Function block library

Modbus_RTU_7

for PLCnext Engineer

Documentation for
PHOENIX CONTACT function blocks
PHOENIX CONTACT GmbH Co. KG
Flachsmarktstrasse 8
D-32825 Blomberg, Germany

This documentation is available in English only.

Table of Contents

1 Installation hint

If you did not specify a different directory during **library** installation all data in the MSI file will be unpacked to

c:\Users\Public\Documents\Phoenix Contact Libraries\PLCnext Engineer (former: PC Worx Engineer)

Please copy the library data to your PLCnext Engineer (former: PC Worx Engineer) working library directory.

If you did not specify a different directory during **PLCnext Engineer** installation the default PLCnext Engineer working library directory is

c:\Users\Public\Documents\PLCnext Engineer\Libraries (former: PC Worx Engineer\Libraries)

2 General information

Modbus is a communication protocol used for serial communication. It is a master/slave protocol. Only one master is connected to the bus at a time. In addition, one or more slaves (247, maximum) are connected to the same serial bus.

Modbus communication is always initiated by the master. The master sends a request, then the slave specified in the request responds. It is possible to send a request to all slaves (broadcast). The slaves will never transmit data without receiving a request from the master. In addition, the slaves do not communicate with each other. The master initiates only one Modbus transaction at a time.

There are four data types stored in a Modbus device memory: discrete inputs (bits), coils (bits), holding registers (16-bit registers), and input registers (16-bit registers).

3 Change notes

Library version	Library build	PLCnext Engineer version	Change notes	Supported PLCs
7	20191002	2019.0 LTS 2019.3 2019.6 2019.9	Adapted to 2019.9	AXC F 2152 (2404267)
6	20190723	2019.0 LTS 2019.3 2019.6	Adapted to 2019.6	AXC F 2152 (2404267)
5	20190701	2019.0 LTS 2019.3	MB_RTU_Master: • Improved handshakes between master and serial driver. MB_RTU_FC23: • Runtime error: "Error while accessing indirect variable address" MB_RTU_FC (all FCs): • Operating FC stops when other FCs are deactivated MB_AXL_RS_UNI_SND: • Bugfix: "Communication error after FB reset during send or receive phase." • Bugfix: "Inter-character time bigger than Modbus specification allows. Communication errors with slow CPU cycle-times or high bussystem cycle-times." MB_AXL_RS_UNI_RCV: • Bugfix: "Communication error after FB reset during send or receive phase."	AXC F 2152 (2404267)
4	20190226	2019.0 LTS	Supports “Allow extended identifiers” = ON	AXC F 2152 (2404267)
4	20190219	2019.0 LTS	Modbus_RTU_4: • Adapted to PLCnext Engineer 2019.0 LTS	AXC F 2152 (2404267)

3	20180928	7.2.3	Adapted to PLCnext Engineer 7.3	AXC F 1050 (2404701) AXC F 2152 (2404267)
2	20180508	7.2.2	Converted from PC Worx 6 Modbus_RTU_1 library. New functionalities: • New udtDiag output at all function blocks for better diagnostics. • Master and Slave function blocks with integrated driver are no longer encrypted for better diagnostics. MB_RTU_Master_2: • "Array out of index" error message with enabled xAuto_CRC input is corrected. • "xNDR stays true after function block is deactivated during send request" error is fixed. • "Execution error of following FCs, if previous FC is in error" error is fixed. MB_RTU_FC1,2,3,4,23: • New diagnostic for"broadcast on reading FBs not possible". MB_RTU_FC2_2: • "Reading wrong count of bits" error is fixed. MB_RTU_FC23_2: • "Reading one register less than requested" error is fixed MB_RTU_FC*_2 (all FCs): • Correction in polltimer execution interval • "wDiagCode goes to 16#0000 after xDone" error is fixed • "Function code invalid" diag code is changed from 16#C110 to 16#C100	AXC F 1050 (2404701) AXC F 2152 (2404267)
1	-	-	Phoenix Contact internal version	-

New version number: Functional changes of at least one function block, incompatibilities (e.g. change of library format)

New build number: No functional changes, but changes in the MSI file (e.g. documentation update, additional examples)

4 Function blocks

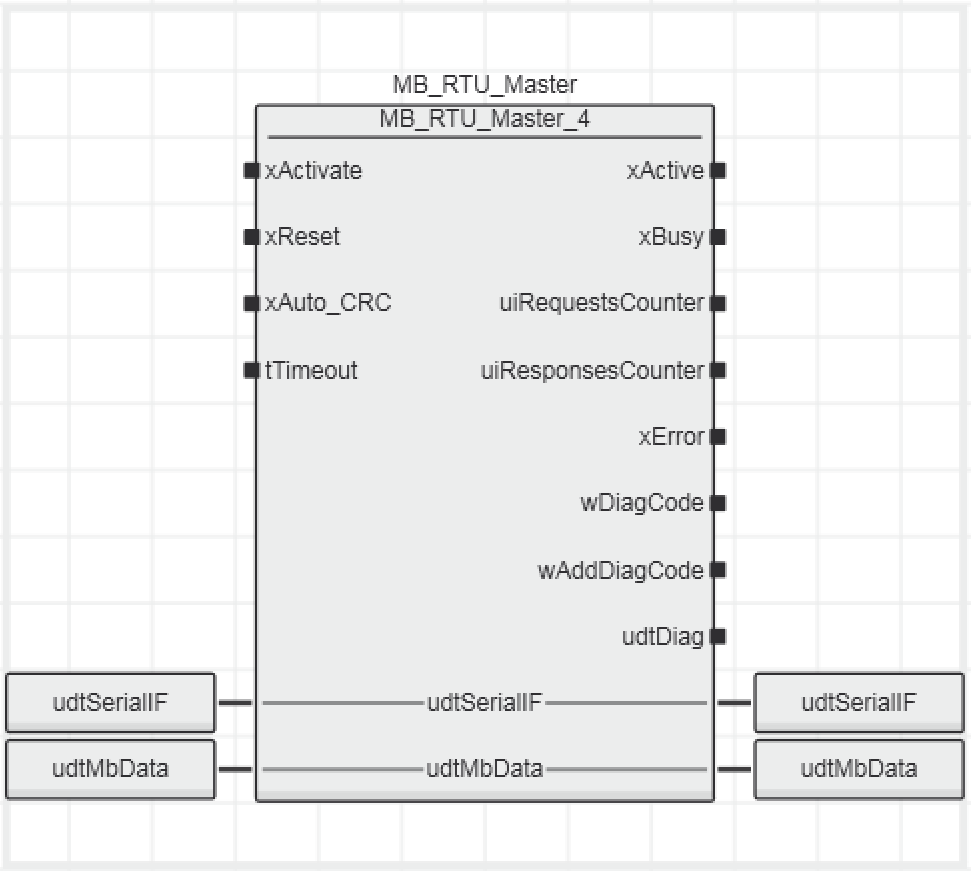
Function block	Description	Version	Supported articles	License
MB_RTU_Master	The function block enables communication as master with Modbus RTU devices.	4	-	none
MB_RTU_Slave	The function block enables communication as slave with Modbus RTU devices.	4	-	none
MB_RTU_FC1	This function block reads the status of discrete outputs from a Modbus slave.	4	-	none
MB_RTU_FC2	This function block reads discrete inputs from a Modbus slave.	4	-	none
MB_RTU_FC3	This function block reads holding registers from a Modbus slave.	4	-	none
MB_RTU_FC4	This function block reads input registers from a Modbus slave.	4	-	none
MB_RTU_FC5	This function block writes a single output bit of a Modbus slave.	4	-	none
MB_RTU_FC6	This function block writes a single holding register of a Modbus slave.	4	-	none
MB_RTU_FC15	This function block writes multiple output bits of a Modbus slave.	4	-	none
MB_RTU_FC16	This function block writes multiple holding registers of a Modbus slave.	4	-	none
MB_RTU_FC23	This function block writes or reads multiple holding registers of a Modbus slave.	4	-	none
MB_RTU_DiagInfo_EN	This optional function block displays diagnostic messages of the Modbus master as clear text in English.	3	-	none
MB_AXL_RS_UNI_REC	This block runs the receiving operations via the AXL F RS UNI 1H (2688666) module.	4	AXL F RS UNI 1H (2688666)	none
MB_AXL_RS_UNI_SND	This block runs the sending operations via the AXL F RS UNI 1H (2688666) module.	3	AXL F RS UNI 1H (2688666)	none

5 MB_RTU_Master

This block controls the requests of the FC blocks and sends the Modbus request to the Modbus network via the connected serial interface. The response is analyzed in this block and forwarded to the requesting block.

Diagnostic information on the Modbus requests is displayed at the respective FC block.

5.1 Function block call



5.2 Input parameters

Name	Type	Description
xActivate	BOOL	Rising edge: Activates the function block. FALSE: Deactivates the function block.
xReset	BOOL	Rising edge: Resets the function block.
tTimeout	TIME	The block monitors the communication to the serial driver block. The default value for times less than 10 ms is set to 1 second. The input is copied by xActivate or xReset if there is a rising edge.
xAuto_CRC	BOOL	TRUE: The CRC is calculated by the module (only AXL F RS UNI). FALSE:The CRC is calculated in the block.

Important:

The function of the Auto CRC calculation from the AXL F RS UNI module with the HW/FW 01/1.00 can be used up to a data length of 17 bytes in the Modbus protocol!

5.3 Output parameters

Name	Type	Description
xActive	BOOL	TRUE: Function block is active. FALSE: Function block is not active.
xBusy	BOOL	TRUE: The block is busy with the service execution.
xError	BOOL	TRUE: An error has occurred. For details refer to wDiagCode and wAddDiagCode.
wDiagCode	WORD	Diagnostic code. Refer to diagnostics table.
wAddDiagCode	WORD	Additional diagnostic code. Refer to diagnostics table.
uiRequestsCounter	UINT	Shows the number of requests transmitted.
uiResponsesCounter	UINT	Shows the number of responses received.
udtDiag	MB_UDT_RTU_MASTER_DIAG	Structure with internal variables for Diagnostic

5.4 Inout parameters

Name	Type	Description
udtSerialIF	MB2_AXL_RSUNI2_UDT_IF	The block communicates via this structure with the Modbus driver block.
udtMBData	udtModbus2_Data	The block communicates via this structure with the FC blocks.

5.5 Diagnosis

wDiagCode	wAddDiagCode	Description
16#0000		Block is not activated.
16#8000		Block is active and operating without errors.
16#8100		HW Reset phase to delete serial driver error

The block shows the diagnostics of the serial driver blocks. (Section: Diagnostics MB_AXL_RS_UNI_REC and MB_AXL_RS_UNI_SND).

5.6 Supported Modbus function codes

	Function code	Description
01	Read coils	This function block reads digital outputs.
02	Read Discrete Inputs	This function block reads digital inputs.
03	Read Holding Registers	This function block reads holding registers.
04	Read Input Registers	This function block reads input registers.
05	Write Single Coil	This function block writes a bit.
06	Write Single Register	This function block writes a register.
15	Write Multiple Coils	This function block writes multiple bits.
16	Write Multiple Registers	This function block writes multiple registers.
23	Read/Write Multiple Registers	This function block reads/writes multiple registers.

Using this block, the request is configured and sent. Input parameters are specified for each function code.

All FC blocks that communicate with the master function block also have to be connected via the same parameter udtMBData.

It is possible to use more than one instance in the same program.

Note regarding FC function blocks:

The xDone and xError outputs indicate completion of send-receive process. Output stays true till the xSendRequest input is set to false.

In case of polling, the response result (xDone and xError) are set for one cycle, then the next request will be executed.

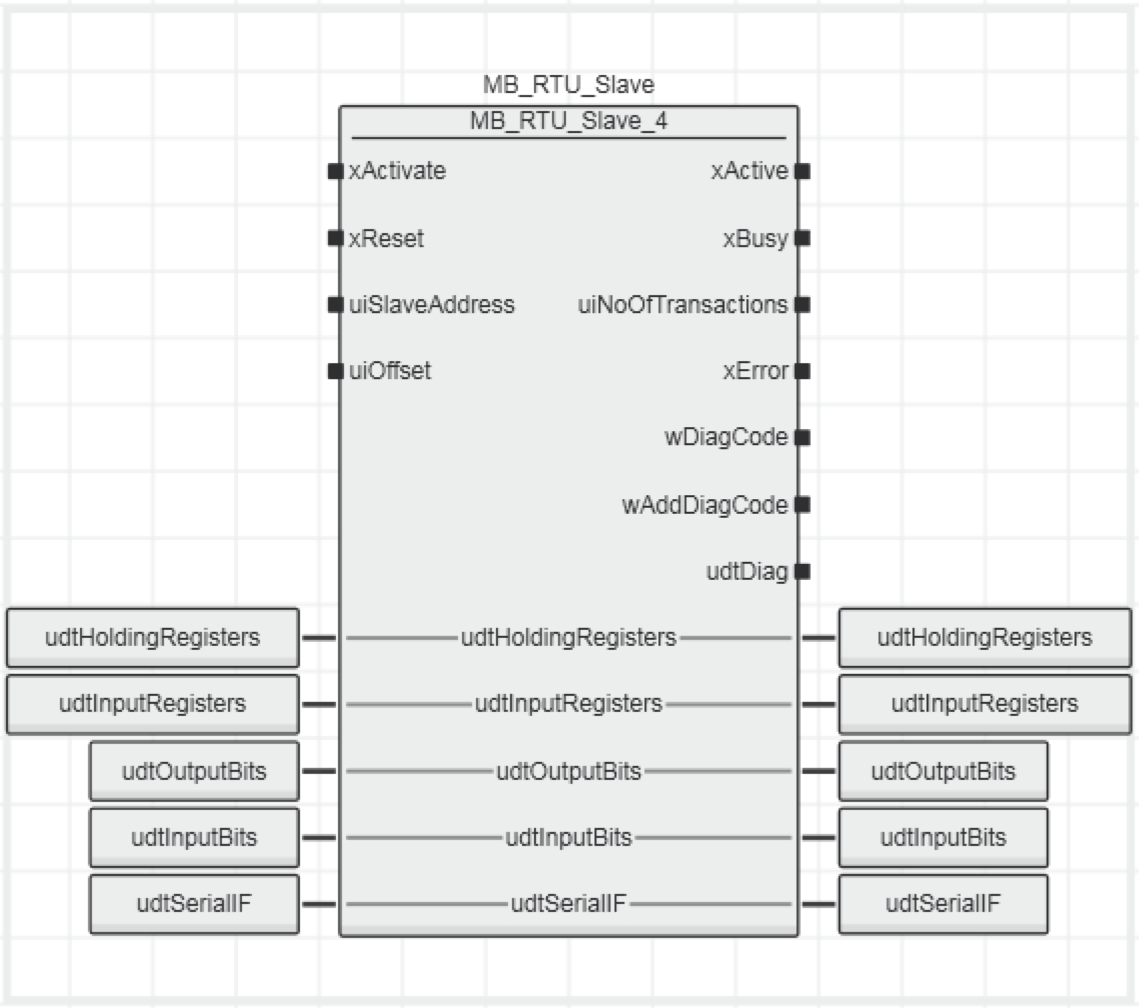
If more than one request is sent at the same time from different instances, they will be executed one after the other.

6 MB_RTU_Slave

This block configures the controller as the Modbus slave. The slave contains data fields that can be retrieved in a Modbus network. The function codes 1, 2, 3, 4, 5, 6, 15, and 16 are supported. A slave can be connected to only one serial interface.

The function block is released for baud rates between 9600 and 38400 baud.

6.1 Function block call



6.2 Input parameters

Name	Type	Description
xActivate	BOOL	Block activation (TRUE = Active).
xReset	BOOL	The input resets the block. All connected FC blocks are reset as well.
uiSlaveAddress	UINT	The input specifies the address of the slave to be communicated with (1 to 255).
uiOffset	UINT	The start address is increased by this value. If the input uiOffset has the value 2000, then the register with address 3 in the request will be addressed with the address 2003 (2000 + 3).

Important:
The function of the Auto CRC calculation from the AXL F RS UNI module with the HW/FW 01/1.00 can be used up

to a data length of 17 bytes in the Modbus protocol!

6.3 Output parameters

Name	Type	Description
xActive	BOOL	TRUE: Function block is active. FALSE: Function block is not active.
xBusy	BOOL	TRUE: The block is busy with the service execution.
uiNoOfTransactions	UINT	Number of processed requests
xError	BOOL	TRUE: An error has occurred. For details refer to wDiagCode and wAddDiagCode.
wDiagCode	WORD	Diagnostic code. Refer to diagnostics table.
wAddDiagCode	WORD	Additional diagnostic code. Refer to diagnostics table.
udtDiag	MB_UDT_RTU_SLAVE_DIAG	Structure with internal variables for Diagnostic

6.4 Inout parameters

Name	Type	Description
udtHoldingRegisters	MB_RTU_w_0_1999	Array with 2000 words representing the holding registers. The address range is 0-1999. Function codes: 3,6 and 16
udtInputRegisters	MB_RTU_w_2000_2999	Array with 1000 words representing the input registers. The address range is 2000-2999. Function code: 4
udtOutputBits	MB_RTU_x_3000_3999	Array of 1000 bits representing the digital outputs. The address range is 3000-3999. Function codes: 1,5 and 15
udtInputBits	MB_RTU_x_4000_4999	Array of 1000 bits representing digital inputs. The address range is 4000-4999. Function code: 2
udtSerialIF	MB2_AXL_RSUNI2_UDT_IF	The block communicates via this structure with the Modbus driver block.

6.5 Diagnosis

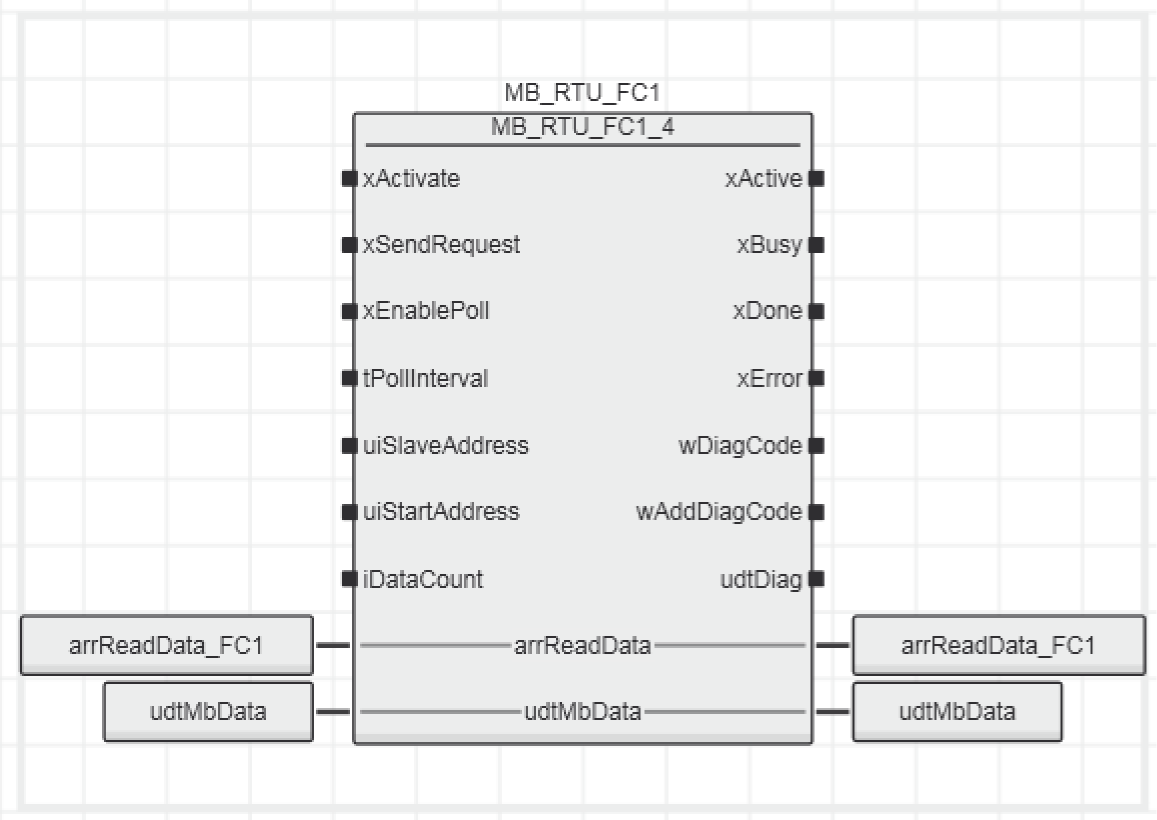
wDiagCode	wAddDiagCode	Description
16#0000	16#0000	Block is not activated.
16#8000	16#0000	Block is active and operating without errors.

The block shows the diagnostics of the serial driver blocks. (Section: Diagnostics MB_AXL_RS_UNI_REC and MB_AXL_RS_UNI_SND).

7 MB_RTU_FC1

This function block reads the status of discrete outputs from a Modbus slave.

7.1 Function block call



7.2 Input parameters

Name	Type	Description
xActivate	BOOL	Block activation (TRUE = Active).
xSendRequest	BOOL	A send request to the master block is activated with a rising edge. A falling edge deletes current Modbus errors and resets the block outputs.
xEnablePoll	BOOL	Cyclical polling is started with a rising edge. A falling edge deactivates the polling. Input xSendRequest triggers an additional request and should be deactivated during poll mode. Note that the outputs xDone and xError are only one cycle true.
tPollIntervall	TIME	If xEnablePoll is activated, then transmission is cyclical in the time interval of the specified value.
uiSlaveAddress	UINT	The input specifies the address of the slave to be communicated with (1 to 255).
uiStartAddress	UINT	The input specifies the start address of the bit to be read on the slave.
iDataCount	INT	The input specifies the number of bits to be read on the slave (1 to 2000).

7.3 Output parameters

Name	Type	Description
xActive	BOOL	TRUE: Function block is active. FALSE: Function block is not active.
xBusy	BOOL	TRUE: The block is busy with the service execution.
xDone	BOOL	Request is sent and response from slave is successfully received.
xError	BOOL	TRUE: An error has occurred. For details refer to wDiagCode and wAddDiagCode.
wDiagCode	WORD	Diagnostic code. Refer to diagnostics table.
wAddDiagCode	WORD	Additional diagnostic code. Refer to diagnostics table.
udtDiag	MB_UDT_RTU_FC_DIAG	Structure with internal variables for Diagnostic

7.4 Inout parameters

Name	Type	Description
arrReadData	arrModbus2_X_1_2000	The parameter contains the requested Modbus data.
udtMBData	udtModbus2_Data	The block communicates via this structure with the FC blocks.

7.5 Diagnosis

The diagnostics contains diagnostic codes on the FC blocks of the library. Modbus errors are indicated at the respective FC block and need to be reset there. Thus the communication in a Modbus network is not disturbed by an error in a request to a slave. An error at the FC block is deleted by a reset of the send input or by renewed activation of the block.

Modbus exception codes are sent by the respective slave and contain messages specific for Modbus.

wDiagCode	wAddDiagCode	Description
16#0000		Block is not activated.
16#8000		Block is active and operating without errors.
16#8300		Block executes a service.
16#C100		Error during configuration (displayed on the FC block).
	16#0001	Slave address is outside the valid range.
	16#0002	Number of the requested data amount invalid (iDataCount).
	16#0003	Function code invalid.
	16#0004	Broadcast not possible. FC supports reading function.
16#C110		Error in the Modbus (displayed on the FC block).
	16#0001	Timeout on master block.
	16#0002	Checksum (CRC) invalid.
16#C120		Modbus Exception Code (shown at the FC block).
	16#0001	Exception Code 1 (Illegal Function).
	16#0002	Exception Code 2 (Illegal Data Address).
	16#0003	Exception Code 3 (Illegal Data Value).
	16#0004	Exception Code 4 (Server Device Failure).
	16#0005	Exception Code 5 (Acknowledge).

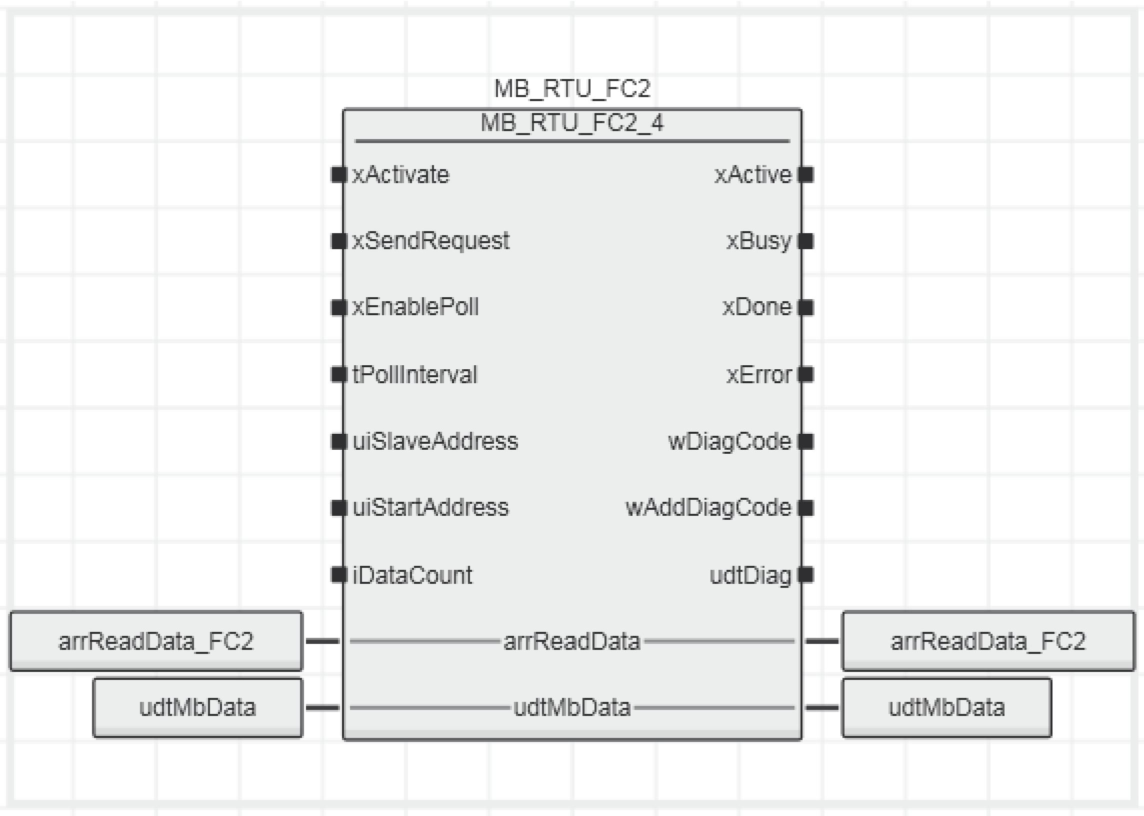
	16#0006	Exception Code 6 (Server Device Busy).
	16#0008	Exception Code 8 (Memory Parity Error).
	16#000A	Exception Code 10 (Gateway Path Unavailable).
	16#000B	Exception Code 11 (Gateway Target Device Failed To Respond).

These diagnostic codes, as well as xError, are reset by a falling edge of xActivate or xSendRequest on an FC block.
The block displays the diagnosis of the master block and thus also the diagnosis of the serial blocks. These errors must be reset by deactivating the affected blocks. For errorcodes 16#C010 - 16#C060 refer to serial block diagnostic.

8 MB_RTU_FC2

This function block reads discrete inputs from a Modbus slave.

8.1 Function block call



8.2 Input parameters

Name	Type	Description
xActivate	BOOL	Block activation (TRUE = Active).
xSendRequest	BOOL	A send request to the master block is activated with a rising edge. A falling edge deletes current Modbus errors and resets the block outputs.
xEnablePoll	BOOL	Cyclical polling is started with a rising edge. A falling edge deactivates the polling. Input xSendRequest triggers an additional request and should be deactivated during poll mode. Note that the outputs xDone and xError are only one cycle true.
tPollIntervall	TIME	If xEnablePoll is activated, then transmission is cyclical in the time interval of the specified value.
uiSlaveAddress	UINT	The input specifies the address of the slave to be communicated with (1 to 255).
uiStartAddress	UINT	The input specifies the start address of the bit to be read on the slave.
iDataCount	INT	The input specifies the number of bits to be read on the slave (1 to 2000).

8.3 Output parameters

Name	Type	Description
xActive	BOOL	TRUE: Function block is active. FALSE: Function block is not active.
xBusy	BOOL	TRUE: The block is busy with the service execution.
xDone	BOOL	Request is sent and response from slave is successfully received.
xError	BOOL	TRUE: An error has occurred. For details refer to wDiagCode and wAddDiagCode.
wDiagCode	WORD	Diagnostic code. Refer to diagnostics table.
wAddDiagCode	WORD	Additional diagnostic code. Refer to diagnostics table.
udtDiag	MB_UDT_RTU_FC_DIAG	Structure with internal variables for Diagnostic

8.4 Inout parameters

Name	Type	Description
arrReadData	arrModbus2_X_1_2000	The parameter contains the requested Modbus data.
udtMBData	udtModbus2_Data	The block communicates via this structure with the FC blocks.

8.5 Diagnosis

The diagnostics contains diagnostic codes on the FC blocks of the library. Modbus errors are indicated at the respective FC block and need to be reset there. Thus the communication in a Modbus network is not disturbed by an error in a request to a slave. An error at the FC block is deleted by a reset of the send input or by renewed activation of the block.

Modbus exception codes are sent by the respective slave and contain messages specific for Modbus.

wDiagCode	wAddDiagCode	Description
16#0000		Block is not activated.
16#8000		Block is active and operating without errors.
16#8300		Block executes a service.
16#C100		Error during configuration (displayed on the FC block).
	16#0001	Slave address is outside the valid range.
	16#0002	Number of the requested data amount invalid (iDataCount).
	16#0003	Function code invalid.
	16#0004	Broadcast not possible. FC supports reading function.
16#C110		Error in the Modbus (displayed on the FC block).
	16#0001	Timeout on master block.
	16#0002	Checksum (CRC) invalid.
16#C120		Modbus Exception Code (shown at the FC block).
	16#0001	Exception Code 1 (Illegal Function).
	16#0002	Exception Code 2 (Illegal Data Address).
	16#0003	Exception Code 3 (Illegal Data Value).
	16#0004	Exception Code 4 (Server Device Failure).
	16#0005	Exception Code 5 (Acknowledge).

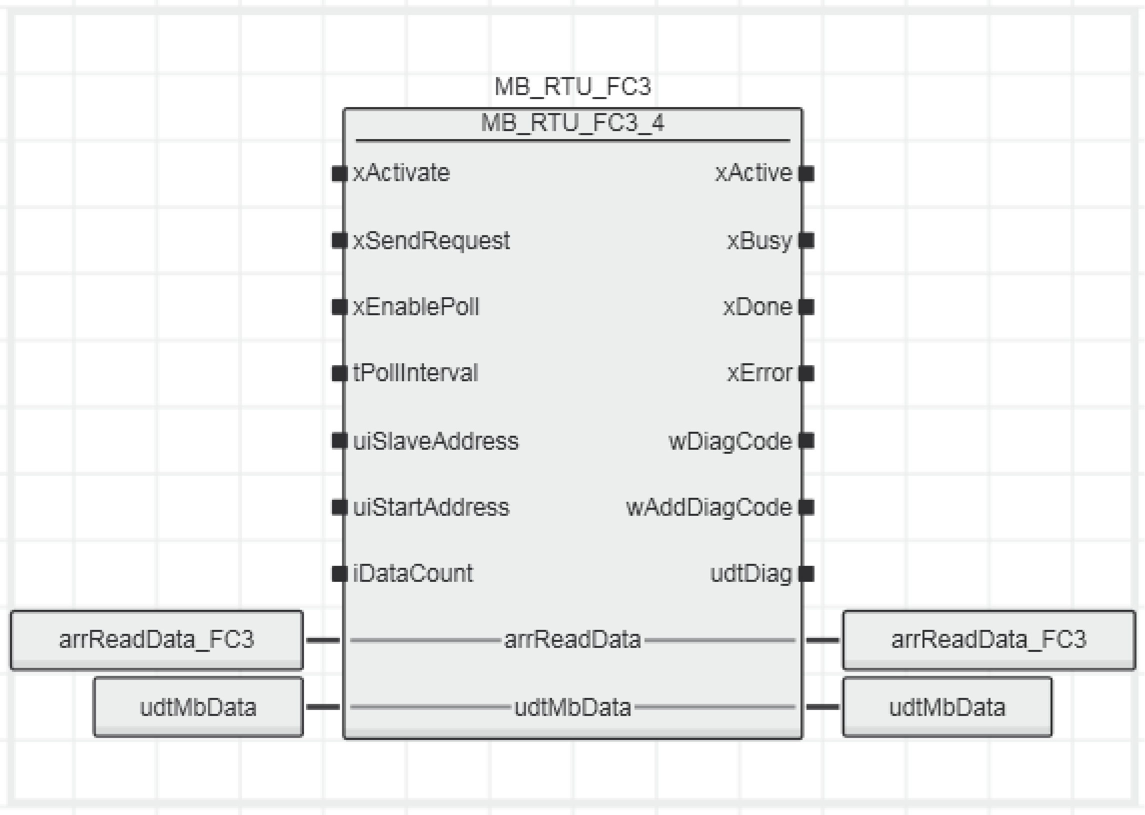
	16#0006	Exception Code 6 (Server Device Busy).
	16#0008	Exception Code 8 (Memory Parity Error).
	16#000A	Exception Code 10 (Gateway Path Unavailable).
	16#000B	Exception Code 11 (Gateway Target Device Failed To Respond).

These diagnostic codes, as well as xError, are reset by a falling edge of xActivate or xSendRequest on an FC block.
The block displays the diagnosis of the master block and thus also the diagnosis of the serial blocks. These errors must be reset by deactivating the affected blocks. For errorcodes 16#C010 - 16#C060 refer to serial block diagnostic.

9 MB_RTU_FC3

This function block reads holding registers from a Modbus slave.

9.1 Function block call



9.2 Input parameters

Name	Type	Description
xActivate	BOOL	Block activation (TRUE = Active).
xSendRequest	BOOL	A send request to the master block is activated with a rising edge. A falling edge deletes current Modbus errors and resets the block outputs.
xEnablePoll	BOOL	Cyclical polling is started with a rising edge. A falling edge deactivates the polling. Input xSendRequest triggers an additional request and should be deactivated during poll mode. Note that the outputs xDone and xError are only one cycle true.
tPollIntervall	TIME	If xEnablePoll is activated, then transmission is cyclical in the time interval of the specified value.
uiSlaveAddress	UINT	The input specifies the address of the slave to be communicated with (1 to 255).
uiStartAddress	UINT	The input specifies the start address of the bit to be read on the slave.
iDataCount	INT	The input specifies the number of bits to be read on the slave (1 to 2000).

9.3 Output parameters

Name	Type	Description
xActive	BOOL	TRUE: Function block is active. FALSE: Function block is not active.
xBusy	BOOL	TRUE: The block is busy with the service execution.
xDone	BOOL	Request is sent and response from slave is successfully received.
xError	BOOL	TRUE: An error has occurred. For details refer to wDiagCode and wAddDiagCode.
wDiagCode	WORD	Diagnostic code. Refer to diagnostics table.
wAddDiagCode	WORD	Additional diagnostic code. Refer to diagnostics table.
udtDiag	MB_UDT_RTU_FC_DIAG	Structure with internal variables for Diagnostic

9.4 Inout parameters

Name	Type	Description
arrReadData	arrModbus2_W_1_125	The parameter contains the requested Modbus data.
udtMBData	udtModbus2_Data	The block communicates via this structure with the FC blocks.

9.5 Diagnosis

The diagnostics contains diagnostic codes on the FC blocks of the library. Modbus errors are indicated at the respective FC block and need to be reset there. Thus the communication in a Modbus network is not disturbed by an error in a request to a slave. An error at the FC block is deleted by a reset of the send input or by renewed activation of the block.

Modbus exception codes are sent by the respective slave and contain messages specific for Modbus.

wDiagCode	wAddDiagCode	Description
16#0000		Block is not activated.
16#8000		Block is active and operating without errors.
16#8300		Block executes a service.
16#C100		Error during configuration (displayed on the FC block).
	16#0001	Slave address is outside the valid range.
	16#0002	Number of the requested data amount invalid (iDataCount).
	16#0003	Function code invalid.
	16#0004	Broadcast not possible. FC supports reading function.
16#C110		Error in the Modbus (displayed on the FC block).
	16#0001	Timeout on master block.
	16#0002	Checksum (CRC) invalid.
16#C120		Modbus Exception Code (shown at the FC block).
	16#0001	Exception Code 1 (Illegal Function).
	16#0002	Exception Code 2 (Illegal Data Address).
	16#0003	Exception Code 3 (Illegal Data Value).
	16#0004	Exception Code 4 (Server Device Failure).
	16#0005	Exception Code 5 (Acknowledge).

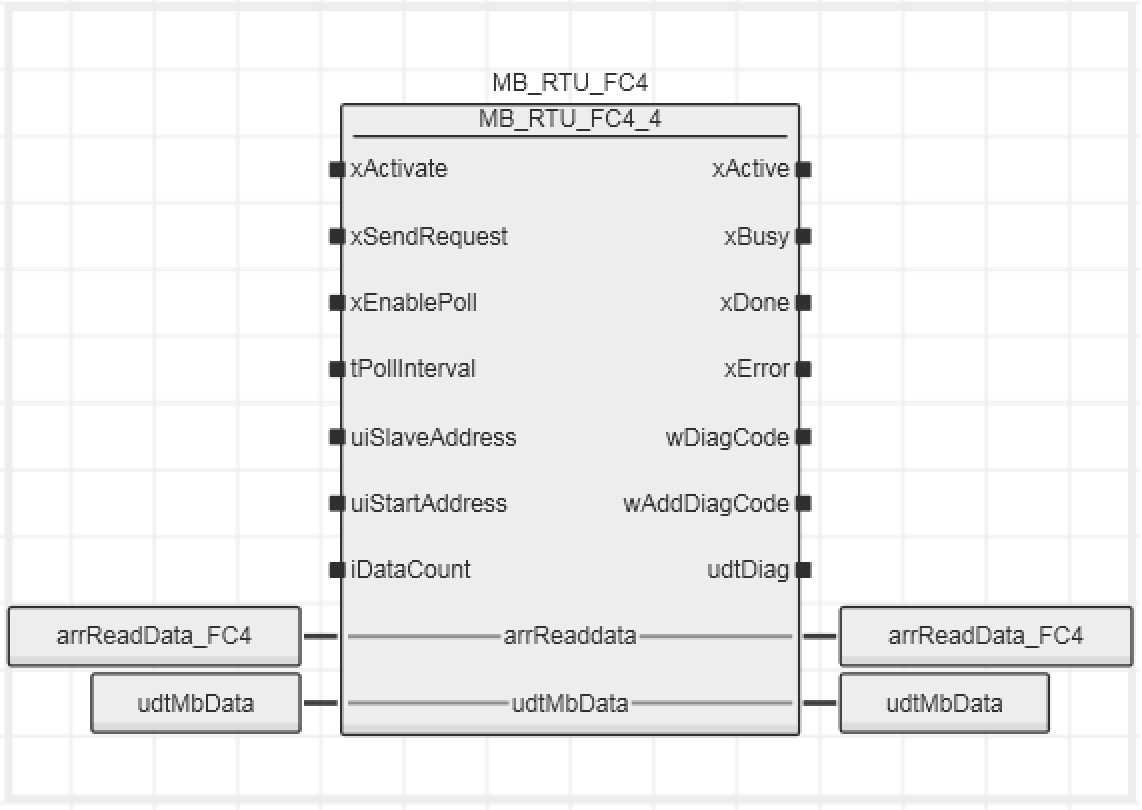
	16#0006	Exception Code 6 (Server Device Busy).
	16#0008	Exception Code 8 (Memory Parity Error).
	16#000A	Exception Code 10 (Gateway Path Unavailable).
	16#000B	Exception Code 11 (Gateway Target Device Failed To Respond).

These diagnostic codes, as well as xError, are reset by a falling edge of xActivate or xSendRequest on an FC block.
The block displays the diagnosis of the master block and thus also the diagnosis of the serial blocks. These errors must be reset by deactivating the affected blocks. For errorcodes 16#C010 - 16#C060 refer to serial block diagnostic.

10 MB_RTU_FC4

This function block reads input registers from a Modbus slave.

10.1 Function block call



10.2 Input parameters

Name	Type	Description
xActivate	BOOL	Block activation (TRUE = Active).
xSendRequest	BOOL	A send request to the master block is activated with a rising edge. A falling edge deletes current Modbus errors and resets the block outputs.
xEnablePoll	BOOL	Cyclical polling is started with a rising edge. A falling edge deactivates the polling. Input xSendRequest triggers an additional request and should be deactivated during poll mode. Note that the outputs xDone and xError are only one cycle true.
tPollIntervall	TIME	If xEnablePoll is activated, then transmission is cyclical in the time interval of the specified value.
uiSlaveAddress	UINT	The input specifies the address of the slave to be communicated with (1 to 255).
uiStartAddress	UINT	The input specifies the start address of the bit to be read on the slave.
iDataCount	INT	The input specifies the number of bits to be read on the slave (1 to 2000).

10.3 Output parameters

Name	Type	Description
xActive	BOOL	TRUE: Function block is active. FALSE: Function block is not active.
xBusy	BOOL	TRUE: The block is busy with the service execution.
xDone	BOOL	Request is sent and response from slave is successfully received.
xError	BOOL	TRUE: An error has occurred. For details refer to wDiagCode and wAddDiagCode.
wDiagCode	WORD	Diagnostic code. Refer to diagnostics table.
wAddDiagCode	WORD	Additional diagnostic code. Refer to diagnostics table.
udtDiag	MB_UDT_RTU_FC_DIAG	Structure with internal variables for Diagnostic

10.4 Inout parameters

Name	Type	Description
arrReadData	arrModbus2_W_1_125	The parameter contains the requested Modbus data.
udtMBData	udtModbus2_Data	The block communicates via this structure with the FC blocks.

10.5 Diagnosis

The diagnostics contains diagnostic codes on the FC blocks of the library. Modbus errors are indicated at the respective FC block and need to be reset there. Thus the communication in a Modbus network is not disturbed by an error in a request to a slave. An error at the FC block is deleted by a reset of the send input or by renewed activation of the block.

Modbus exception codes are sent by the respective slave and contain messages specific for Modbus.

wDiagCode	wAddDiagCode	Description
16#0000		Block is not activated.
16#8000		Block is active and operating without errors.
16#8300		Block executes a service.
16#C100		Error during configuration (displayed on the FC block).
	16#0001	Slave address is outside the valid range.
	16#0002	Number of the requested data amount invalid (iDataCount).
	16#0003	Function code invalid.
	16#0004	Broadcast not possible. FC supports reading function.
16#C110		Error in the Modbus (displayed on the FC block).
	16#0001	Timeout on master block.
	16#0002	Checksum (CRC) invalid.
16#C120		Modbus Exception Code (shown at the FC block).
	16#0001	Exception Code 1 (Illegal Function).
	16#0002	Exception Code 2 (Illegal Data Address).
	16#0003	Exception Code 3 (Illegal Data Value).
	16#0004	Exception Code 4 (Server Device Failure).
	16#0005	Exception Code 5 (Acknowledge).

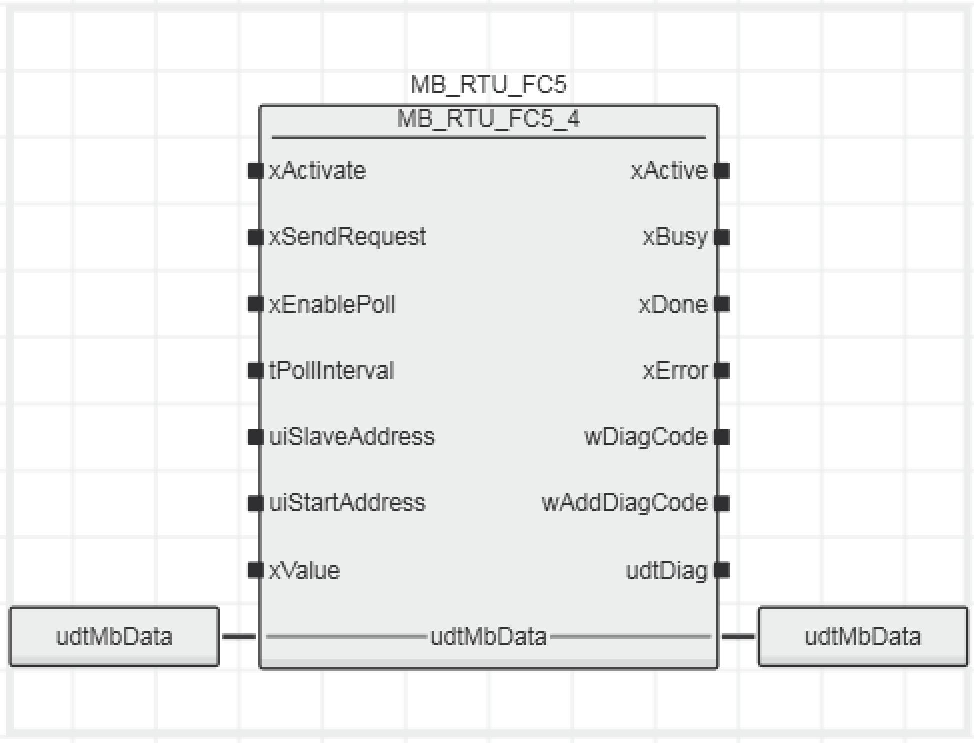
	16#0006	Exception Code 6 (Server Device Busy).
	16#0008	Exception Code 8 (Memory Parity Error).
	16#000A	Exception Code 10 (Gateway Path Unavailable).
	16#000B	Exception Code 11 (Gateway Target Device Failed To Respond).

These diagnostic codes, as well as xError, are reset by a falling edge of xActivate or xSendRequest on an FC block.
The block displays the diagnosis of the master block and thus also the diagnosis of the serial blocks. These errors must be reset by deactivating the affected blocks. For errorcodes 16#C010 - 16#C060 refer to serial block diagnostic.

11 MB_RTU_FC5

This function block writes a single output bit of a Modbus slave.

11.1 Function block call



11.2 Input parameters

Name	Type	Description
xActivate	BOOL	Block activation (TRUE = Active).
xSendRequest	BOOL	A send request to the master block is activated with a rising edge. A falling edge deletes current Modbus errors and resets the block outputs.
xEnablePoll	BOOL	Cyclical polling is started with a rising edge. A falling edge deactivates the polling. Input xSendRequest triggers an additional request and should be deactivated during poll mode. Note that the outputs xDone and xError are only one cycle true.
tPollIntervall	TIME	If xEnablePoll is activated, then transmission is cyclical in the time interval of the specified value.
uiSlaveAddress	UINT	The input specifies the address of the slave to be communicated with (1 to 255).
uiStartAddress	UINT	The input specifies the start address of the bit to be read on the slave.
xValue	BOOL	The status of the input is written in the memory to be written.

11.3 Output parameters

Name	Type	Description
xActive	BOOL	TRUE: Function block is active. FALSE: Function block is not active.
xBusy	BOOL	TRUE: The block is busy with the service execution.
xDone	BOOL	Request is sent and response from slave is successfully received.
xError	BOOL	TRUE: An error has occurred. For details refer to wDiagCode and wAddDiagCode.
wDiagCode	WORD	Diagnostic code. Refer to diagnostics table.
wAddDiagCode	WORD	Additional diagnostic code. Refer to diagnostics table.
udtDiag	MB_UDT_RTU_FC_DIAG	Structure with internal variables for Diagnostic

11.4 Inout parameters

Name	Type	Description
udtMBData	udtModbus2_Data	The block communicates via this structure with the FC blocks.

11.5 Diagnosis

The diagnostics contains diagnostic codes on the FC blocks of the library. Modbus errors are indicated at the respective FC block and need to be reset there. Thus the communication in a Modbus network is not disturbed by an error in a request to a slave. An error at the FC block is deleted by a reset of the send input or by renewed activation of the block.

Modbus exception codes are sent by the respective slave and contain messages specific for Modbus.

wDiagCode	wAddDiagCode	Description
16#0000		Block is not activated.
16#8000		Block is active and operating without errors.
16#8300		Block executes a service.
16#C100		Error during configuration (displayed on the FC block).
	16#0001	Slave address is outside the valid range.
	16#0002	Number of the requested data amount invalid (iDataCount).
	16#0003	Function code invalid.
	16#0004	Broadcast not possible. FC supports reading function.
16#C110		Error in the Modbus (displayed on the FC block).
	16#0001	Timeout on master block.
	16#0002	Checksum (CRC) invalid.
16#C120		Modbus Exception Code (shown at the FC block).
	16#0001	Exception Code 1 (Illegal Function).
	16#0002	Exception Code 2 (Illegal Data Address).
	16#0003	Exception Code 3 (Illegal Data Value).
	16#0004	Exception Code 4 (Server Device Failure).
	16#0005	Exception Code 5 (Acknowledge).

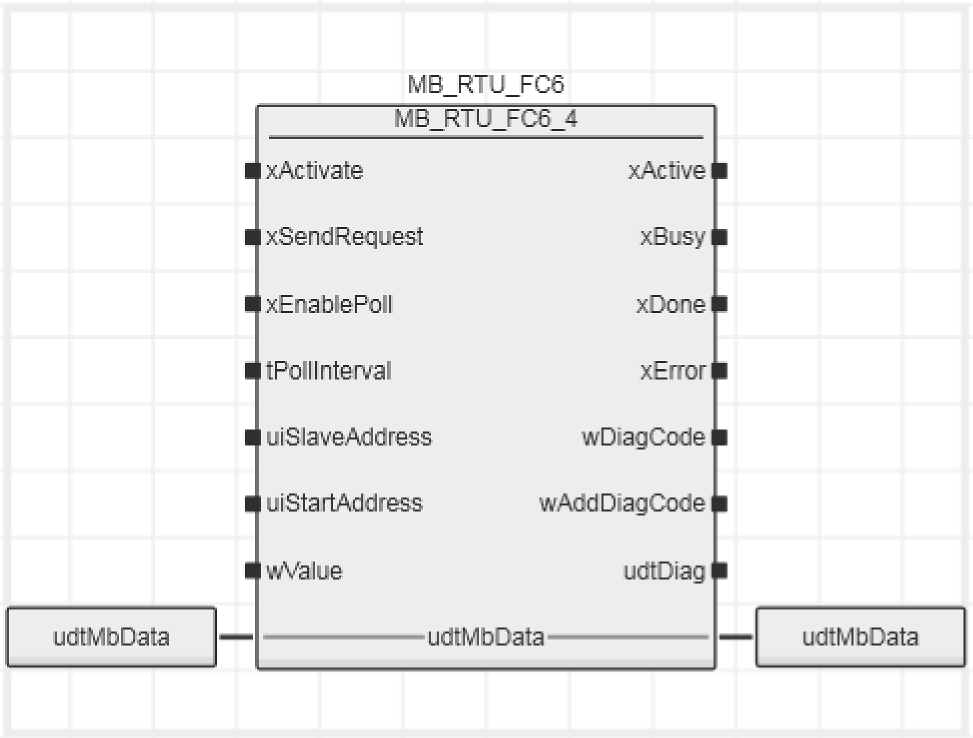
	16#0006	Exception Code 6 (Server Device Busy).
	16#0008	Exception Code 8 (Memory Parity Error).
	16#000A	Exception Code 10 (Gateway Path Unavailable).
	16#000B	Exception Code 11 (Gateway Target Device Failed To Respond).

These diagnostic codes, as well as xError, are reset by a falling edge of xActivate or xSendRequest on an FC block.
The block displays the diagnosis of the master block and thus also the diagnosis of the serial blocks. These errors must be reset by deactivating the affected blocks. For errorcodes 16#C010 - 16#C060 refer to serial block diagnostic.

12 MB_RTU_FC6

This function block writes a single holding register of a Modbus slave.

12.1 Function block call



12.2 Input parameters

Name	Type	Description
xActivate	BOOL	Block activation (TRUE = Active).
xSendRequest	BOOL	A send request to the master block is activated with a rising edge. A falling edge deletes current Modbus errors and resets the block outputs.
xEnablePoll	BOOL	Cyclical polling is started with a rising edge. A falling edge deactivates the polling. Input xSendRequest triggers an additional request and should be deactivated during poll mode. Note that the outputs xDone and xError are only one cycle true.
tPollIntervall	TIME	If xEnablePoll is activated, then transmission is cyclical in the time interval of the specified value.
uiSlaveAddress	UINT	The input specifies the address of the slave to be communicated with (1 to 255).
uiStartAddress	UINT	The input specifies the start address of the bit to be read on the slave.
wValue	WORD	The status of the input is written in the memory to be written.

12.3 Output parameters

Name	Type	Description
xActive	BOOL	TRUE: Function block is active. FALSE: Function block is not active.
xBusy	BOOL	TRUE: The block is busy with the service execution.
xDone	BOOL	Request is sent and response from slave is successfully received.
xError	BOOL	TRUE: An error has occurred. For details refer to wDiagCode and wAddDiagCode.
wDiagCode	WORD	Diagnostic code. Refer to diagnostics table.
wAddDiagCode	WORD	Additional diagnostic code. Refer to diagnostics table.
udtDiag	MB_UDT_RTU_FC_DIAG	Structure with internal variables for Diagnostic

12.4 Inout parameters

Name	Type	Description
udtMBData	udtModbus2_Data	The block communicates via this structure with the FC blocks.

12.5 Diagnosis

The diagnostics contains diagnostic codes on the FC blocks of the library. Modbus errors are indicated at the respective FC block and need to be reset there. Thus the communication in a Modbus network is not disturbed by an error in a request to a slave. An error at the FC block is deleted by a reset of the send input or by renewed activation of the block.

Modbus exception codes are sent by the respective slave and contain messages specific for Modbus.

wDiagCode	wAddDiagCode	Description
16#0000		Block is not activated.
16#8000		Block is active and operating without errors.
16#8300		Block executes a service.
16#C100		Error during configuration (displayed on the FC block).
	16#0001	Slave address is outside the valid range.
	16#0002	Number of the requested data amount invalid (iDataCount).
	16#0003	Function code invalid.
	16#0004	Broadcast not possible. FC supports reading function.
16#C110		Error in the Modbus (displayed on the FC block).
	16#0001	Timeout on master block.
	16#0002	Checksum (CRC) invalid.
16#C120		Modbus Exception Code (shown at the FC block).
	16#0001	Exception Code 1 (Illegal Function).
	16#0002	Exception Code 2 (Illegal Data Address).
	16#0003	Exception Code 3 (Illegal Data Value).
	16#0004	Exception Code 4 (Server Device Failure).
	16#0005	Exception Code 5 (Acknowledge).

	16#0006	Exception Code 6 (Server Device Busy).
	16#0008	Exception Code 8 (Memory Parity Error).
	16#000A	Exception Code 10 (Gateway Path Unavailable).
	16#000B	Exception Code 11 (Gateway Target Device Failed To Respond).

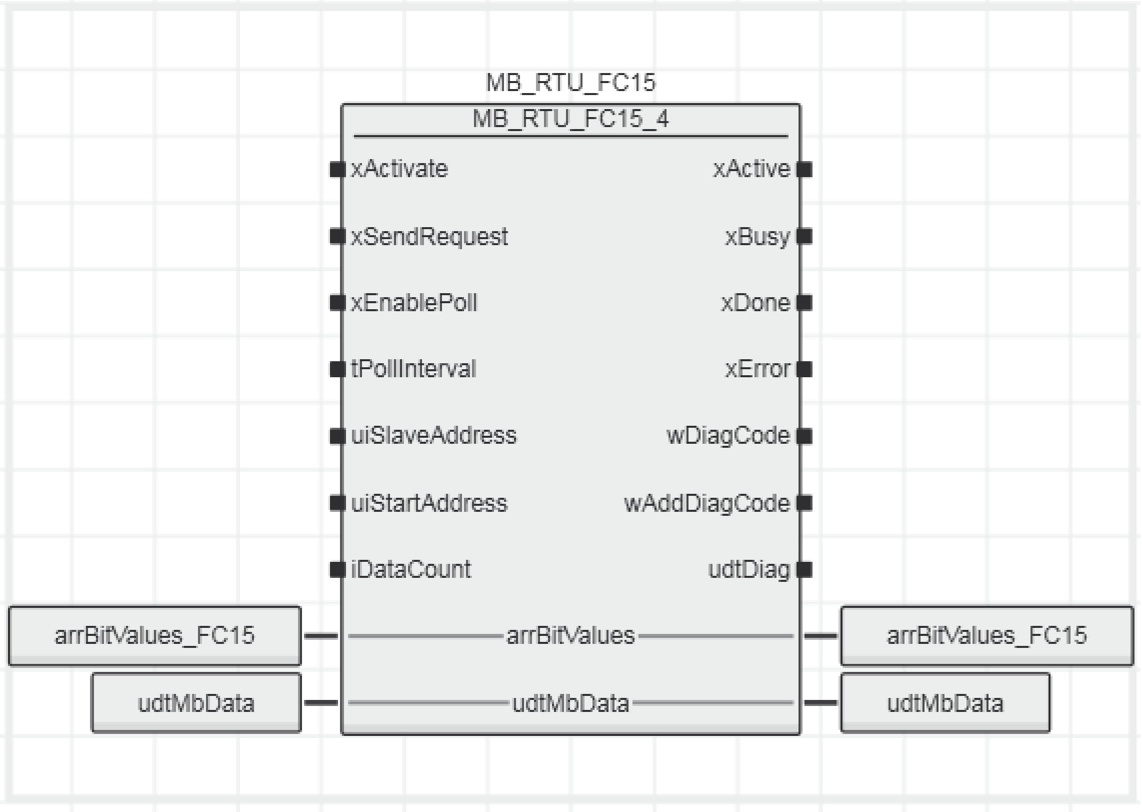
These diagnostic codes, as well as xError, are reset by a falling edge of xActivate or xSendRequest on an FC block.

The block displays the diagnosis of the master block and thus also the diagnosis of the serial blocks. These errors must be reset by deactivating the affected blocks. For errorcodes 16#C010 - 16#C060 refer to serial block diagnostic.

13 MB_RTU_FC15

This function block writes multiple output bits of a Modbus slave.

13.1 Function block call



13.2 Input parameters

Name	Type	Description
xActivate	BOOL	Block activation (TRUE = Active).
xSendRequest	BOOL	A send request to the master block is activated with a rising edge. A falling edge deletes current Modbus errors and resets the block outputs.
xEnablePoll	BOOL	Cyclical polling is started with a rising edge. A falling edge deactivates the polling. Input xSendRequest triggers an additional request and should be deactivated during poll mode. Note that the outputs xDone and xError are only one cycle true.
tPollIntervall	TIME	If xEnablePoll is activated, then transmission is cyclical in the time interval of the specified value.
uiSlaveAddress	UINT	The input specifies the address of the slave to be communicated with (1 to 255).
uiStartAddress	UINT	The input specifies the start address of the bit to be read on the slave.
iDataCount	INT	The input specifies the number of bits to be read on the slave (1 to 1968).

13.3 Output parameters

Name	Type	Description
xActive	BOOL	TRUE: Function block is active. FALSE: Function block is not active.
xBusy	BOOL	TRUE: The block is busy with the service execution.
xDone	BOOL	Request is sent and response from slave is successfully received.
xError	BOOL	TRUE: An error has occurred. For details refer to wDiagCode and wAddDiagCode.
wDiagCode	WORD	Diagnostic code. Refer to diagnostics table.
wAddDiagCode	WORD	Additional diagnostic code. Refer to diagnostics table.
udtDiag	MB_UDT_RTU_FC_DIAG	Structure with internal variables for Diagnostic

13.4 Inout parameters

Name	Type	Description
arrBitValues	arrModbus2_X_1_1968	The array of 1968 bits contains the desired values of the addressed bits.
udtMBData	udtModbus2_Data	The block communicates via this structure with the FC blocks.

13.5 Diagnosis

The diagnostics contains diagnostic codes on the FC blocks of the library. Modbus errors are indicated at the respective FC block and need to be reset there. Thus the communication in a Modbus network is not disturbed by an error in a request to a slave. An error at the FC block is deleted by a reset of the send input or by renewed activation of the block.

Modbus exception codes are sent by the respective slave and contain messages specific for Modbus.

wDiagCode	wAddDiagCode	Description
16#0000		Block is not activated.
16#8000		Block is active and operating without errors.
16#8300		Block executes a service.
16#C100		Error during configuration (displayed on the FC block).
	16#0001	Slave address is outside the valid range.
	16#0002	Number of the requested data amount invalid (iDataCount).
	16#0003	Function code invalid.
	16#0004	Broadcast not possible. FC supports reading function.
16#C110		Error in the Modbus (displayed on the FC block).
	16#0001	Timeout on master block.
	16#0002	Checksum (CRC) invalid.
16#C120		Modbus Exception Code (shown at the FC block).
	16#0001	Exception Code 1 (Illegal Function).
	16#0002	Exception Code 2 (Illegal Data Address).
	16#0003	Exception Code 3 (Illegal Data Value).
	16#0004	Exception Code 4 (Server Device Failure).

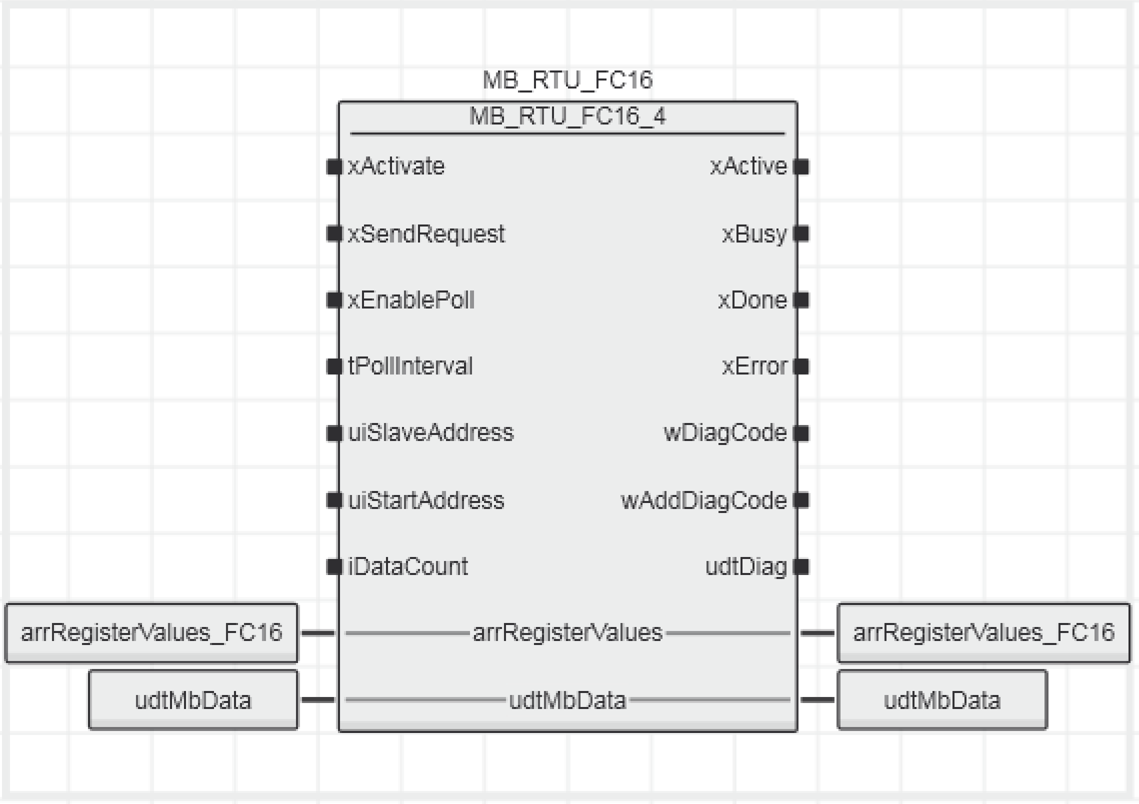
	16#0005	Exception Code 5 (Acknowledge).
	16#0006	Exception Code 6 (Server Device Busy).
	16#0008	Exception Code 8 (Memory Parity Error).
	16#000A	Exception Code 10 (Gateway Path Unavailable).
	16#000B	Exception Code 11 (Gateway Target Device Failed To Respond).

These diagnostic codes, as well as xError, are reset by a falling edge of xActivate or xSendRequest on an FC block.
The block displays the diagnosis of the master block and thus also the diagnosis of the serial blocks. These errors must be reset by deactivating the affected blocks. For errorcodes 16#C010 - 16#C060 refer to serial block diagnostic.

14 MB_RTU_FC16

This function block writes multiple holding registers of a Modbus slave.

14.1 Function block call



14.2 Input parameters

Name	Type	Description
xActivate	BOOL	Block activation (TRUE = Active).
xSendRequest	BOOL	A send request to the master block is activated with a rising edge. A falling edge deletes current Modbus errors and resets the block outputs.
xEnablePoll	BOOL	Cyclical polling is started with a rising edge. A falling edge deactivates the polling. Input xSendRequest triggers an additional request and should be deactivated during poll mode. Note that the outputs xDone and xError are only one cycle true.
tPollIntervall	TIME	If xEnablePoll is activated, then transmission is cyclical in the time interval of the specified value.
uiSlaveAddress	UINT	The input specifies the address of the slave to be communicated with (1 to 255).
uiStartAddress	UINT	The input specifies the start address of the bit to be read on the slave.
iDataCount	INT	The input specifies the number of bits to be written on the slave (1 to 123).

14.3 Output parameters

Name	Type	Description
xActive	BOOL	TRUE: Function block is active. FALSE: Function block is not active.
xBusy	BOOL	TRUE: The block is busy with the service execution.
xDone	BOOL	Request is sent and response from slave is successfully received.
xError	BOOL	TRUE: An error has occurred. For details refer to wDiagCode and wAddDiagCode.
wDiagCode	WORD	Diagnostic code. Refer to diagnostics table.
wAddDiagCode	WORD	Additional diagnostic code. Refer to diagnostics table.
udtDiag	MB_UDT_RTU_FC_DIAG	Structure with internal variables for Diagnostic

14.4 Inout parameters

Name	Type	Description
arrRegisterValues	arrModbus2_W_1_123	The array of 123 words contains the desired values of the addressed register.
udtMBData	udtModbus2_Data	The block communicates via this structure with the FC blocks.

14.5 Diagnosis

The diagnostics contains diagnostic codes on the FC blocks of the library. Modbus errors are indicated at the respective FC block and need to be reset there. Thus the communication in a Modbus network is not disturbed by an error in a request to a slave. An error at the FC block is deleted by a reset of the send input or by renewed activation of the block.

Modbus exception codes are sent by the respective slave and contain messages specific for Modbus.

wDiagCode	wAddDiagCode	Description
16#0000		Block is not activated.
16#8000		Block is active and operating without errors.
16#8300		Block executes a service.
16#C100		Error during configuration (displayed on the FC block).
	16#0001	Slave address is outside the valid range.
	16#0002	Number of the requested data amount invalid (iDataCount).
	16#0003	Function code invalid.
	16#0004	Broadcast not possible. FC supports reading function.
16#C110		Error in the Modbus (displayed on the FC block).
	16#0001	Timeout on master block.
	16#0002	Checksum (CRC) invalid.
16#C120		Modbus Exception Code (shown at the FC block).
	16#0001	Exception Code 1 (Illegal Function).
	16#0002	Exception Code 2 (Illegal Data Address).
	16#0003	Exception Code 3 (Illegal Data Value).
	16#0004	Exception Code 4 (Server Device Failure).

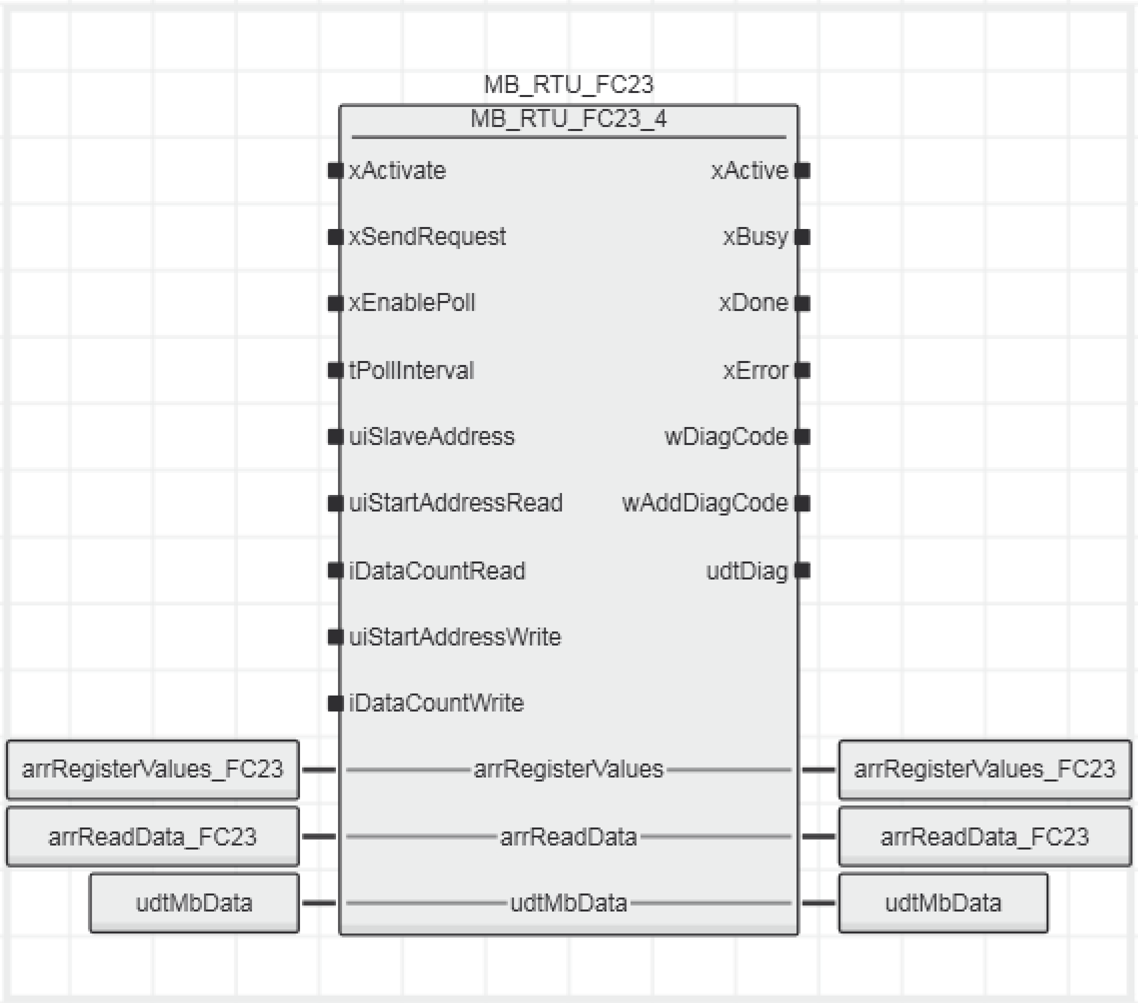
	16#0005	Exception Code 5 (Acknowledge).
	16#0006	Exception Code 6 (Server Device Busy).
	16#0008	Exception Code 8 (Memory Parity Error).
	16#000A	Exception Code 10 (Gateway Path Unavailable).
	16#000B	Exception Code 11 (Gateway Target Device Failed To Respond).

These diagnostic codes, as well as xError, are reset by a falling edge of xActivate or xSendRequest on an FC block.
The block displays the diagnosis of the master block and thus also the diagnosis of the serial blocks. These errors must be reset by deactivating the affected blocks. For errorcodes 16#C010 - 16#C060 refer to serial block diagnostic.

15 MB_RTU_FC23

This function block writes or reads multiple holding registers of a Modbus slave.

15.1 Function block call



15.2 Input parameters

Name	Type	Description
xActivate	BOOL	Block activation (TRUE = Active).
xSendRequest	BOOL	A send request to the master block is activated with a rising edge. A falling edge deletes current Modbus errors and resets the block outputs.
xEnablePoll	BOOL	Cyclical polling is started with a rising edge. A falling edge deactivates the polling. Input xSendRequest triggers an additional request and should be deactivated during poll mode. Note that the outputs xDone and xError are only one cycle true.
tPollIntervall	TIME	If xEnablePoll is activated, then transmission is cyclical in the time interval of the specified value.
uiSlaveAddress	UINT	The input specifies the address of the slave to be communicated with (1 to 255).

uiStartAddressRead	UINT	The input specifies the start address of the data to be read on the slave.
iDataCountRead	INT	The input specifies the amount of data to be read on the slave (1..125).
uiStartAddressWrite	UINT	The input specifies the start address of the data to be written on the slave.
iDataCountWrite	INT	The input specifies the amount of the data to be written on the slave (1..121).

15.3 Output parameters

Name	Type	Description
xActive	BOOL	TRUE: Function block is active. FALSE: Function block is not active.
xBusy	BOOL	TRUE: The block is busy with the service execution.
xDone	BOOL	Request is sent and response from slave is successfully received.
xError	BOOL	TRUE: An error has occurred. For details refer to wDiagCode and wAddDiagCode.
wDiagCode	WORD	Diagnostic code. Refer to diagnostics table.
wAddDiagCode	WORD	Additional diagnostic code. Refer to diagnostics table.
udtDiag	MB_UDT_RTU_FC_DIAG	Structure with internal variables for Diagnostic

15.4 Inout parameters

Name	Type	Description
arrRegisterValues	arrModbus2_W_1_123	The array of 123 words contains the desired values of the addressed register.
arrReadData	arrModbus2_W_1_125	The parameter contains the requested Modbus data.
udtMBData	udtModbus2_Data	The block communicates via this structure with the FC blocks.

15.5 Diagnosis

The diagnostics contains diagnostic codes on the FC blocks of the library. Modbus errors are indicated at the respective FC block and need to be reset there. Thus the communication in a Modbus network is not disturbed by an error in a request to a slave. An error at the FC block is deleted by a reset of the send input or by renewed activation of the block.

Modbus exception codes are sent by the respective slave and contain messages specific for Modbus.

wDiagCode	wAddDiagCode	Description
16#0000		Block is not activated.
16#8000		Block is active and operating without errors.
16#8300		Block executes a service.
16#C100		Error during configuration (displayed on the FC block).
	16#0001	Slave address is outside the valid range.
	16#0002	Number of the requested data amount invalid (iDataCount).
	16#0003	Function code invalid.
	16#0004	Broadcast not possible. FC supports reading function.
16#C110		Error in the Modbus (displayed on the FC block).
	16#0001	Timeout on master block.

	16#0002	Checksum (CRC) invalid.
16#C120		Modbus Exception Code (shown at the FC block).
	16#0001	Exception Code 1 (Illegal Function).
	16#0002	Exception Code 2 (Illegal Data Address).
	16#0003	Exception Code 3 (Illegal Data Value).
	16#0004	Exception Code 4 (Server Device Failure).
	16#0005	Exception Code 5 (Acknowledge).
	16#0006	Exception Code 6 (Server Device Busy).
	16#0008	Exception Code 8 (Memory Parity Error).
	16#000A	Exception Code 10 (Gateway Path Unavailable).
	16#000B	Exception Code 11 (Gateway Target Device Failed To Respond).

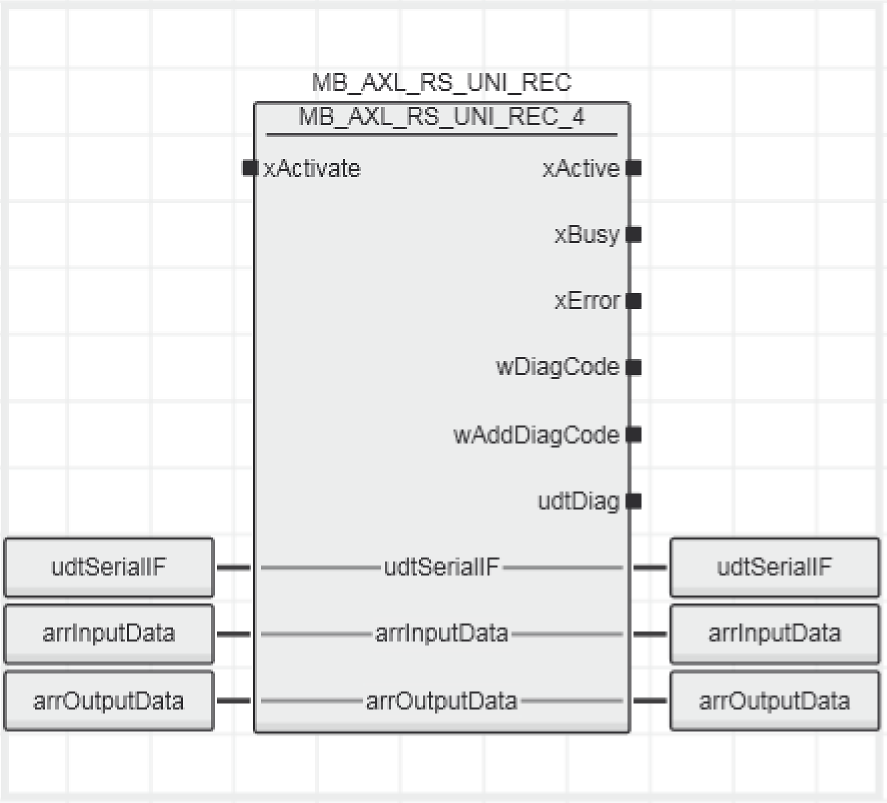
These diagnostic codes, as well as xError, are reset by a falling edge of xActivate or xSendRequest on an FC block.

The block displays the diagnosis of the master block and thus also the diagnosis of the serial blocks. These errors must be reset by deactivating the affected blocks. For errorcodes 16#C010 - 16#C060 refer to serial block diagnostic.

16 MB_AXL_RS_UNI_REC

This block runs the receiving operations via the AXL F RS UNI 1H (2688666) module. The process data width for serial communication is 20 bytes. Of this, 17 bytes are reserved for the user data.

16.1 Function block call



16.2 Input parameters

Name	Type	Description
xActivate	BOOL	Block activation (TRUE = Active).

16.3 Output parameters

Name	Type	Description
xActive	BOOL	TRUE: Function block is active. FALSE: Function block is not active.
xBusy	BOOL	TRUE: The block is busy with the service execution.
xError	BOOL	TRUE: An error has occurred. For details refer to wDiagCode and wAddDiagCode.
wDiagCode	WORD	Diagnostic code. Refer to diagnostics table.
wAddDiagCode	WORD	Additional diagnostic code. Refer to diagnostics table.
udtDiag	MB_UDT_RTU_REC_DIAG	Structure with internal variables for Diagnostic

16.4 Input and output parameters

Name	Type	Description
udtSerialIF	MB2_AXL_RSUNI2_UDT_IF	The block communicates via this structure with the Modbus driver block.
arrInputDataAXL_RSUNI	MB2_AXL_RSUNI2_ARR_B_0_19	Connection of the input process data of the serial interface.
arrOutputDataAXL_RSUNI	MB2_AXL_RSUNI2_ARR_B_0_19	Connection of the output process data of the serial interface.

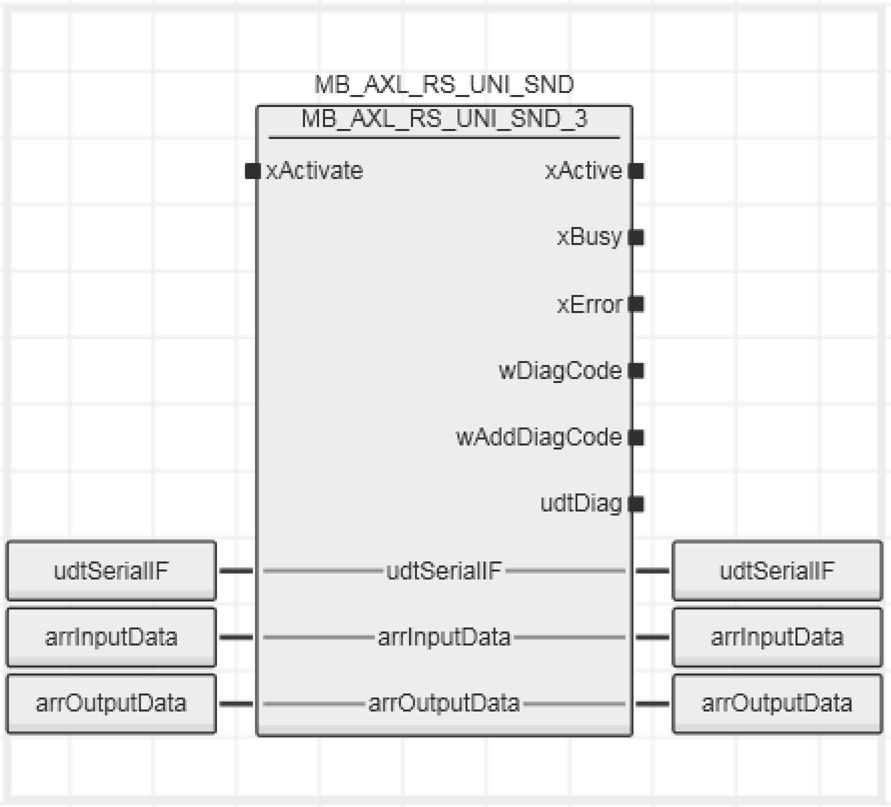
16.5 Diagnosis

wDiagCode	wAddDiagCode	Description
16#0000	16#0000	Block is not activated.
16#8000	16#0000	Block is active and operating without errors.
16#C030		Error when receiving.
	16#0010	Timeout when receiving.
	16#0030	uiRcvLength is larger than the memory available in the receive buffer.
	16#0040	uiRcvLength <> 0 for end-to-end protocol.
	16#0060	Communication error when receiving.
16#C040		Error in intermediate storage.
	16#0010	Timeout in intermediate storage.
16#C050	16#0000	Error from module: • Failure of the peripheral voltage • Invalid parameter for specified command

17 MB_AXL_RS_UNI_SND

This block runs the sending operations via the AXL F RS UNI 1H (2688666) module. The process data width for serial communication is 20 bytes. Of this, 17 bytes are reserved for the user data.

17.1 Function block call



17.2 Input parameters

Name	Type	Description
xActivate	BOOL	Block activation (TRUE = Active).

17.3 Output parameters

Name	Type	Description
xActive	BOOL	TRUE: Function block is active. FALSE: Function block is not active.
xBusy	BOOL	TRUE: The block is busy with the service execution.
xError	BOOL	TRUE: An error has occurred. For details refer to wDiagCode and wAddDiagCode.
wDiagCode	WORD	Diagnostic code. Refer to diagnostics table.
wAddDiagCode	WORD	Additional diagnostic code. Refer to diagnostics table.
udtDiag	MB_UDT_RTU_SND_DIAG	Structure with internal variables for Diagnostic

17.4 Inout parameters

Name	Type	Description
udtSerialIF	MB2_AXL_RSUNI2_UDT_IF	The block communicates via this structure with the Modbus driver block.
arrInputDataAXL_RSUNI	MB2_AXL_RSUNI2_ARR_B_0_19	Connection of the input process data of the serial interface.
arrOutputDataAXL_RSUNI	MB2_AXL_RSUNI2_ARR_B_0_19	Connection of the output process data of the serial interface.

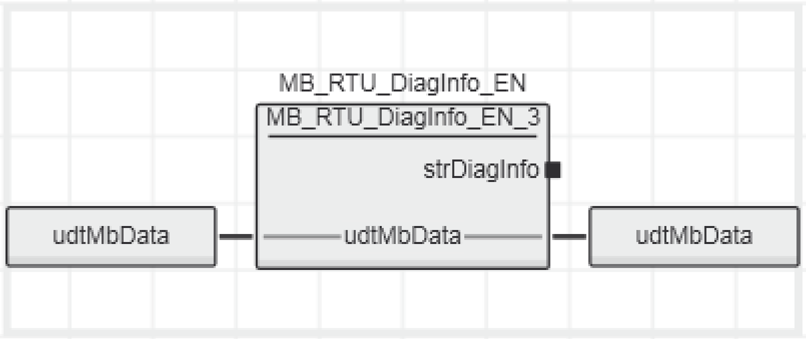
17.5 Diagnosis

wDiagCode	wAddDiagCode	Description
16#0000	16#0000	Block is not activated.
16#8000	16#0000	Block is active and operating without errors.
16#C020		Error when sending.
	16#0020	Maximum size exceeded when sending.
	16#0060	Data send error in module.
16#C030		Error when receiving.
	16#0060	Communication error when receiving.
16#C040		Error in intermediate storage.
	16#0010	Timeout in intermediate storage.
16#C050	16#0000	Error from module: • Failure of the peripheral voltage • Invalid parameter for specified command

18 MB_RTU_DiagInfo_EN

If there is an error, this block shows the diagnostics of the master block as a text in English. The source code of the block can be read and modified. To show the diagnostic messages in other languages, copy the block and translate the diagnostic text into the desired language. The text output (strDiagInfo) is limited to 80 characters.

18.1 Function block call



18.2 Input parameters

None

18.3 Output parameters

Name	Type	Description
strDiagInfo	STRING	If there is an error, the variable shows the description for the current wDiagCode and wAddDiagCode in English.

18.4 Inout parameters

Name	Type	Description
udtMbData	udtModbus2_Data	The block communicates via this structure with the FC blocks.

19 Examples

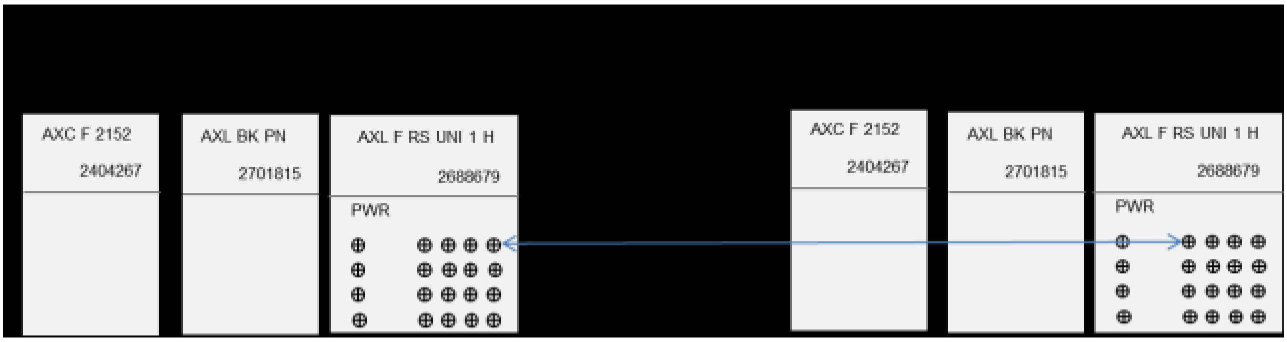
For the startup instruction of the Modbus_RTU function block please find the following examples:

- MB_RTU_7_EXA_AXL_UNI_MA_AXC_F_2152.pcwex
- MB_RTU_7_EXA_AXL_UNI_SL_AXC_F_2152.pcwex

These examples are located in the “Examples” folder of the unzipped msi file of the library.

They describes the communication between Modbus_Master and Modbus_Slave.

The serial interface from example 1 (Modbus_RTU Master) must be connected with the serial interface from example 2 (Modbus_RTU Slave) via RS485 (two wires and termination at each end).



19.1 Example 1: Modbus_RTU master functionality

19.1.1 Plant

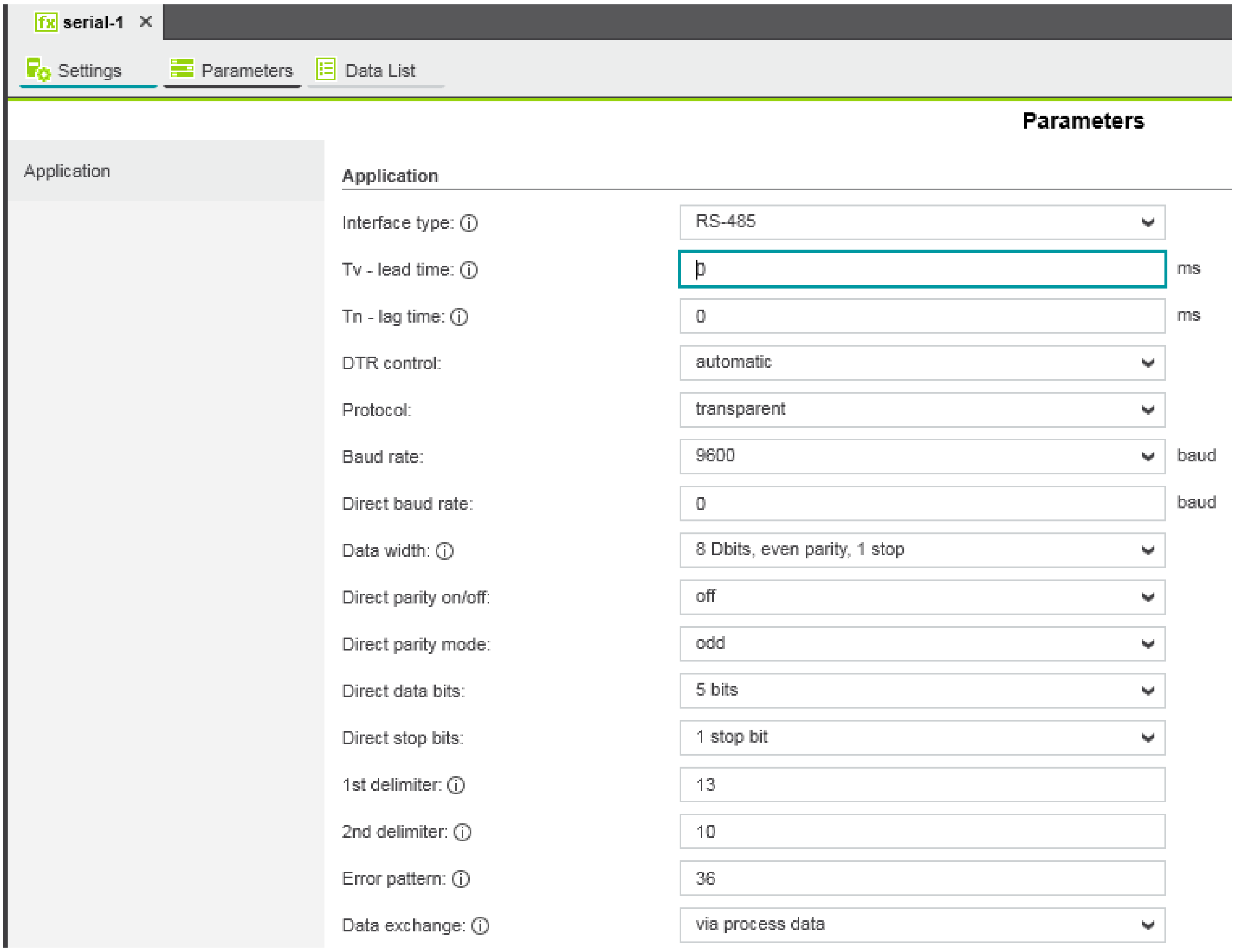
For this example, the following hardware is used:

- AXC F 2152 (2404267)
- AXL F BK PN (2701815)
- AXL F RS UNI 1H (2688666)

19.1.2 Modbus master with AXL F RS UNI 1H (2688666)

AXL F RS UNI 1H startup parameters:

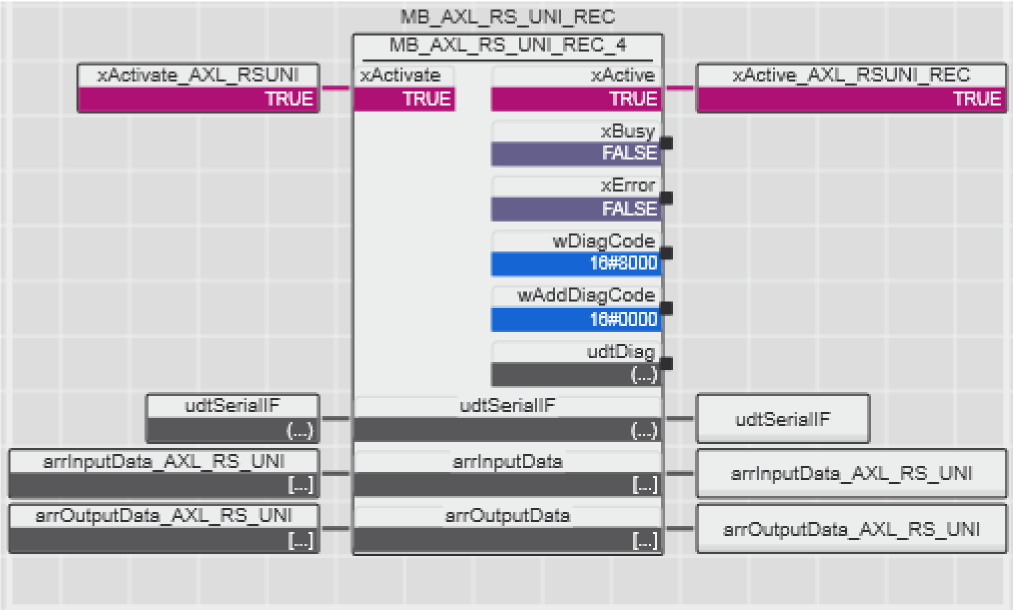
If the xAuto_CRC input is activated, the selected protocol must be "Modbus RTU". If the xAuto_CRC input is deactivated, the selected protocol must be "Transparent".

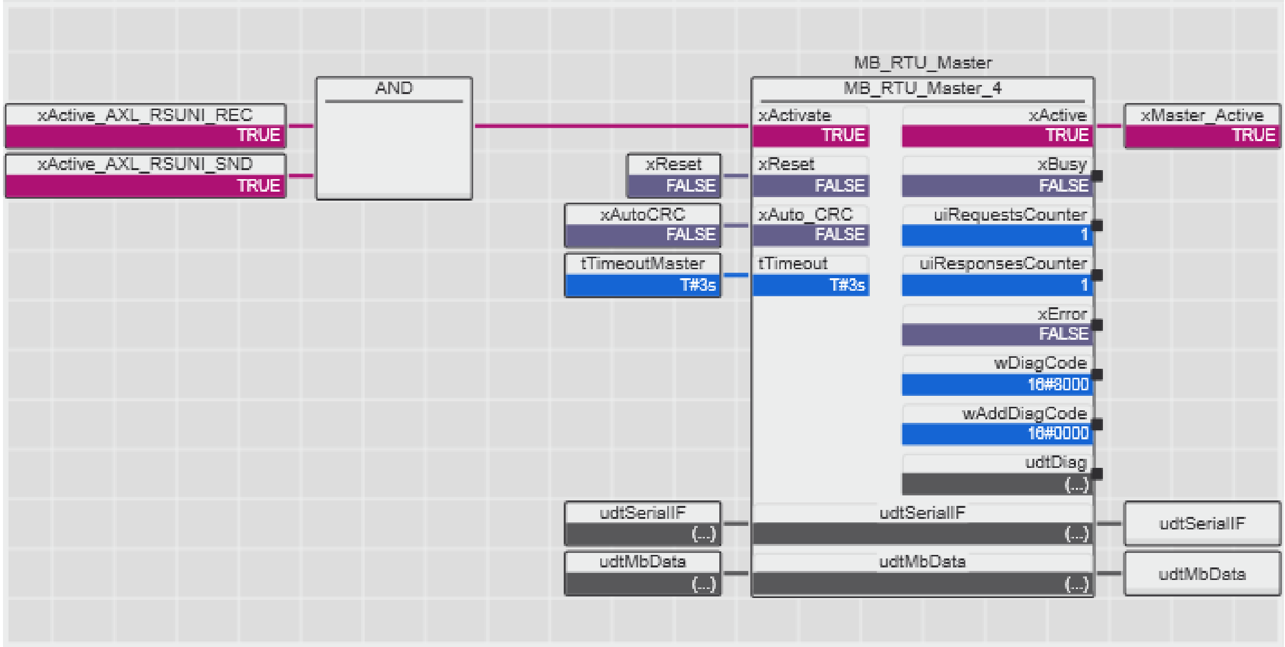


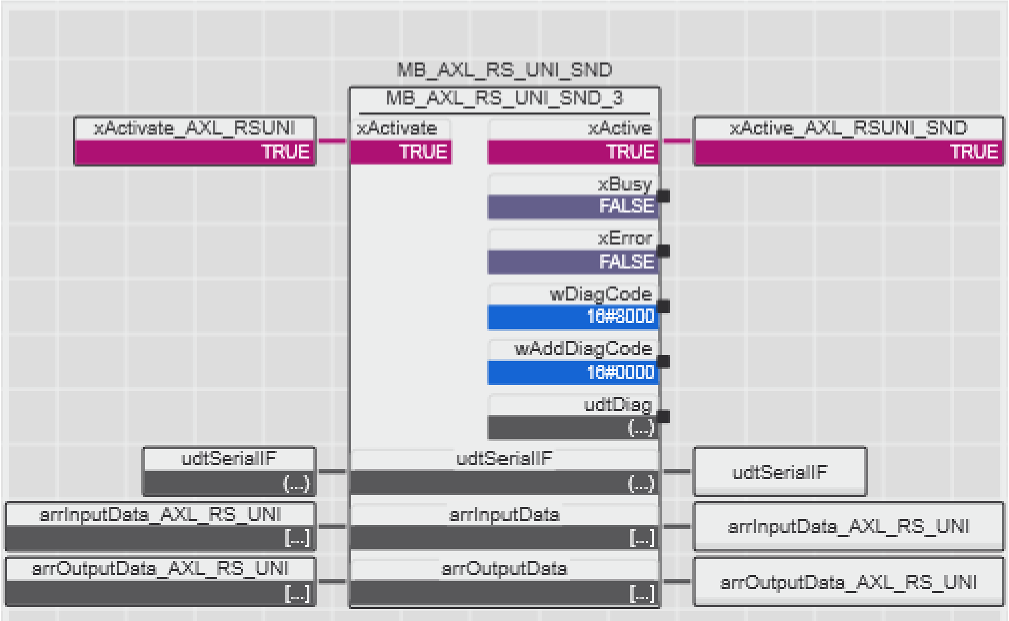
Application

Interface type: ⓘ	RS-485	
Tv - lead time: ⓘ	0	ms
Tn - lag time: ⓘ	0	ms
DTR control:	automatic	
Protocol:	transparent	
Baud rate:	9600	baud
Direct baud rate:	0	baud
Data width: ⓘ	8 Dbits, even parity, 1 stop	
Direct parity on/off:	off	
Direct parity mode:	odd	
Direct data bits:	5 bits	
Direct stop bits:	1 stop bit	
1st delimiter: ⓘ	13	
2nd delimiter: ⓘ	10	
Error pattern: ⓘ	36	
Data exchange: ⓘ	via process data	

Modbus master with AXL drivers:

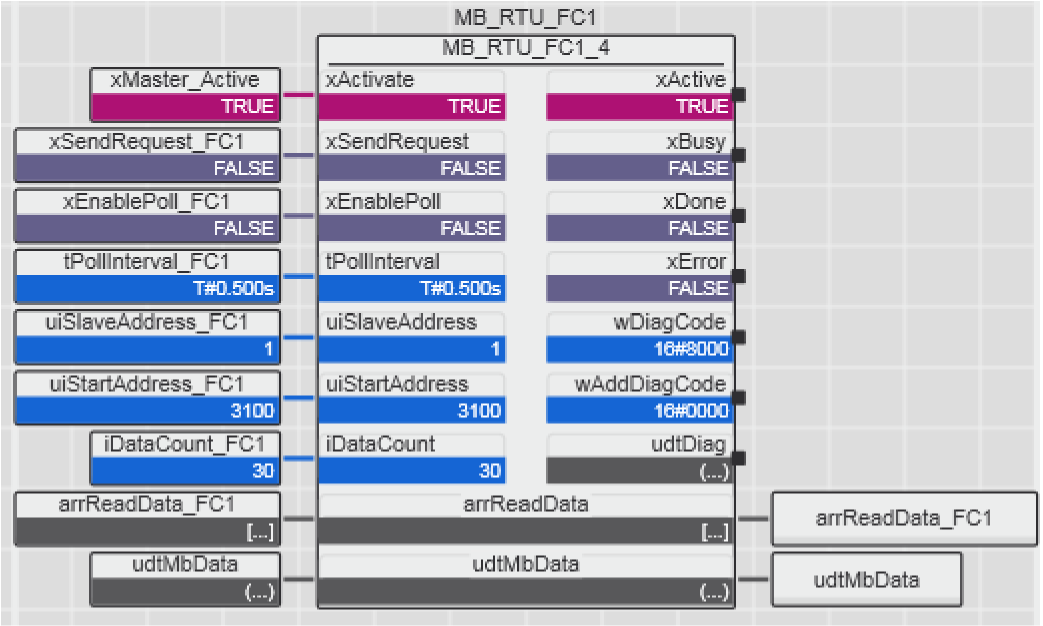






MB_AXL_RS_UNI_SND
MB_AXL_RS_UNI_SND_3

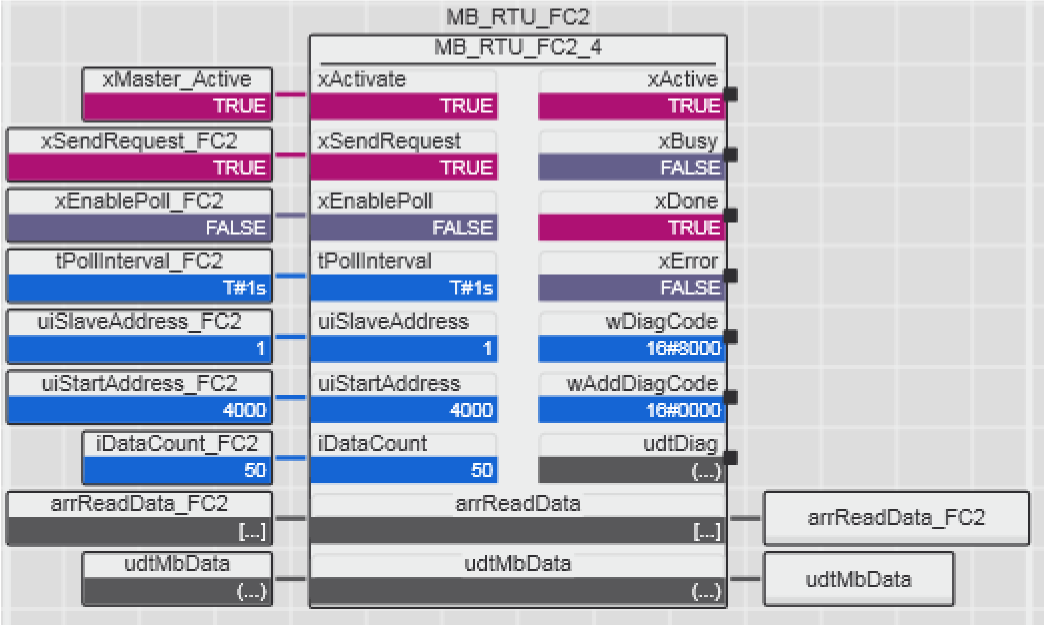
xActivate_AXL_RSUNI TRUE → xActivate TRUE
xActive TRUE → xActive_AXL_RSUNI_SND TRUE
xBusy FALSE
xError FALSE
wDiagCode 16#8000
wAddDiagCode 16#0000
udtDiag (...)
udtSerialIF (...) → udtSerialIF (...) → udtSerialIF
arrInputData_AXL_RS_UNI [...] → arrInputData [...] → arrInputData_AXL_RS_UNI
arrOutputData_AXL_RS_UNI [...] → arrOutputData [...] → arrOutputData_AXL_RS_UNI



MB_RTU_FC1
MB_RTU_FC1_4

xMaster_Active TRUE → xActivate TRUE
xSendRequest_FC1 FALSE → xSendRequest FALSE
xEnablePoll_FC1 FALSE → xEnablePoll FALSE
tPollInterval_FC1 T#0.500s → tPollInterval T#0.500s
uiSlaveAddress_FC1 1 → uiSlaveAddress 1
uiStartAddress_FC1 3100 → uiStartAddress 3100
iDataCount_FC1 30 → iDataCount 30
arrReadData_FC1 [...] → arrReadData [...] → arrReadData_FC1
udtMbData (...) → udtMbData (...) → udtMbData

xActive TRUE
xBusy FALSE
xDone FALSE
xError FALSE
wDiagCode 16#8000
wAddDiagCode 16#0000
udtDiag (...)



MB_RTU_FC2
MB_RTU_FC2_4

xMaster_Active TRUE → xActivate TRUE
xSendRequest_FC2 TRUE → xSendRequest TRUE
xEnablePoll_FC2 FALSE → xEnablePoll FALSE
tPollInterval_FC2 T#1s → tPollInterval T#1s
uiSlaveAddress_FC2 1 → uiSlaveAddress 1
uiStartAddress_FC2 4000 → uiStartAddress 4000
iDataCount_FC2 50 → iDataCount 50
arrReadData_FC2 [...] → arrReadData [...] → arrReadData_FC2
udtMbData (...) → udtMbData (...) → udtMbData

xActive TRUE
xBusy FALSE
xDone TRUE
xError FALSE
wDiagCode 16#8000
wAddDiagCode 16#0000
udtDiag (...)

Execution order:

To process the blocks optimally and for a fast Modbus communication, the serial block for receiving has to be placed at the beginning of the program and the serial block for sending has to be placed at the end of the program. In between the master block is inserted.

Creating structures:

The blocks of the serial interface and the Modbus master have to be connected with each other via the same structure. The Modbus master as well as the
FC blocks are also connected with each other via a structure.

Instantiation:

A master block can be connected only to one serial driver (here: udtSerialIF). For a second instance, a second serial interface needs to be implemented. Every Modbus FC can be instantiated multiple times and connected multiple times with the same Modbus master via udtMbData structure.

19.2 Example 2: Modbus_RTU slave functionality

19.2.1 Plant

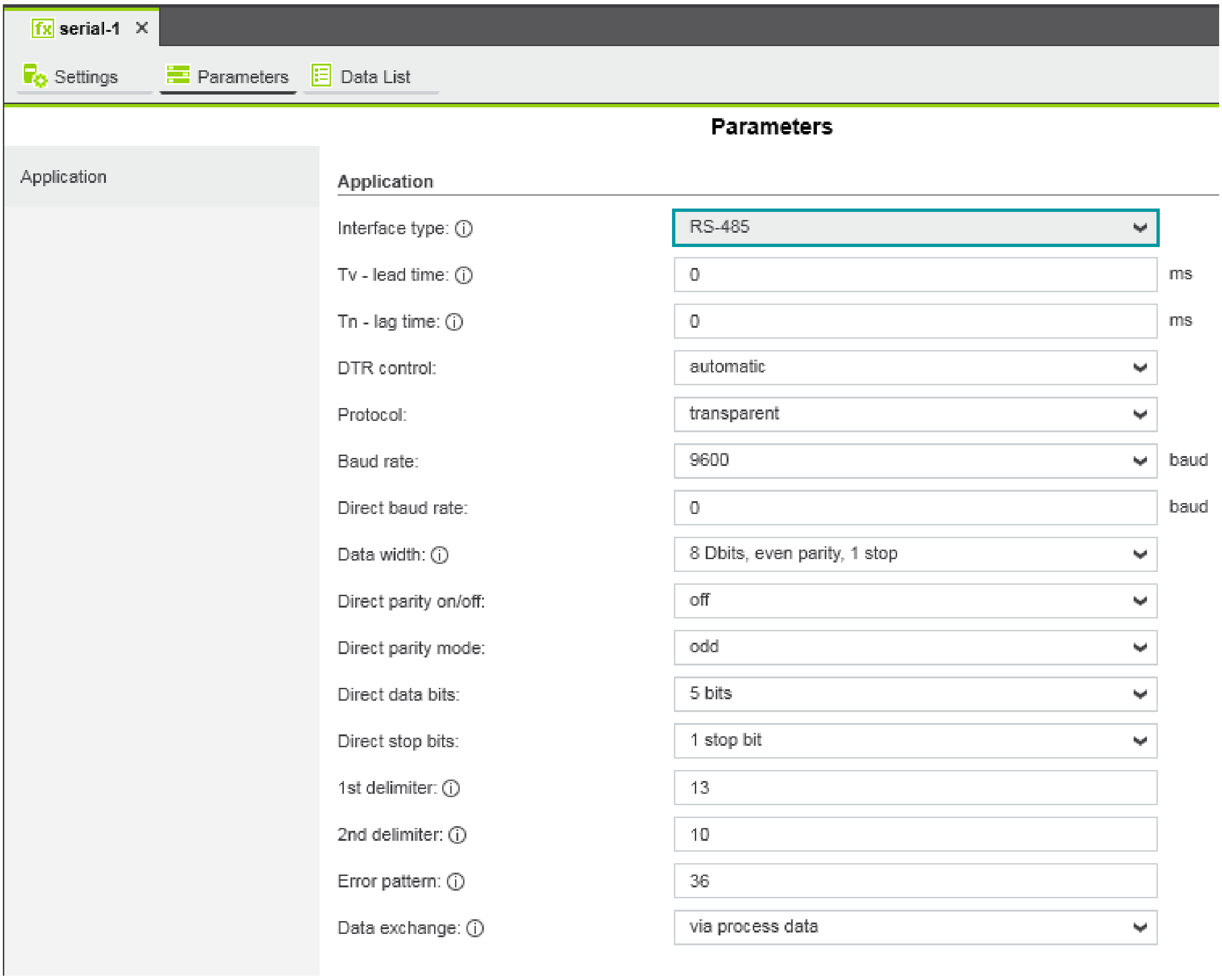
For this example, the following hardware is used:

- AXC F 2152 (2404267)
- AXL F BK PN (2701815)
- AXL F RS UNI 1H (2688666)

19.2.2 Modbus slave with AXL F RS UNI 1H (2688666)

AXL F RS UNI 1H startup parameters:

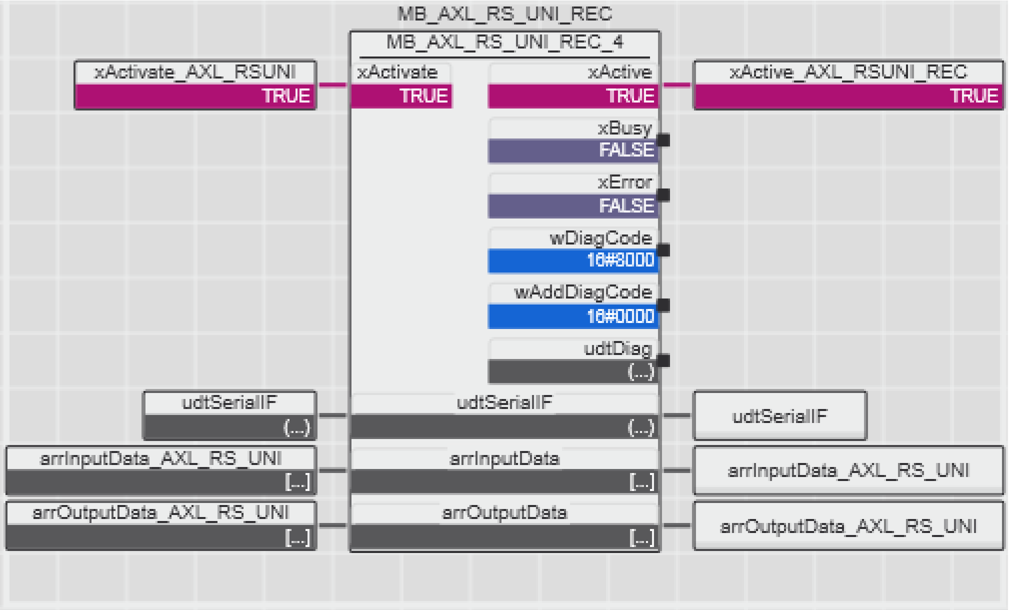
The selected protocol must be "Transparent".

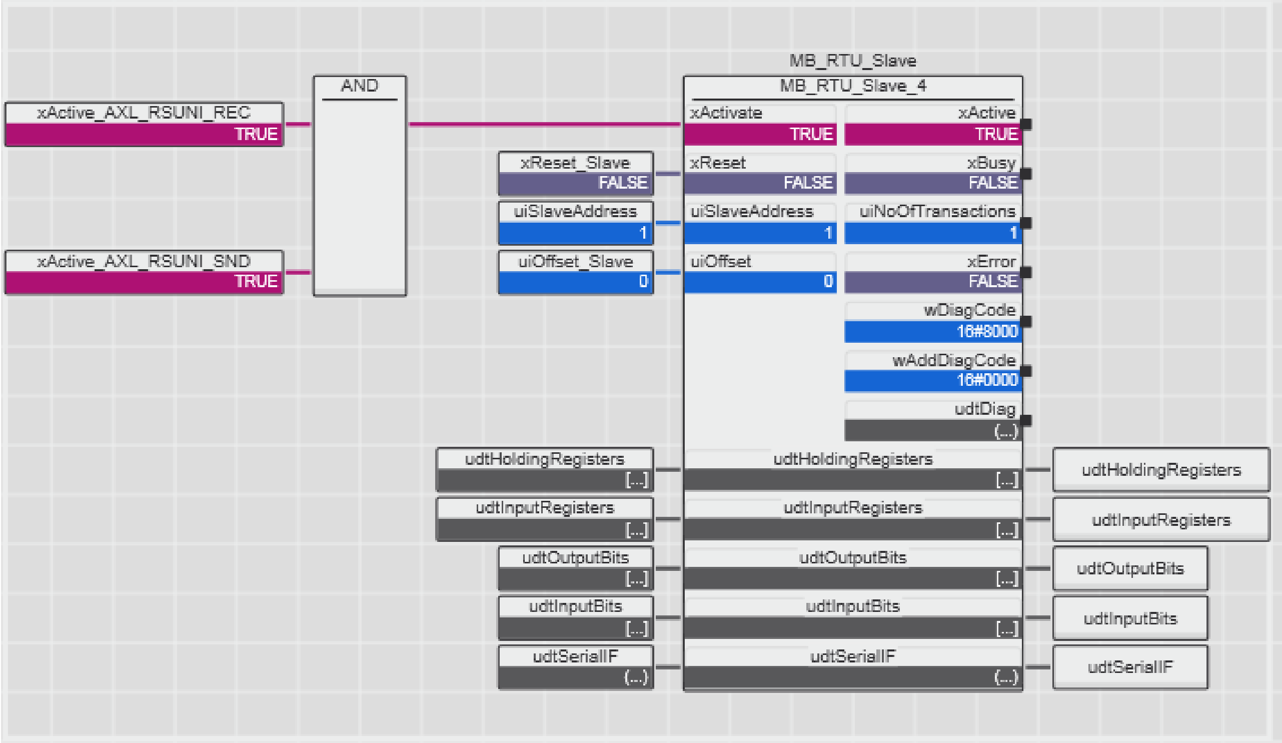


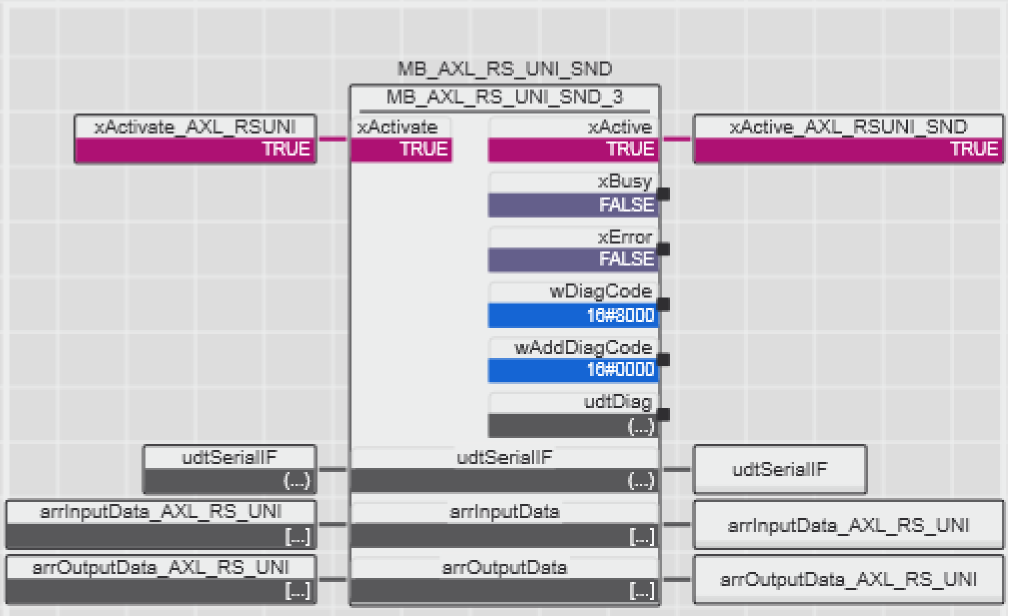
Application

Interface type:	RS-485	
Tv - lead time:	0	ms
Tn - lag time:	0	ms
DTR control:	automatic	
Protocol:	transparent	
Baud rate:	9600	baud
Direct baud rate:	0	baud
Data width:	8 Dbits, even parity, 1 stop	
Direct parity on/off:	off	
Direct parity mode:	odd	
Direct data bits:	5 bits	
Direct stop bits:	1 stop bit	
1st delimiter:	13	
2nd delimiter:	10	
Error pattern:	36	
Data exchange:	via process data	

Modbus slave with AXL drivers:







The slave block is connected to the serial driver just like the master block via the udtSerialIF structure. It should be located between the driver function blocks.

20 Appendix

20.1 Data types

```
TYPE
    arrModbus2_W_1_126      : ARRAY [1..126]    OF WORD;
    arrModbus2_W_1_125      : ARRAY [1..125]    OF WORD;
    arrModbus2_W_1_123      : ARRAY [1..123]    OF WORD;
    arrModbus2_B_1_330      : ARRAY [1..330]    OF BYTE;
    arrModbus2_X_1_2000     : ARRAY [1..2000]   OF BOOL;
    arrModbus2_X_1_1968     : ARRAY [1..1968]   OF BOOL;
    arrModbus2_X_1_16       : ARRAY [1..16]     OF BOOL;

    arrModbus2_w_0_1999     : ARRAY [0..1999]       OF WORD;
    arrModbus2_x_3000_3999  : ARRAY [3000..4015]    OF BOOL;
        (*additional 16 bits were added to avoid out of range
        error when processing the last 16 bits*)
    arrModbus2_x_4000_4999  : ARRAY [4000..5015]    OF BOOL;
        (*additional 16 bits were added to avoid out of range
        error when processing the last 16 bits*)
    arrModbus2_w_2000_2999  : ARRAY [2000..2999]    OF WORD;
    arrModbus2_w_0_124      : ARRAY [0..124]        OF WORD;
    arrModbus2_x_0_15       : ARRAY [0..15]         OF BOOL;
    arrModbus2_B_0_256      : ARRAY [0..257]        OF BYTE;

    udtModbus2_Data : STRUCT
        (* Modbus Handling *)
        (* Send Modbus request *)
        xSendRequest        : BOOL;
            (* Indicates FC wants to send a Modbus request *)
        xNDR                : BOOL; (* New modbus response received *)
        xBusy               : BOOL; (* FC only operates if not busy *)
        xReset              : BOOL; (* Reset from input on master FB *)
        (* General Modbus data *)
        uiSlaveAddress      : UINT; (* Address of the Modbus slave *)
        iFunctionCode       : INT;  (* Function Code by the Master *)
        uiStartAddress      : UINT;
            (* Starting address in the Modbus register table *)
        iSndDataCount       : INT;  (* Required data length from FC *)
        iExpDataCount       : INT;  (* Expected data length depending
            of the function code number OF bits or words *)
        uiRcvdDataCount     : UINT; (* Received bytes from Serial IF
            / UINT for the range higher than 127 *)
        arrData             : arrModbus2_W_1_125;  (* modbus telegram *)
        (* Failure handling (master outputs) *)
        xMasterActive       : BOOL; (* interface is ready *)
        xMasterBusy         : BOOL; (* interface is busy *)
        xMasterError        : BOOL; (* error indication *)
        wMasterDiagCode     : WORD; (* diagnostics code *)
        wMasterAddDiagCode  : WORD; (* additional diagnostics code *)
        xMB_Error           : BOOL; (* Exception Code Response *)
        xFC_Busy            : BOOL; (* FC catches bit IF request and not
            xFC_Busy *)
    END_STRUCT;

(* Diagnostic structures udtDiag *)

    MB_UDT_RTU_MASTER_DIAG : STRUCT
        iState          : INT;
        wDiagCode       : WORD;
        wAddDiagCode    : WORD;
    END_STRUCT;
```

```
MB_UDT_RTU_SLAVE_DIAG : STRUCT
    iState          : INT;
    wDiagCode       : WORD;
    wAddDiagCode    : WORD;
END_STRUCT;

MB_UDT_RTU_REC_DIAG : STRUCT
    iState          : INT;
    wDiagCode       : WORD;
    wAddDiagCode    : WORD;
    bControlByte0   : BYTE;
    bStatusByte0    : BYTE;
END_STRUCT;

MB_UDT_RTU_SND_DIAG : STRUCT
    iState          :   INT;
    wDiagCode       :   WORD;
    wAddDiagCode    :   WORD;
    bControlByte0   :   BYTE;
    bStatusByte0    :   BYTE;
END_STRUCT;

MB_UDT_RTU_FC_DIAG : STRUCT
    iState          : INT;
    wDiagCode       : WORD;
    wAddDiagCode    : WORD;
END_STRUCT;

(* *** AXL F RS UNI 1H *** *)

(* Input and output array for processdata of the module *)
MB2_AXL_RSUNI2_ARR_B_0_19 : ARRAY [0..19] OF BYTE;

(* Buffer for temporary saving of received data *)
MB2_AXL_RSUNI2_ARR_B_1_17 : ARRAY [1..17] OF BYTE;

(* Maximum buffer for outgoing user data *)
MB2_AXL_RSUNI2_ARR_B_1_1023 : ARRAY [1..1023] OF BYTE;

(* Maximum buffer for incoming user data *)
MB2_AXL_RSUNI2_ARR_B_1_4096 : ARRAY [1..4096] OF BYTE;

(* Status of the serial interface *)
MB2_AXL_RSUNI2_UDT_STATUS : STRUCT
    xErrorModule        : BOOL; (*Error in module - peripheral fault or
        invalid command*)

    (* additional status OF the module *)
    xDSR                : BOOL; (* TRUE -> Data set ready. Opposite
        side is ready for communication *)
    xDCD                : BOOL; (* TRUE -> Data carrier detect. Opposite
        side detecting incoming data *)

    (*status OF the receiving part OF the module*)
    xErrorRcv           : BOOL; (* TRUE -> Error during data receive
        operation OF the module *)
    xRcvBufferFull      : BOOL; (* TRUE -> Receive buffer of module
        full *)
    xRcvBufferNotEmpty  : BOOL; (* TRUE -> Receive buffer of module not
        empty *)

    (* status of the sending part OF the module *)
    xErrorSend          : BOOL; (* TRUE -> Error during data send
        operation of the module *)
    xSendBufferFull     : BOOL; (* TRUE -> Send buffer of module full *)
    xSendBufferNotEmpty : BOOL; (* TRUE -> Send buffer of module not
        empty *)

    uiRcvBufferModule   : UINT; (* Number OF characters in the receive
        buffer of the module *)
    wFirmwareVersion    : WORD; (* Firmware version OF the module *)
END_STRUCT;

(* counter of the serial interface *)
MB2_AXL_RSUNI2_UDT_COUNTER : STRUCT
    uiRcvCharValid      : UINT; (* Number OF valid received characters *)
    uiRcvCharInvalid    : UINT; (* Number OF invalid received characters *)
    uiSendChar          : UINT; (* Number OF sent characters *)
END_STRUCT;

MB2_AXL_RSUNI2_UDT_IF   : STRUCT
    xActive                     : BOOL; (* TRUE -> Serial Driver is
        Activated *)
    xReady                      : BOOL; (* TRUE -> Serial IL Driver is
        Ready to send / receive *)
    xEndToEnd                   : BOOL; (* TRUE -> End to end protocoll is
        used for communication *)
    xAck                        : BOOL; (* TRUE -> Acknowledge incoming
        errors *)
    xAutoAck                    : BOOL; (* TRUE -> Reset communication
        errors automatically *)
    xDTR                        : BOOL; (* TRUE -> Turn on DTR function of
        module *)
    xReadStatusCounter          : BOOL; (* TRUE -> Read status counters of
        the module *)
    xSend                       : BOOL; (* TRUE -> Send send request to
        module *)
    uiSendLength                : UINT; (* Number of bytes to be sent *)
    xResetRecBuf                : BOOL; (* TRUE -> Reset receive buffer of
        function block *)
    uiRcvLength                 : UINT; (* Number of characters to be read
        in *)
    xFBSending                  : BOOL; (* Function block in sending mode *)
    xFBReceiving                : BOOL; (* Function block in receiving
        mode *)

    udtStatusCounter            : MB2_AXL_RSUNI2_UDT_COUNTER;
        (* Structure containing status counters *)
    udtStatusSerialInterface    : MB2_AXL_RSUNI2_UDT_STATUS;
        (* Structure containing status OF serial interface *)
    xStatusFailure              : BOOL; (* Receive or send error
        existing *)
    xRcvBufferNotEmpty          : BOOL; (* TRUE -> Receive buffer
        containing data *)
    xRecBufFull                 : BOOL; (* TRUE -> Software receive buffer
        full *)
    xReadCounterDone            : BOOL; (* Finished reading in status
        counter same cycle *)
    xSendDone                   : BOOL; (* Finished sending of data same
        cycle *)
    xNDR                        : BOOL; (* Finished reading of data same
        cycle *)
    uiRcvDataLength             : UINT; (* Number of read in characters *)

    tTimeout                    : TIME; (* Timeout value for timeout in
        case of freezed receiving, sending, buffering operation *)
    arrRcvData                  : MB2_AXL_RSUNI2_ARR_B_1_4096;
```

```
            (* Array containing received data *)
        arrSendData                 : MB2_AXL_RSUNI2_ARR_B_1_1023;
            (* Array containing data to be sent *)

        (* Mirroring for observing in Modbus Master FB *)

        xActive_REC                 : BOOL;
        xBusy_REC                   : BOOL;
        xError_REC                  : BOOL;
        wDiagCode_REC               : WORD;
        wAddDiagCode_REC            : WORD;

        xActive_SND                 : BOOL;
        xBusy_SND                   : BOOL;
        xError_SND                  : BOOL;
        wDiagCode_SND               : WORD;
        wAddDiagCode_SND            : WORD;

        xComSerial_IL               : BOOL; (* TRUE: Inline Module *)
        xRCV_ComSerial_IL           : BOOL; (* xReceive for Inline modules *)
    END_STRUCT;

    (* Common arrays *)

    MB2_COM_ARR_B_1_12      : ARRAY [1..12] OF BYTE;

    MB2_COM_ARR_B_1_128     : ARRAY [1..128] OF BYTE;

    MB2_COM_ARR_B_1_330     : ARRAY [1..330] OF BYTE;

    MB2_COM_ARR_B_1_2048    : ARRAY [1..2048] OF BYTE;

    MB2_RSUNI_ARR_B_1_14    : ARRAY [1..14] OF BYTE;
    MB2_RSUNI_ARR_B_1_30    : ARRAY [1..30] OF BYTE;
    MB2_RSUNI_ARR_B_1_62    : ARRAY [1..62] OF BYTE;
    MB2_RSUNI_ARR_B_1_256   : ARRAY [1..256] OF BYTE;

END_TYPE
```

21 Support

For technical support please contact your local PHOENIX CONTACT agency

at https://www.phoenixcontact.com

Owner:

PHOENIX CONTACT Electronics GmbH
Business Unit Automation Systems
System Services
Library Services