Function block library

Modbus_RTU_5

for PC Worx

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\PC Worx 6

Please copy the library data to your PC Worx 6 working library directory.

If you did not specify a different directory during **PC Worx 6** installation the default PC Worx 6 working library directory is

c:\Users\Public\Documents\PC WORX\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	PC Worx version	Change notes	Supported PLCs
5	20200611	6.30.2907	MB_RTU_FC (all FCs): • Bugfix: The FC shows neither xError nor xDone when xReset of the MB_xxx_Master block is set in the same cycle, where xError or xDone should come. MB_AXL_F_RSUNI_Master: • Bugfix: Modbus transmission error messages with the terminal parameterization “Modbus_RTU” must be output at the FC and not at the MB_AXL_F_RSUNI_Master. • Bugfix: xActive may only become TRUE if all internal blocks are active. MB_AXL_SE_RS485_Master: • Bugfix: xActive may only become TRUE if all internal blocks are active. MB_IL_485E_Master and Slave, MB_IL_232E_Master and Slave, MB_IL_UNIxx_Master and Slave: • Correction of the precision of the process data timeout timer. • Increased occurrence of Modbus timeouts with slow bus cycle time between PLC and module and simultaneously fast PLC cycle time.	Refer to “Supported PLCs”

4	20200430	6.30.2907	MB_RTU_Master: • Modified timeout handling. • Deactivation after PD-Timeout. MB_RTU_FCx: • New timeout between FC and Modbus_Master. New function blocks: • MB_AXL_F_RSUNI_Master • MB_AXL_F_RSUNI_Slave • MB_RTU_AXL_F_RSUNI_Master • MB_RTU_AXL_F_RSUNI_Slave NOTE: The function blocks: • MB_RTU_Master • MB_RTU_Slave • MB_AXL_RS_UNI_RCV • MB_AXL_RS_UNI_SND were replaced by the compatible blocks: • MB_RTU_AXL_F_RSUNI_Master • MB_RTU_AXL_F_RSUNI_Slave	"

3	20200128	6.30.2519	MB_RTU_Master: • Enable communication after error without FB restart. MB_RTU_FC (all FCs) and MB_RTU_Master: • Resetting the FC by resetting the MB_RTU_Master. • At deactivation request of the master or FC during execution of a Modbus request wait for response or timeout before deactivation. MB_RTU_FC (all FCs) • Additional check of the Modbus response for validity (inside the FC). Response is consistent with the request. MB_AXL_RS_UNI_SND and MB_AXL_RS_UNI_RCV: • Modified timeout and Error handling. • Correction of problems when receiving more than 17 bytes and the parameterization Modbus_RTU.	"
2	20190521	6.10.200	New functionalities: • New function blocks MB_IL_232E_Master and MB_IL_232E_Slave MB_RTU_Master_5: • Improved handshakes between master and serial driver. • INLINE mode: Master block shows wrong diagnosis code 16#8100 in case of parameterization error of the terminal. 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_IL_232P_Master: • xActive = TRUE, if Driver and Modbus Master are active. MB_IL_232P_Slave: • xActive = TRUE, if Driver and Modbus Master are active. MB_IL_485E_Master: • xActive = TRUE, if Driver and Modbus Master are active. • Bugfix: "Inter-character time bigger than Modbus specification allows. Communication errors with slow CPU cycle-times or high bussystem cycle-times." MB_IL_485E_Slave: • xActive = TRUE, if Driver and Modbus Master are active. • Bugfix: "Inter-character time bigger than Modbus specification allows. Communication errors with slow CPU cycle-times or high bussystem cycle-times." MB_IL_485P_Master: • xActive = TRUE, if Driver and Modbus Master are active. MB_IL_485P_Slave: • xActive = TRUE, if Driver and Modbus Master are active. MB_IL_UNIxx_Master: • xActive = TRUE, if Driver and Modbus Master are active. • Bugfix: "Inter-character time bigger than Modbus specification allows. Communication errors with slow		

			CPU cycle-times or high bussystem cycle-times." MB_IL_UNIxx_Slave: • xActive = TRUE, if Driver and Modbus Master are active. • Bugfix: "Inter-character time bigger than Modbus specification allows. Communication errors with slow CPU cycle-times or high bussystem cycle-times."	
1	20180412	6.10.200	Extracted from Modbus_3 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_4: • "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: • "Reading wrong count of bits" error is fixed. MB_RTU_FC23: • "Reading one register less than requested" error is fixed MB_RTU_FC (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	"

4 Supported PLCs

AXC 1050 (2700988)
AXC 1050 XC (2701295)
AXC 3050 (2700989)
ILC 130 ETH (2988803)
ILC 131 ETH (2700973)
ILC 131 ETH/XC (2701034)
ILC 150 ETH (2985330)
ILC 151 ETH (2700974)
ILC 151 GSM/GPRS (2700977)
ILC 170 ETH 2TX (2916532)
ILC 171 ETH 2TX (2700975)
ILC 190 ETH 2TX (2700527)
ILC 191 ETH 2TX (2700976)
ILC 191 ME/AN (2700074)
ILC 191 ME/INC (2700075)
ILC 330 ETH (2737193)
ILC 330 PN (2988191)
ILC 350 ETH (2737203)
ILC 350 ETH/M (2985819)
ILC 350 PN (2876928)
ILC 370 ETH 2TX-IB (2876999)
ILC 370 PN 2TX-IB (2876915)
ILC 370 PN 2TX-IB/M (2985576)
ILC 390 PN 2TX-IB (2985314)
RFC 430 ETH-IB (2730190)
RFC 450 ETH-IB (2730200)
RFC 470S PN 3TX (2916794)
RFC 470 PN 3TX (2916600)
RFC 480S PN 4TX (2404577)

Note: The library is released for baud rates from 1200 baud.

5 Function blocks

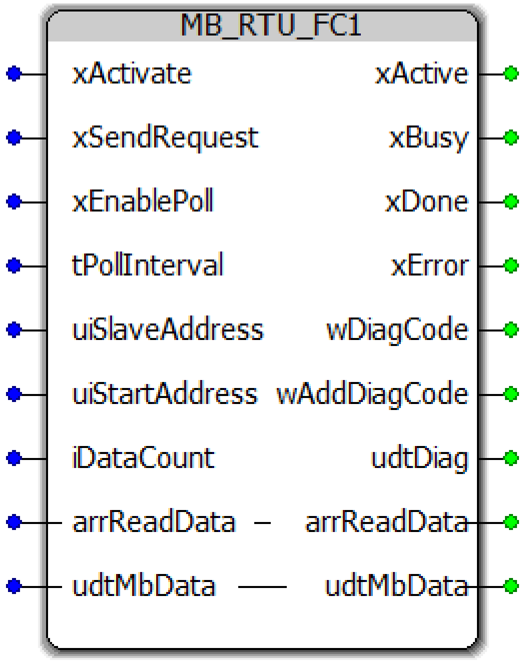
Function block	Description	Version	Supported articles	License
MB_RTU_FC1	This function block reads the status of discrete outputs from a Modbus slave.	7	-	none
MB_RTU_FC2	This function block reads discrete inputs from a Modbus slave.	8	-	none
MB_RTU_FC3	This function block reads holding registers from a Modbus slave.	7	-	none
MB_RTU_FC4	This function block reads input registers from a Modbus slave.	7	-	none
MB_RTU_FC5	This function block writes a single output bit of a Modbus slave.	7	-	none
MB_RTU_FC6	This function block writes a single holding register of a Modbus slave.	7	-	none
MB_RTU_FC15	This function block writes multiple output bits of a Modbus slave.	7	-	none
MB_RTU_FC16	This function block writes multiple holding registers of a Modbus slave.	7	-	none
MB_RTU_FC23	This function block writes or reads multiple holding registers of a Modbus slave.	7	-	none
MB_RTU_DiagInfo_DE	This optional function block displays diagnostic messages of the Modbus master as clear text in German.	3	-	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_SE_RS485_Master	This block runs the sending operations via the AXL SE RS485 (1088128) module.	2	AXL SE RS485 (1088128)	none
MB_AXL_SE_RS485_Slave	This block runs the sending operations via the AXL SE RS485 (1088128) module.	1	AXL SE RS485 (1088128)	none
MB_AXL_F_RSUNI_Master	This block runs the sending operations via the AXL F RS UNI 1H (2688666) module.	2	AXL F RS UNI 1H (2688666)	none
MB_AXL_F_RSUNI_Slave	This block runs the sending operations via the AXL F RS UNI 1H (2688666) module.	1	AXL F RS UNI 1H (2688666)	none
MB_IL_232P_Master	This function block is used to implement a Modbus Master for the specified module type.	6	IB IL RS 232-PRO-PAC (2878722)	none
MB_IL_232P_Slave	This function block is used to implement a Modbus Slave for the specified module type.	4	IB IL RS 232-PRO-PAC (2878722)	none
MB_IL_232E_Master	This function block is used to implement a Modbus Master for the specified module type.	4	IB IL RS 232-ECO (2702141)	none

MB_IL_232E_Slave	This function block is used to implement a Modbus Slave for the specified module type.	3	IB IL RS 232-ECO (2702141)	none
MB_IL_485P_Master	This function block is used to implement a Modbus Master for the specified module type.	6	IB IL RS 485/422-PRO-PAC (2863627)	none
MB_IL_485P_Slave	This function block is used to implement a Modbus Slave for the specified module type.	4	IB IL RS 485/422-PRO-PAC (2863627)	none
MB_IL_485E_Master	This function block is used to implement a Modbus Master for the specified module type.	7	IB IL RS 485-ECO (2702795)	none
MB_IL_485E_Slave	This function block is used to implement a Modbus Slave for the specified module type.	5	IB IL RS 485-ECO (2702795)	none
MB_IL_UNIxx_Master	This function block is used to implement a Modbus Master for the specified module type.	7	IB IL RS UNI-PAC (2700893)	none
MB_IL_UNIxx_Slave	This function block is used to implement a Modbus Slave for the specified module type.	5	IB IL RS UNI-PAC (2700893)	none

6 MB_RTU_FC1

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

6.1 Block call



6.2 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).

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.
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_xxx_DIAG	Structure with internal variables for Diagnostic

6.4 Input and output 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.

6.5 Diagnostic

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.

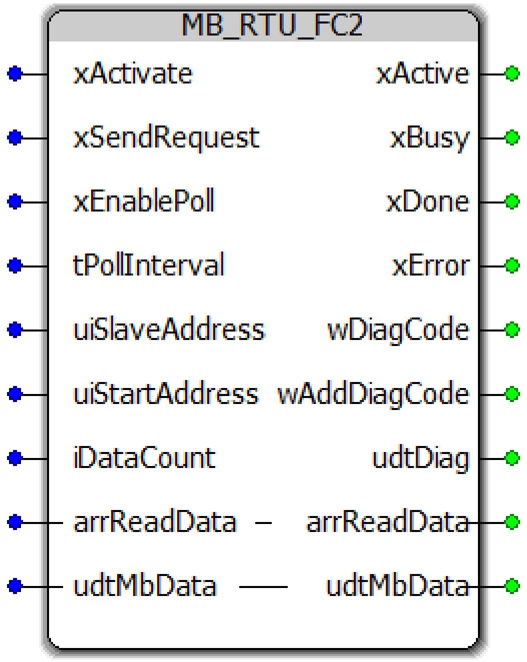
wDiagCode	wAddDiagCode	Description
16#0000		Block is not activated.
16#8000		Block is active and operating without errors.
16#8300		Block executes a service.
16#C100		Error during configuration (displayed on the FC block).
	16#0001	Slave address is outside the valid range.
	16#0002	Number of the requested data amount invalid (iDataCount).
	16#0003	Function code invalid.
	16#0004	Broadcast not possible. FC supports reading function.
16#C110		Modbus error.
	16#0001	Modbus communication timeout.
	16#0002	Modbus 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).
16#C130		Invalid Response.
	16#0001	Slave address of Response invalid.
	16#0002	Function code of Response invalid.
	16#0003	Length of Response invalid.
16#C416		Internal timeout.
	16#0001	Timeout between FC and Modbus_Master.

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 the diagcode description of 16#C010 - 16#C060 refer to the serial master block diagnostic (MB_xx_xx_Master).

7 MB_RTU_FC2

This function block reads discrete inputs from a Modbus slave.

7.1 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_xxx_DIAG	Structure with internal variables for Diagnostic

7.4 Input and output 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 Diagnostic

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.

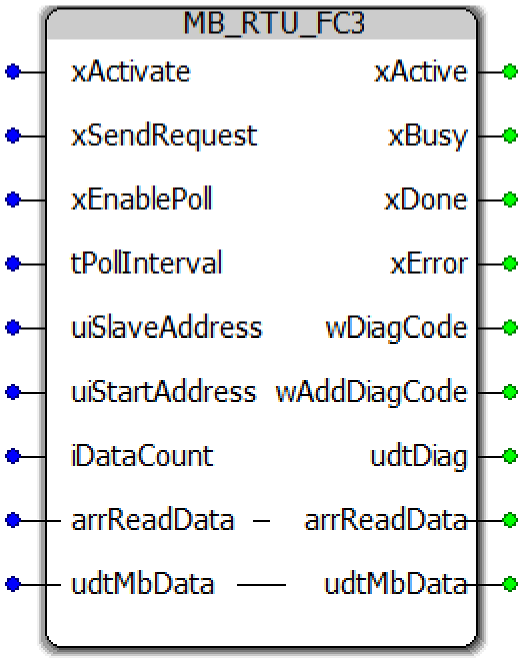
wDiagCode	wAddDiagCode	Description
16#0000		Block is not activated.
16#8000		Block is active and operating without errors.
16#8300		Block executes a service.
16#C100		Error during configuration (displayed on the FC block).
	16#0001	Slave address is outside the valid range.
	16#0002	Number of the requested data amount invalid (iDataCount).
	16#0003	Function code invalid.
	16#0004	Broadcast not possible. FC supports reading function.
16#C110		Modbus error.
	16#0001	Modbus communication timeout.
	16#0002	Modbus 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).
16#C130		Invalid Response.
	16#0001	Slave address of Response invalid.
	16#0002	Function code of Response invalid.
	16#0003	Length of Response invalid.
16#C416		Internal timeout.
	16#0001	Timeout between FC and Modbus_Master.

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 the diagcode description of 16#C010 - 16#C060 refer to the serial master block diagnostic (MB_xx_xx_Master).

8 MB_RTU_FC3

This function block reads holding registers from a Modbus slave.

8.1 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_xxx_DIAG	Structure with internal variables for Diagnostic

8.4 Input and output 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.

8.5 Diagnostic

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.

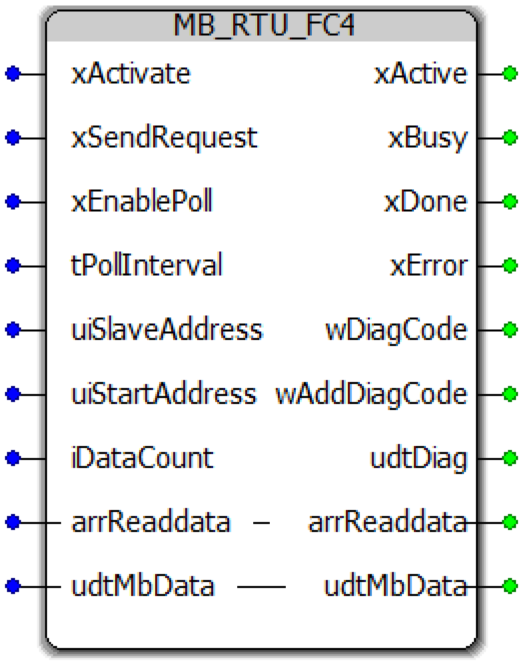
wDiagCode	wAddDiagCode	Description
16#0000		Block is not activated.
16#8000		Block is active and operating without errors.
16#8300		Block executes a service.
16#C100		Error during configuration (displayed on the FC block).
	16#0001	Slave address is outside the valid range.
	16#0002	Number of the requested data amount invalid (iDataCount).
	16#0003	Function code invalid.
	16#0004	Broadcast not possible. FC supports reading function.
16#C110		Modbus error.
	16#0001	Modbus communication timeout.
	16#0002	Modbus 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).
16#C130		Invalid Response.
	16#0001	Slave address of Response invalid.
	16#0002	Function code of Response invalid.
	16#0003	Length of Response invalid.
16#C416		Internal timeout.
	16#0001	Timeout between FC and Modbus_Master.

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 the diagcode description of 16#C010 - 16#C060 refer to the serial master block diagnostic (MB_xx_xx_Master).

9 MB_RTU_FC4

This function block reads input registers from a Modbus slave.

9.1 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_xxx_DIAG	Structure with internal variables for Diagnostic

9.4 Input and output 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 Diagnostic

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.

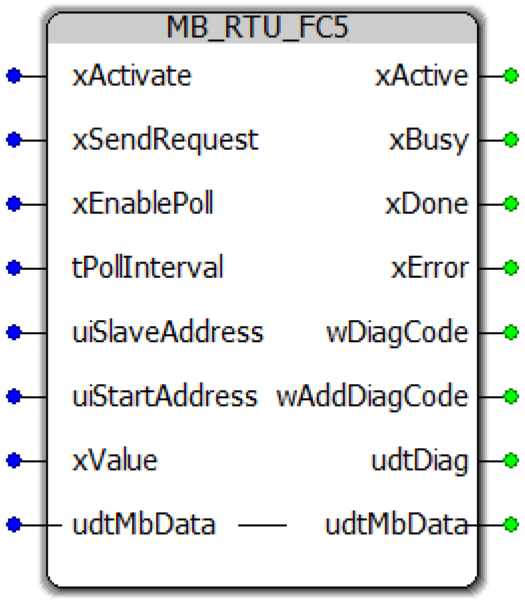
wDiagCode	wAddDiagCode	Description
16#0000		Block is not activated.
16#8000		Block is active and operating without errors.
16#8300		Block executes a service.
16#C100		Error during configuration (displayed on the FC block).
	16#0001	Slave address is outside the valid range.
	16#0002	Number of the requested data amount invalid (iDataCount).
	16#0003	Function code invalid.
	16#0004	Broadcast not possible. FC supports reading function.
16#C110		Modbus error.
	16#0001	Modbus communication timeout.
	16#0002	Modbus 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).
16#C130		Invalid Response.
	16#0001	Slave address of Response invalid.
	16#0002	Function code of Response invalid.
	16#0003	Length of Response invalid.
16#C416		Internal timeout.
	16#0001	Timeout between FC and Modbus_Master.

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 the diagcode description of 16#C010 - 16#C060 refer to the serial master block diagnostic (MB_xx_xx_Master).

10 MB_RTU_FC5

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

10.1 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.
xValue	BOOL	The status of the input is written in the memory to be written.

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_xxx_DIAG	Structure with internal variables for Diagnostic

10.4 Input and output parameters

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

10.5 Diagnostic

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.

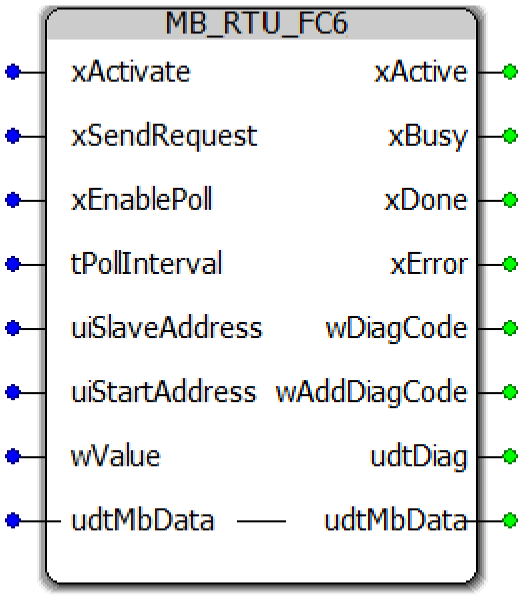
wDiagCode	wAddDiagCode	Description
16#0000		Block is not activated.
16#8000		Block is active and operating without errors.
16#8300		Block executes a service.
16#C100		Error during configuration (displayed on the FC block).
	16#0001	Slave address is outside the valid range.
	16#0002	Number of the requested data amount invalid (iDataCount).
	16#0003	Function code invalid.
	16#0004	Broadcast not possible. FC supports reading function.
16#C110		Modbus error.
	16#0001	Modbus communication timeout.
	16#0002	Modbus 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).
16#C130		Invalid Response.
	16#0001	Slave address of Response invalid.
	16#0002	Function code of Response invalid.
	16#0003	Length of Response invalid.
16#C416		Internal timeout.
	16#0001	Timeout between FC and Modbus_Master.

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 the diagcode description of 16#C010 - 16#C060 refer to the serial master block diagnostic (MB_xx_xx_Master).

11 MB_RTU_FC6

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

11.1 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.
wValue	WORD	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_xxx_DIAG	Structure with internal variables for Diagnostic

11.4 Input and output parameters

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

11.5 Diagnostic

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.

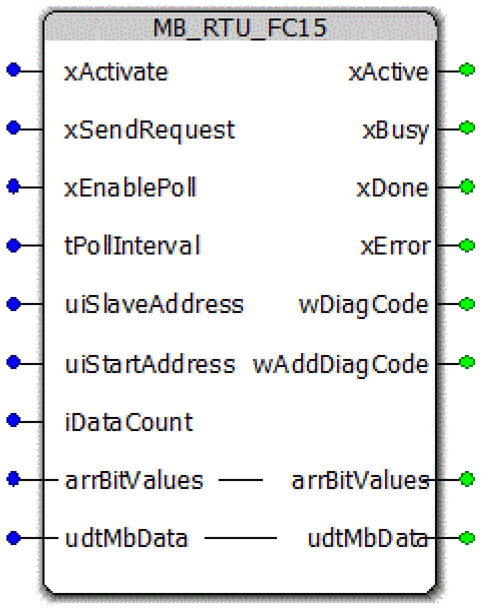
wDiagCode	wAddDiagCode	Description
16#0000		Block is not activated.
16#8000		Block is active and operating without errors.
16#8300		Block executes a service.
16#C100		Error during configuration (displayed on the FC block).
	16#0001	Slave address is outside the valid range.
	16#0002	Number of the requested data amount invalid (iDataCount).
	16#0003	Function code invalid.
	16#0004	Broadcast not possible. FC supports reading function.
16#C110		Modbus error.
	16#0001	Modbus communication timeout.
	16#0002	Modbus 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).
16#C130		Invalid Response.
	16#0001	Slave address of Response invalid.
	16#0002	Function code of Response invalid.
	16#0003	Length of Response invalid.
16#C416		Internal timeout.
	16#0001	Timeout between FC and Modbus_Master.

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 the diagcode description of 16#C010 - 16#C060 refer to the serial master block diagnostic (MB_xx_xx_Master).

12 MB_RTU_FC15

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

12.1 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.
iDataCount	INT	The input specifies the number of bits to be read on the slave (1 to 1968).

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_xxx_DIAG	Structure with internal variables for Diagnostic

12.4 Input and output 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.

12.5 Diagnostic

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.

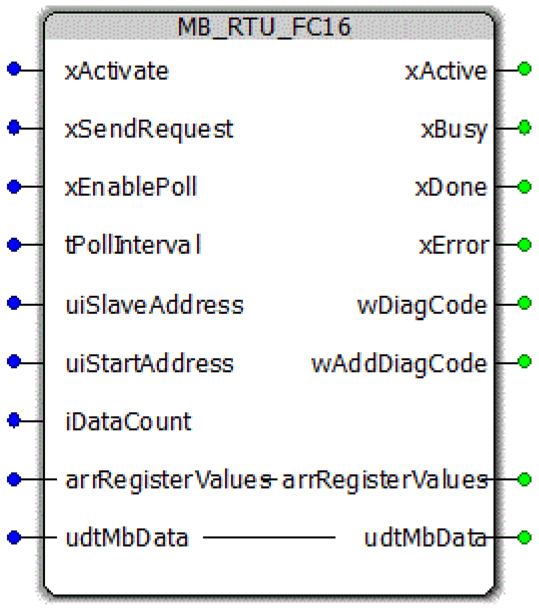
wDiagCode	wAddDiagCode	Description
16#0000		Block is not activated.
16#8000		Block is active and operating without errors.
16#8300		Block executes a service.
16#C100		Error during configuration (displayed on the FC block).
	16#0001	Slave address is outside the valid range.
	16#0002	Number of the requested data amount invalid (iDataCount).
	16#0003	Function code invalid.
	16#0004	Broadcast not possible. FC supports reading function.
16#C110		Modbus error.
	16#0001	Modbus communication timeout.
	16#0002	Modbus 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).
16#C130		Invalid Response.
	16#0001	Slave address of Response invalid.
	16#0002	Function code of Response invalid.
	16#0003	Length of Response invalid.
16#C416		Internal timeout.
	16#0001	Timeout between FC and Modbus_Master.

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 the diagcode description of 16#C010 - 16#C060 refer to the serial master block diagnostic (MB_xx_xx_Master).

13 MB_RTU_FC16

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

13.1 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 written on the slave (1 to 123).

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_xxx_DIAG	Structure with internal variables for Diagnostic

13.4 Input and output 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.

13.5 Diagnostic

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.

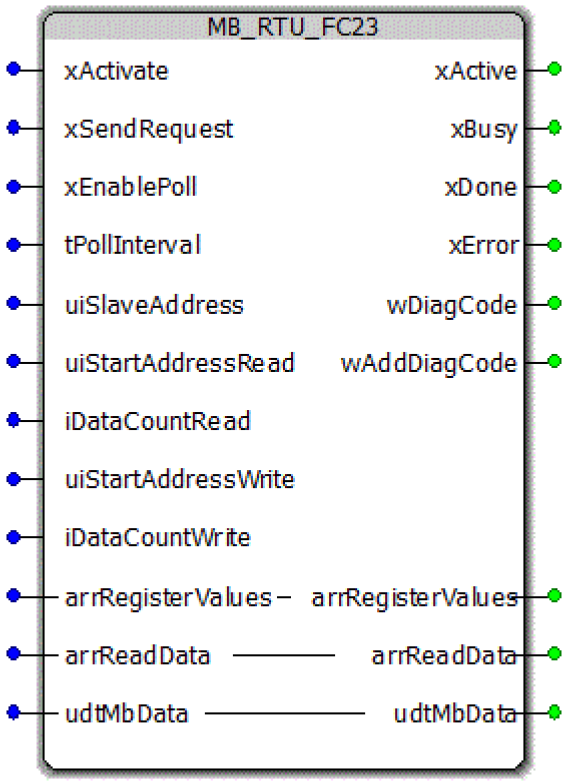
wDiagCode	wAddDiagCode	Description
16#0000		Block is not activated.
16#8000		Block is active and operating without errors.
16#8300		Block executes a service.
16#C100		Error during configuration (displayed on the FC block).
	16#0001	Slave address is outside the valid range.
	16#0002	Number of the requested data amount invalid (iDataCount).
	16#0003	Function code invalid.
	16#0004	Broadcast not possible. FC supports reading function.
16#C110		Modbus error.
	16#0001	Modbus communication timeout.
	16#0002	Modbus 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).
16#C130		Invalid Response.
	16#0001	Slave address of Response invalid.
	16#0002	Function code of Response invalid.
	16#0003	Length of Response invalid.
16#C416		Internal timeout.
	16#0001	Timeout between FC and Modbus_Master.

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 the diagcode description of 16#C010 - 16#C060 refer to the serial master block diagnostic (MB_xx_xx_Master).

14 MB_RTU_FC23

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

14.1 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).
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).

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_xxx_DIAG	Structure with internal variables for Diagnostic

14.4 Input and output 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.

14.5 Diagnostic

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.

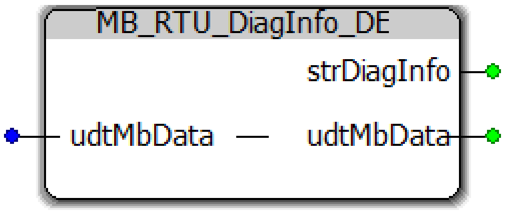
wDiagCode	wAddDiagCode	Description
16#0000		Block is not activated.
16#8000		Block is active and operating without errors.
16#8300		Block executes a service.
16#C100		Error during configuration (displayed on the FC block).
	16#0001	Slave address is outside the valid range.
	16#0002	Number of the requested data amount invalid (iDataCount).
	16#0003	Function code invalid.
	16#0004	Broadcast not possible. FC supports reading function.
16#C110		Modbus error.
	16#0001	Modbus communication timeout.
	16#0002	Modbus 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).
16#C130		Invalid Response.
	16#0001	Slave address of Response invalid.
	16#0002	Function code of Response invalid.
	16#0003	Length of Response invalid.
16#C416		Internal timeout.
	16#0001	Timeout between FC and Modbus_Master.

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 the diagcode description of 16#C010 - 16#C060 refer to the serial master block diagnostic (MB_xx_xx_Master).

15 MB_RTU_DiagInfo_DE

If there is an error, this block shows the diagnostics of the master block as a text in German. 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.

15.1 Block call



15.2 Input parameters

None

15.3 Output parameters

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

15.4 Input and output parameters

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

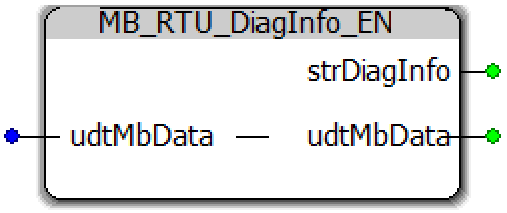
15.5 Diagnostic

None

16 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.

16.1 Block call



16.2 Input parameters

None

16.3 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.

16.4 Input and output parameters

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

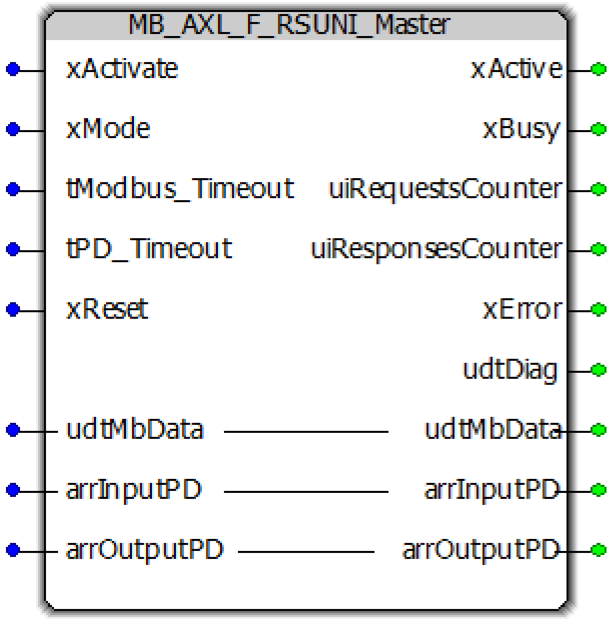
16.5 Diagnostic

None

17 MB_AXL_F_RSUNI_Master

This function block is used to implement a Modbus Master for the specified module type. Accordingly the function blocks are connected inside. The required parameters have to be parameterized on this function block. The associated parameter description refers to the description of the included function blocks.

17.1 Function block call



17.2 Input parameters

Name	Type	Description
xActivate	BOOL	Block activation (TRUE = Active).
xMode	BOOL	TRUE: AXL F RS UNI Module parameterized in Modbus RTU mode. FALSE: AXL F RS UNI Module parameterized in Transparent mode.
tModbus_Timeout	TIME	Defines the maximum time until the Modbus response from the slave must arrive. The default value: TIME#5s (if input is 0s). The input is copied by xActivate or xReset if there is a rising edge.
tPD_Timeout	TIME	Timeout for process data communication between PLC and module. The input is copied by xActivate or xReset if there is a rising edge. Default: TIME#2s
xReset	BOOL	The input resets the block. All connected FC blocks are reset as well.

17.3 Output parameters

Name	Type	Description
xActive	BOOL	TRUE: Function block is active. FALSE: Function block is not active.
xError	BOOL	TRUE: An error has occurred. For details refer to udtDiag strucure “wDiagCode” and “wAddDiagCode”.
xBusy	BOOL	TRUE: The block is busy with the service execution.
uiRequestsCounter	UINT	Shows the number of requests transmitted.
uiResponsesCounter	UINT	Shows the number of responses received.
udtDiag	MB_UDT_AXL_RSUNI_DIAG_MASTER	Structure with internal structures for Diagnostic

17.4 Inout parameters

Name	Type	Description
udtMBData	udtModbus2_Data	The block communicates via this structure with the FC blocks.
arrInputPD	MB2_AXL_RSUNI2_ARR_B_0_19	IN process data.
arrOutputPD	MB2_AXL_RSUNI2_ARR_B_0_19	OUT process data.

17.5 Diagnosis

17.5.1 MB_RTU_Master

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

17.5.2 MB_AXL_RS_UNI_REC

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	Process data timeout during the receive process between PLC and module.
	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#0070	Error could not acknowledged.
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

17.5.3 MB_AXL_RS_UNI_SND

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#0010	Process data timeout during the send process between PLC and module.
	16#0020	Maximum size exceeded when sending.
	16#0060	Data send error in module.
	16#0070	Error could not acknowledged.
16#C030		Error when receiving.
	16#0060	Communication error when receiving.
16#C040		Error in intermediate storage.
	16#0010	Timeout in intermediate storage.

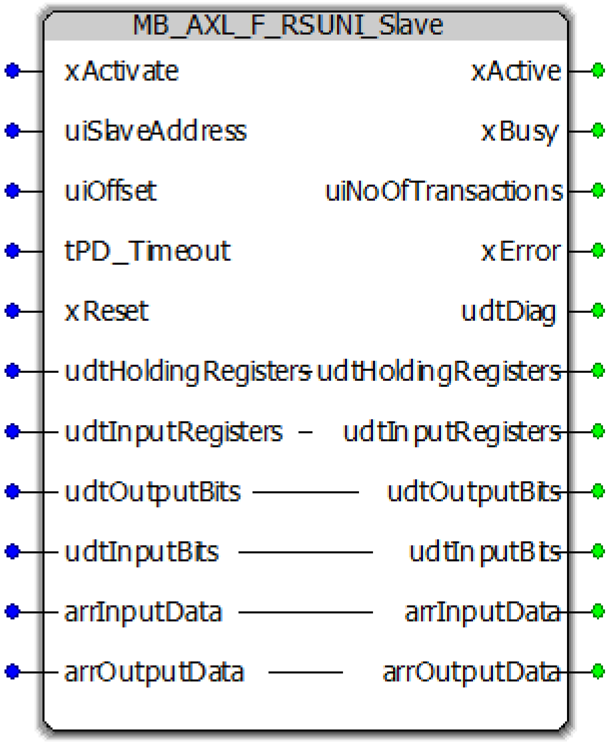
16#C050	16#0000	Error from module: • Failure of the peripheral voltage • Invalid parameter for specified command

18 MB_AXL_F_RSUNI_Slave

This function block is used to implement a Modbus Slave for the specified module type. Accordingly the function blocks are connected inside. The required parameters have to be parameterized on this function block. The associated parameter description refers to the description of the included function blocks.

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

18.1 Function block call



18.2 Input parameters

Name	Type	Description
xActivate	BOOL	Block activation (TRUE = Active).
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).
tPD_Timeout	TIME	Timeout for process data communication between PLC and module. The input is copied by xActivate or xReset if there is a rising edge. Default: TIME#2s
xReset	BOOL	The input resets the block. All connected FC blocks are reset as well.

18.3 Output parameters

Name	Type	Description
xActive	BOOL	TRUE: Function block is active. FALSE: Function block is not active.
xError	BOOL	TRUE: An error has occurred. For details refer to udtDiag strucure “wDiagCode” and “wAddDiagCode”.
xBusy	BOOL	TRUE: The block is busy with the service execution.
uiNoOfTransactions	UINT	Number of processed requests
udtDiag	MB_UDT_AXL_RSUNI_DIAG_SLAVE	Structure with internal structures for Diagnostic

18.4 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
arrInputPD	MB2_AXL_RSUNI2_ARR_B_0_19	IN process data.
arrOutputPD	MB2_AXL_RSUNI2_ARR_B_0_19	OUT process data.

18.5 Diagnosis

18.5.1 MB_RTU_Slave

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

18.5.2 MB_AXL_RS_UNI_REC

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	Process data timeout during the receive process between PLC and module.
	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#0070	Error could not acknowledged.
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

18.5.3 MB_AXL_RS_UNI_SND

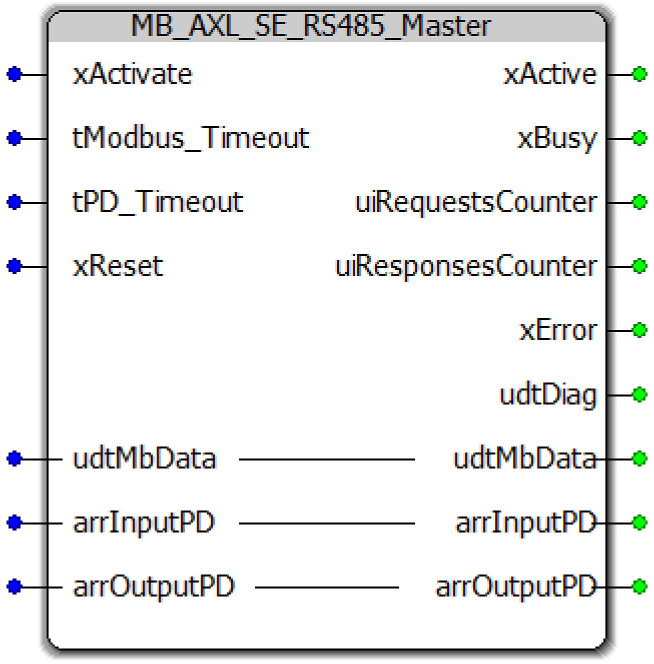
wDiagCode	wAddDiagCode	Description
16#0000	16#0000	Block is not activated.
16#8000	16#0000	Block is active and operating without errors.
16#C020		Error when sending.
	16#0010	Process data timeout during the send process between PLC and module.
	16#0020	Maximum size exceeded when sending.
	16#0060	Data send error in module.
	16#0070	Error could not acknowledged.
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

19 MB_AXL_SE_RS485_Master

This function block is used to implement a Modbus Master for the specified module type. Accordingly the function blocks are connected inside. The required parameters have to be parameterized on this function block. The associated parameter description refers to the description of the included function blocks.

19.1 Function block call



19.2 Input parameters

Name	Type	Description
xActivate	BOOL	Block activation (TRUE = Active).
tModbus_Timeout	TIME	Defines the maximum time until the Modbus response from the slave must arrive. The default value: TIME#5s (if input is 0s). The input is copied by xActivate or xReset if there is a rising edge.
tPD_Timeout	TIME	Timeout for process data communication between PLC and module. The input is copied by xActivate or xReset if there is a rising edge. Default: TIME#2s
xReset	BOOL	The input resets the block. All connected FC blocks are reset as well.

19.3 Output parameters

Name	Type	Description
xActive	BOOL	TRUE: Function block is active. FALSE: Function block is not active.
xError	BOOL	TRUE: An error has occurred. For details refer to udtDiag strucure “wDiagCode” and “wAddDiagCode”.
xBusy	BOOL	TRUE: The block is busy with the service execution.
uiRequestsCounter	UINT	Shows the number of requests transmitted.
uiResponsesCounter	UINT	Shows the number of responses received.
udtDiag	MB_UDT_AXL_SE_RS485_DIAG_MASTER	Structure with internal structures for Diagnostic

19.4 Inout parameters

Name	Type	Description
udtMBData	udtModbus2_Data	The block communicates via this structure with the FC blocks.
arrInputAddress	MB2_RSUNI_ARR_B_1_xx	IN process data.
arrOutputAddress	MB2_RSUNI_ARR_B_1_xx	OUT process data.

19.5 Diagnosis

19.5.1 MB_RTU_Master

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

19.5.2 MB_AXL_RS_UNI_REC

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	Process data timeout during the receive process between PLC and module.
	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#0070	Error could not acknowledged.
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

19.5.3 MB_AXL_RS_UNI_SND

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#0010	Process data timeout during the send process between PLC and module.
	16#0020	Maximum size exceeded when sending.
	16#0060	Data send error in module.
	16#0070	Error could not acknowledged.
16#C030		Error when receiving.
	16#0060	Communication error when receiving.
16#C040		Error in intermediate storage.
	16#0010	Timeout in intermediate storage.

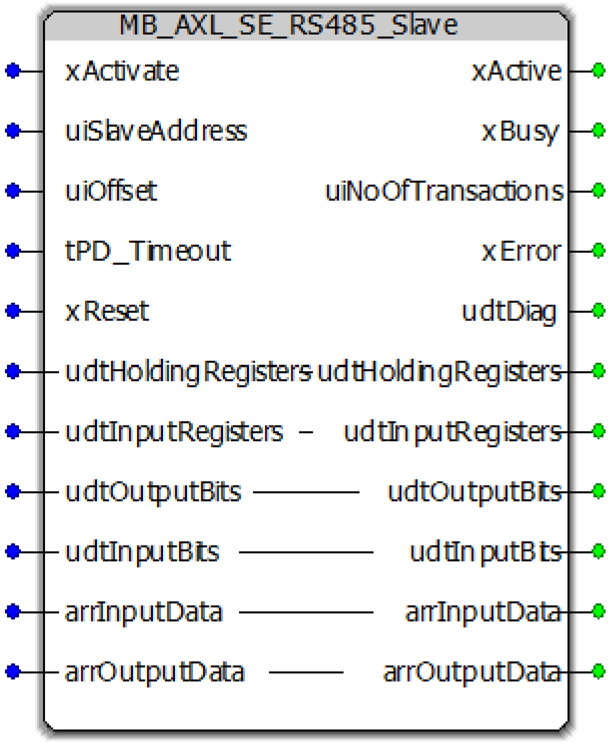
16#C050	16#0000	Error from module: • Failure of the peripheral voltage • Invalid parameter for specified command

20 MB_AXL_SE_RS485_Slave

This function block is used to implement a Modbus Slave for the specified module type. Accordingly the function blocks are connected inside. The required parameters have to be parameterized on this function block. The associated parameter description refers to the description of the included function blocks.

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

20.1 Function block call



20.2 Input parameters

Name	Type	Description
xActivate	BOOL	Block activation (TRUE = Active).
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).
tPD_Timeout	TIME	Timeout for process data communication between PLC and module. The input is copied by xActivate or xReset if there is a rising edge. Default: TIME#2s
xReset	BOOL	The input resets the block. All connected FC blocks are reset as well.

20.3 Output parameters

Name	Type	Description
xActive	BOOL	TRUE: Function block is active. FALSE: Function block is not active.
xError	BOOL	TRUE: An error has occurred. For details refer to udtDiag strucure “wDiagCode” and “wAddDiagCode”.
xBusy	BOOL	TRUE: The block is busy with the service execution.
uiNoOfTransactions	UINT	Number of processed requests
udtDiag	MB_UDT_AXL_SE_RS485_DIAG_SLAVE	Structure with internal structures for Diagnostic

20.4 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
arrInputAddress	MB2_RSUNI_ARR_B_1_xx	IN process data.
arrOutputAddress	MB2_RSUNI_ARR_B_1_xx	OUT process data.

20.5 Diagnosis

20.5.1 MB_RTU_Slave

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

20.5.2 MB_AXL_RS_UNI_REC

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	Process data timeout during the receive process between PLC and module.
	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#0070	Error could not acknowledged.
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

20.5.3 MB_AXL_RS_UNI_SND

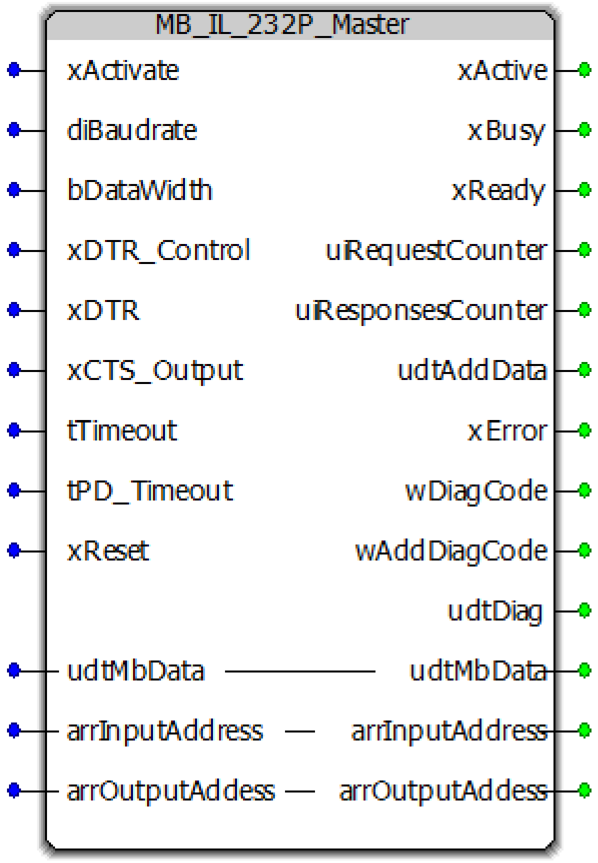
wDiagCode	wAddDiagCode	Description
16#0000	16#0000	Block is not activated.
16#8000	16#0000	Block is active and operating without errors.
16#C020		Error when sending.
	16#0010	Process data timeout during the send process between PLC and module.
	16#0020	Maximum size exceeded when sending.
	16#0060	Data send error in module.
	16#0070	Error could not acknowledged.
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

21 MB_IL_232P_Master

This function block is used to implement a Modbus Master for the specified module type. Accordingly the function blocks are connected inside. The required parameters have to be parameterized on this function block. The associated parameter description refers to the description of the included function blocks.

21.1 Block call



21.2 Input parameters

Name	Type	Description
xActivate	BOOL	Block activation (TRUE = Active).
diBaudrate	DINT	Baud rate in the range from 110 baud to 500 kbaud.
bDataWidth	BYTE	0 hex = 7 data bits, even, 1 stop bit 1 hex = 7 data bits, odd, 1 stop bit 2 hex = 8 data bits, even, 1 stop bit 3 hex = 8 data bits, odd, 1 stop bit 4 hex = 8 data bits, none, 1 stop bit 5 hex = 7 data bits, none, 1 stop bit 6 hex = 7 data bits, even, 2 stop bits 7 hex = 7 data bits, odd, 2 stop bits 8 hex = 8 data bits, even, 2 stop bits 9 hex = 8 data bits, odd, 2 stop bits A hex = 8 data bits, none, 2 stop bits B hex = 7 data bits, none, 2 stop bits
xDTR_Control	BOOL	FALSE: The DTR signal is controlled automatically. TRUE: The DTR signal is controlled by the user.
xDTR	BOOL	The DTR signal is controlled. Only active if the corresponding mode is activated (implemented via the parameterization blocks).
xCTS_Output	BOOL	FALSE: CTS signal is not output. TRUE: CTS signal is output.
tTimeout	TIME	Defines the maximum time until the Modbus response from the slave must arrive. The default value: TIME#5s (if input is 0s). The input is copied by xActivate or xReset if there is a rising edge.
tPD_Timeout	TIME	Timeout for process data communication between PLC and module. The input is copied by xActivate or xReset if there is a rising edge. Default: TIME#2s
xReset	BOOL	The input resets the block. All connected FC blocks are reset as well.

21.3 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.
xReady	BOOL	The block is ready to execute services. When executing services, this parameter is FALSE.
uiRequestsCounter	UINT	Shows the number of requests transmitted.
uiResponsesCounter	UINT	Shows the number of responses received.
udtAddData	MB2_COM_UDT_R232P_DATA_V1	Structure with additional status variables.
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_xxx_DIAG	Structure with internal variables for Diagnostic

21.4 Input and output parameters

Name	Type	Description
udtMBData	udtModbus2_Data	The block communicates via this structure with the FC blocks.
arrInputAddress	MB2_COM_ARR_B_1_12	IN process data.
arrOutputAddress	MB2_COM_ARR_B_1_12	OUT process data.

21.5 Diagnostic

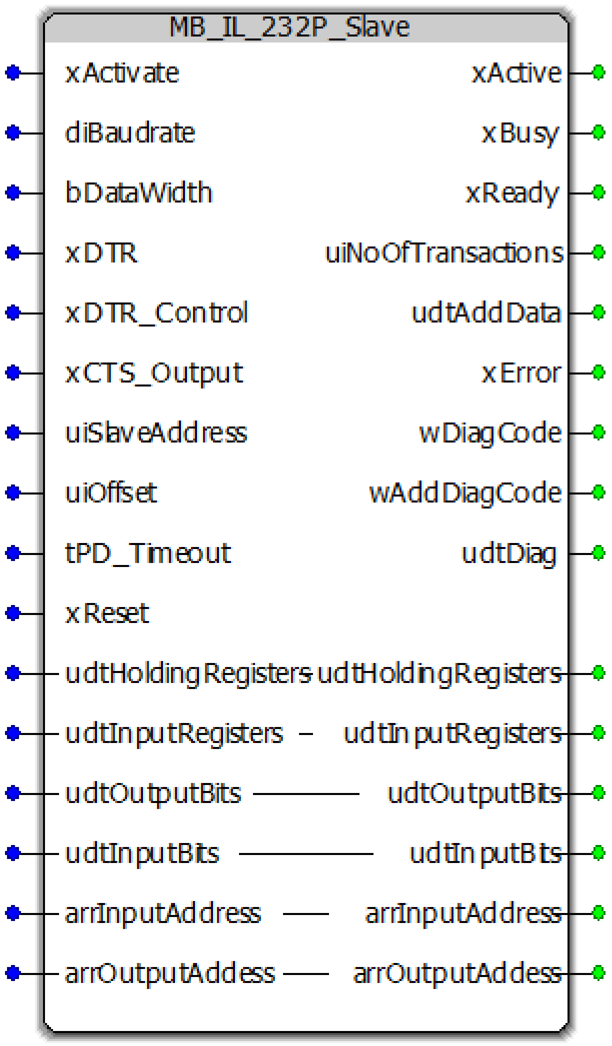
wDiagCode	wAddDiagCode	Description
16#0000		Block is not activated.
16#8000		Block is active and operating without errors.
16#8100		HW Reset phase to delete serial driver error
16#C010		Incorrect terminal type connected.
	16#00XX	Read terminal type.
	16#FFFF	Terminal is not responding.
16#C020		Incorrect parameter.
	16#0010	Baud rate.
	16#0020	Data width.
	16#0030	Protocol.
	16#0040	Terminal configuration error.
	16#0100	xReceive and xSend inputs are set at the same time.
	16#0110	xReceive input is set during send procedure.
	16#0120	xSend input is set during receive procedure.
16#C030		Error when sending.
	16#0010	Process data timeout during the send process between PLC and module.
	16#0020	Maximum size when sending exceeded.
	16#0030	uiSendLength too large.
16#C040		Error when receiving.
	16#0010	Process data timeout during the receive process between PLC and module.
	16#0020	Maximum size when receiving exceeded.
	16#0030	uiRcvLength too large.
	16#0040	uiRcvLength <> 0 for the end-to-end, 3964R, and dual buffer protocols.
16#C050		3964R protocol error.
	16#0010	Error when sending a 3964R telegram.
	16#0020	Error when receiving a 3964R telegram.

22 MB_IL_232P_Slave

This function block is used to implement a Modbus Slave for the specified module type. Accordingly the function blocks are connected inside. The required parameters have to be parameterized on this function block. The associated parameter description refers to the description of the included function blocks.

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

22.1 Block call



22.2 Input parameters

Name	Type	Description
xActivate	BOOL	Block activation (TRUE = Active).
diBaudrate	DINT	Baud rate in the range from 110 baud to 500 kbaud.
bDataWidth	BYTE	0 hex = 7 data bits, even, 1 stop bit 1 hex = 7 data bits, odd, 1 stop bit 2 hex = 8 data bits, even, 1 stop bit 3 hex = 8 data bits, odd, 1 stop bit 4 hex = 8 data bits, none, 1 stop bit 5 hex = 7 data bits, none, 1 stop bit 6 hex = 7 data bits, even, 2 stop bits 7 hex = 7 data bits, odd, 2 stop bits 8 hex = 8 data bits, even, 2 stop bits 9 hex = 8 data bits, odd, 2 stop bits A hex = 8 data bits, none, 2 stop bits B hex = 7 data bits, none, 2 stop bits
xDTR	BOOL	The DTR signal is controlled. Only active if the corresponding mode is activated (implemented via the parameterization blocks).
xDTR_Control	BOOL	FALSE: The DTR signal is controlled automatically. TRUE: The DTR signal is controlled by the user.
xCTS_Output	BOOL	FALSE: CTS signal is not output. TRUE: CTS signal is output.
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).
tPD_Timeout	TIME	Timeout for process data communication between PLC and module. The input is copied by xActivate or xReset if there is a rising edge. Default: TIME#2s
xReset	BOOL	The input resets the block. All connected FC blocks are reset as well.

22.3 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.
xReady	BOOL	The block is ready to execute services. When executing services, this parameter is FALSE.
uiNoOfTransactions	UINT	Number of processed requests
udtAddData	MB2_COM_UDT_R232P_DATA_V1	Structure with additional status variables.
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_xxx_DIAG	Structure with internal variables for Diagnostic

22.4 Input and output 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
arrInputAddress	MB2_COM_ARR_B_1_12	IN process data.
arrOutputAddress	MB2_COM_ARR_B_1_12	OUT process data.

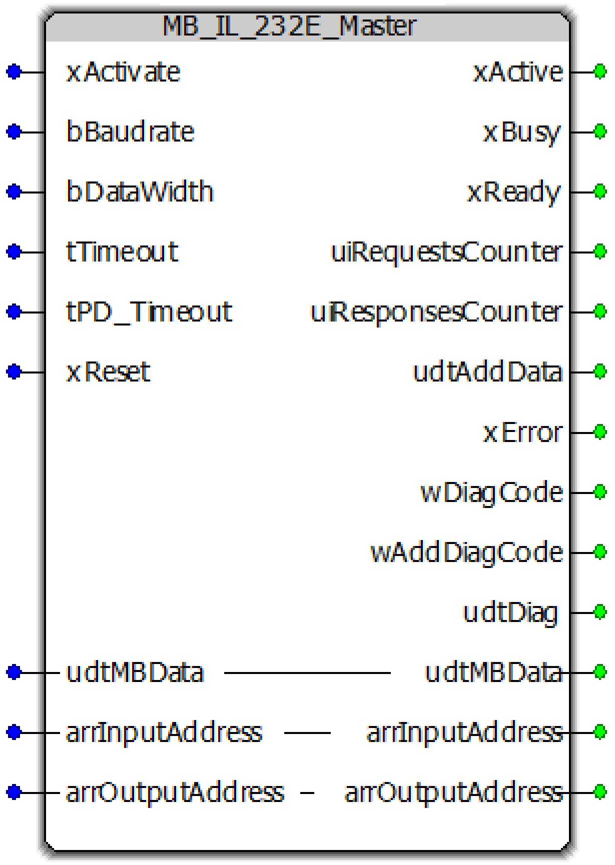
22.5 Diagnostic

wDiagCode	wAddDiagCode	Description
16#0000		Block is not activated.
16#8000		Block is active and operating without errors.
16#C010		Incorrect terminal type connected.
	16#00XX	Read terminal type.
	16#FFFF	Terminal is not responding.
16#C020		Incorrect parameter.
	16#0010	Baud rate.
	16#0020	Data width.
	16#0030	Protocol.
	16#0040	Terminal configuration error.
	16#0100	xReceive and xSend inputs are set at the same time.
	16#0110	xReceive input is set during send procedure.
	16#0120	xSend input is set during receive procedure.
16#C030		Error when sending.
	16#0010	Process data timeout during the send process between PLC and module.
	16#0020	Maximum size when sending exceeded.
	16#0030	uiSendLength too large.
16#C040		Error when receiving.
	16#0010	Process data timeout during the receive process between PLC and module.
	16#0020	Maximum size when receiving exceeded.
	16#0030	uiRcvLength too large.
	16#0040	uiRcvLength <> 0 for the end-to-end, 3964R, and dual buffer protocols.
16#C050		3964R protocol error.
	16#0010	Error when sending a 3964R telegram.
	16#0020	Error when receiving a 3964R telegram.

23 MB_IL_232E_Master

This function block is used to implement a Modbus Master for the specified module type. Accordingly the function blocks are connected inside. The required parameters have to be parameterized on this function block. The associated parameter description refers to the description of the included function blocks.

23.1 Block call



23.2 Input parameters

Name	Type	Description
xActivate	BOOL	Block activation (TRUE = Active).
bBaudrate	BYTE	0 hex = Baudrate 110 1 hex = Baudrate 300 2 hex = Baudrate 600 3 hex = Baudrate 1200 4 hex = Baudrate 1800 5 hex = Baudrate 2400 6 hex = Baudrate 4800 7 hex = Baudrate 9600 8 hex = Baudrate 15625 9 hex = Baudrate 19200 A hex = Baudrate 38400 B-F hex = reserved

bDataWidth	BYTE	0 hex = 7 data bits, even, 1 stop bit 1 hex = 7 data bits, odd, 1 stop bit 2 hex = 8 data bits, even, 1 stop bit 3 hex = 8 data bits, odd, 1 stop bit 4 hex = 8 data bits, none, 1 stop bit 5 hex = 7 data bits, none, 1 stop bit 6 hex = 7 data bits, even, 2 stop bits 7 hex = 7 data bits, odd, 2 stop bits 8 hex = 8 data bits, even, 2 stop bits 9 hex = 8 data bits, odd, 2 stop bits A hex = 8 data bits, none, 2 stop bits B hex = 7 data bits, none, 2 stop bits C hex = 8 data bits, constant at 0, 1 stop bits D hex = 8 data bits, constant at 1, 1 stop bits E hex = 6 data bits, none, 1 stop bits F hex = Reserved
tTimeout	TIME	Defines the maximum time until the Modbus response from the slave must arrive. The default value: TIME#5s (if input is 0s). The input is copied by xActivate or xReset if there is a rising edge.
tPD_Timeout	TIME	Timeout for process data communication between PLC and module. The input is copied by xActivate or xReset if there is a rising edge. Default: TIME#2s
xReset	BOOL	The input resets the block. All connected FC blocks are reset as well.

23.3 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.
xReady	BOOL	The block is ready to execute services. When executing services, this parameter is FALSE.
uiRequestsCounter	UINT	Shows the number of requests transmitted.
uiResponsesCounter	UINT	Shows the number of responses received.
udtAddData	MB2_RSUNI_UDT_DATA_V1	Structure with variables for diagnostics
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_xxx_DIAG	Structure with internal variables for Diagnostic

23.4 Input and output parameters

Name	Type	Description
udtMBData	udtModbus2_Data	The block communicates via this structure with the FC blocks.
arrInputAddress	MB2_RSUNI_ARR_B_1_xx	IN process data.
arrOutputAddress	MB2_RSUNI_ARR_B_1_xx	OUT process data.

23.5 Diagnostic

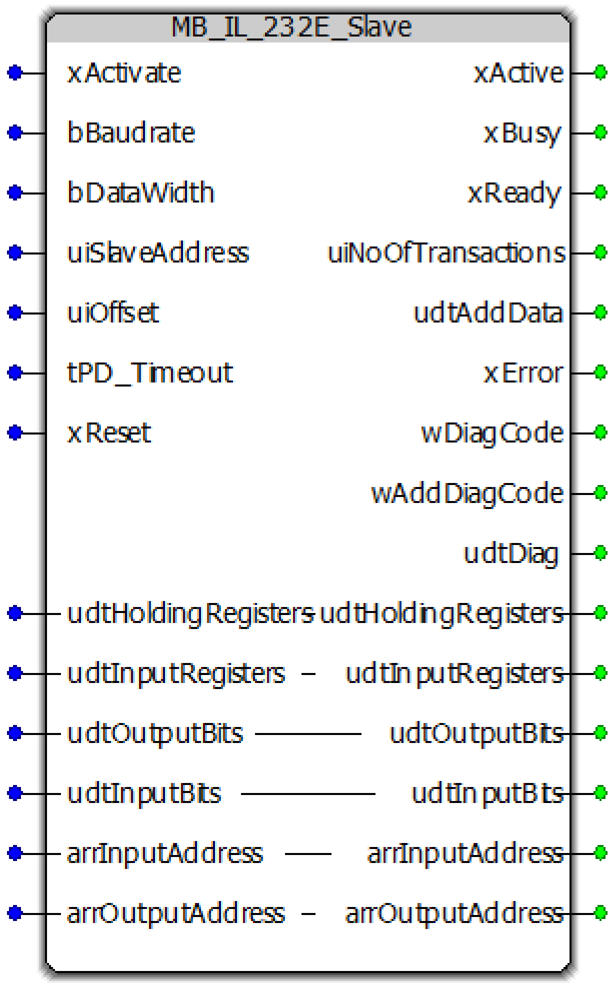
wDiagCode	wAddDiagCode	Description
16#0000		Block is not activated.
16#8000		Block is active and operating without errors.
16#8100		HW Reset phase to delete serial driver error
16#C010		Incorrect parameters.
	16#0010	Baud rate.
	16#0020	Data width.
	16#0050	Terminal configuration error.
	16#0060	Communication error.
	16#0080	xReceive and xSend inputs are set at the same time.
	16#0090	xReceive input is set during send procedure.
	16#0100	xSend input is set during receive procedure.
16#C020		Error when sending.
	16#0020	Maximum size exceeded.
	16#0060	Process data timeout during the send process between PLC and module.
16#C030		Error when receiving.
	16#0030	uiRcvLength is larger than the memory available in the receive buffer.
	16#0060	Process data timeout during the receive process between PLC and module.

24 MB_IL_232E_Slave

This function block is used to implement a Modbus Slave for the specified module type. Accordingly the function blocks are connected inside. The required parameters have to be parameterized on this function block. The associated parameter description refers to the description of the included function blocks.

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

24.1 Block call



24.2 Input parameters

Name	Type	Description
xActivate	BOOL	Block activation (TRUE = Active).
diBaudrate	DINT	Baud rate in the range from 110 baud to 500 kbaud.
bDataWidth	BYTE	0 hex = 7 data bits, even, 1 stop bit 1 hex = 7 data bits, odd, 1 stop bit 2 hex = 8 data bits, even, 1 stop bit 3 hex = 8 data bits, odd, 1 stop bit 4 hex = 8 data bits, none, 1 stop bit 5 hex = 7 data bits, none, 1 stop bit 6 hex = 7 data bits, even, 2 stop bits 7 hex = 7 data bits, odd, 2 stop bits 8 hex = 8 data bits, even, 2 stop bits 9 hex = 8 data bits, odd, 2 stop bits A hex = 8 data bits, none, 2 stop bits B hex = 7 data bits, none, 2 stop bits C hex = 8 data bits, constant at 0, 1 stop bits D hex = 8 data bits, constant at 1, 1 stop bits E hex = 6 data bits, none, 1 stop bits F hex = Reserved
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).
tPD_Timeout	TIME	Timeout for process data communication between PLC and module. The input is copied by xActivate or xReset if there is a rising edge. Default: TIME#2s
xReset	BOOL	The input resets the block. All connected FC blocks are reset as well.

24.3 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.
xReady	BOOL	The block is ready to execute services. When executing services, this parameter is FALSE.
uiNoOfTransactions	UINT	Number of processed requests
udtAddData	MB2_RSUNI_UDT_DATA_V1	Structure with variables for diagnostics
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_xxx_DIAG	Structure with internal variables for Diagnostic

24.4 Input and output 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
arrInputAddress	MB2_RSUNI_ARR_B_1_xx	IN process data.
arrOutputAddress	MB2_RSUNI_ARR_B_1_xx	OUT process data.

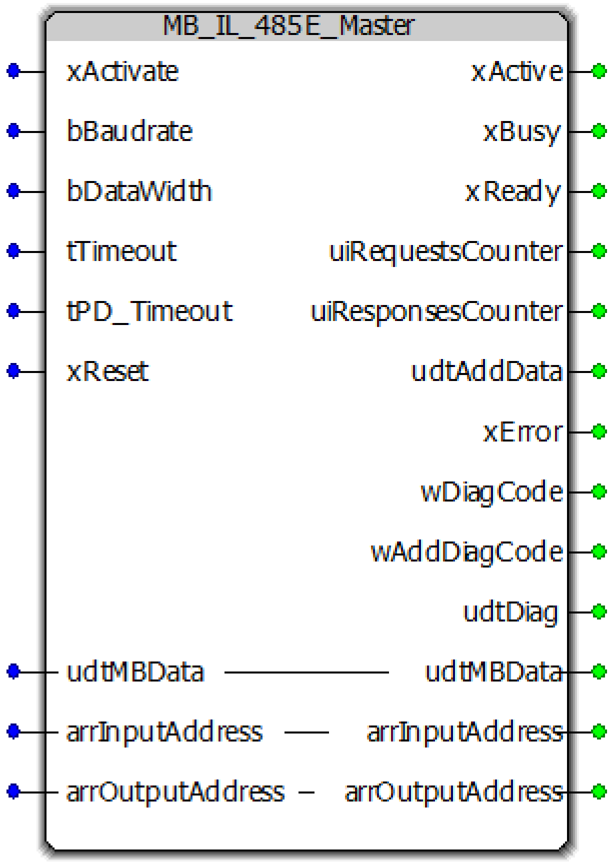
24.5 Diagnostic

wDiagCode	wAddDiagCode	Description
16#0000		Block is not activated.
16#8000		Block is active and operating without errors.
16#C010		Incorrect parameters.
	16#0010	Baud rate.
	16#0020	Data width.
	16#0050	Terminal configuration error.
	16#0060	Communication error.
	16#0080	xReceive and xSend inputs are set at the same time.
	16#0090	xReceive input is set during send procedure.
	16#0100	xSend input is set during receive procedure.
16#C020		Error when sending.
	16#0020	Maximum size exceeded.
	16#0060	Process data timeout during the send process between PLC and module.
16#C030		Error when receiving.
	16#0030	uiRcvLength is larger than the memory available in the receive buffer.
	16#0060	Process data timeout during the receive process between PLC and module.

25 MB_IL_485E_Master

This function block is used to implement a Modbus Master for the specified module type. Accordingly the function blocks are connected inside. The required parameters have to be parameterized on this function block. The associated parameter description refers to the description of the included function blocks.

25.1 Block call



25.2 Input parameters

Name	Type	Description
xActivate	BOOL	Block activation (TRUE = Active).
bBaudrate	BYTE	0 hex = Baudrate 110 1 hex = Baudrate 300 2 hex = Baudrate 600 3 hex = Baudrate 1200 4 hex = Baudrate 1800 5 hex = Baudrate 2400 6 hex = Baudrate 4800 7 hex = Baudrate 9600 8 hex = Baudrate 15625 9 hex = Baudrate 19200 A hex = Baudrate 38400 B-F hex = reserved

bDataWidth	BYTE	0 hex = 7 data bits, even, 1 stop bit 1 hex = 7 data bits, odd, 1 stop bit 2 hex = 8 data bits, even, 1 stop bit 3 hex = 8 data bits, odd, 1 stop bit 4 hex = 8 data bits, none, 1 stop bit 5 hex = 7 data bits, none, 1 stop bit 6 hex = 7 data bits, even, 2 stop bits 7 hex = 7 data bits, odd, 2 stop bits 8 hex = 8 data bits, even, 2 stop bits 9 hex = 8 data bits, odd, 2 stop bits A hex = 8 data bits, none, 2 stop bits B hex = 7 data bits, none, 2 stop bits C hex = 8 data bits, constant at 0, 1 stop bits D hex = 8 data bits, constant at 1, 1 stop bits E hex = 6 data bits, none, 1 stop bits F hex = Reserved
xReset	BOOL	The input resets the block. All connected FC blocks are reset as well.
tTimeout	TIME	Defines the maximum time until the Modbus response from the slave must arrive. The default value: TIME#5s (if input is 0s). The input is copied by xActivate or xReset if there is a rising edge.
tPD_Timeout	TIME	Timeout for process data communication between PLC and module. The input is copied by xActivate or xReset if there is a rising edge. Default: TIME#2s
xReset	BOOL	The input resets the block. All connected FC blocks are reset as well.

25.3 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.
xReady	BOOL	The block is ready to execute services. When executing services, this parameter is FALSE.
uiRequestsCounter	UINT	Shows the number of requests transmitted.
uiResponsesCounter	UINT	Shows the number of responses received.
udtAddData	MB2_RSUNI_UDT_DATA_V1	Structure with variables for diagnostics
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_xxx_DIAG	Structure with internal variables for Diagnostic

25.4 Input and output parameters

Name	Type	Description
udtMBData	udtModbus2_Data	The block communicates via this structure with the FC blocks.
arrInputAddress	MB2_RSUNI_ARR_B_1_xx	IN process data.
arrOutputAddress	MB2_RSUNI_ARR_B_1_xx	OUT process data.

25.5 Diagnostic

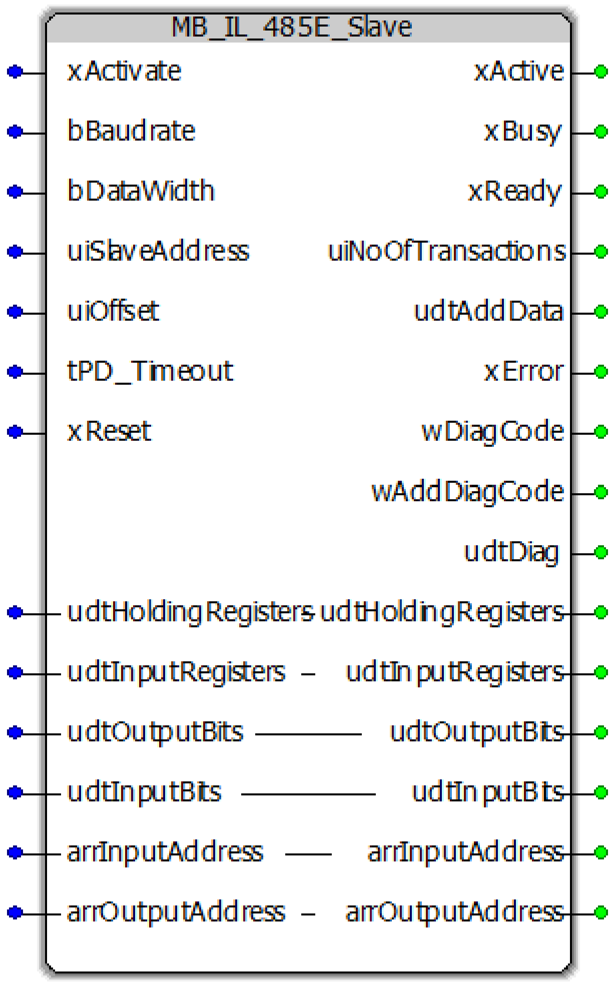
wDiagCode	wAddDiagCode	Description
16#0000		Block is not activated.
16#8000		Block is active and operating without errors.
16#8100		HW Reset phase to delete serial driver error
16#C010		Incorrect parameters.
	16#0010	Baud rate.
	16#0020	Data width.
	16#0050	Terminal configuration error.
	16#0060	Communication error.
	16#0080	xReceive and xSend inputs are set at the same time.
	16#0090	xReceive input is set during send procedure.
	16#0100	xSend input is set during receive procedure.
16#C020		Error when sending.
	16#0020	Maximum size exceeded.
	16#0060	Process data timeout during the send process between PLC and module.
16#C030		Error when receiving.
	16#0030	uiRcvLength is larger than the memory available in the receive buffer.
	16#0060	Process data timeout during the receive process between PLC and module.

26 MB_IL_485E_Slave

This function block is used to implement a Modbus Slave for the specified module type. Accordingly the function blocks are connected inside. The required parameters have to be parameterized on this function block. The associated parameter description refers to the description of the included function blocks.

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

26.1 Block call



26.2 Input parameters

Name	Type	Description
xActivate	BOOL	Block activation (TRUE = Active).
diBaudrate	DINT	Here, the baud rate can be specified freely up to 500 000. Standard values are 110, 300, 600, 1200, 1800, 2400, 4800, 9600, 19200, 38400
bDataWidth	BYTE	0 hex = 7 data bits, even, 1 stop bit 1 hex = 7 data bits, odd, 1 stop bit 2 hex = 8 data bits, even, 1 stop bit 3 hex = 8 data bits, odd, 1 stop bit 4 hex = 8 data bits, none, 1 stop bit 5 hex = 7 data bits, none, 1 stop bit 6 hex = 7 data bits, even, 2 stop bits 7 hex = 7 data bits, odd, 2 stop bits 8 hex = 8 data bits, even, 2 stop bits 9 hex = 8 data bits, odd, 2 stop bits A hex = 8 data bits, none, 2 stop bits B hex = 7 data bits, none, 2 stop bits C hex = 8 data bits, constant at 0, 1 stop bits D hex = 8 data bits, constant at 1, 1 stop bits E hex = 6 data bits, none, 1 stop bits F hex = Reserved
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).
tPD_Timeout	TIME	Timeout for process data communication between PLC and module. The input is copied by xActivate or xReset if there is a rising edge. Default: TIME#2s
xReset	BOOL	The input resets the block. All connected FC blocks are reset as well.

26.3 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.
xReady	BOOL	The block is ready to execute services. When executing services, this parameter is FALSE.
uiNoOfTransactions	UINT	Number of processed requests
udtAddData	MB2_RSUNI_UDT_DATA_V1	Structure with variables for diagnostics
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_xxx_DIAG	Structure with internal variables for Diagnostic

26.4 Input and output 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
arrInputAddress	MB2_RSUNI_ARR_B_1_xx	IN process data.
arrOutputAddress	MB2_RSUNI_ARR_B_1_xx	OUT process data.

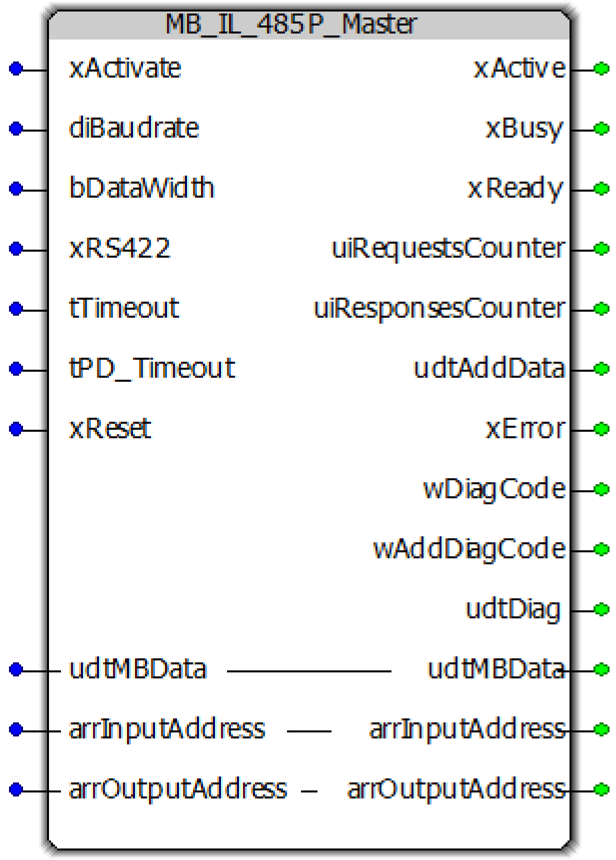
26.5 Diagnostic

wDiagCode	wAddDiagCode	Description
16#0000		Block is not activated.
16#8000		Block is active and operating without errors.
16#C010		Incorrect parameters.
	16#0010	Baud rate.
	16#0020	Data width.
	16#0050	Terminal configuration error.
	16#0060	Communication error.
	16#0080	xReceive and xSend inputs are set at the same time.
	16#0090	xReceive input is set during send procedure.
	16#0100	xSend input is set during receive procedure.
16#C020		Error when sending.
	16#0020	Maximum size exceeded.
	16#0060	Process data timeout during the send process between PLC and module.
16#C030		Error when receiving.
	16#0030	uiRcvLength is larger than the memory available in the receive buffer.
	16#0060	Process data timeout during the receive process between PLC and module.

27 MB_IL_485P_Master

This function block is used to implement a Modbus Master for the specified module type. Accordingly the function blocks are connected inside. The required parameters have to be parameterized on this function block. The associated parameter description refers to the description of the included function blocks.

27.1 Block call



27.2 Input parameters

Name	Type	Description
xActivate	BOOL	Block activation (TRUE = Active).
diBaudrate	DINT	Here, the baud rate can be specified freely up to 500 000. Standard values are 110, 300, 600, 1200, 1800, 2400, 4800, 9600, 19200, 38400
bDataWidth	BYTE	0 hex = 7 data bits, even, 1 stop bit 1 hex = 7 data bits, odd, 1 stop bit 2 hex = 8 data bits, even, 1 stop bit 3 hex = 8 data bits, odd, 1 stop bit 4 hex = 8 data bits, none, 1 stop bit 5 hex = 7 data bits, none, 1 stop bit 6 hex = 7 data bits, even, 2 stop bits 7 hex = 7 data bits, odd, 2 stop bits 8 hex = 8 data bits, even, 2 stop bits 9 hex = 8 data bits, odd, 2 stop bits A hex = 8 data bits, none, 2 stop bits B hex = 7 data bits, none, 2 stop bits
xRS422	BOOL	FALSE: RS485 TRUE: RS422
tTimeout	TIME	Defines the maximum time until the Modbus response from the slave must arrive. The default value: TIME#5s (if input is 0s). The input is copied by xActivate or xReset if there is a rising edge.
tPD_Timeout	TIME	Timeout for process data communication between PLC and module. The input is copied by xActivate or xReset if there is a rising edge. Default: TIME#2s
xReset	BOOL	The input resets the block. All connected FC blocks are reset as well.

27.3 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.
xReady	BOOL	The block is ready to execute services. When executing services, this parameter is FALSE.
uiRequestsCounter	UINT	Shows the number of requests transmitted.
uiResponsesCounter	UINT	Shows the number of responses received.
udtAddData	MB2_COM_UDT_R485P_DATA_V1	Structure with additional status variables.
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_xxx_DIAG	Structure with internal variables for Diagnostic

27.4 Input and output parameters

Name	Type	Description
udtMBData	udtModbus2_Data	The block communicates via this structure with the FC blocks.
arrInputAddress	MB2_COM_ARR_B_1_12	IN process data.
arrOutputAddress	MB2_COM_ARR_B_1_12	OUT process data.

27.5 Diagnostic

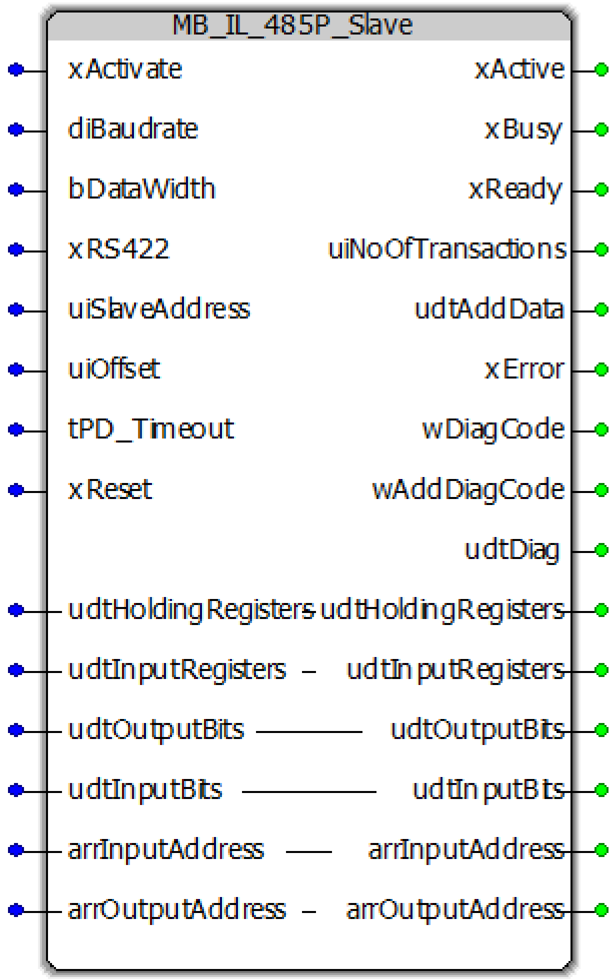
wDiagCode	wAddDiagCode	Description
16#0000		Block is not activated.
16#8000		Block is active and operating without errors.
16#8100		HW Reset phase to delete serial driver error
16#C010		Incorrect terminal type connected.
	16#00XX	Read terminal type.
	16#FFFF	Terminal is not responding.
16#C020		Incorrect parameter.
	16#0010	Baud rate exceeded.
	16#0020	Data width.
	16#0030	Protocol.
	16#0040	Terminal error.
	16#0050	Baudrate <= 0.
	16#0070	Terminal configuration error.
	16#0100	xReceive and xSend inputs are set at the same time.
	16#0110	xReceive input is set during send procedure.
	16#0120	xSend input is set during receive procedure.
16#C030		Error when sending.
	16#0010	Process data timeout during the send process between PLC and module.
	16#0020	Maximum size when sending exceeded.
	16#0030	uiSendLength exceeded.
	16#0040	uiSendLength > 255 with 3964R-Protocol.
16#C040		Error when receiving.
	16#0010	Process data timeout during the receive process between PLC and module.
	16#0020	Maximum size when receiving exceeded.
	16#0030	uiRcvLength exceeded.
	16#0040	uiRcvLength <> 0 for the end-to-end, 3964R, and dual buffer protocols.
16#C050		3964R protocol error.
	16#0010	Error when sending a 3964R telegram.
	16#0020	Error when receiving a 3964R telegram.

28 MB_IL_485P_Slave

This function block is used to implement a Modbus Slave for the specified module type. Accordingly the function blocks are connected inside. The required parameters have to be parameterized on this function block. The associated parameter description refers to the description of the included function blocks.

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

28.1 Block call



28.2 Input parameters

Name	Type	Description
xActivate	BOOL	Block activation (TRUE = Active).
diBaudrate	DINT	Here, the baud rate can be specified freely up to 500 000. Standard values are 110, 300, 600, 1200, 1800, 2400, 4800, 9600, 19200, 38400
bDataWidth	BYTE	0 hex = 7 data bits, even, 1 stop bit 1 hex = 7 data bits, odd, 1 stop bit 2 hex = 8 data bits, even, 1 stop bit 3 hex = 8 data bits, odd, 1 stop bit 4 hex = 8 data bits, none, 1 stop bit 5 hex = 7 data bits, none, 1 stop bit 6 hex = 7 data bits, even, 2 stop bits 7 hex = 7 data bits, odd, 2 stop bits 8 hex = 8 data bits, even, 2 stop bits 9 hex = 8 data bits, odd, 2 stop bits A hex = 8 data bits, none, 2 stop bits B hex = 7 data bits, none, 2 stop bits
xRS422	BOOL	FALSE: RS485 TRUE: RS422
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).
tPD_Timeout	TIME	Timeout for process data communication between PLC and module. The input is copied by xActivate or xReset if there is a rising edge. Default: TIME#2s
xReset	BOOL	The input resets the block. All connected FC blocks are reset as well.

28.3 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.
xReady	BOOL	The block is ready to execute services. When executing services, this parameter is FALSE.
uiNoOfTransactions	UINT	Number of processed requests
udtAddData	MB2_COM_UDT_R485P_DATA_V1	Structure with additional status variables.
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_xxx_DIAG	Structure with internal variables for Diagnostic

28.4 Input and output 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
arrInputAddress	MB2_COM_ARR_B_1_12	IN process data.
arrOutputAddress	MB2_COM_ARR_B_1_12	OUT process data.

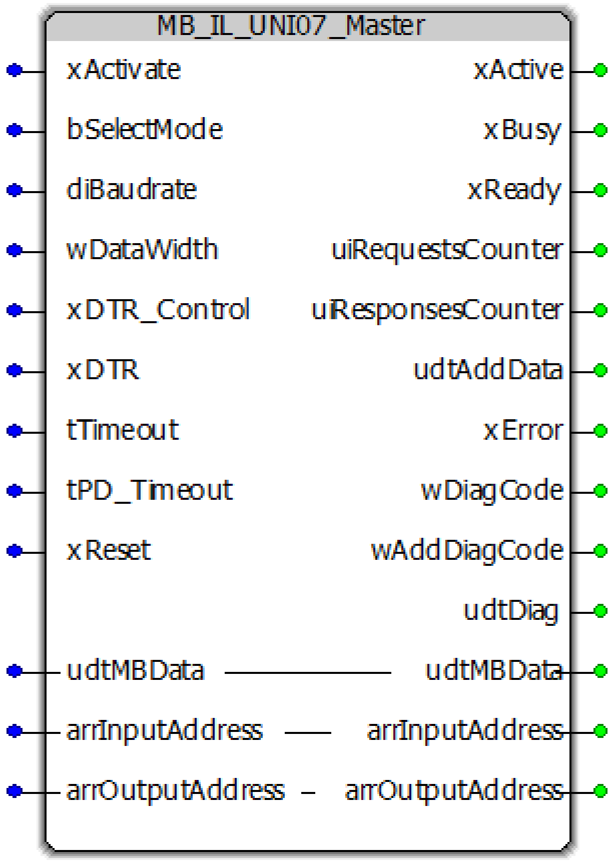
28.5 Diagnostic

wDiagCode	wAddDiagCode	Description
16#0000		Block is not activated.
16#8000		Block is active and operating without errors.
16#C010		Incorrect terminal type connected.
	16#00XX	Read terminal type.
	16#FFFF	Terminal is not responding.
16#C020		Incorrect parameter.
	16#0010	Baud rate exceeded.
	16#0020	Data width.
	16#0030	Protocol.
	16#0040	Terminal error.
	16#0050	Baudrate <= 0.
	16#0070	Terminal configuration error.
	16#0100	xReceive and xSend inputs are set at the same time.
	16#0110	xReceive input is set during send procedure.
	16#0120	xSend input is set during receive procedure.
16#C030		Error when sending.
	16#0010	Process data timeout during the send process between PLC and module.
	16#0020	Maximum size when sending exceeded.
	16#0030	uiSendLength exceeded.
	16#0040	uiSendLength > 255 with 3964R-Protocol.
16#C040		Error when receiving.
	16#0010	Process data timeout during the receive process between PLC and module.
	16#0020	Maximum size when receiving exceeded.
	16#0030	uiRcvLength exceeded.
	16#0040	uiRcvLength <> 0 for the end-to-end, 3964R, and dual buffer protocols.
16#C050		3964R protocol error.
	16#0010	Error when sending a 3964R telegram.
	16#0020	Error when receiving a 3964R telegram.

29 MB_IL_UNI07_Master

This function block is used to implement a Modbus Master for the specified module type. Accordingly the function blocks are connected inside. The required parameters have to be parameterized on this function block. The associated parameter description refers to the description of the included function blocks.

29.1 Block call



29.2 Input parameters

Name	Type	Description
xActivate	BOOL	Block activation (TRUE = Active).
bSelectMode	BYTE	0 hex = RS-232 1 hex = RS-485 2 hex = RS-422
diBaudrate	DINT	Here, the baud rate can be specified freely from 110 baud to 262143 baud. Standard values are 110, 300, 600, 1200, 1800, 2400, 4800, 9600, 19200, 38400, 57600, 115200, 230400, 250000.
wDataWidth	WORD	Write the code for the data width combination to the low byte. If you wish to use a data width different from the standard combination, then write value 0xF to the low byte and your desired value to the high byte. The combinations can be found in the data sheet. Direct specification: Bit15 to Bit8 Code: Bit7 to Bit0 0 hex = 7 data bits, even, 1 stop bit 1 hex = 7 data bits, odd, 1 stop bit 2 hex = 8 data bits, even, 1 stop bit 3 hex = 8 data bits, odd, 1 stop bit 4 hex = 8 data bits, none, 1 stop bit 5 hex = 7 data bits, none, 1 stop bit 6 hex = 7 data bits, even, 2 stop bits 7 hex = 7 data bits, odd, 2 stop bits 8 hex = 8 data bits, even, 2 stop bits 9 hex = 8 data bits, odd, 2 stop bits A hex = 8 data bits, none, 2 stop bits B hex = 7 data bits, none, 2 stop bits C hex = 8 data bits, constant at 0, 1 stop bits D hex = 8 data bits, constant at 1, 1 stop bits E hex = 6 data bits, none, 1 stop bits F hex = Reserved
xDTR_Control	BOOL	FALSE: The DTR signal is controlled automatically. TRUE: The DTR signal is controlled by the user.

xDTR	BOOL	The DTR signal is controlled. Only active if the corresponding mode is activated (implemented via the parameterization blocks).
xReset	BOOL	The input resets the block. All connected FC blocks are reset as well.
tTimeout	TIME	Defines the maximum time until the Modbus response from the slave must arrive. The default value: TIME#5s (if input is 0s). The input is copied by xActivate or xReset if there is a rising edge.
tPD_Timeout	TIME	Timeout for process data communication between PLC and module. The input is copied by xActivate or xReset if there is a rising edge. Default: TIME#2s
xReset	BOOL	The input resets the block. All connected FC blocks are reset as well.

29.3 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.
xReady	BOOL	The block is ready to execute services. When executing services, this parameter is FALSE.
uiRequestsCounter	UINT	Shows the number of requests transmitted.
uiResponsesCounter	UINT	Shows the number of responses received.
udtAddData	MB2_COM_UDT_R485P_DATA_V1	Structure with additional status variables.
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_xxx_DIAG	Structure with internal variables for Diagnostic

29.4 Input and output parameters

Name	Type	Description
udtMBData	udtModbus2_Data	The block communicates via this structure with the FC blocks.
arrInputAddress	MB2_RSUNI_ARR_B_1_xx	IN process data.
arrOutputAddress	MB2_RSUNI_ARR_B_1_xx	OUT process data.

29.5 Diagnostic

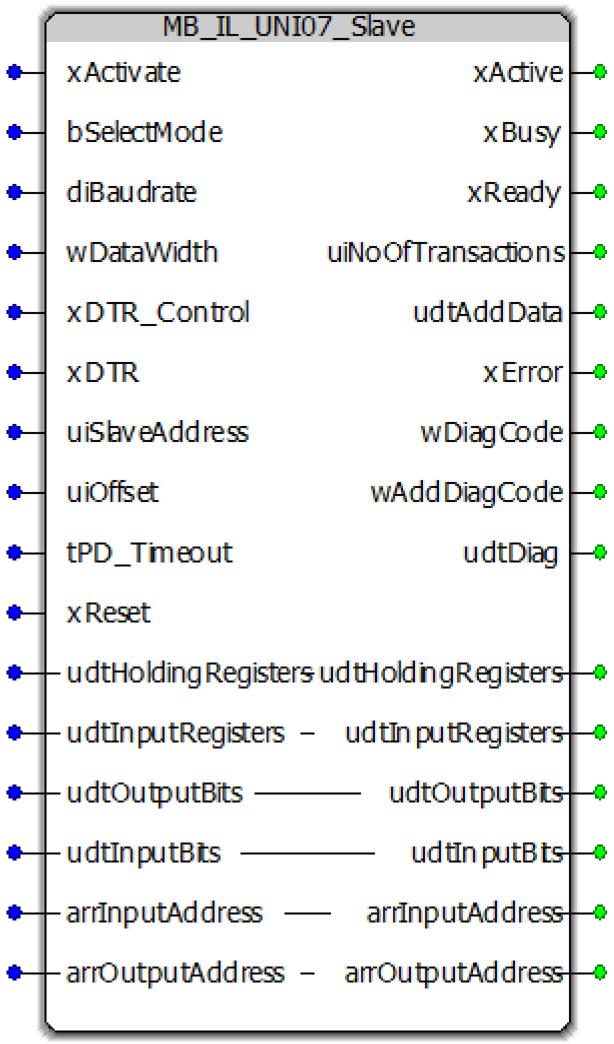
wDiagCode	wAddDiagCode	Description
16#0000		Block is not activated.
16#8000		Block is active and operating without errors.
16#8100		HW Reset phase to delete serial driver error
16#C010		Incorrect parameters.
	16#0010	Baud rate.
	16#0020	Data width.
	16#0030	Protocol.
	16#0040	Interface.
	16#0050	Terminal configuration error.
	16#0060	Communication error.
	16#0070	Communication error during reset of module.
	16#0080	xReceive and xSend inputs are set at the same time.
	16#0090	xReceive input is set during send procedure.
	16#0100	xSend input is set during receive procedure.
16#C020		Error when sending.
	16#0020	Maximum size exceeded.
	16#0060	Process data timeout during the send process between PLC and module.
16#C030		Error when receiving.
	16#0030	uiRcvLength is longer than the memory available in the receive buffer.
	16#0040	uiRcvLength <> 0 for end-to-end protocol.
	16#0060	Process data timeout during the receive process between PLC and module.

30 MB_IL_UNI07_Slave

This function block is used to implement a Modbus Slave for the specified module type. Accordingly the function blocks are connected inside. The required parameters have to be parameterized on this function block. The associated parameter description refers to the description of the included function blocks.

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

30.1 Block call



30.2 Input parameters

Name	Type	Description
xActivate	BOOL	Block activation (TRUE = Active).
bSelectMode	BYTE	0 hex = RS-232 1 hex = RS-485 2 hex = RS-422
diBaudrate	DINT	Here, the baud rate can be specified freely from 110 baud to 262143 baud. Standard values are 110, 300, 600, 1200, 1800, 2400, 4800, 9600, 19200, 38400, 57600, 115200, 230400, 250000.
wDataWidth	WORD	Write the code for the data width combination to the low byte. If you wish to use a data width different from the standard combination, then write value 0xF to the low byte and your desired value to the high byte. The combinations can be found in the data sheet. Direct specification: Bit15 to Bit8 Code: Bit7 to Bit0 0 hex = 7 data bits, even, 1 stop bit 1 hex = 7 data bits, odd, 1 stop bit 2 hex = 8 data bits, even, 1 stop bit 3 hex = 8 data bits, odd, 1 stop bit 4 hex = 8 data bits, none, 1 stop bit 5 hex = 7 data bits, none, 1 stop bit 6 hex = 7 data bits, even, 2 stop bits 7 hex = 7 data bits, odd, 2 stop bits 8 hex = 8 data bits, even, 2 stop bits 9 hex = 8 data bits, odd, 2 stop bits A hex = 8 data bits, none, 2 stop bits B hex = 7 data bits, none, 2 stop bits C hex = 8 data bits, constant at 0, 1 stop bits D hex = 8 data bits, constant at 1, 1 stop bits E hex = 6 data bits, none, 1 stop bits F hex = Reserved
xDTR_Control	BOOL	FALSE: The DTR signal is controlled automatically. TRUE: The DTR signal is controlled by the user.

xDTR	BOOL	The DTR signal is controlled. Only active if the corresponding mode is activated (implemented via the parameterization blocks).
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).
tPD_Timeout	TIME	Timeout for process data communication between PLC and module. The input is copied by xActivate or xReset if there is a rising edge. Default: TIME#2s
xReset	BOOL	The input resets the block. All connected FC blocks are reset as well.

30.3 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.
xReady	BOOL	The block is ready to execute services. When executing services, this parameter is FALSE.
uiNoOfTransactions	UINT	Number of processed requests
udtAddData	MB2_RSUNI_UDT_DATA_V1	Structure with variables for diagnostics
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_xxx_DIAG	Structure with internal variables for Diagnostic

30.4 Input and output 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
arrInputAddress	MB2_RSUNI_ARR_B_1_xx	IN process data.
arrOutputAddress	MB2_RSUNI_ARR_B_1_xx	OUT process data.

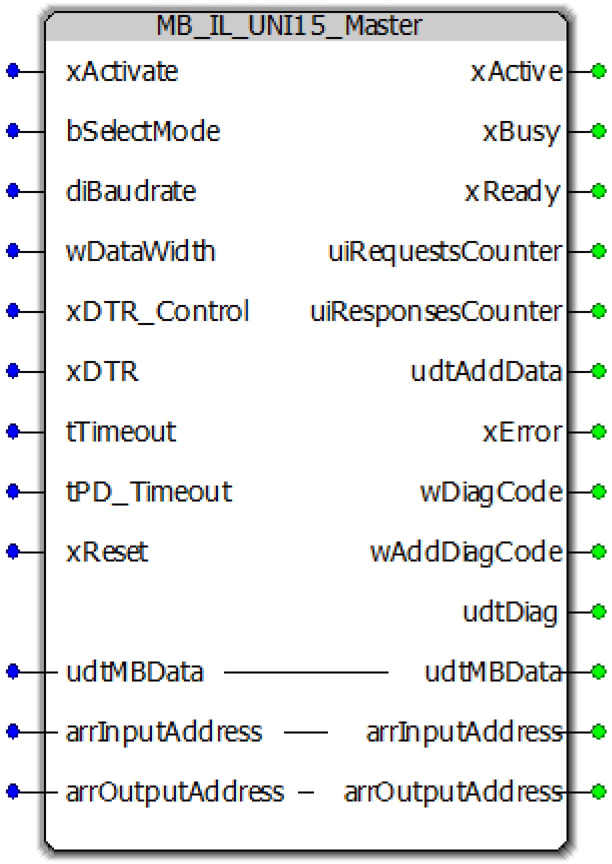
30.5 Diagnostic

wDiagCode	wAddDiagCode	Description
16#0000		Block is not activated.
16#8000		Block is active and operating without errors.
16#C010		Incorrect parameters.
	16#0010	Baud rate.
	16#0020	Data width.
	16#0030	Protocol.
	16#0040	Interface.
	16#0050	Terminal configuration error.
	16#0060	Communication error.
	16#0070	Communication error during reset of module.
	16#0080	xReceive and xSend inputs are set at the same time.
	16#0090	xReceive input is set during send procedure.
	16#0100	xSend input is set during receive procedure.
16#C020		Error when sending.
	16#0020	Maximum size exceeded.
	16#0060	Process data timeout during the send process between PLC and module.
16#C030		Error when receiving.
	16#0030	uiRcvLength is longer than the memory available in the receive buffer.
	16#0040	uiRcvLength <> 0 for end-to-end protocol.
	16#0060	Process data timeout during the receive process between PLC and module.

31 MB_IL_UNI15_Master

This function block is used to implement a Modbus Master for the specified module type. Accordingly the function blocks are connected inside. The required parameters have to be parameterized on this function block. The associated parameter description refers to the description of the included function blocks.

31.1 Block call



31.2 Input parameters

Name	Type	Description
xActivate	BOOL	Block activation (TRUE = Active).
bSelectMode	BYTE	0 hex = RS-232 1 hex = RS-485 2 hex = RS-422
diBaudrate	DINT	Here, the baud rate can be specified freely from 110 baud to 262143 baud. Standard values are 110, 300, 600, 1200, 1800, 2400, 4800, 9600, 19200, 38400, 57600, 115200, 230400, 250000.
wDataWidth	WORD	Write the code for the data width combination to the low byte. If you wish to use a data width different from the standard combination, then write value 0xF to the low byte and your desired value to the high byte. The combinations can be found in the data sheet. Direct specification: Bit15 to Bit8 Code: Bit7 to Bit0 0 hex = 7 data bits, even, 1 stop bit 1 hex = 7 data bits, odd, 1 stop bit 2 hex = 8 data bits, even, 1 stop bit 3 hex = 8 data bits, odd, 1 stop bit 4 hex = 8 data bits, none, 1 stop bit 5 hex = 7 data bits, none, 1 stop bit 6 hex = 7 data bits, even, 2 stop bits 7 hex = 7 data bits, odd, 2 stop bits 8 hex = 8 data bits, even, 2 stop bits 9 hex = 8 data bits, odd, 2 stop bits A hex = 8 data bits, none, 2 stop bits B hex = 7 data bits, none, 2 stop bits C hex = 8 data bits, constant at 0, 1 stop bits D hex = 8 data bits, constant at 1, 1 stop bits E hex = 6 data bits, none, 1 stop bits F hex = Reserved
xDTR_Control	BOOL	FALSE: The DTR signal is controlled automatically. TRUE: The DTR signal is controlled by the user.

xDTR	BOOL	The DTR signal is controlled. Only active if the corresponding mode is activated (implemented via the parameterization blocks).
tTimeout	TIME	Defines the maximum time until the Modbus response from the slave must arrive. The default value: TIME#5s (if input is 0s). The input is copied by xActivate or xReset if there is a rising edge.
tPD_Timeout	TIME	Timeout for process data communication between PLC and module. The input is copied by xActivate or xReset if there is a rising edge. Default: TIME#2s
xReset	BOOL	The input resets the block. All connected FC blocks are reset as well.

31.3 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.
xReady	BOOL	The block is ready to execute services. When executing services, this parameter is FALSE.
uiRequestsCounter	UINT	Shows the number of requests transmitted.
uiResponsesCounter	UINT	Shows the number of responses received.
udtAddData	MB2_COM_UDT_R485P_DATA_V1	Structure with additional status variables.
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_xxx_DIAG	Structure with internal variables for Diagnostic

31.4 Input and output parameters

Name	Type	Description
udtMBData	udtModbus2_Data	The block communicates via this structure with the FC blocks.
arrInputAddress	MB2_RSUNI_ARR_B_1_xx	IN process data.
arrOutputAddress	MB2_RSUNI_ARR_B_1_xx	OUT process data.

31.5 Diagnostic

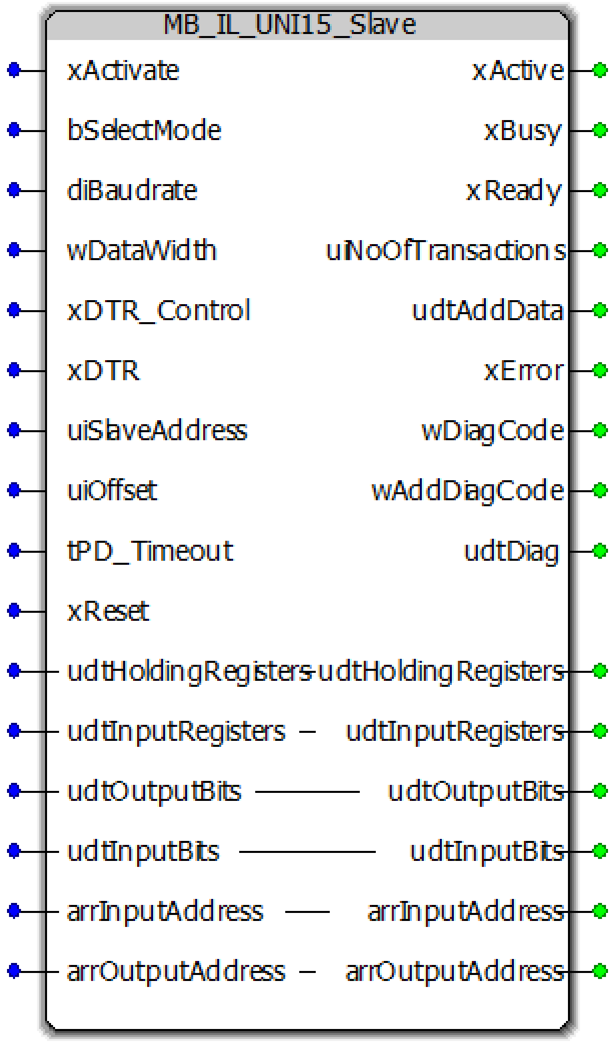
wDiagCode	wAddDiagCode	Description
16#0000		Block is not activated.
16#8000		Block is active and operating without errors.
16#8100		HW Reset phase to delete serial driver error
16#C010		Incorrect parameters.
	16#0010	Baud rate.
	16#0020	Data width.
	16#0030	Protocol.
	16#0040	Interface.
	16#0050	Terminal configuration error.
	16#0060	Communication error.
	16#0070	Communication error during reset of module.
	16#0080	xReceive and xSend inputs are set at the same time.
	16#0090	xReceive input is set during send procedure.
	16#0100	xSend input is set during receive procedure.
16#C020		Error when sending.
	16#0020	Maximum size exceeded.
	16#0060	Process data timeout during the send process between PLC and module.
16#C030		Error when receiving.
	16#0030	uiRcvLength is longer than the memory available in the receive buffer.
	16#0040	uiRcvLength <> 0 for end-to-end protocol.
	16#0060	Process data timeout during the receive process between PLC and module.

32 MB_IL_UNI15_Slave

This function block is used to implement a Modbus Slave for the specified module type. Accordingly the function blocks are connected inside. The required parameters have to be parameterized on this function block. The associated parameter description refers to the description of the included function blocks.

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

32.1 Block call



32.2 Input parameters

Name	Type	Description
xActivate	BOOL	Block activation (TRUE = Active).
bSelectMode	BYTE	0 hex = RS-232 1 hex = RS-485 2 hex = RS-422
diBaudrate	DINT	Here, the baud rate can be specified freely from 110 baud to 262143 baud. Standard values are 110, 300, 600, 1200, 1800, 2400, 4800, 9600, 19200, 38400, 57600, 115200, 230400, 250000.
wDataWidth	WORD	Write the code for the data width combination to the low byte. If you wish to use a data width different from the standard combination, then write value 0xF to the low byte and your desired value to the high byte. The combinations can be found in the data sheet. Direct specification: Bit15 to Bit8 Code: Bit7 to Bit0 0 hex = 7 data bits, even, 1 stop bit 1 hex = 7 data bits, odd, 1 stop bit 2 hex = 8 data bits, even, 1 stop bit 3 hex = 8 data bits, odd, 1 stop bit 4 hex = 8 data bits, none, 1 stop bit 5 hex = 7 data bits, none, 1 stop bit 6 hex = 7 data bits, even, 2 stop bits 7 hex = 7 data bits, odd, 2 stop bits 8 hex = 8 data bits, even, 2 stop bits 9 hex = 8 data bits, odd, 2 stop bits A hex = 8 data bits, none, 2 stop bits B hex = 7 data bits, none, 2 stop bits C hex = 8 data bits, constant at 0, 1 stop bits D hex = 8 data bits, constant at 1, 1 stop bits E hex = 6 data bits, none, 1 stop bits F hex = Reserved
xDTR_Control	BOOL	FALSE: The DTR signal is controlled automatically. TRUE: The DTR signal is controlled by the user.

xDTR	BOOL	The DTR signal is controlled. Only active if the corresponding mode is activated (implemented via the parameterization blocks).
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).
tPD_Timeout	TIME	Timeout for process data communication between PLC and module. The input is copied by xActivate or xReset if there is a rising edge. Default: TIME#2s
xReset	BOOL	The input resets the block. All connected FC blocks are reset as well.

32.3 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.
xReady	BOOL	The block is ready to execute services. When executing services, this parameter is FALSE.
uiNoOfTransactions	UINT	Number of processed requests
udtAddData	MB2_RSUNI_UDT_DATA_V1	Structure with variables for diagnostics
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_xxx_DIAG	Structure with internal variables for Diagnostic

32.4 Input and output 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
arrInputAddress	MB2_RSUNI_ARR_B_1_xx	IN process data.
arrOutputAddress	MB2_RSUNI_ARR_B_1_xx	OUT process data.

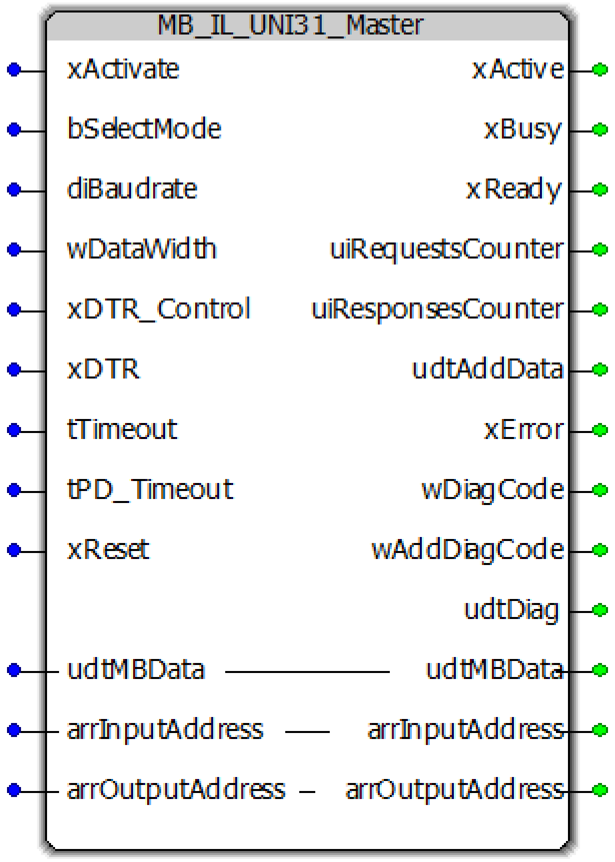
32.5 Diagnostic

wDiagCode	wAddDiagCode	Description
16#0000		Block is not activated.
16#8000		Block is active and operating without errors.
16#C010		Incorrect parameters.
	16#0010	Baud rate.
	16#0020	Data width.
	16#0030	Protocol.
	16#0040	Interface.
	16#0050	Terminal configuration error.
	16#0060	Communication error.
	16#0070	Communication error during reset of module.
	16#0080	xReceive and xSend inputs are set at the same time.
	16#0090	xReceive input is set during send procedure.
	16#0100	xSend input is set during receive procedure.
16#C020		Error when sending.
	16#0020	Maximum size exceeded.
	16#0060	Process data timeout during the send process between PLC and module.
16#C030		Error when receiving.
	16#0030	uiRcvLength is longer than the memory available in the receive buffer.
	16#0040	uiRcvLength <> 0 for end-to-end protocol.
	16#0060	Process data timeout during the receive process between PLC and module.

33 MB_IL_UNI31_Master

This function block is used to implement a Modbus Master for the specified module type. Accordingly the function blocks are connected inside. The required parameters have to be parameterized on this function block. The associated parameter description refers to the description of the included function blocks.

33.1 Block call



33.2 Input parameters

Name	Type	Description
xActivate	BOOL	Block activation (TRUE = Active).
bSelectMode	BYTE	0 hex = RS-232 1 hex = RS-485 2 hex = RS-422
diBaudrate	DINT	Here, the baud rate can be specified freely from 110 baud to 262143 baud. Standard values are 110, 300, 600, 1200, 1800, 2400, 4800, 9600, 19200, 38400, 57600, 115200, 230400, 250000.
wDataWidth	WORD	Write the code for the data width combination to the low byte. If you wish to use a data width different from the standard combination, then write value 0xF to the low byte and your desired value to the high byte. The combinations can be found in the data sheet. Direct specification: Bit15 to Bit8 Code: Bit7 to Bit0 0 hex = 7 data bits, even, 1 stop bit 1 hex = 7 data bits, odd, 1 stop bit 2 hex = 8 data bits, even, 1 stop bit 3 hex = 8 data bits, odd, 1 stop bit 4 hex = 8 data bits, none, 1 stop bit 5 hex = 7 data bits, none, 1 stop bit 6 hex = 7 data bits, even, 2 stop bits 7 hex = 7 data bits, odd, 2 stop bits 8 hex = 8 data bits, even, 2 stop bits 9 hex = 8 data bits, odd, 2 stop bits A hex = 8 data bits, none, 2 stop bits B hex = 7 data bits, none, 2 stop bits C hex = 8 data bits, constant at 0, 1 stop bits D hex = 8 data bits, constant at 1, 1 stop bits E hex = 6 data bits, none, 1 stop bits F hex = Reserved
xDTR_Control	BOOL	FALSE: The DTR signal is controlled automatically. TRUE: The DTR signal is controlled by the user.

xDTR	BOOL	The DTR signal is controlled. Only active if the corresponding mode is activated (implemented via the parameterization blocks).
xReset	BOOL	The input resets the block. All connected FC blocks are reset as well.
tTimeout	TIME	Defines the maximum time until the Modbus response from the slave must arrive. The default value: TIME#5s (if input is 0s). The input is copied by xActivate or xReset if there is a rising edge.
tPD_Timeout	TIME	Timeout for process data communication between PLC and module. The input is copied by xActivate or xReset if there is a rising edge. Default: TIME#2s
xReset	BOOL	The input resets the block. All connected FC blocks are reset as well.

33.3 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.
xReady	BOOL	The block is ready to execute services. When executing services, this parameter is FALSE.
uiRequestsCounter	UINT	Shows the number of requests transmitted.
uiResponsesCounter	UINT	Shows the number of responses received.
udtAddData	MB2_COM_UDT_R485P_DATA_V1	Structure with additional status variables.
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_xxx_DIAG	Structure with internal variables for Diagnostic

33.4 Input and output parameters

Name	Type	Description
udtMBData	udtModbus2_Data	The block communicates via this structure with the FC blocks.
arrInputAddress	MB2_RSUNI_ARR_B_1_xx	IN process data.
arrOutputAddress	MB2_RSUNI_ARR_B_1_xx	OUT process data.

33.5 Diagnostic

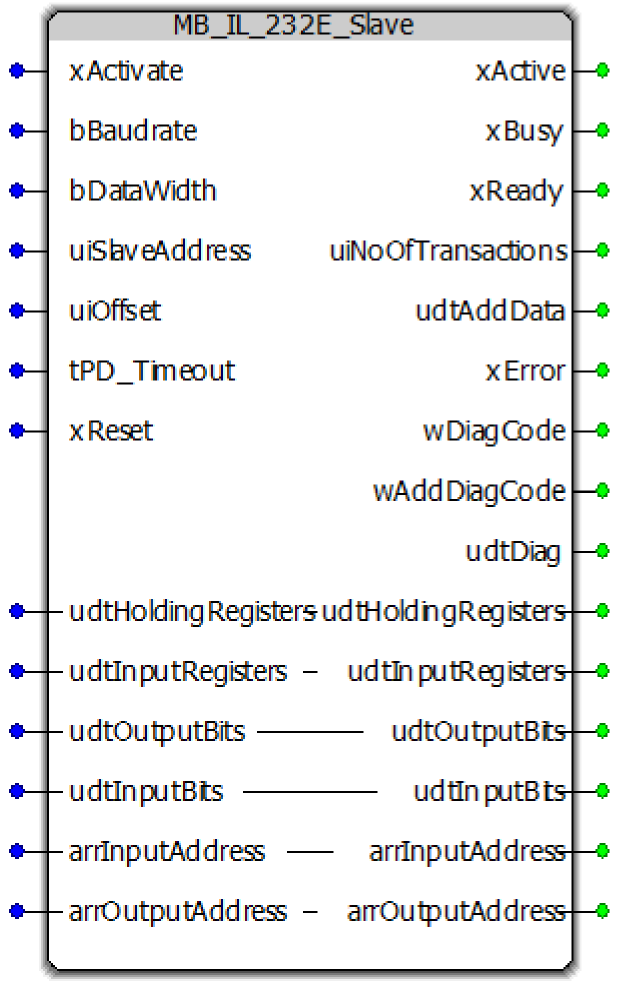
wDiagCode	wAddDiagCode	Description
16#0000		Block is not activated.
16#8000		Block is active and operating without errors.
16#8100		HW Reset phase to delete serial driver error
16#C010		Incorrect parameters.
	16#0010	Baud rate.
	16#0020	Data width.
	16#0030	Protocol.
	16#0040	Interface.
	16#0050	Terminal configuration error.
	16#0060	Communication error.
	16#0070	Communication error during reset of module.
	16#0080	xReceive and xSend inputs are set at the same time.
	16#0090	xReceive input is set during send procedure.
	16#0100	xSend input is set during receive procedure.
16#C020		Error when sending.
	16#0020	Maximum size exceeded.
	16#0060	Process data timeout during the send process between PLC and module.
16#C030		Error when receiving.
	16#0030	uiRcvLength is longer than the memory available in the receive buffer.
	16#0040	uiRcvLength <> 0 for end-to-end protocol.
	16#0060	Process data timeout during the receive process between PLC and module.

34 MB_IL_UNI31_Slave

This function block is used to implement a Modbus Slave for the specified module type. Accordingly the function blocks are connected inside. The required parameters have to be parameterized on this function block. The associated parameter description refers to the description of the included function blocks.

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

34.1 Block call



34.2 Input parameters

Name	Type	Description
xActivate	BOOL	Block activation (TRUE = Active).
bSelectMode	BYTE	0 hex = RS-232 1 hex = RS-485 2 hex = RS-422
diBaudrate	DINT	Here, the baud rate can be specified freely from 110 baud to 262143 baud. Standard values are 110, 300, 600, 1200, 1800, 2400, 4800, 9600, 19200, 38400, 57600, 115200, 230400, 250000.
wDataWidth	WORD	Write the code for the data width combination to the low byte. If you wish to use a data width different from the standard combination, then write value 0xF to the low byte and your desired value to the high byte. The combinations can be found in the data sheet. Direct specification: Bit15 to Bit8 Code: Bit7 to Bit0 0 hex = 7 data bits, even, 1 stop bit 1 hex = 7 data bits, odd, 1 stop bit 2 hex = 8 data bits, even, 1 stop bit 3 hex = 8 data bits, odd, 1 stop bit 4 hex = 8 data bits, none, 1 stop bit 5 hex = 7 data bits, none, 1 stop bit 6 hex = 7 data bits, even, 2 stop bits 7 hex = 7 data bits, odd, 2 stop bits 8 hex = 8 data bits, even, 2 stop bits 9 hex = 8 data bits, odd, 2 stop bits A hex = 8 data bits, none, 2 stop bits B hex = 7 data bits, none, 2 stop bits C hex = 8 data bits, constant at 0, 1 stop bits D hex = 8 data bits, constant at 1, 1 stop bits E hex = 6 data bits, none, 1 stop bits F hex = Reserved
xDTR_Control	BOOL	FALSE: The DTR signal is controlled automatically. TRUE: The DTR signal is controlled by the user.

xDTR	BOOL	The DTR signal is controlled. Only active if the corresponding mode is activated (implemented via the parameterization blocks).
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).
tPD_Timeout	TIME	Timeout for process data communication between PLC and module. The input is copied by xActivate or xReset if there is a rising edge. Default: TIME#2s
xReset	BOOL	The input resets the block. All connected FC blocks are reset as well.

34.3 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.
xReady	BOOL	The block is ready to execute services. When executing services, this parameter is FALSE.
uiNoOfTransactions	UINT	Number of processed requests
udtAddData	MB2_RSUNI_UDT_DATA_V1	Structure with variables for diagnostics
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_xxx_DIAG	Structure with internal variables for Diagnostic

34.4 Input and output 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
arrInputAddress	MB2_RSUNI_ARR_B_1_xx	IN process data.
arrOutputAddress	MB2_RSUNI_ARR_B_1_xx	OUT process data.

34.5 Diagnostic

wDiagCode	wAddDiagCode	Description
16#0000		Block is not activated.
16#8000		Block is active and operating without errors.
16#C010		Incorrect parameters.
	16#0010	Baud rate.
	16#0020	Data width.
	16#0030	Protocol.
	16#0040	Interface.
	16#0050	Terminal configuration error.
	16#0060	Communication error.
	16#0070	Communication error during reset of module.
	16#0080	xReceive and xSend inputs are set at the same time.
	16#0090	xReceive input is set during send procedure.
	16#0100	xSend input is set during receive procedure.
16#C020		Error when sending.
	16#0020	Maximum size exceeded.
	16#0060	Process data timeout during the send process between PLC and module.
16#C030		Error when receiving.
	16#0030	uiRcvLength is longer than the memory available in the receive buffer.
	16#0040	uiRcvLength <> 0 for end-to-end protocol.
	16#0060	Process data timeout during the receive process between PLC and module.

35 Examples

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

- MB_RTU_5_EXA_AXL_MA.zwt
- MB_RTU_5_EXA_AXL_SL.zwt
- MB_RTU_5_EXA_IL_MA.zwt
- MB_RTU_5_EXA_IL_SL.zwt

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

35.1 Example 1: Modbus_RTU AXL master functionality

35.1.1 Plant

For this example, the following hardware is used:

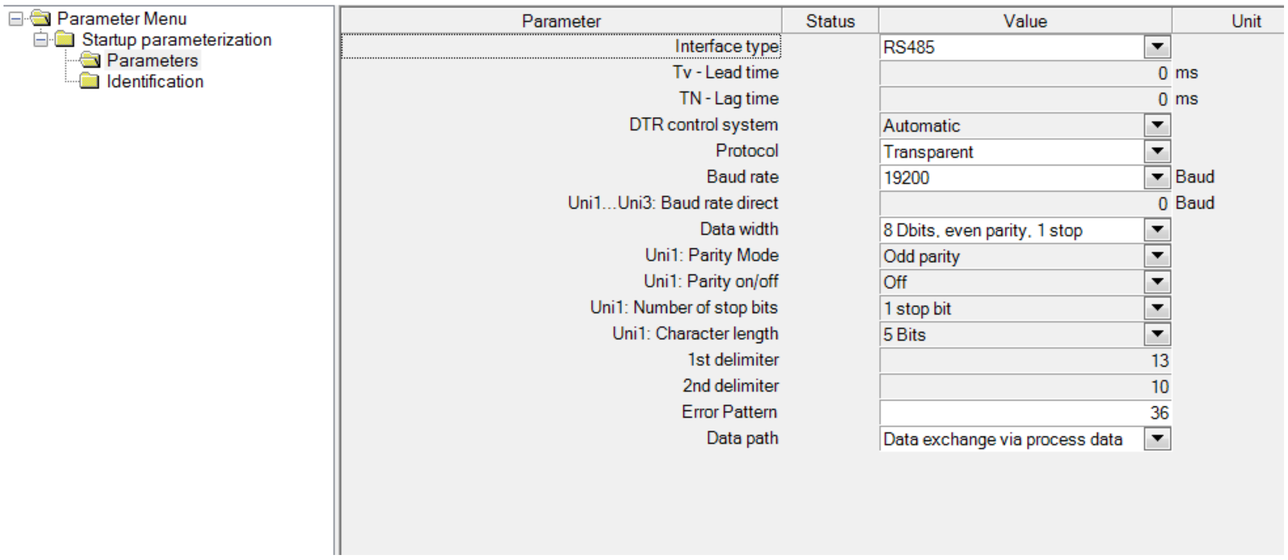
- AXC 1050 (2700988)
- AXL F RS UNI 1H (2688666)

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

This project shows one example for the startup of MB_AXL_F_RSUNI_Master function block.

AXL F RS UNI 1H startup parameters:

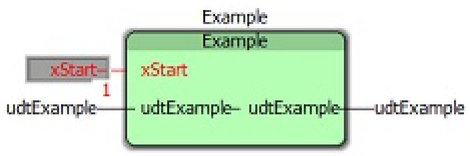
If the xMode input is activated, the selected protocol must be "Modbus RTU". If the xMode input is deactivated, the selected protocol must be "Transparent".

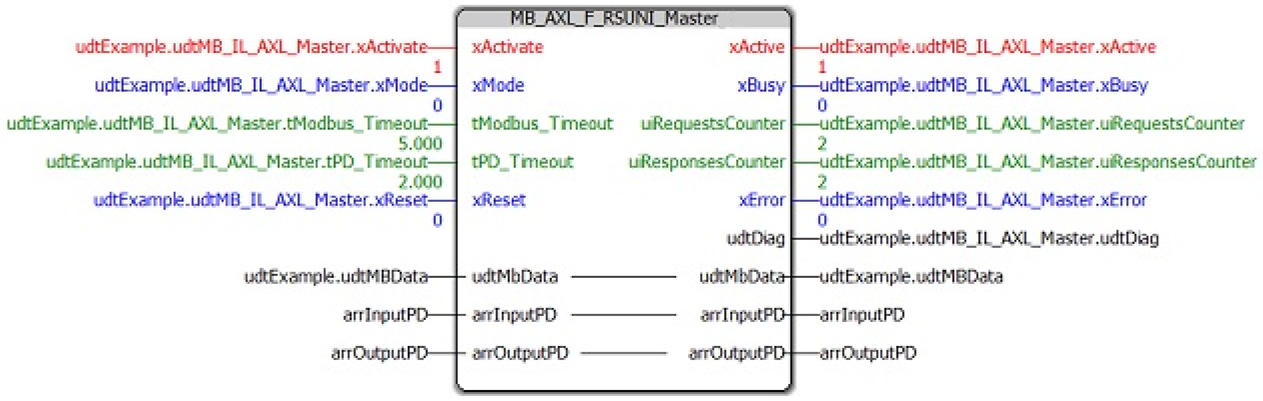


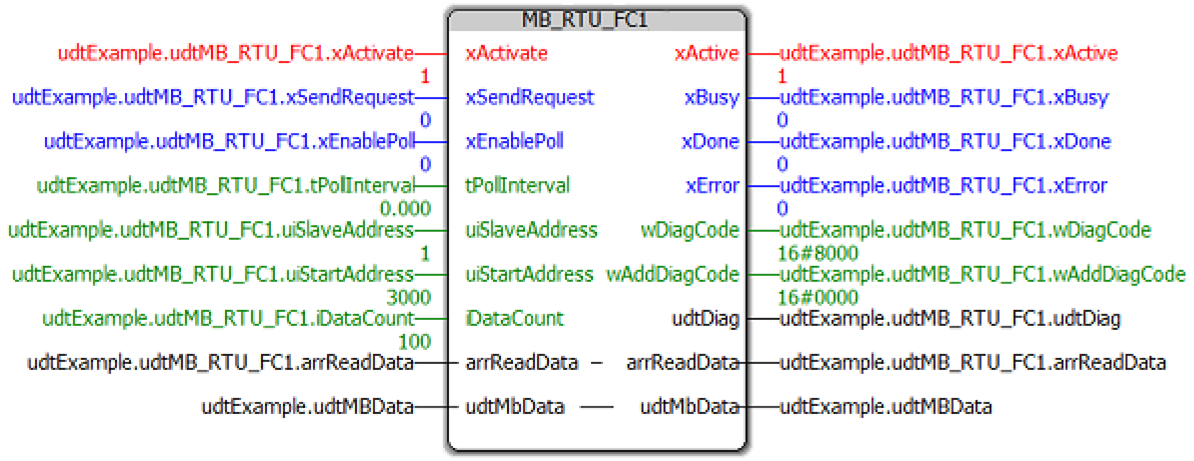
- Parameter Menu
  - Startup parameterization
    - Parameters
    - Identification

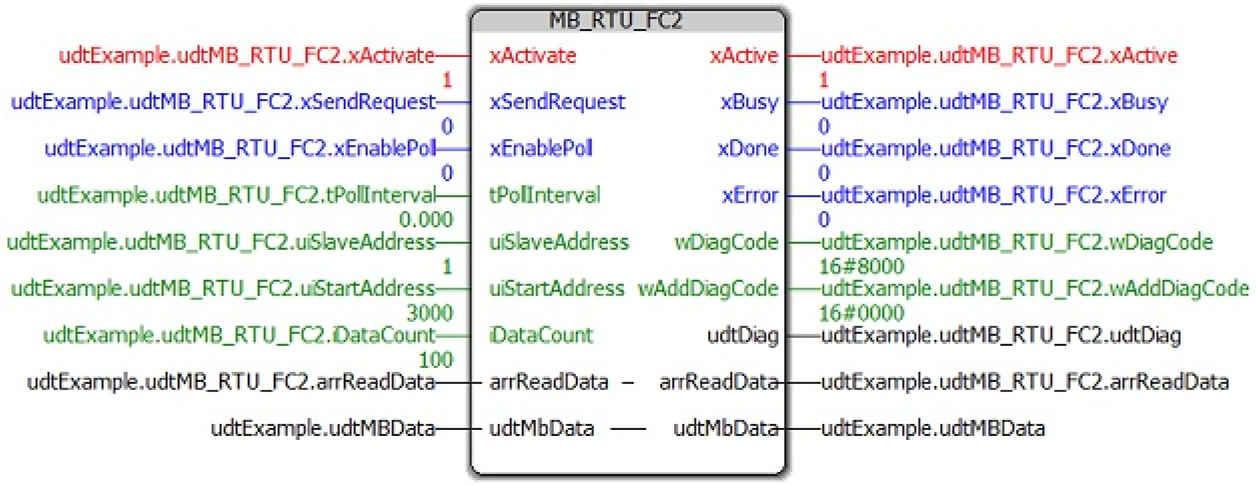
Parameter	Status	Value	Unit
Interface type		RS485	
Tv - Lead time		0	ms
TN - Lag time		0	ms
DTR control system		Automatic	
Protocol		Transparent	
Baud rate		19200	Baud
Uni1...Uni3: Baud rate direct		0	Baud
Data width		8 Dbits, even parity, 1 stop	
Uni1: Parity Mode		Odd parity	
Uni1: Parity on/off		Off	
Uni1: Number of stop bits		1 stop bit	
Uni1: Character length		5 Bits	
1st delimiter		13	
2nd delimiter		10	
Error Pattern		36	
Data path		Data exchange via process data	

MB_AXL_F_RSUNI_MASTER:




MB_AXL_F_RSUNI_Master
udtExample.udtMB_IL_AXL_Master.xActivate
1
xActivate
xActive
udtExample.udtMB_IL_AXL_Master.xActive
1
udtExample.udtMB_IL_AXL_Master.xMode
0
xMode
xBusy
udtExample.udtMB_IL_AXL_Master.xBusy
0
udtExample.udtMB_IL_AXL_Master.tModbus_Timeout
5.000
tModbus_Timeout
uiRequestsCounter
udtExample.udtMB_IL_AXL_Master.uiRequestsCounter
2
udtExample.udtMB_IL_AXL_Master.tPD_Timeout
2.000
tPD_Timeout
uiResponsesCounter
udtExample.udtMB_IL_AXL_Master.uiResponsesCounter
2
udtExample.udtMB_IL_AXL_Master.xReset
0
xReset
xError
udtExample.udtMB_IL_AXL_Master.xError
0
udtDiag
udtExample.udtMB_IL_AXL_Master.udtDiag
udtExample.udtMBData
udtMbData
udtMbData
udtExample.udtMBData
arrInputPD
arrInputPD
arrInputPD
arrInputPD
arrOutputPD
arrOutputPD
arrOutputPD
arrOutputPD


MB_RTU_FC1
udtExample.udtMB_RTU_FC1.xActivate
1
xActivate
xActive
udtExample.udtMB_RTU_FC1.xActive
1
udtExample.udtMB_RTU_FC1.xSendRequest
0
xSendRequest
xBusy
udtExample.udtMB_RTU_FC1.xBusy
0
udtExample.udtMB_RTU_FC1.xEnablePoll
0
xEnablePoll
xDone
udtExample.udtMB_RTU_FC1.xDone
0
udtExample.udtMB_RTU_FC1.tPollInterval
0.000
tPollInterval
xError
udtExample.udtMB_RTU_FC1.xError
0
udtExample.udtMB_RTU_FC1.uiSlaveAddress
1
uiSlaveAddress
wDiagCode
udtExample.udtMB_RTU_FC1.wDiagCode
16#8000
udtExample.udtMB_RTU_FC1.uiStartAddress
3000
uiStartAddress
wAddDiagCode
udtExample.udtMB_RTU_FC1.wAddDiagCode
16#0000
udtExample.udtMB_RTU_FC1.iDataCount
100
iDataCount
udtDiag
udtExample.udtMB_RTU_FC1.udtDiag
udtExample.udtMB_RTU_FC1.arrReadData
arrReadData
arrReadData
udtExample.udtMB_RTU_FC1.arrReadData
udtExample.udtMBData
udtMbData
udtMbData
udtExample.udtMBData


MB_RTU_FC2
udtExample.udtMB_RTU_FC2.xActivate
1
xActivate
xActive
udtExample.udtMB_RTU_FC2.xActive
1
udtExample.udtMB_RTU_FC2.xSendRequest
0
xSendRequest
xBusy
udtExample.udtMB_RTU_FC2.xBusy
0
udtExample.udtMB_RTU_FC2.xEnablePoll
0
xEnablePoll
xDone
udtExample.udtMB_RTU_FC2.xDone
0
udtExample.udtMB_RTU_FC2.tPollInterval
0.000
tPollInterval
xError
udtExample.udtMB_RTU_FC2.xError
0
udtExample.udtMB_RTU_FC2.uiSlaveAddress
1
uiSlaveAddress
wDiagCode
udtExample.udtMB_RTU_FC2.wDiagCode
16#8000
udtExample.udtMB_RTU_FC2.uiStartAddress
3000
uiStartAddress
wAddDiagCode
udtExample.udtMB_RTU_FC2.wAddDiagCode
16#0000
udtExample.udtMB_RTU_FC2.iDataCount
100
iDataCount
udtDiag
udtExample.udtMB_RTU_FC2.udtDiag
udtExample.udtMB_RTU_FC2.arrReadData
arrReadData
arrReadData
udtExample.udtMB_RTU_FC2.arrReadData
udtExample.udtMBData
udtMbData
udtMbData
udtExample.udtMBData

Creating structures:

The Modbus master as well as the FC blocks are connected with each other via a structure.

To start the example set the xStart input of the Example function block to TRUE.

```
CASE udtExample.iState OF
    0: (* wait for xStart *)
        IF xStart = TRUE THEN
            udtExample.iState   := 10;
        END_IF;
    10: (* Start Master *)
        udtExample.udtMB_IL_AXL_Master.tModbus_Timeout  := TIME#5s;
        udtExample.udtMB_IL_AXL_Master.tPD_Timeout      := TIME#2s;
        udtExample.udtMB_IL_AXL_Master.xMode        := FALSE;
        udtExample.udtMB_IL_AXL_Master.xActivate    := TRUE;
        udtExample.iState                           := 20;
    20: (* wait for Master xActive and xReady *)
        IF  udtExample.udtMB_IL_AXL_Master.xActive = TRUE AND
            udtExample.udtMB_IL_AXL_Master.xError = FALSE
        THEN
            udtExample.iState   := 30;
        ELSIF udtExample.udtMB_IL_AXL_Master.xError THEN
            udtExample.iState   := 998;
        END_IF;
    30: (* Activate FC1 *)
        udtExample.udtMB_RTU_FC1.xActivate          := TRUE;
        udtExample.udtMB_RTU_FC1.xSendRequest       := FALSE;
        udtExample.udtMB_RTU_FC1.uiSlaveAddress     := UINT#1;
        udtExample.udtMB_RTU_FC1.xEnablePoll        := FALSE;
        udtExample.udtMB_RTU_FC1.tPollInterval      := T#0s;
        udtExample.udtMB_RTU_FC1.uiStartAddress     := UINT#3000;
        udtExample.udtMB_RTU_FC1.iDataCount         := 100;
        IF  udtExample.udtMB_RTU_FC1.xActive = TRUE AND
            udtExample.udtMB_RTU_FC1.xError = FALSE AND
            udtExample.udtMB_RTU_FC1.xBusy = FALSE
        THEN
            udtExample.iState   := 40;
        ELSIF udtExample.udtMB_RTU_FC1.xError THEN
            udtExample.iState   := 998;
        END_IF;
    40: (* Activate FC2 *)
        udtExample.udtMB_RTU_FC2.xActivate          := TRUE;
        udtExample.udtMB_RTU_FC2.xSendRequest       := FALSE;
        udtExample.udtMB_RTU_FC2.uiSlaveAddress     := UINT#1;
        udtExample.udtMB_RTU_FC2.xEnablePoll        := FALSE;
        udtExample.udtMB_RTU_FC2.tPollInterval      := T#0s;
        udtExample.udtMB_RTU_FC2.uiStartAddress     := UINT#3000;
        udtExample.udtMB_RTU_FC2.iDataCount         := 100;
        IF  udtExample.udtMB_RTU_FC2.xActive = TRUE AND
            udtExample.udtMB_RTU_FC2.xError = FALSE AND
            udtExample.udtMB_RTU_FC2.xBusy = FALSE
        THEN
            udtExample.iState   := 50;
        ELSIF udtExample.udtMB_RTU_FC2.xError THEN
            udtExample.iState   := 998;
        END_IF;
    50: (* start send request FC1 *)
        udtExample.udtMB_RTU_FC1.xSendRequest   := TRUE;
        IF  udtExample.udtMB_RTU_FC1.xActive = TRUE AND
            udtExample.udtMB_RTU_FC1.xError = FALSE AND
            udtExample.udtMB_RTU_FC1.xBusy = FALSE AND
            udtExample.udtMB_RTU_FC1.xDone = TRUE
        THEN
            udtExample.udtMB_RTU_FC1.xSendRequest   := FALSE;
            udtExample.iState   := 60;
        ELSIF udtExample.udtMB_RTU_FC1.xError THEN
            udtExample.iState   := 998;
        END_IF;
    60: (* start send request FC2 *)
        udtExample.udtMB_RTU_FC2.xSendRequest   := TRUE;
        IF  udtExample.udtMB_RTU_FC2.xActive = TRUE AND
            udtExample.udtMB_RTU_FC2.xError = FALSE AND
            udtExample.udtMB_RTU_FC2.xBusy = FALSE AND
            udtExample.udtMB_RTU_FC2.xDone = TRUE
        THEN
            udtExample.udtMB_RTU_FC2.xSendRequest   := FALSE;
            udtExample.iState   := 70;
        ELSIF udtExample.udtMB_RTU_FC2.xError THEN
            udtExample.iState   := 998;
        END_IF;
    70: (* wait for deactivation - xStart = FALSE *)
        IF xStart = FALSE THEN
            udtExample.udtMB_IL_AXL_Master.xActivate    := FALSE;
            udtExample.udtMB_RTU_FC1.xActivate          := FALSE;
            udtExample.udtMB_RTU_FC2.xActivate          := FALSE;
            udtExample.iState   := 80;
        END_IF;
    80: (* go back to State 0 *)
        IF  udtExample.udtMB_IL_AXL_Master.xActive = FALSE AND
            udtExample.udtMB_RTU_FC1.xActive = FALSE AND
            udtExample.udtMB_RTU_FC2.xActive = FALSE
        THEN
            udtExample.iState   := 0;
        END_IF;
    998: (* Error state *)
        IF xStart = FALSE THEN
            udtExample.udtMB_IL_AXL_Master.xActivate    := FALSE;
            udtExample.udtMB_RTU_FC1.xActivate          := FALSE;
            udtExample.udtMB_RTU_FC2.xActivate          := FALSE;
            udtExample.iState   := 999;
        END_IF;
    999: (* go back to State 0 *)
        IF  udtExample.udtMB_IL_AXL_Master.xActive = FALSE AND
            udtExample.udtMB_RTU_FC1.xActive = FALSE AND
            udtExample.udtMB_RTU_FC2.xActive = FALSE
        THEN
            udtExample.iState   := 0;
        END_IF;
END_CASE;
```

35.2 Example 2: Modbus_RTU AXL slave functionality

35.2.1 Plant

For this example, the following hardware is used:

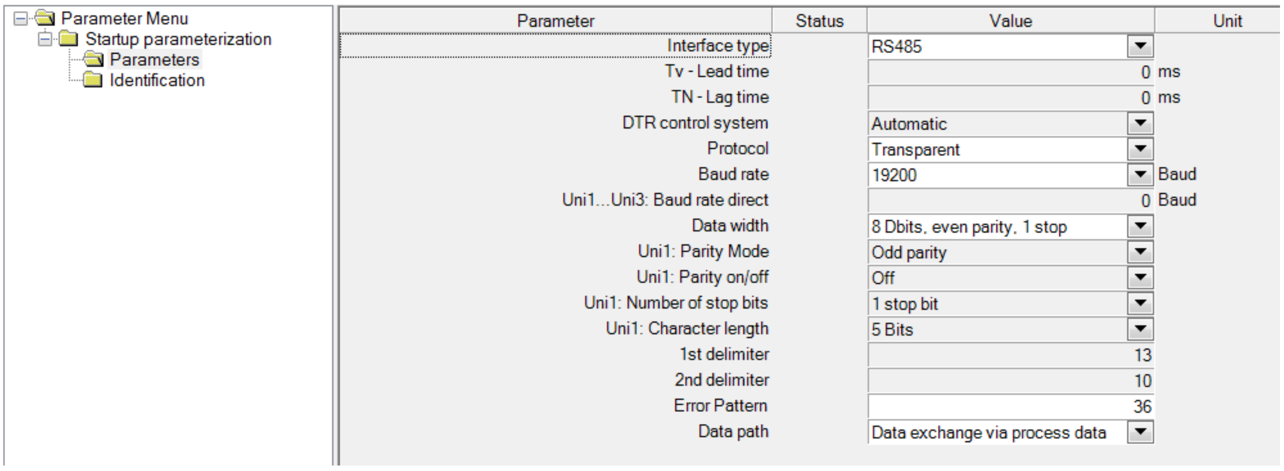
- AXC 1050 (2700988)
- AXL F BK PN (2701815)
- AXL F RS UNI 1H (2688666)

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

This project shows one example for the startup of MB_AXL_F_RSUNI_Slave function block.

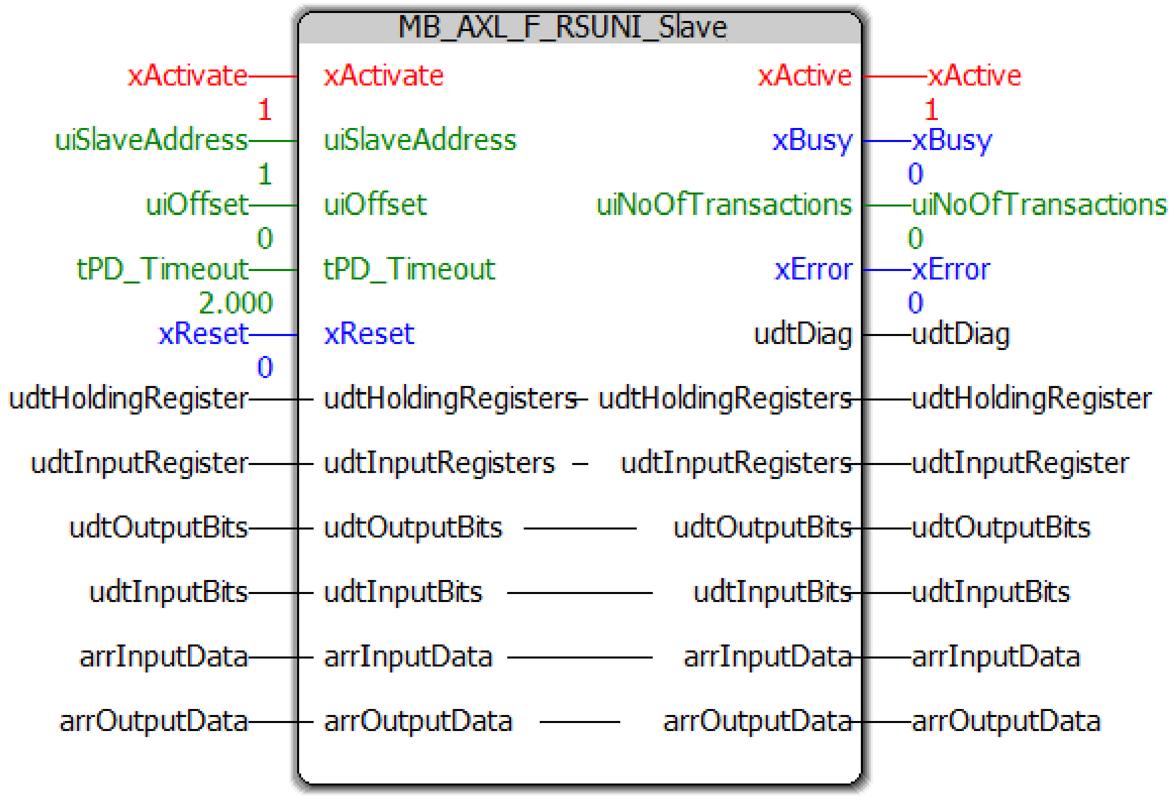
AXL F RS UNI 1H startup parameters:

The selected protocol must be "Transparent".



Parameter	Status	Value	Unit
Interface type		RS485	
Tv - Lead time		0	ms
TN - Lag time		0	ms
DTR control system		Automatic	
Protocol		Transparent	
Baud rate		19200	Baud
Uni1...Uni3: Baud rate direct		0	Baud
Data width		8 Dbits, even parity, 1 stop	
Uni1: Parity Mode		Odd parity	
Uni1: Parity on/off		Off	
Uni1: Number of stop bits		1 stop bit	
Uni1: Character length		5 Bits	
1st delimiter		13	
2nd delimiter		10	
Error Pattern		36	
Data path		Data exchange via process data	

MB_AXL_F_RSUNI_SLAVE:



35.3 Example 3: Modbus_RTU IL master functionality

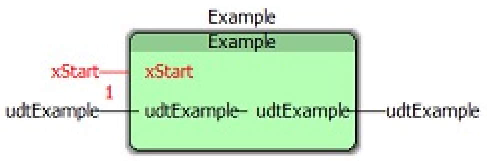
35.3.1 Plant

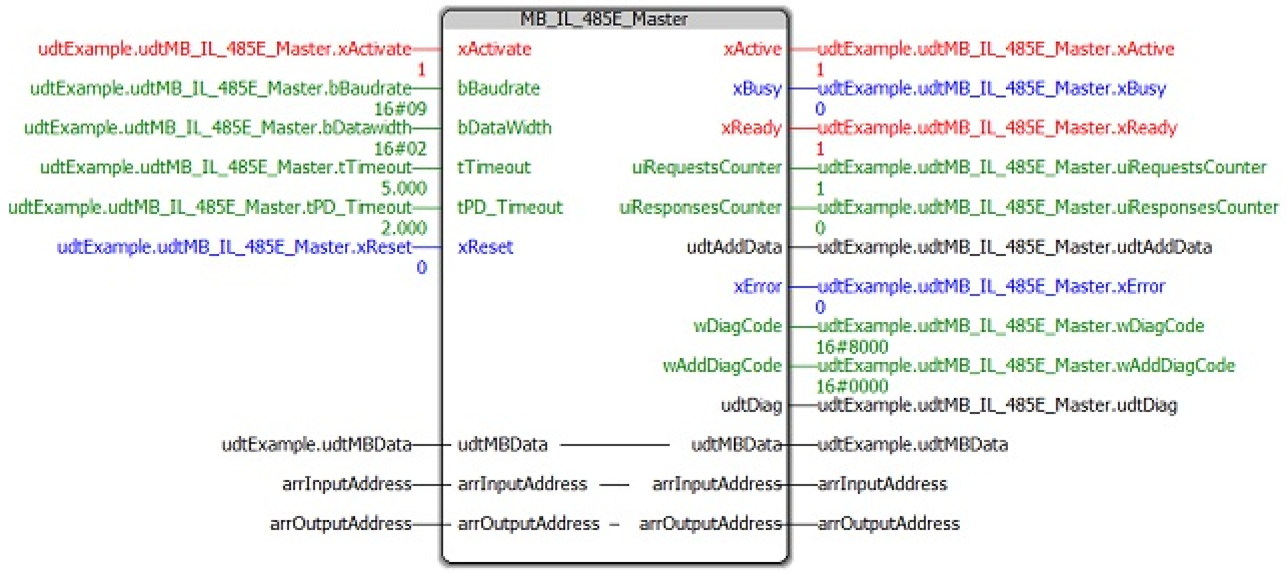
For this example, the following hardware is used:

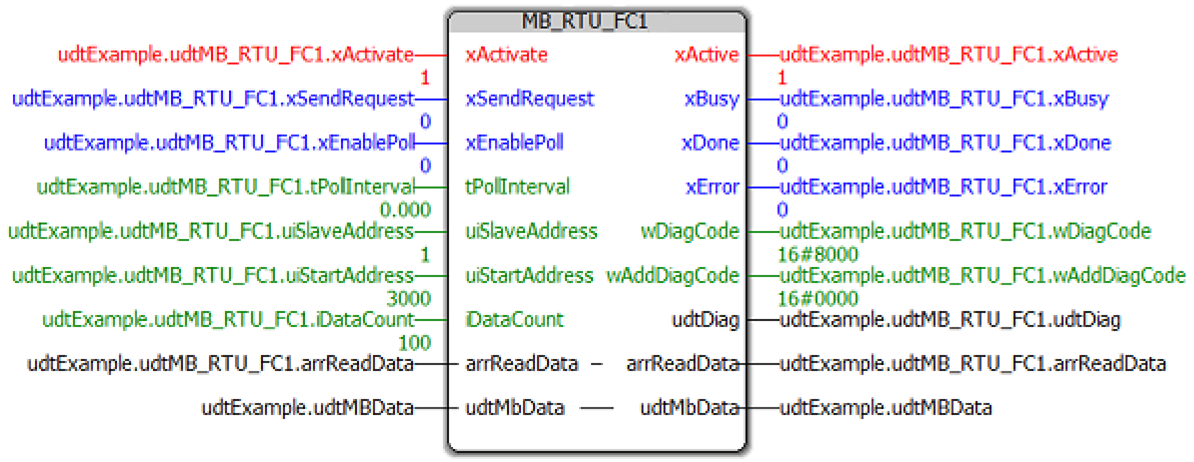
- ILC 370 PN 2TX-IB (2876915)
- IB IL RS 485-ECO (2702795)

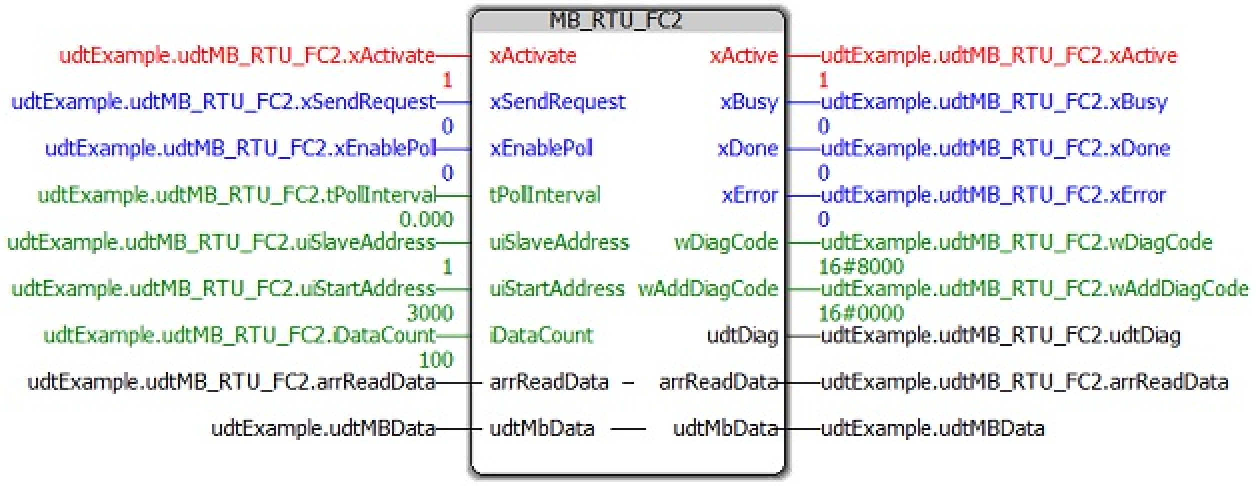
35.3.2 Modbus master with IB IL RS 485-ECO (2702795)

MB_IL_485E_MASTER:




MB_IL_485E_Master
udtExample.udtMB_IL_485E_Master.xActivate
1
xActivate
udtExample.udtMB_IL_485E_Master.bBaudrate
16#09
bBaudrate
udtExample.udtMB_IL_485E_Master.bDatawidth
16#02
bDataWidth
udtExample.udtMB_IL_485E_Master.tTimeout
5.000
tTimeout
udtExample.udtMB_IL_485E_Master.tPD_Timeout
2.000
tPD_Timeout
udtExample.udtMB_IL_485E_Master.xReset
0
xReset
udtExample.udtMBData
udtMBData
arrInputAddress
arrInputAddress
arrOutputAddress
arrOutputAddress
xActive
udtExample.udtMB_IL_485E_Master.xActive
1
xBusy
udtExample.udtMB_IL_485E_Master.xBusy
0
xReady
udtExample.udtMB_IL_485E_Master.xReady
1
uiRequestsCounter
udtExample.udtMB_IL_485E_Master.uiRequestsCounter
1
uiResponsesCounter
udtExample.udtMB_IL_485E_Master.uiResponsesCounter
0
udtAddData
udtExample.udtMB_IL_485E_Master.udtAddData
xError
udtExample.udtMB_IL_485E_Master.xError
0
wDiagCode
udtExample.udtMB_IL_485E_Master.wDiagCode
16#8000
wAddDiagCode
udtExample.udtMB_IL_485E_Master.wAddDiagCode
16#0000
udtDiag
udtExample.udtMB_IL_485E_Master.udtDiag
udtMBData
udtExample.udtMBData
arrInputAddress
arrInputAddress
arrOutputAddress
arrOutputAddress


MB_RTU_FC1
udtExample.udtMB_RTU_FC1.xActivate
1
xActivate
udtExample.udtMB_RTU_FC1.xSendRequest
0
xSendRequest
udtExample.udtMB_RTU_FC1.xEnablePoll
0
xEnablePoll
udtExample.udtMB_RTU_FC1.tPollInterval
0.000
tPollInterval
udtExample.udtMB_RTU_FC1.uiSlaveAddress
1
uiSlaveAddress
udtExample.udtMB_RTU_FC1.uiStartAddress
3000
uiStartAddress
udtExample.udtMB_RTU_FC1.iDataCount
100
iDataCount
udtExample.udtMB_RTU_FC1.arrReadData
arrReadData
udtExample.udtMBData
udtMbData
xActive
udtExample.udtMB_RTU_FC1.xActive
1
xBusy
udtExample.udtMB_RTU_FC1.xBusy
0
xDone
udtExample.udtMB_RTU_FC1.xDone
0
xError
udtExample.udtMB_RTU_FC1.xError
0
wDiagCode
udtExample.udtMB_RTU_FC1.wDiagCode
16#8000
wAddDiagCode
udtExample.udtMB_RTU_FC1.wAddDiagCode
16#0000
udtDiag
udtExample.udtMB_RTU_FC1.udtDiag
arrReadData
udtExample.udtMB_RTU_FC1.arrReadData
udtMbData
udtExample.udtMBData


MB_RTU_FC2
udtExample.udtMB_RTU_FC2.xActivate
1
udtExample.udtMB_RTU_FC2.xSendRequest
0
udtExample.udtMB_RTU_FC2.xEnablePoll
0
udtExample.udtMB_RTU_FC2.tPollInterval
0.000
udtExample.udtMB_RTU_FC2.uiSlaveAddress
1
udtExample.udtMB_RTU_FC2.uiStartAddress
3000
udtExample.udtMB_RTU_FC2.iDataCount
100
udtExample.udtMB_RTU_FC2.arrReadData
udtExample.udtMBData
xActivate
xSendRequest
xEnablePoll
tPollInterval
uiSlaveAddress
uiStartAddress
iDataCount
arrReadData
udtMbData
xActive
xBusy
xDone
xError
wDiagCode
wAddDiagCode
udtDiag
arrReadData
udtMbData
udtExample.udtMB_RTU_FC2.xActive
1
udtExample.udtMB_RTU_FC2.xBusy
0
udtExample.udtMB_RTU_FC2.xDone
0
udtExample.udtMB_RTU_FC2.xError
0
udtExample.udtMB_RTU_FC2.wDiagCode
16#8000
udtExample.udtMB_RTU_FC2.wAddDiagCode
16#0000
udtExample.udtMB_RTU_FC2.udtDiag
udtExample.udtMB_RTU_FC2.arrReadData
udtExample.udtMBData

Creating structures:

The Modbus master as well as the FC blocks are connected with each other via a structure.

To start the example set the xStart input of the Example function block to TRUE.

```
CASE udtExample.iState OF
    0: (* wait for xStart *)
        IF xStart = TRUE THEN
            udtExample.iState   := 10;
        END_IF;
    10: (* Start Master *)
        udtExample.udtMB_IL_485E_Master.bBaudrate    := BYTE#09;
        udtExample.udtMB_IL_485E_Master.bDatawidth   := BYTE#02;
        udtExample.udtMB_IL_485E_Master.tTimeout     := TIME#5s;
        udtExample.udtMB_IL_485E_Master.tPD_Timeout  := TIME#2s;
        udtExample.udtMB_IL_485E_Master.xActivate    := TRUE;
        udtExample.iState                            := 20;
    20: (* wait for Master xActive and xReady *)
        IF  udtExample.udtMB_IL_485E_Master.xActive = TRUE AND
            udtExample.udtMB_IL_485E_Master.xReady = TRUE AND
            udtExample.udtMB_IL_485E_Master.xError = FALSE
        THEN
            udtExample.iState   := 30;
        ELSIF udtExample.udtMB_IL_485E_Master.xError THEN
            udtExample.iState   := 998;
        END_IF;
    30: (* Activate FC1 *)
        udtExample.udtMB_RTU_FC1.xActivate           := TRUE;
        udtExample.udtMB_RTU_FC1.xSendRequest        := FALSE;
        udtExample.udtMB_RTU_FC1.uiSlaveAddress      := UINT#1;
        udtExample.udtMB_RTU_FC1.xEnablePoll         := FALSE;
        udtExample.udtMB_RTU_FC1.tPollInterval       := T#0s;
        udtExample.udtMB_RTU_FC1.uiStartAddress      := UINT#3000;
        udtExample.udtMB_RTU_FC1.iDataCount          := 100;
        IF  udtExample.udtMB_RTU_FC1.xActive = TRUE AND
            udtExample.udtMB_RTU_FC1.xError = FALSE AND
            udtExample.udtMB_RTU_FC1.xBusy = FALSE
        THEN
            udtExample.iState   := 40;
        ELSIF udtExample.udtMB_RTU_FC1.xError THEN
```

```
            udtExample.iState   := 998;
        END_IF;
    40: (* Activate FC2 *)
        udtExample.udtMB_RTU_FC2.xActivate          := TRUE;
        udtExample.udtMB_RTU_FC2.xSendRequest       := FALSE;
        udtExample.udtMB_RTU_FC2.uiSlaveAddress     := UINT#1;
        udtExample.udtMB_RTU_FC2.xEnablePoll        := FALSE;
        udtExample.udtMB_RTU_FC2.tPollInterval      := T#0s;
        udtExample.udtMB_RTU_FC2.uiStartAddress     := UINT#3000;
        udtExample.udtMB_RTU_FC2.iDataCount         := 100;
        IF  udtExample.udtMB_RTU_FC2.xActive = TRUE AND
            udtExample.udtMB_RTU_FC2.xError = FALSE AND
            udtExample.udtMB_RTU_FC2.xBusy = FALSE
        THEN
            udtExample.iState   := 50;
        ELSIF udtExample.udtMB_RTU_FC2.xError THEN
            udtExample.iState   := 998;
        END_IF;
    50: (* start send request FC1 *)
        udtExample.udtMB_RTU_FC1.xSendRequest   := TRUE;
        IF  udtExample.udtMB_RTU_FC1.xActive = TRUE AND
            udtExample.udtMB_RTU_FC1.xError = FALSE AND
            udtExample.udtMB_RTU_FC1.xBusy = FALSE AND
            udtExample.udtMB_RTU_FC1.xDone = TRUE
        THEN
            udtExample.udtMB_RTU_FC1.xSendRequest   := FALSE;
            udtExample.iState   := 60;
        ELSIF udtExample.udtMB_RTU_FC1.xError THEN
            udtExample.iState   := 998;
        END_IF;
    60: (* start send request FC2 *)
        udtExample.udtMB_RTU_FC2.xSendRequest   := TRUE;
        IF  udtExample.udtMB_RTU_FC2.xActive = TRUE AND
            udtExample.udtMB_RTU_FC2.xError = FALSE AND
            udtExample.udtMB_RTU_FC2.xBusy = FALSE AND
            udtExample.udtMB_RTU_FC2.xDone = TRUE
        THEN
            udtExample.udtMB_RTU_FC2.xSendRequest   := FALSE;
            udtExample.iState   := 70;
        ELSIF udtExample.udtMB_RTU_FC2.xError THEN
            udtExample.iState   := 998;
        END_IF;
    70: (* wait for deactivation - xStart = FALSE *)
        IF xStart = FALSE THEN
            udtExample.udtMB_IL_485E_Master.xActivate   := FALSE;
            udtExample.udtMB_RTU_FC1.xActivate          := FALSE;
            udtExample.udtMB_RTU_FC2.xActivate          := FALSE;
            udtExample.iState   := 80;
        END_IF;
    80: (* go back to State 0 *)
        IF  udtExample.udtMB_IL_485E_Master.xActive = FALSE AND
            udtExample.udtMB_RTU_FC1.xActive = FALSE AND
            udtExample.udtMB_RTU_FC2.xActive = FALSE
        THEN
            udtExample.iState   := 0;
        END_IF;
    998: (* Error state *)
        IF xStart = FALSE THEN
            udtExample.udtMB_IL_485E_Master.xActivate   := FALSE;
            udtExample.udtMB_RTU_FC1.xActivate          := FALSE;
            udtExample.udtMB_RTU_FC2.xActivate          := FALSE;
            udtExample.iState   := 999;
        END_IF;
    999: (* go back to State 0 *)
        IF  udtExample.udtMB_IL_485E_Master.xActive = FALSE AND
            udtExample.udtMB_RTU_FC1.xActive = FALSE AND
            udtExample.udtMB_RTU_FC2.xActive = FALSE
        THEN
            udtExample.iState   := 0;
        END_IF;
END_CASE;
```

35.4 Example 4: Modbus_RTU IL slave functionality

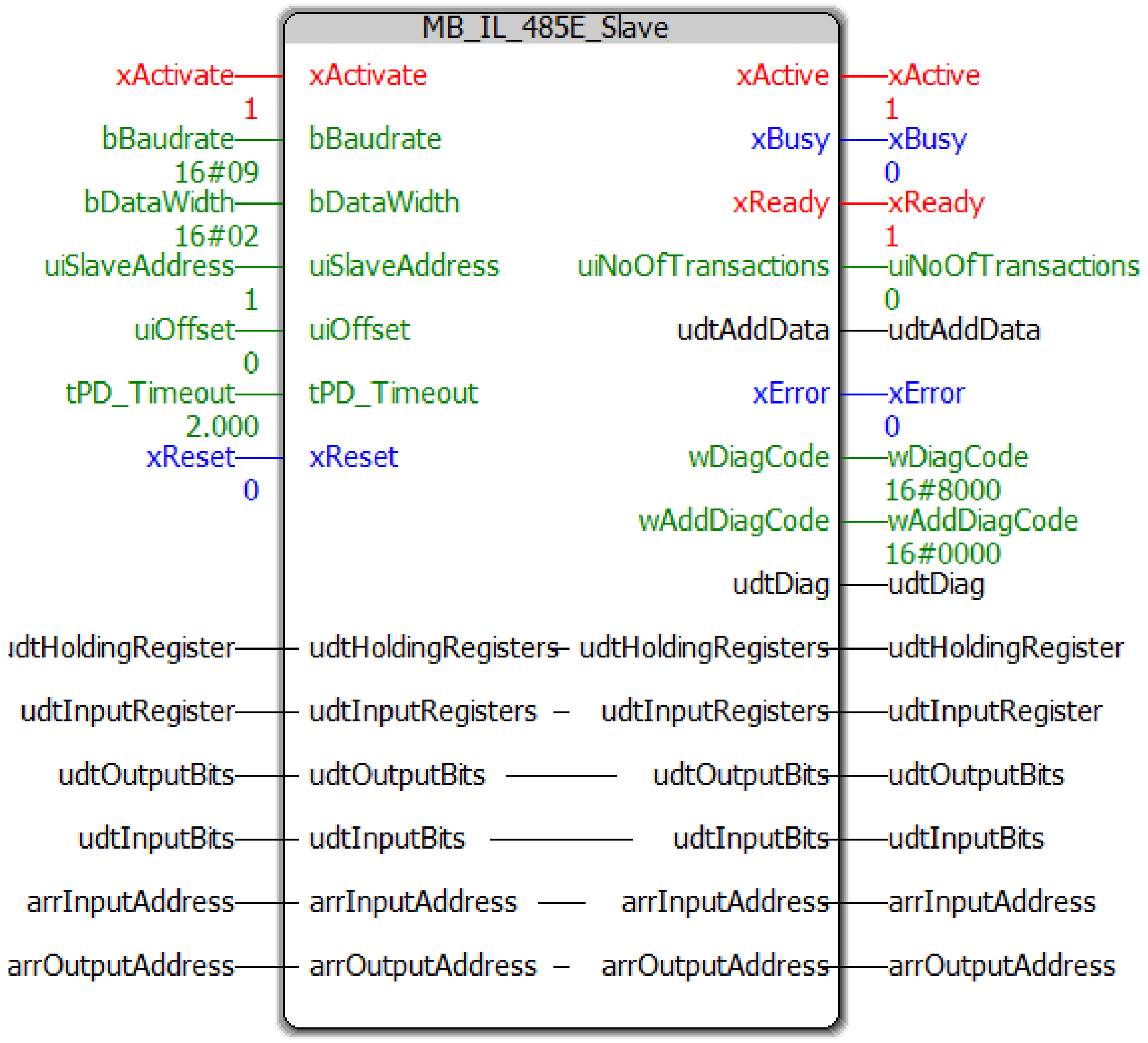
35.4.1 Plant

For this example, the following hardware is used:

- ILC 370 PN 2TX-IB (2876915)
- IL PN BK DI8 DO4 2TX-PAC (2703994)
- IB IL RS 485-ECO (2702795)

35.4.2 Modbus slave with IB IL RS 485-ECO (2702795)

MB_IL_485E_SLAVE:



36 Appendix

36.1 Data types

```
(* Modbus *)
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;
    (* additional 16 bits were added to avoid out of range error when
        processing the last 16 bits *)
    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;
    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 *)
        tTimeout                :   TIME;   (* input tTimeout of the Modbus_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_126;  (* 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;
END_TYPE

(* Diagnostic Structures udtDiag *)
TYPE
    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;

MB_UDT_ILRS232P_DIAG    :
STRUCT
    iState          :   INT;
    wDiagCode       :   WORD;
    wAddDiagCode    :   WORD;
END_STRUCT;

MB_UDT_ILRS232ECO_DIAG  :
STRUCT
    iState          :   INT;    (* Current state of statemachine *)
    wDiagCode       :   WORD;   (* Diag Code *)
    wAddDiagCode    :   WORD;   (* Additional Diag Code *)
END_STRUCT;

MB_UDT_ILRS485ECO_DIAG  :
STRUCT
    iState          :   INT;
    wDiagCode       :   WORD;
    wAddDiagCode    :   WORD;
END_STRUCT;

MB_UDT_ILRS485P_DIAG    :
STRUCT
    iState          :   INT;
    wDiagCode       :   WORD;
    wAddDiagCode    :   WORD;
```

```
END_STRUCT;

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

MB_UDT_IL_UNI_MASTER_DIAG   :
STRUCT
    udtMB_ILRSUNI_Diag      :   MB_UDT_ILRSUNI_DIAG;
    udtMB_RTU_Master_Diag   :   MB_UDT_RTU_MASTER_DIAG;
END_STRUCT;

MB_UDT_IL_RS232P_MASTER_DIAG    :
STRUCT
    udtMB_ILRS232P_Diag     :   MB_UDT_ILRS232P_DIAG;
    udtMB_RTU_Master_Diag   :   MB_UDT_RTU_MASTER_DIAG;
END_STRUCT;

MB_UDT_IL_232E_MASTER_DIAG      :
STRUCT
    udtMB_ILRS232ECO_Diag   :   MB_UDT_ILRS232ECO_DIAG;
    udtMB_RTU_Master_Diag   :   MB_UDT_RTU_MASTER_DIAG;
END_STRUCT;

MB_UDT_IL_485E_MASTER_DIAG  :
STRUCT
    udtMB_ILRS485ECO_Diag   :   MB_UDT_ILRS485ECO_DIAG;
    udtMB_RTU_Master_Diag   :   MB_UDT_RTU_MASTER_DIAG;
END_STRUCT;

MB_UDT_IL_485P_MASTER_DIAG  :
STRUCT
    udtMB_ILRS485P_Diag     :   MB_UDT_ILRS485P_DIAG;
    udtMB_RTU_Master_Diag   :   MB_UDT_RTU_MASTER_DIAG;
END_STRUCT;

MB_UDT_IL_UNI_SLAVE_DIAG    :
STRUCT
    udtMB_ILRSUNI_Diag      :   MB_UDT_ILRSUNI_DIAG;
    udtMB_RTU_Slave_Diag    :   MB_UDT_RTU_SLAVE_DIAG;
END_STRUCT;

MB_UDT_IL_232P_SLAVE_DIAG   :
STRUCT
    udtMB_ILRS232P_Diag     :   MB_UDT_ILRS232P_DIAG;
    udtMB_RTU_Slave_Diag    :   MB_UDT_RTU_SLAVE_DIAG;
END_STRUCT;

MB_UDT_IL_232E_SLAVE_DIAG       :
STRUCT
    udtMB_ILRS232ECO_Diag   :   MB_UDT_ILRS232ECO_DIAG;
    udtMB_RTU_Slave_Diag    :   MB_UDT_RTU_SLAVE_DIAG;
END_STRUCT;

MB_UDT_IL_485E_SLAVE_DIAG   :
STRUCT
    udtMB_ILRS485ECO_Diag   :   MB_UDT_ILRS485ECO_DIAG;
    udtMB_RTU_Slave_Diag    :   MB_UDT_RTU_SLAVE_DIAG;
END_STRUCT;

MB_UDT_IL_485P_SLAVE_DIAG   :
    STRUCT
        udtMB_ILRS485P_Diag     :   MB_UDT_ILRS485P_DIAG;
        udtMB_RTU_Slave_Diag    :   MB_UDT_RTU_SLAVE_DIAG;
    END_STRUCT;


    MB_UDT_AXL_RSUNI_DIAG_MASTER : STRUCT
        udtMB_AXL_RS_UNI_REC_Diag   :   MB_UDT_RTU_REC_DIAG;
        udtMB_AXL_RS_UNI_SND_Diag   :   MB_UDT_RTU_SND_DIAG;
        udtMB_RTU_Master_Diag       :   MB_UDT_RTU_MASTER_DIAG;
    END_STRUCT;
    MB_UDT_AXL_RSUNI_DIAG_SLAVE : STRUCT
        udtMB_AXL_RS_UNI_REC_Diag   :   MB_UDT_RTU_REC_DIAG;
        udtMB_AXL_RS_UNI_SND_Diag   :   MB_UDT_RTU_SND_DIAG;
        udtMB_RTU_Slave_Diag        :   MB_UDT_RTU_SLAVE_DIAG;
    END_STRUCT;

    MB_UDT_AXL_SE_RS485_DIAG_MA : STRUCT
        udtMB_AXL_RS_UNI_REC_Diag   :   MB_UDT_RTU_REC_DIAG;
        udtMB_AXL_RS_UNI_SND_Diag   :   MB_UDT_RTU_SND_DIAG;
        udtMB_RTU_Master_Diag       :   MB_UDT_RTU_MASTER_DIAG;
    END_STRUCT;
    MB_UDT_AXL_SE_RS485_DIAG_SLAVE : STRUCT
        udtMB_AXL_RS_UNI_REC_Diag   :   MB_UDT_RTU_REC_DIAG;
        udtMB_AXL_RS_UNI_SND_Diag   :   MB_UDT_RTU_SND_DIAG;
        udtMB_RTU_Slave_Diag        :   MB_UDT_RTU_SLAVE_DIAG;
    END_STRUCT;
END_TYPE
END_TYPE


(* AXL RS UNI *)
TYPE
    (* *** 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;


    (* rs uni by processdata *)
    (* 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

        (* Error in module - peripheral fault or invalid command *)
        xErrorModule        : BOOL;

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

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

        (* status of the sending part of the module *)
        (* TRUE -> Error during data send operation of the module *)
        xErrorSend          : BOOL;
        (* TRUE -> Send buffer of module full *)
        xSendBufferFull     : BOOL;
        (* TRUE -> Send buffer of module not empty *)
        xSendBufferNotEmpty : BOOL;
        (* Number of characters in the receive buffer of the module *)
        uiRcvBufferModule   : UINT;
        (* Firmware version of the module *)
        wFirmwareVersion    : WORD;

    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
        (* TRUE -> Serial Driver is Activated *)
        xActive                     : BOOL;
        (* TRUE -> Serial IL Driver is Ready to send / receive *)
        xReady                      : BOOL;
        (* TRUE -> End to end protocoll is used for communication *)
        xEndToEnd                   : BOOL;
        (* TRUE -> Acknowledge incoming errors *)
        xAck                        : BOOL;
        (* TRUE -> Reset communication errors automatically *)
        xAutoAck                    : BOOL;
        (* TRUE -> Turn on DTR function of module *)
        xDTR                        : BOOL;
        (* TRUE -> Read status counters of the module *)
        xReadStatusCounter          : BOOL;
        (* TRUE -> Send send request to module *)
        xSend                       : BOOL;
        (* Number of bytes to be sent *)
        uiSendLength                : UINT;
        (* TRUE -> Reset receive buffer of function block *)
        xResetRecBuf                : BOOL;
        (* Number of characters to be read in *)
        uiRcvLength                 : UINT;
        (* Function block in sending mode *)
        xFBSending                  : BOOL;
        (* Function block in receiving mode *)
        xFBReceiving                : BOOL;
        (* Structure containing status counters *)
        udtStatusCounter            : MB2_AXL_RSUNI2_UDT_COUNTER;
        (* Structure containing status of serial interface *)
        udtStatusSerialInterface    : MB2_AXL_RSUNI2_UDT_STATUS;
        (* Receive or send error existing *)
        xStatusFailure              : BOOL;
        (* TRUE -> Receive buffer containing data *)
        xRcvBufferNotEmpty          : BOOL;
        (* TRUE -> Software receive buffer full *)
        xRecBufFull                 : BOOL;
        (* Finished reading in status counter same cycle *)
        xReadCounterDone            : BOOL;
        (* Finished sending of data same cycle *)
        xSendDone                   : BOOL;
        (* Finished reading of data same cycle *)
        xNDR                        : BOOL;
        (* Number of read in characters *)
        uiRcvDataLength             : UINT;
        (* Timeout value for timeout in CASE of freezed receiving, sending, buffering operation
        tTimeout                    : TIME;
        (* Array containing received data *)
        arrRcvData                  : MB2_AXL_RSUNI2_ARR_B_1_4096;
        (* Array containing data to be sent *)
        arrSendData                 : MB2_AXL_RSUNI2_ARR_B_1_1023;
        (* 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;

END_TYPE

TYPE

(* IL RS UNI *)

    (* IB IL RS485 PRO *)
    MB2_COM_UDT_RS485P_PARA_V1  :               (* Padding-Bytes *)
    STRUCT
        diBaudrate          :   DINT;
        bDataWidth          :   BYTE;
        bErrorPattern       :   BYTE;
        xOutputTyp          :   BOOL;
        bFirstDelimeter     :   BYTE;
        bSecondDelimeter    :   BYTE;
        bProtocol           :   BYTE;
        bDummy1             :   BYTE;
        bDummy2             :   BYTE;
    END_STRUCT;

    MB2_COM_UDT_RS485P_DATA_V1  :
    STRUCT
        xErrorRcv           :   BOOL;
        xErrorSend          :   BOOL;
        xRcvBufferFull      :   BOOL;
        xSendBufferFull     :   BOOL;
        xSendBufferNotEmpty :   BOOL;
        uiRcvCounter        :   UINT;
        wFWVersion          :   WORD;
    END_STRUCT;

    (* ** Datentypen RS232P_xxx*********************************** *)
    MB2_COM_UDT_RS232P_PARA_V1  :                (* Padding-Bytes *)
    STRUCT
        diBaudrate          :   DINT;
        bDataWidth          :   BYTE;
        bErrorPattern       :   BYTE;
        xDTR_Control        :   BOOL;
        xCTS_Output         :   BOOL;
        bFirstDelimeter     :   BYTE;
        bSecondDelimeter    :   BYTE;
        bProtocol           :   BYTE;
        bDummy1             :   BYTE;

    END_STRUCT;

    MB2_COM_UDT_RS232P_DATA_V1  :
    STRUCT
        xCTS                :   BOOL;
        xErrorRcv3964R      :   BOOL;
        xErrorSend3964R     :   BOOL;
        xRcvBufferFull      :   BOOL;
        xSendBufferFull     :   BOOL;
        xSendBufferNotEmpty :   BOOL;
        uiRcvCounter        :   UINT;
        wFWVersion          :   WORD;
    END_STRUCT;

(* Parameterization , IL_RSUNI_xxx_V1_01 *)      (* Padding-Bytes *)
    MB2_RSUNI_UDT_PARA_V2       :
    STRUCT
        bSelectMode         :   BYTE;
        xDTR_Control        :   BOOL;
        diBaudrate          :   DINT;
        wDatawidth          :   WORD;
        bErrorPattern       :   BYTE;
        bSecondDelimeter    :   BYTE;
        bFirstDelimeter     :   BYTE;
        bProtocol           :   BYTE;
        bDummy1             :   BYTE;
        bDummy2             :   BYTE;
    END_STRUCT;

(*  (* Additional Diagnostics *)
    MB2_RSUNI_UDT_DATA_V1       :
    STRUCT
        xDCD                :   BOOL;
        xDSR                :   BOOL;
        xRcvBufferNotEmpty  :   BOOL;
        xRcvBufferFull      :   BOOL;
        xSendBufferFull     :   BOOL;
        xSendBufferNotEmpty :   BOOL;
        wRcvBufferHW        :   WORD;
        wRcvBufferSW        :   WORD;
        wRcvCounter         :   WORD;
        wRcvCounterFailed   :   WORD;
        wSendCounter        :   WORD;
        wFWVersion          :   WORD;
        iState              :   INT;     (* Current state of statemachine *)
        wDiagCode           :   WORD;    (* Diag Code *)
        wAddDiagCode        :   WORD;    (* Additional Diag Code *)
    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;
    (* process data width 11 BYTE *)
    MB2_RSUNI_ARR_B_1_14    : ARRAY [1..14]     OF BYTE;
    (* process data width 27 BYTE *)
    MB2_RSUNI_ARR_B_1_30    : ARRAY [1..30]     OF BYTE;
    (* process data width 59 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
(* *)
```

37 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