell: Manual start.

bBlockadeSecAdr: Lock in secondary addressing.

usiAddress: Primary address.

byCField: C field.

stSecAdr: Secondary address [▶ 223].

stSecAdr. udiIdNumber: Serial number of the meter.

stSecAdr. uiManufacturer: Manufacturer code.

stSecAdr. usiVersion: Meter software version.

stSecAdr. usiMedium: Medium.

eError: Error massage [▶ 216].

eBaudrate: Baudrate [▶ 216].

arrMBusLongFrame: Received bytes.

bySendByte: Number of bytes.

uiMaxCount: Maximum number of read commands.

uiCount: Current read command.

stKomRxBuffer: Receive buffer.

stKomTxBuffer: Send buffer.

6.37.8 ST_MBus_Data

Value information.

```
TYPE ST_MBus_Data :
STRUCT
  sValue   : STRING(25);
  sUnit    : STRING(20);
  sInfo    : STRING;
  eFct     : E_MBus_Fct;
  iTariff  : INT;
  iStorNo  : INT;
  iUnit    : INT;
  byVIFE   : BYTE;
END_STRUCT
END_TYPE
```

sValue: Value.

sUnit: Unit.

sInfo: Information.

eFct: Funktion [▶ 219].

iTariff: Tariff.

iStorNo: Storage number.

iUnit: Unit (integer).

byVIFE: VIFE.

6.37.9 ST_MBus_Data2

Structure of the output values in the block FB_MBUS_General_Ext [▶ 35].

```
TYPE ST_MBus_Data2 :
STRUCT
  arrData  : ARRAY[1..cMBUS_MaxData] OF ST_MBus_Data;
END_STRUCT
END_TYPE
```

arrData: Values.

6.37.10 ST_MBus_Info

Value information.

```
TYPE ST_MBus_Info :
STRUCT
  sValue   : STRING(25);
  sUnit    : STRING(20);
  eFct     : E_MBus_Fct;
END_STRUCT
END_TYPE
```

sValue: Value.

sUnit: Unit.

eFct: Function [▶ 219].

M-Bus devices may supply very large values, which cannot be displayed or can only be displayed inaccurately as numbers on BC/BX systems. The values are therefore supplied as strings (sValue).

> LREAL is the preferred format for the conversion to a number format, since conversion to REAL (STRING_TO_REAL) would return inaccurate/invalid values if the DWORD value range is exceed. LREAL values cannot be used on BC/BX systems.

Example view of the values of a heat meter:

```
   iGEN = 47
⊟··stEnergy
     ├···.sValue = '0.140'
     ├···.sUnit = 'MWh'
     └···.eFct = eMBUS_InstantaneousValue
⊟··stTarrif1
     ├···.sValue = '0.0'
     ├···.sUnit = 'MWh'
     └···.eFct = eMBUS_InstantaneousValue
⊟··stTarrif2
     ├···.sValue = '27.227'
     ├···.sUnit = 'm³'
     └···.eFct = eMBUS_InstantaneousValue
⊟··stVolume
     ├···.sValue = '100.883'
     ├···.sUnit = 'm³'
     └···.eFct = eMBUS_InstantaneousValue
⊟··stPower
     ├···.sValue = '141.41512'
     ├···.sUnit = 'MW'
     └···.eFct = eMBUS_ValueDuringErrorState
⊞··stFlow
⊞··stForwardTemp
⊞··stReturnTemp
```

6.37.11 ST_MBus_SecAdr

Secondary address.

```
TYPE ST_MBus_SecAdr :
STRUCT
  udiIdNumber     : UDINT:=16#FFFFFFFF;
  uiManufacturer  : UINT:=16#FFFF;
  usiVersion      : USINT:=16#FF;
  usiMedium       : USINT:=16#FF;
END_STRUCT
END_TYPE
```

udiIdNumber: Serial number of the meter.

uiManufacturer: Manufacturer code.

usiVersion: Meter software version.

usiMedium: Medium.

6.37.12 ST_MBus_Scan

Information while scanning.

```
TYPE ST_MBus_Scan :
STRUCT
  usiAddress  : USINT;
```

```
  dwIdNumber   : DWORD;
  byStatus     : BYTE;
  eMedium      : E_MBUS_Medium;
  sMan         : STRING(3);
  byGEN        : BYTE;
END_STRUCT
END_TYPE
```

usiAddress: Primary address [▶ 14] of the meter.

dwIdNumber: Serial number of the meter.

byStatus: Status of the meter. Please refer to device description for meanings.

eMedium: Medium [▶ 219].

sMan: Manufacturer code.

byGEN: Meter software version.

6.37.13 ST_MBus_DueDayHYD1

Structure of due day values in the block FB_MBUS_HYD_Sharky_00 [▶ 120].

```
TYPE ST_MBus_DueDayHYD1 :
STRUCT
  stEnergy            : ST_MBus_Info;
  stVolume            : ST_MBus_Info;
  stTariff1           : ST_MBus_Info;
  stTariff2           : ST_MBus_Info;
  stDate              : ST_MBus_Info;
  stDateFutureDueDay  : ST_MBus_Info;
END_STRUCT
END_TYPE
```

stEnergy: Meter reading [▶ 222], energy.

stEnergy. sValue: Value.

stEnergy. sUnit: Unit.

stEnergy. eFct: E_MBus_Fct [▶ 219] / function.

stVolume: Meter reading, volume.

stVolume. sValue: Value.

stVolume. sUnit: Unit.

stVolume. eFct: E_MBus_Fct [▶ 219] / function.

stTariff1: Meter reading, tariff 1.

stTariff1. sValue: Value.

stTariff1. sUnit: Unit.

stTariff1. eFct: E_MBus_Fct [▶ 219] / function.

stTariff2: Meter reading, tariff 2.

stTariff2. sValue: Value.

stTariff2. sUnit: Unit.

stTariff2. eFct E_MBus_Fct [▶ 219] / function.

stDate: Date.

stDate. sValue Value.

stDate. sUnit: Unit.

stDate. eFct: E_MBus_Fct [▶ 219] / function.

stDateFutureDueDay: Date future due day.

stDateFutureDueDay. sValue: Value.

stDateFutureDueDay. sUnit: Unit.

stDateFutureDueDay. eFct E_MBus_Fct [▶ 219] / function.

6.37.14 ST_MBus_F22

Structure of the monthly values in the block FB_MBUS_SVM_F22_Ext [▶ 163].

```
TYPE ST_MBus_F22 :
STRUCT
  stEnergy          : ST_MBus_Info;
  stVolume          : ST_MBus_Info;
  stVolume2         : ST_MBus_Info;
  stPulsecounter1   : ST_MBus_Info;
  stPulsecounter2   : ST_MBus_Info;
  stDate            : ST_MBus_Info;
END_STRUCT
END_TYPE
```

stEnergy: Meter reading [▶ 222], energy.

stEnergy. sValue: Value.

stEnergy. sUnit: Unit.

stEnergy. eFct: E_MBus_Fct [▶ 219] / function.

stVolume: Meter reading, volume.

stVolume. sValue: Value.

stVolume. sUnit: Unit.

stVolume. eFct: E_MBus_Fct [▶ 219] / function.

stVolume2: Meter reading, volume.

stVolume2. sValue: Value.

stVolume2. sUnit: Unit.

stVolume2. eFct: E_MBus_Fct [▶ 219] / function.

stPulsecounter1: Meter reading, pulse counter 1.

stPulsecounter1. sValue: Value.

stPulsecounter1. sUnit: Unit.

stPulsecounter1. eFct: E_MBus_Fct [▶ 219] / function.

stPulsecounter2: Meter reading, pulse counter 2.

stPulsecounter2. sValue: Value.

stPulsecounter2. sUnit: Unit.

stPulsecounter2. eFct: E_MBus_Fct [▶ 219] / function.

stDate: Date.

stDate. sValue: Value.

stDate. sUnit: Unit.

stDate. eFct: E_MBus_Fct [▶ 219] / function.

6.38 Globale_Variablen_MBUS

This constants only included in TcMBus.lib.

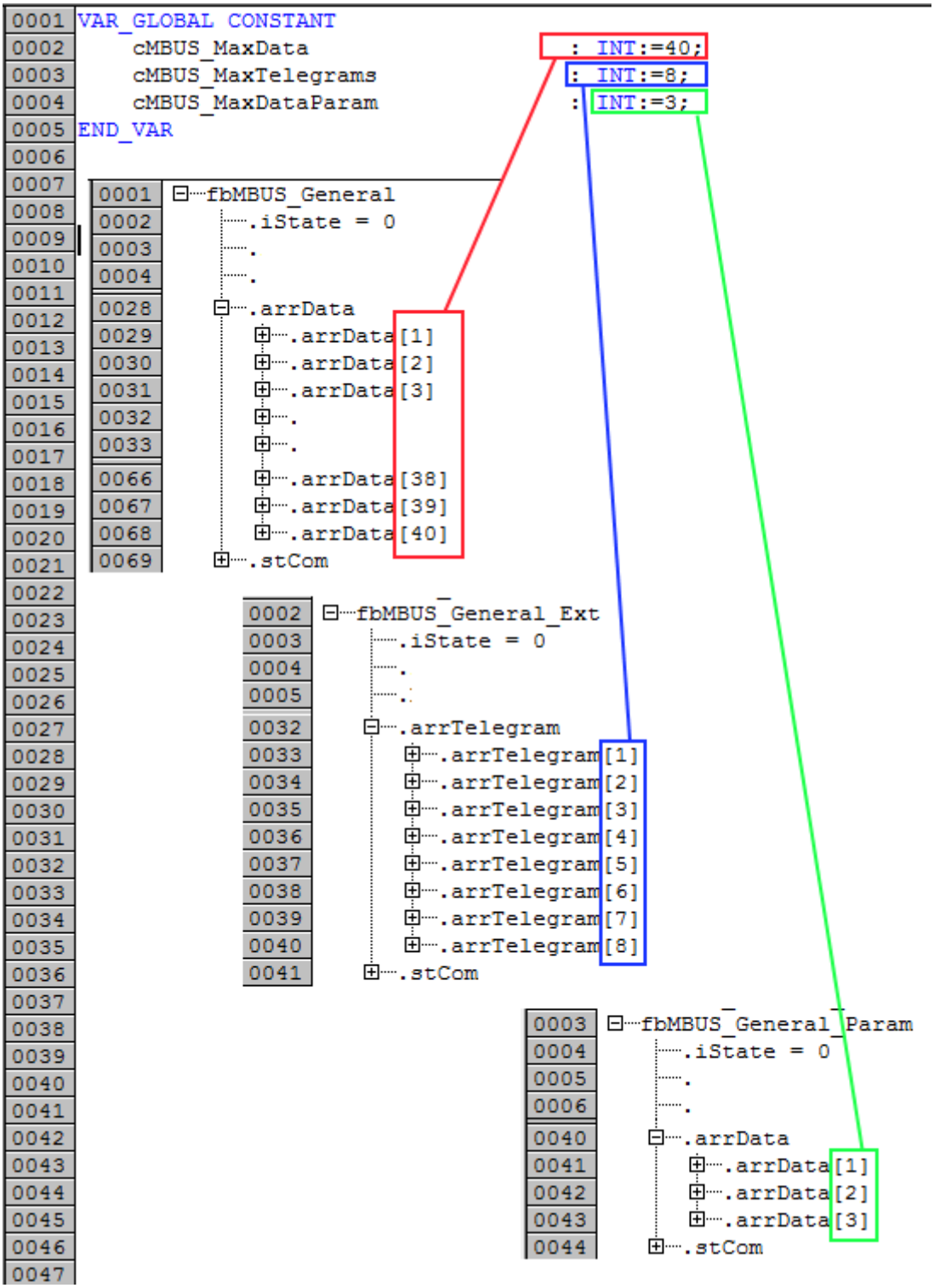
If they are being declared in the program, a warning message is generated during program compilation, since the constant already exists. This warning can be ignored.

```
VAR_GLOBAL CONSTANT
  cMBUS_MaxData       := 40,
  cMBUS_MaxTelegrams  := 5,
  cMBUS_MaxDataParam  := 10,
END_VAR
```

cMBUS_MaxData**:** The constant applies to all instances of the blocks FB_MBUS_General() [▶ 31], FB_MBUS_General_Ext() [▶ 35] and FB_MBUS_General_Param() [▶ 41]. This constant indicates the maximum data volume expected in a meter telegram.

cMBUS_MaxTelegrams**:** The constant applies to all instances of the FB_MBUS_General_Ext() [▶ 35] block. This constant indicates the maximum number of telegrams to be expected.

cMBUS_MaxDataParam**:** The constant applies to all instances of the FB_MBUS_General_Param() [▶ 41] blocks. This constant indicates the maximum number of values to be displayed by the instances of FB_MBUS_General_Param() [▶ 41].


0001 VAR_GLOBAL CONSTANT
0002 cMBUS_MaxData : INT:=40;
0003 cMBUS_MaxTelegrams : INT:=8;
0004 cMBUS_MaxDataParam : INT:=3;
0005 END_VAR
0001 fbMBUS_General
.iState = 0
.arrData
.arrData[1]
.arrData[2]
.arrData[3]
.arrData[38]
.arrData[39]
.arrData[40]
.stCom
0002 fbMBUS_General_Ext
.iState = 0
.arrTelegram
.arrTelegram[1]
.arrTelegram[2]
.arrTelegram[3]
.arrTelegram[4]
.arrTelegram[5]
.arrTelegram[6]
.arrTelegram[7]
.arrTelegram[8]
.stCom
0003 fbMBUS_General_Param
.iState = 0
.arrData
.arrData[1]
.arrData[2]
.arrData[3]
.stCom

7 Error codes

Value (hex)	Value (dec)	Value (enum)	Description
0x0000	0	eMBUS_no_error	No error is present at the block. The block is currently not querying a counter.
0x0001	1	eMBUS_busy	The block is querying a meter.
0x0003	3	eMBUS_Disabled	FB is disabled.
0x0004	4	eMBUS_FBKL6781_Disabled	Function block FB_MBUSKL6781() [▶ 29] is disabled.
0x0065	101	eMBUSERROR_CIField_wrong_72hex_expected	The 7th byte in the response telegram contains the CI field. In this byte the hexadecimal number 72 is expected. It stands for variable data structure, low byte is sent first. Only this data structure is supported.
0x0066	102	eMBUSERROR_no_data_received	No data was received. This can different have causes, e.g. invalid address, invalid baud rate, incorrect wiring.
0x0067	103	eMBUSERROR_error_checksum	The response telegram includes a checksum (sum of all bytes from byte 5). The received checksum does not match the calculated checksum. This happens if the protocol was not received cleanly (e.g. in the event of interference on the cable or if the cable is too long).
0x0068	104	eMBUSERROR_error_in_head_data	The first 4 bytes are not included in the checksum. These 4 bytes are monitored separately.
0x0069	105	eMBUSERROR_usiAddress_ovr_250	Addresses higher than 250 are not permitted. The input *usiAddress* of the meter block was assigned a higher value than 250.
0x006A	106	eMBUSERROR_send_error	Error message for error during sending.
0x006C	108	eMBUSERROR_received_address_wrong	Received address does not match the sent address.
0x006D	109	eMBUSERROR_cMBUS_MaxCom_below_1	Reserve.
0x006E	110	eMBUSERROR_iComId_over_cMBUS_MaxCom	Reserve.
0x006F	111	eMBUSERROR_manufacturer_sign_wrong	The response telegram includes a manufacturer code. This code is allocated to the meter blocks. This message appears if the received manufacturer code does not match the block used.
0x0070	112	eMBUSERROR_baudrate_wrong	Eingang *eBaudrate* Input *eBaudrate* of the block was assigned invalid values. Only E_MBUS_Baudrate [▶ 216] are allowed.
0x0071	113	eMBUSERROR_ReceiveBufferFull	The receive buffer of the serial interface is full. This may happen with long telegrams and/or long cycle times. The PLC is unable to read the

Value (hex)	Value (dec)	Value (enum)	Description
			data quick enough from the receive buffer, resulting in data loss. The situation may be resolved by reducing the cycle time.
0x0072	114	eMBUSERROR_E5hex_no_received	E5 no received.
0x0073	115	eMBUSERROR_no_stop_character	No stop character.
0x0074	116	eMBUSERROR_length_wrong	Data lenght wrong.
0x0075	117	eMBUSERROR_wrong_terminal	Wrong terminal.
0x0076	118	eMBUSERROR_Terminal_is_not_initialized	Terminal is not initilalized.
0x0077	119	eMBUSERROR_stSecAdr_udiIdNumber_wrong	The input variable *stSecAdr.udiIdNumber* is not used.
0x0078	120	eMBUSERROR_missing_parts_telegram	Values (bytes) are missing.
0x0079	121	eMBUSERROR_no_stop_character_received	Stop sign was not received (16hex).
0x007A	122	eMBUSERROR_too_many_characters	Too many characters have been received.
0x007B	123	eMBUSERROR_TimeOut_FB_KL6781	Timeout *FB_KL6781*.
0x007C	124	eMBUSERROR_TimeOut_MeterFB	Timeout meter function block.
0x00C9	201	eMBUSERROR_COM_PARAMETERCHANGED	Input parameters changed during reception.
0x00CA	202	eMBUSERROR_COM_TXBUFFOVERRUN	String > transmit buffer.
0x00D2	210	eMBUSERROR_COM_STRINGOVERRUN	End of string.
0x00D3	211	eMBUSERROR_COM_ZEROCHARINVALID	String cannot receive zero characters.
0x00DC	220	eMBUSERROR_COM_INVALIDPOINTER	Invalid data pointer, e. g. zero.
0x00DD	221	eMBUSERROR_COM_INVALIDRXPOINTER	Invalid data pointer for *ReceiveData*.
0x00DE	222	eMBUSERROR_COM_INVALIDRXLENGTH	Invalid length for *ReceiveData*, e. g. zero
0x00DF	223	eMBUSERROR_COM_DATASIZEOVERRUN	End of data block.
0x1001	4097	eMBUSERROR_COM_INVALIDBAUDRATE	Invalid baudrate.
0x1002	4098	eMBUSERROR_COM_INVALIDNUMDATABITS	Invalid data bits.
0x1003	4099	eMBUSERROR_COM_INVALIDNUMSTOPBITS	Invalid stop bits.
0x1004	4100	eMBUSERROR_COM_INVALIDPARITY	Invalid parity.
0x1005	4101	eMBUSERROR_COM_INVALIDHANDSHAKE	Invalid handshake.
0x1006	4102	eMBUSERROR_COM_INVALIDNUMREGISTERS	Invalid numregister.
0x1007	4103	eMBUSERROR_COM_INVALIDREGISTER	Invalid register.

Value (hex)	Value (dec)	Value (enum)	Description
0x1008	4109	eMBUSERROR_COM_TIMEOUT	COM timeout.

8 Appendix

8.1 Task Configuration

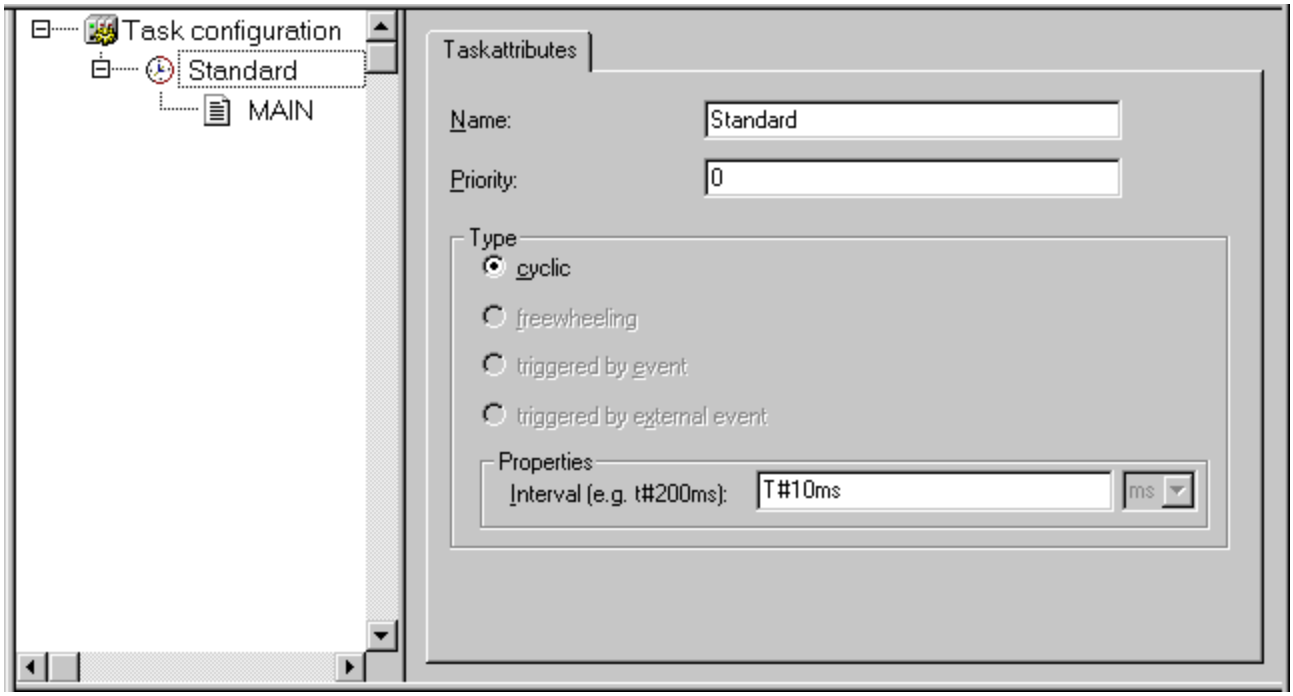
A **Task** is a time unit in the processing of an IEC program. It is defined by a name, a priority and by a type determining which condition will trigger the start of the task. This condition can be defined by a time only (cyclic).

For each task you can now specify a series of programs that will be started by the task.

The combination of priority and condition will determine in which choronological order the tasks will be executed.

In the Online Mode the task processing can be monitored in a diagram.

The Task Configuration is found as an object in the **Resources** register card the Object Organizer. The Task editor is opened in a bipartited window.



In the left part of the window the tasks are represented in a configuration tree. At the topmost position you will always find the entry 'Taskconfiguration'. Below there are the entries for the tasks, represented by the task name. Below each task entry the assigned program calls are inserted.

In the right part of the window a dialog will be displayed which belongs to the currently marked entry in the configuration tree. Here you can configure the tasks and program calls.

Please do not use the same string function in several tasks, because this may cause program faults by overwriting.

Working with the Task Configuration

The most important commands you find in the **context menu** (right mouse button).

At the heading of the task configuration are the words "Task Configuration." If a plus sign is located before the words, then the sequence list is closed. By doubleclicking on the list or pressing <Enter>, you can open the list. A minus sign now appears. By doubleclicking once more, you can close the list again.

- For every task, there is a list of program call-ups attached. Likewise, you can open and close this list the same way.

- With the **"Insert" "Insert Task"** command, you can insert a task.
- With the "**Insert" "Append Task"** command, you can insert a task at the end of the configuration tree.
- With the "**Insert" "Insert Program Call"**, a program call will be inserted.

Furtheron for each entry in the configuration tree an appropriate configuration dialog will appear in the right part of the window. There options can be activated/deactivated resp. inputs to editor fields can be made. Depending on which entry is selected in the configuration tree, there will be the dialog for defining the 'Taskattributes' or the dialog for defining a 'Program Call'. The settings made in the dialogs will be taken over to the configuration tree as soon as the focus is set to the tree again.

A task name or program name can also get edited in the configuration tree. For this perform a mouseclick on the name or select the entry and press the <Space> button to open an edit frame.

You can use the arrow keys to select the previous or next entry in the configuration tree.

'Insert' 'Insert Task' or 'Insert' 'Append Task'

With this command you can insert a new task into the task configuration. If a task is selected, then the **"Insert Task"** command will be at your disposal. The new task will be inserted after the selected one. If the words Task Configuration are selected, then the **"Append Task"** is available, and the new task will be appended to the end of the existing list. The dialog box will open for you to set the task attributes.

Insert the desired attributes:

- **Name**: a name for the task; with this name the task is represented in the configuration tree; the name can be edited there after a mouseclick on the entry or after pressing the <Space> key when the entry is selected.
- **Priority (0-3)**: (a number between 0 and 31; 0 is the highest priority, 31 is the lowest),
- **Type**:
  - **cyclic**: The task will be processed cyclic according to the time definition given in the field 'Interval'.

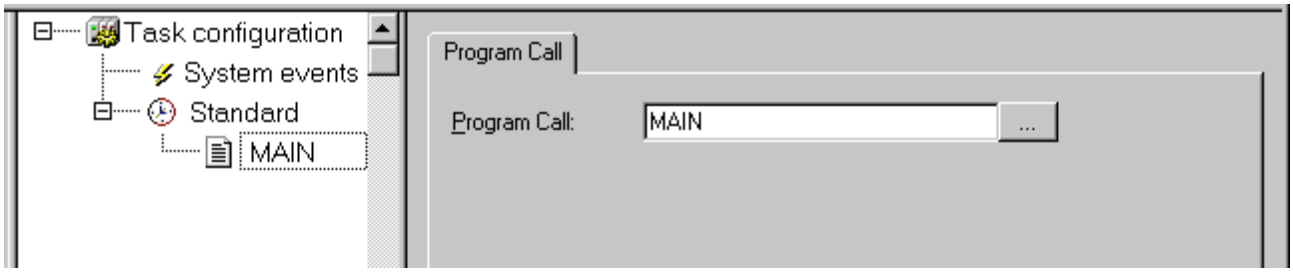
Freewheeling, triggered by event or triggered by external event: These task types are not supported!

Properties:

- **Interval** (for type 'cyclic'): the period of time, after which the task should be restarted. If you enter a number, then you can choose the desired unit in the selection box behind the edit field: milliseconds [ms] or microseconds [µs]. Inputs in [ms]-format will be shown in the TIME format (e.g. "t#200ms") as soon as the window gets repainted; but you also can directly enter the value in TIME format. Inputs in [ms] will always be displayed as a pure number (e.g. "300").

'Insert' 'Insert Program Call' or 'Insert' 'Append Program Call'

With these commands you will open the dialog box for entering a program call to a task in the task configuration. With **"Insert Program Call"**, the new program call is inserted in front of the cursor, and with **"Append Program Call"**, the program call is appended to the end of the existing list.



In the field, specify a valid program name for your project, or open the Input Assistant with the button... or with <F2> to select a valid program name. The program name can also get edited in the configuration tree. For this perform a mouseclick on the name or select the entry and press the <Space> button to open an edit frame. If the selected program requires input variables, then enter these in their usual form and of the declared type (for example, prg(invar:=17)).

'Extras' 'Set Debug Task'

With this command a debugging task can be set in Online mode in the task configuration. The text [DEBUG] will appear after the set task.
The debugging capabilities apply, then, only to this task. In other words, the program only stops at a breakpoint if the program is gone through by the set task. The setting of the Debug Task is stored in the project and will be set again automatically at log in / download.

Cancel the Debug Mode

To cancel the "Debug-Mode"

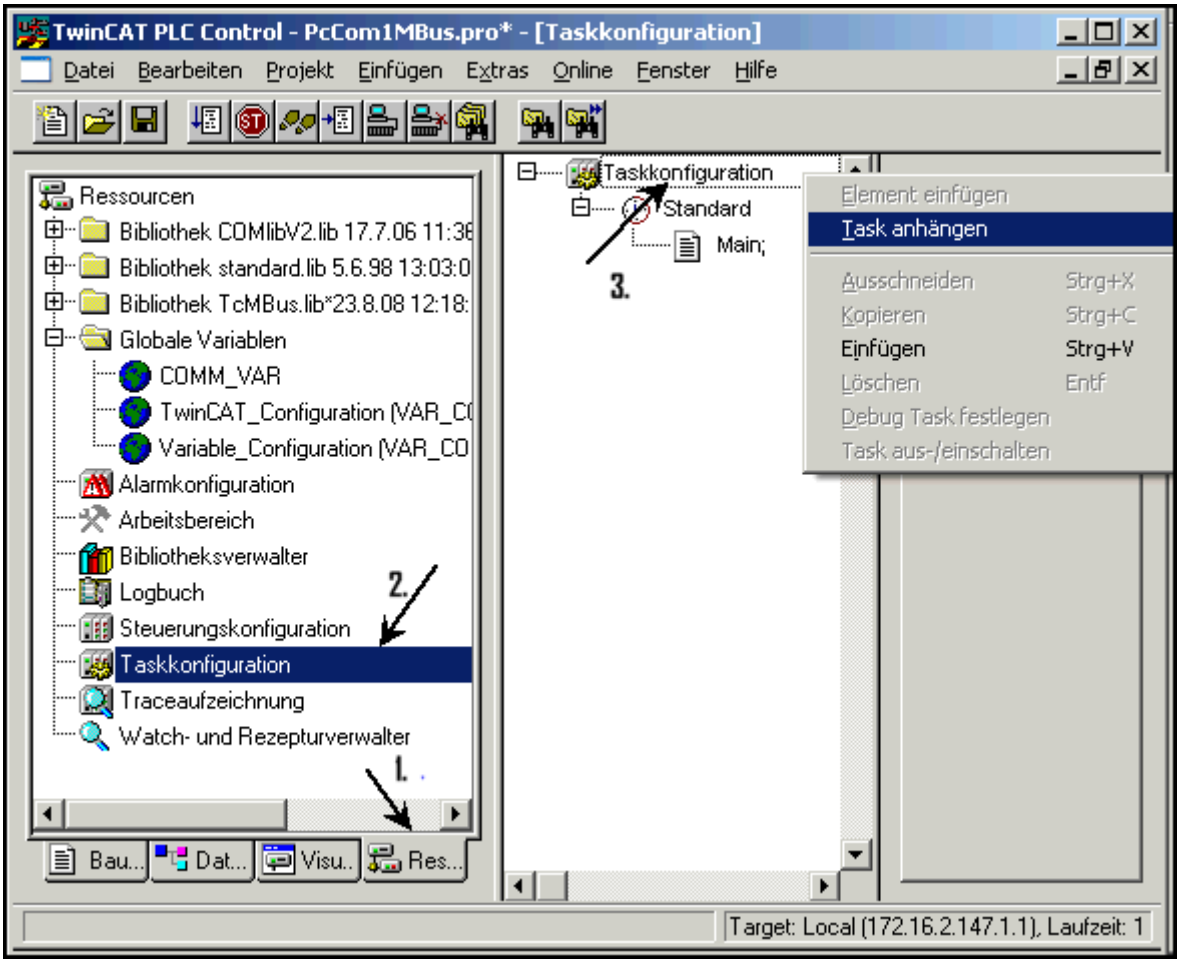
- choose "Task configuration"
- open the Context menu
- choose „Set Debug Task“

'Extras' 'Display Callstack'

If the program is stopped at a breakpoint during debugging, then this command can be used to show the callstack of the corresponding POU. For this purpose the debug task must be selected in the task configuration tree. The window 'Callstack of task <task name>' will open. There you get the name of the POU and the breakpoint position (e.g. "prog_x (2)" for line 2 of POU prog_x) . Below the complete call stack is shown in backward order. If you press button 'Go To', the focus will jump to that position in the POU which is currently marked in the callstack.

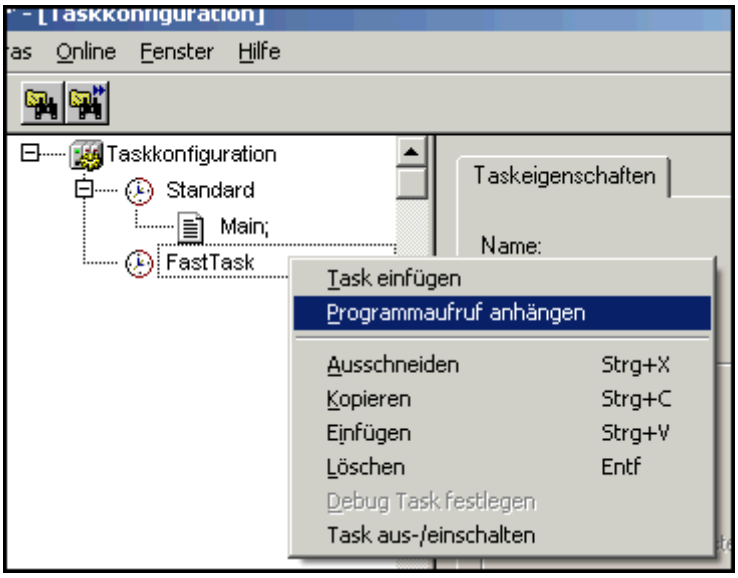
8.2 Configuration with 2 tasks

Right-click Resources (1)/Task configuration (2)/Task configuration (3) and select "Append Task".

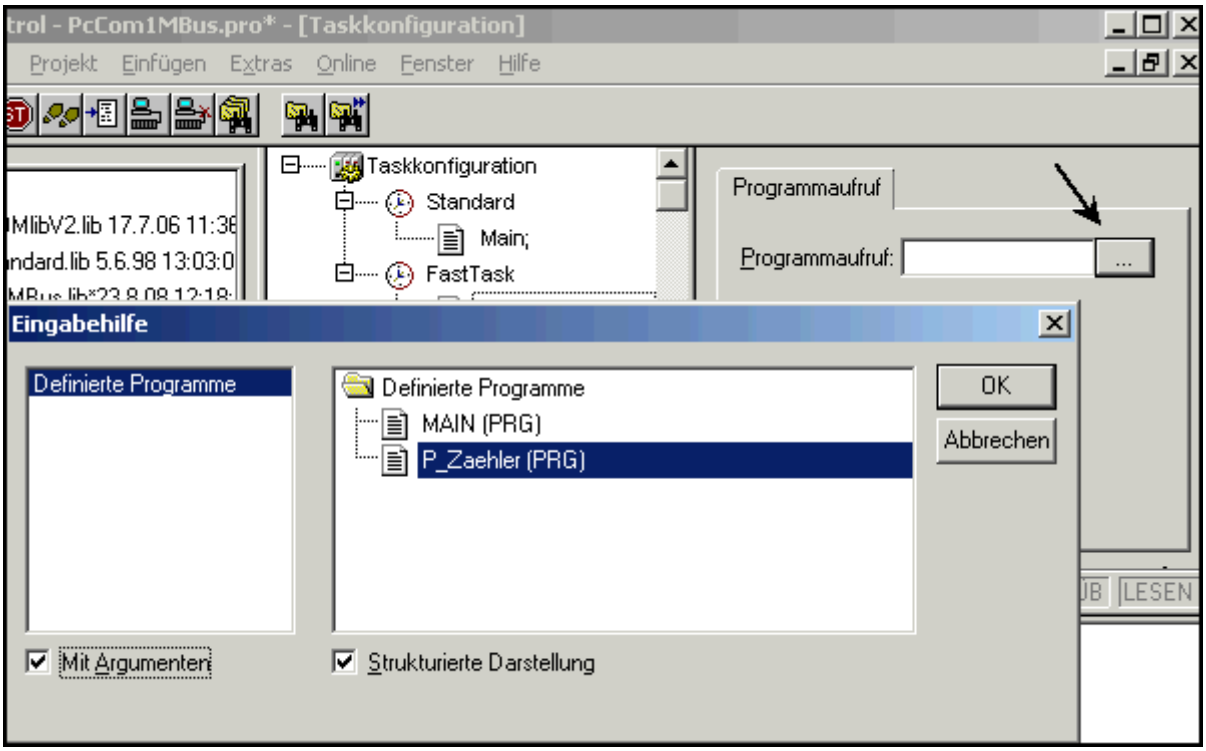


Perhaps rename the new task (here "FastTask")

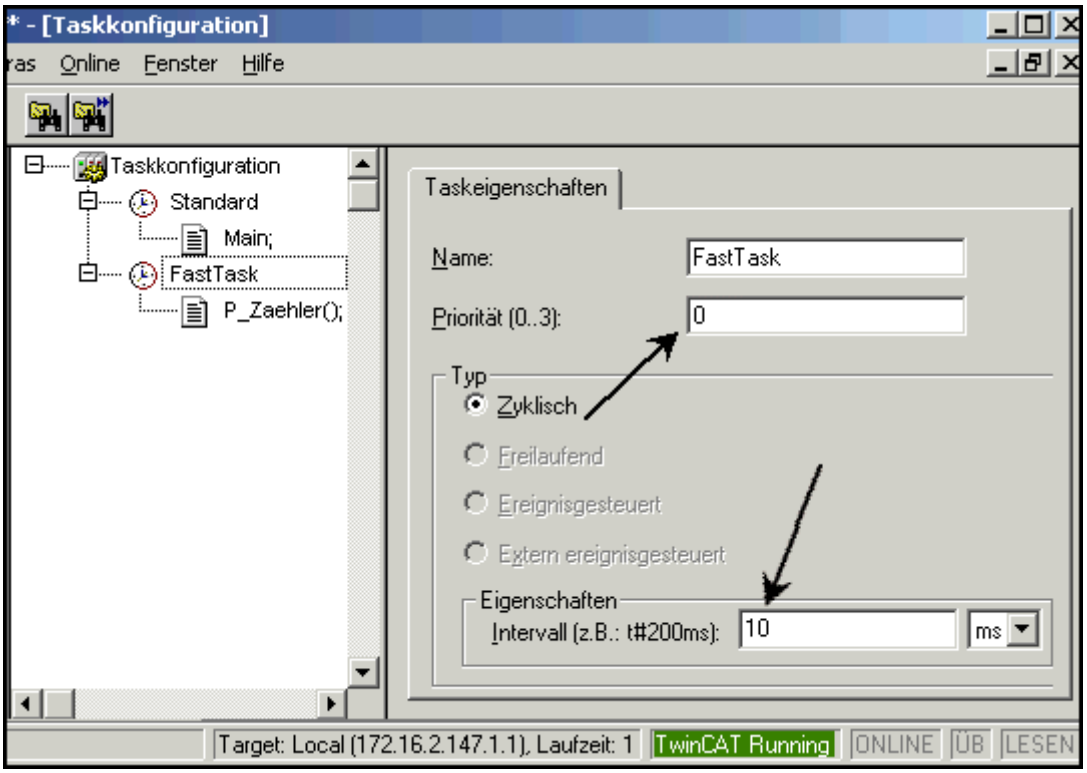
Right-click on "FastTask" and select "Append program call".



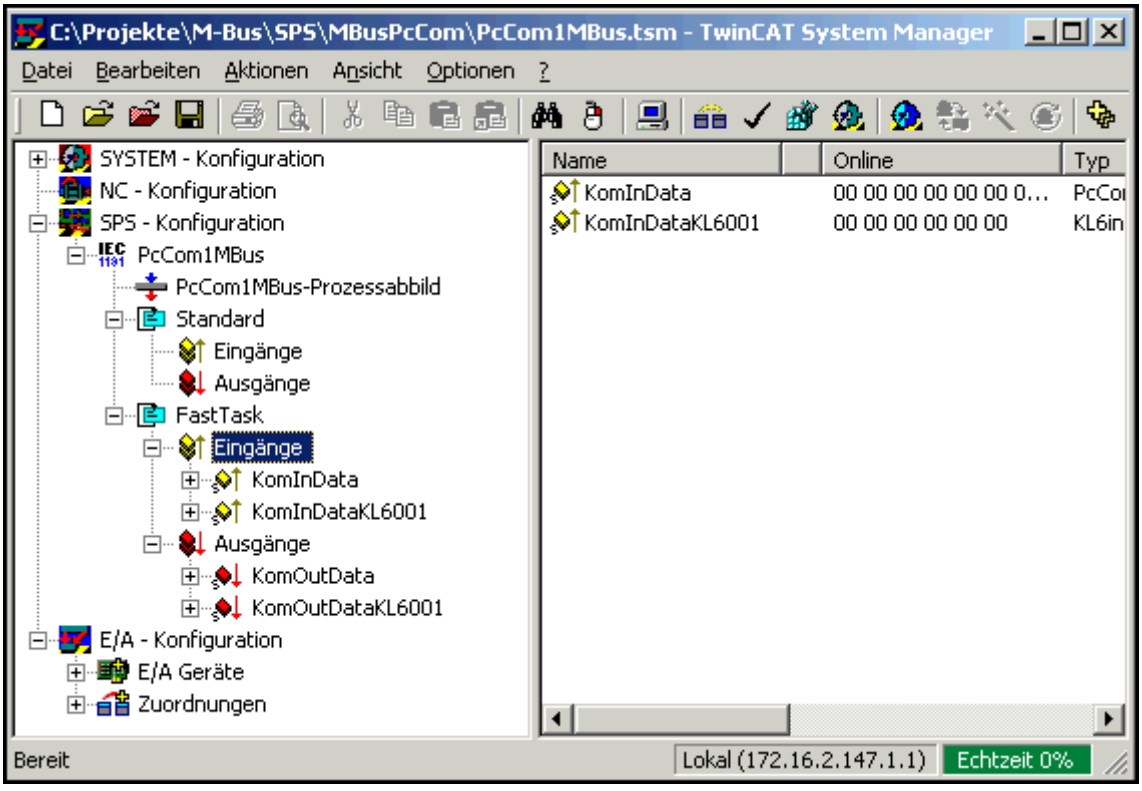
Now click on the button to right of the program call and select the program block that calls the block FB_MBUSKL6781() [▶ 29] (here "P_Zaehler").



Then set the task time (here 10 ms) and the priority (standard 1, FastTask 0).



After a new task was added in PLC Control, the project has to be recompiled and reloaded into the System Manager. The inputs and outputs of the serial interface are then inserted manually into the fast task (here "FastTask").

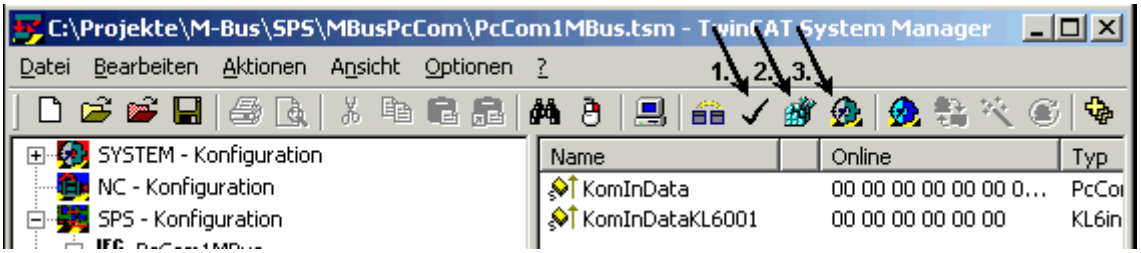


Finally you have to select the three marked buttons:

1.Check configuration

2.Actvate configuration

3.Set/Reset TwinCAT to Run Mode



Further information on task configuration [▶ 231]

8.3 Examples for PC/CX

Example	Description
https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055566347/.zip	Call of the General [▶ 30] function blocks.
https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055574795/.zip	Call of the function blocks of manufacturer ABB. [▶ 49]
https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055579019/.zip	Call of the function blocks of manufacturer Actaris [▶ 52].
https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055583243/.zip	Call of the function blocks of manufacturer Aquametro [▶ 56].
https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055587467/.zip	Call of the function blocks of manufacturer Berg [▶ 68].
https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055591691/.zip	Call of the function blocks of manufacturer Brunata [▶ 72].
https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055595915/.zip	Call of the function blocks of manufacturer Elster [▶ 80].
https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055600139/.zip	Call of the function blocks of manufacturer EMH [▶ 84].
https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055604363/.zip	Call of the function blocks of manufacturer EMU [▶ 93].
https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055608587/.zip	Call of the function blocks of manufacturer Engelmann [▶ 103].
https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055612811/.zip	Call of the function blocks of manufacturer Gossen Metrawatt [▶ 106].
https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055617035/.zip	Call of the function blocks of manufacturer GWF [▶ 109].
https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055621259/.zip	Call of the function blocks of manufacturer Hydrometer [▶ 111].
https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055625483/.zip	Call of the function blocks of manufacturer ista [▶ 123].
https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055629707/.zip	Call of the function blocks of manufacturer Janitza [▶ 134].
https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055633931/.zip	Call of the function blocks of manufacturer Kamstrup [▶ 137].
https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055638155/.zip	Call of the function blocks of manufacturer KUNDO [▶ 151].
https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055642379/.zip	Call of the function blocks of manufacturer Landis & Gyr [▶ 158].

Example	Description
https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055646603/.zip	Call of the function blocks of manufacturer Metrima [▶ 160].
https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055650827/.zip	Call of the function blocks of manufacturer NZR [▶ 165].
https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055655051/.zip	Call of the function blocks of manufacturer OPTEC [▶ 169].
https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055659275/.zip	Call of the function blocks of manufacturer Relay [▶ 172].
https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055663499/.zip	Call of the function blocks of manufacturer Saia [▶ 183].
https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055671947/.zip	Call of the function blocks of manufacturer Schlumberger [▶ 191].
https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055676171/.zip	Call of the function blocks of manufacturer Sensus [▶ 196].
https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055680395/.zip	Call of the function blocks of manufacturer Sontex [▶ 204].
https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055684619/.zip	Read 20 meters with one instance of a meter block.
https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055686027/.zip	Read 20x3 meters.
https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055570571/.zip	Demonstrating the use of secondary addressing.

8.4 Examples for BX

Example	Description
https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055569163/.zip	Call of the General [▶ 30] function blocks.
https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055576203/.zip	Call of the function blocks of manufacturer ABB. [▶ 49]
https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055580427/.zip	Call of the function blocks of manufacturer Actaris [▶ 52].
https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055584651/.zip	Call of the function blocks of manufacturer Aquametro [▶ 56].
https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055588875/.zip	Call of the function blocks of manufacturer Berg [▶ 68].
https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055593099/.zip	Call of the function blocks of manufacturer Brunata [▶ 72].
https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055597323/.zip	Call of the function blocks of manufacturer Elster [▶ 80].
https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055601547/.zip	Call of the function blocks of manufacturer EMH [▶ 84].
https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055605771/.zip	Call of the function blocks of manufacturer EMU [▶ 93].
https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055609995/.zip	Call of the function blocks of manufacturer Engelmann [▶ 103].
https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055614219/.zip	Call of the function blocks of manufacturer Gossen Metrawatt [▶ 106].

Example	Description
https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055618443/.zip	Call of the function blocks of manufacturer GWF [▶ 109].
https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055622667/.zip	Call of the function blocks of manufacturer Hydrometer [▶ 111].
https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055626891/.zip	Call of the function blocks of manufacturer ista [▶ 123].
https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055631115/.zip	Call of the function blocks of manufacturer Janitza [▶ 134].
https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055635339/.zip	Call of the function blocks of manufacturer Kamstrup [▶ 137].
https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055639563/.zip	Call of the function blocks of manufacturer KUNDO [▶ 151].
https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055645195/.zip	Call of the function blocks of manufacturer Landis & Gyr [▶ 158].
https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055648011/.zip	Call of the function blocks of manufacturer Metrima [▶ 160].
https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055652235/.zip	Call of the function blocks of manufacturer NZR [▶ 165].
https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055656459/.zip	Call of the function blocks of manufacturer OPTEC [▶ 169].
https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055660683/.zip	Call of the function blocks of manufacturer Relay [▶ 172].
https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055664907/.zip	Call of the function blocks of manufacturer Saia [▶ 183].
https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055673355/.zip	Call of the function blocks of manufacturer Schlumberger [▶ 191].
https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055677579/.zip	Call of the function blocks of manufacturer Sensus [▶ 196].
https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055681803/.zip	Call of the function blocks of manufacturer Sontex [▶ 204].

8.5 Examples for BC

BCxx00 must be set to "Large Model" in the TwinCAT PLC Control under "Project/Options/Controller Settings".

Example	Description
https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055567755/.zip	Call of the General [▶ 30] function blocks.
https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055577611/.zip	Call of the function blocks of manufacturer ABB. [▶ 49]
https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055581835/.zip	Call of the function blocks of manufacturer Actaris [▶ 52].
https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055586059/.zip	Call of the function blocks of manufacturer Aquametro [▶ 56].
https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055590283/.zip	Call of the function blocks of manufacturer Berg [▶ 68].
https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055594507/.zip	Call of the function blocks of manufacturer Brunata [▶ 72].

Example	Description
https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055598731/.zip	Call of the function blocks of manufacturer Elster [▶ 80].
https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055602955/.zip	Call of the function blocks of manufacturer EMH [▶ 84].
https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055607179/.zip	Call of the function blocks of manufacturer EMU [▶ 93].
https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055611403/.zip	Call of the function blocks of manufacturer Engelmann [▶ 103].
https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055615627/.zip	Call of the function blocks of manufacturer Gossen Metrawatt [▶ 106].
https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055619851/.zip	Call of the function blocks of manufacturer GWF [▶ 109].
https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055624075/.zip	Call of the function blocks of manufacturer Hydrometer [▶ 111].
https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055628299/.zip	Call of the function blocks of manufacturer ista [▶ 123].
https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055632523/.zip	Call of the function blocks of manufacturer Janitza [▶ 134].
https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055636747/.zip	Call of the function blocks of manufacturer Kamstrup [▶ 137].
https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055640971/.zip	Call of the function blocks of manufacturer KUNDO [▶ 151].
https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055643787/.zip	Call of the function blocks of manufacturer Landis & Gyr [▶ 158].
https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055649419/.zip	Call of the function blocks of manufacturer Metrima [▶ 160].
https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055653643/.zip	Call of the function blocks of manufacturer NZR [▶ 165].
https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055657867/.zip	Call of the function blocks of manufacturer OPTEC [▶ 169].
https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055662091/.zip	Call of the function blocks of manufacturer Relay [▶ 172].
https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055666315/.zip	Call of the function blocks of manufacturer Saia [▶ 183].
https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055674763/.zip	Call of the function blocks of manufacturer Schlumberger [▶ 191].
https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055678987/.zip	Call of the function blocks of manufacturer Sensus [▶ 196].
https://infosys.beckhoff.com/content/1033/tcplclibmbus/Resources/12055683211/.zip	Call of the function blocks of manufacturer Sontex [▶ 204].

8.6 Support and Service

Beckhoff and their partners around the world offer comprehensive support and service, making available fast and competent assistance with all questions related to Beckhoff products and system solutions.

Download finder

Our download finder contains all the files that we offer you for downloading. You will find application reports, technical documentation, technical drawings, configuration files and much more.

The downloads are available in various formats.

Beckhoff's branch offices and representatives

Please contact your Beckhoff branch office or representative for local support and service on Beckhoff products!

The addresses of Beckhoff's branch offices and representatives round the world can be found on our internet page: www.beckhoff.com

You will also find further documentation for Beckhoff components there.

Beckhoff Support

Support offers you comprehensive technical assistance, helping you not only with the application of individual Beckhoff products, but also with other, wide-ranging services:

- support
- design, programming and commissioning of complex automation systems
- and extensive training program for Beckhoff system components

Hotline:	+49 5246 963-157
e-mail:	support@beckhoff.com

Beckhoff Service

The Beckhoff Service Center supports you in all matters of after-sales service:

- on-site service
- repair service
- spare parts service
- hotline service

Hotline:	+49 5246 963-460
e-mail:	service@beckhoff.com

Beckhoff Headquarters

Beckhoff Automation GmbH & Co. KG

Huelshorstweg 20
33415 Verl
Germany

Phone:	+49 5246 963-0
e-mail:	info@beckhoff.com
web:	www.beckhoff.com

More Information:
www.beckhoff.com/tx1200

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