Function block library

EEM_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
- 2 General information
- 3 Change notes
- 4 Supported PLCs
- 5 Function blocks
- 6 EEM_*_RTU
  - 6.1 EEM_0000_RTU
  - 6.2 EEM_0370_RTU
  - 6.3 EEM_0741_RTU
  - 6.4 EEM_1000_RTU
  - 6.5 EEM_1100_RTU
  - 6.6 EEM_2300_RTU
  - 6.7 EEM_2400_RTU
  - 6.8 EEM_8000_RTU
  - 6.9 EEM_8100_RTU
  - 6.10 EEM_8200_RTU
  - 6.11 EEM_8300_RTU
  - 6.12 EEM_8340_RTU
  - 6.13 EEM_8500_RTU
  - 6.14 EEM_8540_RTU
  - 6.15 EEM_8600_RTU
  - 6.16 EEM_8700_RTU
  - 6.17 EEM_8800_RTU
  - 6.18 EEM_9000_RTU
  - 6.19 EEM_9046_RTU
  - 6.20 EEM_9100_RTU
  - 6.21 EEM_9146_RTU
  - 6.22 EEM_9200_RTU
  - 6.23 EEM_9300_RTU
  - 6.24 EEM_9380_RTU
  - 6.25 EEM_9400_RTU
  - 6.26 EEM_9400_RTU
  - 6.27 EEM_9500_RTU
  - 6.28 EEM_9580_RTU
  - 6.29 EEM_C550_RTU
  - 6.30 EEM_C650_RTU
  - 6.31 EEM_C750_RTU
  - 6.32 EEM_C850_RTU
  - 6.33 EEM_C900_RTU
  - 6.34 EEM_C950_RTU
  - 6.35 EEM_C95A_RTU
  - 6.36 EEM_CB00_RTU
- 7 EEM_*_TCP
  - 7.1 EEM_0000_TCP
  - 7.2 EEM_0370_TCP
  - 7.3 EEM_0741_TCP
  - 7.4 EEM_1000_TCP
  - 7.5 EEM_1100_TCP
  - 7.6 EEM_2300_TCP
  - 7.7 EEM_2400_TCP
  - 7.8 EEM_8000_TCP
  - 7.9 EEM_8100_TCP
  - 7.10 EEM_8200_TCP

- 7.11 EEM_8300_TCP
- 7.12 EEM_8340_TCP
- 7.13 EEM_8500_TCP
- 7.14 EEM_8540_TCP
- 7.15 EEM_8600_TCP
- 7.16 EEM_8700_TCP
- 7.17 EEM_8800_TCP
- 7.18 EEM_9000_TCP
- 7.19 EEM_9046_TCP
- 7.20 EEM_9100_TCP
- 7.21 EEM_9146_TCP
- 7.22 EEM_9200_TCP
- 7.23 EEM_9300_TCP
- 7.24 EEM_9380_TCP
- 7.25 EEM_9400_TCP
- 7.26 EEM_9480_TCP
- 7.27 EEM_9500_TCP
- 7.28 EEM_9580_TCP
- 7.29 EEM_C550_TCP
- 7.30 EEM_C650_TCP
- 7.31 EEM_C750_TCP
- 7.32 EEM_C850_TCP
- 7.33 EEM_C900_TCP
- 7.34 EEM_C950_TCP
- 7.35 EEM_C95A_TCP
- 7.36 EEM_CB00_TCP

- 8 EEM_*_Values
  - 8.1 EEM_0000_Values
  - 8.2 EEM_0370_Values
  - 8.3 EEM_0741_Values
  - 8.4 EEM_1000_Values
  - 8.5 EEM_1100_Values
  - 8.6 EEM_2300_Values
  - 8.7 EEM_2400_Values
  - 8.8 EEM_8000_Values
  - 8.9 EEM_8100_Values
  - 8.10 EEM_8200_Values
  - 8.11 EEM_8300_Values
  - 8.12 EEM_8340_Values
  - 8.13 EEM_8500_Values
  - 8.14 EEM_8540_Values
  - 8.15 EEM_8600_Values
  - 8.16 EEM_8700_Values
  - 8.17 EEM_8800_Values
  - 8.18 EEM_9000_Values
  - 8.19 EEM_9046_Values
  - 8.20 EEM_9100_Values
  - 8.21 EEM_9146_Values
  - 8.22 EEM_9200_Values
  - 8.23 EEM_9300_Values
  - 8.24 EEM_9380_Values
  - 8.25 EEM_9400_Values
  - 8.26 EEM_9480_Values
  - 8.27 EEM_9500_Values
  - 8.28 EEM_9580_Values
  - 8.29 EEM_C550_Values
  - 8.30 EEM_C650_Values
  - 8.31 EEM_C750_Values

- 8.32 EEM_C850_Values
- 8.33 EEM_C900_Values
- 8.34 EEM_C950_Values
- 8.35 EEM_CB00_Values

- 9 Startup examples
  - 9.1 Example 1: EEM_*_EXA_AXL_RSUNI
  - 9.2 Example 2: EEM_*_EXA_TCP
  - 9.3 Example 3: EEM_*_EXA_AXL_RSUNI_PN

- 10 Appendix
  - 10.1 Data types

- 11 In case of problems
- 12 Support

1 Installation hint

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

The EEM function block library provides function blocks for communication with Electric Energy Meters (EEM). Communication can take place via Modbus RTU, Modbus TCP or RS232. To use the library, also the Modbus_RTU or Modbus_TCP libraries are needed. They are used in this library.

Note:

Always use the current Modbus_RTU and Modbus_TCP libraries available on the Phoenix Contact website. Integrate all the EEM and the Modbus_RTU or Mobus_TCP libraries into your project. Make sure that the current Modbus_RTU or Modbus_TCP libraries are also integrated in the EEM library. If the EEM library cannot be transferred, even though the current Modbus_RTU or Modbus_TCP libraries are integrated, check the path of the integrated library and change it if the path does not match the storage location of the current library.

3 Change notes

Library version	Library build	PC Worx version	Change notes
5	20220901	6.30.2972	Include of Modbus_RTU_8 and Modbus_TCP_5 libraries
4	20210422	6.30.2972	• EEM_9380_RTU: Bug fix - revised double parameter • EEM_8000_*: Bug fix - data type INT instead of SINT16 (data sheet) • EEM_8800_*: Bug fix - data type INT instead of SINT16 (data sheet)
3	20210317	6.30.2907	Added new function blocks for the following register tables: EEM_2300_* : Tab2300 EEM_2400_* : Tab2400 EEM_9000_* : Tab9000 EEM_9046_* : Tab9046 EEM_9100_* : Tab9100 EEM_9200_* : Tab9200 EEM_9146_* : Tab9146 EEM_9300_* : Tab9300 EEM_9380_* : Tab9380 EEM_9400_* : Tab9400 EEM_9480_* : Tab9480 EEM_9500_* : Tab9500 EEM_9580_* : Tab9580
2	20201110	6.30.2907	Revised documentation for EEM_*_TCP: • Changed wrong data type of arrReadData from MB2_TCP_ARR_W_1_125 to MB_TCP_ARR_W_1_125
2	20200923	6.30.2907	• Instead of Modbus, Modbus_RTU and Modbus_TCP must be used. • ComSerial is not longer necessary as the driver FBs have been integrated into the Modbus* libraries.
1.00	–	6.10.200	Initial version

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)

5 Function blocks

Function block	Description	Version	Supported articles	License
EEM_0000_RTU	Function block for reading and standardizing instantaneous values of register addresses 16#0000 – 16#0052.	1	EEM-EM355 (2908578) EEM-EM357 (2908588)	none
EEM_0370_RTU	Function block for reading and standardizing values of register addresses 16#0370 - 16#0374. The register addresses display current average current/voltage value based on current and voltage transducer (under consideration of transmission ratios).	1	EEM-MA600 (2901366)[*2], [*10]	none
EEM_0741_RTU	Function block for reading and standardizing values of register addresses 16#0741 – 16#0743. The register addresses display current average current/voltage value not based on current and voltage transducer (without consideration of transmission ratios).	1	"	none
EEM_1000_RTU	Function block for reading and standardizing instantaneous values of register addresses 16#1000 – 16#103A.	1	EEM-EM355 (2908578) EEM-EM357 (2908588)	none
EEM_1100_RTU	Function block for reading and standardizing meter values of register addresses 16#1100 – 16#114E.	1	"	none
EEM_2300_RTU	Function block is used for extern modules/extern gateway Proficloud using register addresses 16#2300 – 16#2322.	1	EEM-SB370-C (1158951) EEM-SB371-C (1158947)	none
EEM_2400_RTU	Function block is used for extern modules / extern Gateway Profinet / Ethernet/IP using register addresses 16#2400 – 16#240B.	1	EEM-MA770-PN (2907946) EEM-MA771-PN (2908301) EEM-MA770-EIP (2907953) EEM-MA771-EIP (2908302) EEM-MB370-PN (2907984) EEM-MB371-PN (2908308) EEM-MB370-EIP (2907971) EEM-MB371-EIP (2907976)	none

EEM_8000_RTU	Function block for reading and standardizing instantaneous values of register addresses 16#8000 – 16#8057.	2	EEM-MA250 (2901363)[*10] EEM-MA370 (2907983) EEM-MA371-R (29079850) EEM-MA400 (2901364)[*1], [*10] EEM-MA600 (2901366)[*2], [*10] EEM-MA600-24DC (2902352)[*2], [*10] EEM-MA770-R (2907944) EEM-MA771-R (2908285)	none
EEM_8100_RTU	Function block for reading and standardizing energy meter values of register addresses 16#8100 – 16#8116.	1	"	none
EEM_8200_RTU	Function block for reading and standardizing tariff meter values of register addresses 16#8200 – 16#822F.	1	"	none
EEM_8300_RTU	Function block for reading and standardizing operating hours of register addresses 16#8300 – 16#8302.	1	"	none
EEM_8340_RTU	Function block for reading and standardizing the state of impulse counter of digital input of register address 16#8340.	1	"	none
EEM_8500_RTU	Function block for reading and standardizing statistics (prediction) of register addresses 16#8500 - 16#850E.	1	"	none
EEM_8540_RTU	Function block for reading and standardizing statistics (mean values) of register addresses 16#8540 - 16#855E.	1	"	none
EEM_8600_RTU	Function block for reading and standardizing min / max values for mean values of register addresses 16#8600 - 16#863E.	1	"	none
EEM_8700_RTU	Function block for reading and standardizing min / max values for instantaneous values of register addresses 16#8700 - 16#873E.	1	"	none
EEM_8800_RTU	Function block for reading and standardizing total harmonic distortion of register addresses 16#8800 - 16#8809.	2	"	none

EEM_9000_RTU	Function block for reading out voltage dips of register addresses 16#9000 - 16#9045.	1	EEM-SB370-C (1158951) EEM-SB371-C (1158947) EEM-MA370 (2907983) EEM-MA371 (2908307) EEM-MA370-R (2907980) EEM-MA371-R (29079850) EEM-MA770 (2907945) EEM-MA771 (2908286) EEM-MA770-R (2907944) EEM-MA771-R (2908285) EEM-MA770-PN (2907946) EEM-MA771-PN (2908301) EEM-MA770-EIP (2907953) EEM-MA771-EIP (2908302) EEM-MB370 (2907954) EEM-MB371 (2907955) EEM-MB370-PN (2907984) EEM-MB371-PN (2908308) EEM-MB370-EIP (2907971) EEM-MB371-EIP (2907976)	none
EEM_9046_RTU	Function block for reading out voltage dips of register addresses 16#9046 - 16#908B.	1	"	none
EEM_9100_RTU	Function block for reading out voltage swells of register addresses 16#9100 - 16#9145.	1	"	none
EEM_9146_RTU	Function block for reading out voltage swells of register addresses 16#9146 - 16#918B.	1	"	none
EEM_9300_RTU	Function block for reading energy meters (not resetable) of register addresses 16#9300 - 16#9356.	1	"	none
EEM_9380_RTU	Function block for reading energy meters (resetable) of register addresses 16#9380 - 16#93D6.	2	"	none

EEM_9400_RTU	Function block for reading energy meter tariff 1 of register addresses 16#9400 - 16#9456.	1	"	none
EEM_9480_RTU	Function block for reading energy meter tariff 2 of register addresses 16#9480 - 16#94D6.	1	"	none
EEM_9500_RTU	Function block for reading energy meter tariff 3 of register addresses 16#9500 - 16#9556.	1	"	none
EEM_9580_RTU	Function block for reading energy meter tariff 4 of register addresses 16#9580 - 16#95D6.	1	"	none
EEM_C550_RTU	Function block for reading and standardizing values of register addresses 16#C550 – 16#C58C. The register addresses display the current main measured values based on current and voltage transducer (under consideration of transmission ratios).	1	EEM-MA250 (2901363)[*10] EEM-EM355 (2908578) EEM-EM357 (2908588) EEM-MA400 (2901364)[*1], [*10] EEM-MA600 (2901366)[*2], [*10] EEM-MA600-24DC (2902352)[*2], [*10]	none
EEM_C650_RTU	Function block for reading and standardizing values of register addresses 16#C650 - 16#C690. The register addresses display energy values.	1	"	none
EEM_C750_RTU	Function block for reading and standardizing values of register addresses 16#C750 - 16#C795. The register addresses display average values based on current and voltage transducer via the set measurement duration (under consideration of transmission ratios).	1	EEM-MA250 (2901363)[*10] EEM-MA400 (2901364)[*1], [*10] EEM-MA600 (2901366)[*2], [*10] EEM-MA600-24DC (2902352)[*2], [*10]	none
EEM_C850_RTU	Function block for reading and standardizing values of register addresses 16#C850 - 16#C872. The register addresses display current main measured values not based on current and voltage transducer (without consideration of transmission ratios).	1	"	none
EEM_C900_RTU	Function block for reading and standardizing values of register addresses 16#C900 - 16#C907. The register addresses display temperature inputs.	1	"	none
EEM_C950_RTU	Function block for reading and standardizing values of register addresses 16#C950 - 16#C959. The register addresses display current harmonic distortion and harmonics.	1	"	none

EEM_C95A_RTU	Function block for reading and standardizing values of register addresses 16#C95A - 16#CA92. The register addresses display individual harmonic content for currents.	1	EEM-MA600 (2901366) [*2],[*10] EEM-MA600-24DC (2902352) [*2],[*10]	none
EEM_CB00_RTU	Function block for reading and standardizing values of register addresses 16#CB00 - CB03. The register addresses display the status of inputs/outputs.	1	EEM-MA250 (2901363)[*10] EEM-MA400 (2901364)[*1], [*10] EEM-MA600 (2901366)[*2], [*10] EEM-MA600-24DC (2902352)[*2], [*10]	none
EEM_0000_TCP	Function for reading and standardizing instantaneous values of register addresses 16#0000 – 16#0052.	1	EEM-MA375 (2908581) EEM-MA377 (2908590)	none
EEM_0370_TCP	Function block for reading and standardizing current average current/voltage value based on current and voltage transducer (under consideration of transmission ratios) of register addresses 16#0370 – 16#0374.	1	EEM-MA600 (2901366)[*2], [*10]	none
EEM_0741_TCP	Function block for reading and standardizing current average current/voltage value not based on current and voltage transducer (without consideration of transmission ratios) of register addresses 16#0741 – 16#0743.	1	"	none
EEM_1000_TCP	Function block for reading and standardizing instantaneous values of register addresses 16#1000 – 16#103A.	1	EEM-MA375 (2908581) EEM-MA377 (2908590)	none
EEM_1100_TCP	Function block for reading and standardizing meter values of register addresses 16#1100 – 16#114E.	1	"	none
EEM_2300_TCP	Function block is used for extern modules/extern gateway Proficloud using register addresses 16#2300 – 16#2322.	1	EEM-SB370-C (1158951) EEM-SB371-C (1158947)	none

EEM_2400_TCP	Function block is used for extern modules / extern Gateway Profinet / Ethernet/IP using register addresses 16#2400 – 16#240B.	1	EEM-MA770-PN (2907946) EEM-MA771-PN (2908301) EEM-MA770-EIP (2907953) EEM-MA771-EIP (2908302) EEM-MB370-PN (2907984) EEM-MB371-PN (2908308) EEM-MB370-EIP (2907971) EEM-MB371-EIP (2907976)	none

EEM_8000_TCP	Function block for reading and standardizing instantaneous values of register addresses 16#8000 – 16#8057.	2	EEM-MA370 (2907983) EEM-MA370-R (2907980) EEM-MA371 (2908307) EEM-MA371-R (29079850) EEM-MA600 (2901366)[*3], [*10] EEM-MA600-24DC (2902352)[*3], [*10] EEM-MA770 (2907945) EEM-MA770-R (2907944) EEM-MA770-EIP (2907953) EEM-MA770-PN (2907946) EEM-MA771 (2908286) EEM-MA771-R (2908285) EEM-MA771-EIP (2908302) EEM-MA771-PN (2908301) EEM-MB370 (2907954) EEM-MB370-EIP (2907971) EEM-MB370-PN (2907984) EEM-MB371 (2907955) EEM-MB371-EIP (2907976) EEM-MB371-PN (2908308)	none
EEM_8100_TCP	Function block for reading and standardizing energy meter values of register addresses 16#8100 – 16#8116.	1	"	none
EEM_8200_TCP	Function block for reading and standardizing tariff meter values of register addresses 16#8200 – 16#822F.	1	"	none
EEM_8300_TCP	Function block for reading and standardizing operating hours of register addresses 16#8300 – 16#8302.	1	"	none

EEM_8340_TCP	Function block for reading and standardizing the state of impulse counter of digital input of register address 16#8340.	1	"	none
EEM_8500_TCP	Function block for reading and standardizing statistics (prediction) of register addresses 16#8500 - 16#850E.	1	"	none
EEM_8540_TCP	Function block for reading and standardizing statistics (mean values) of register addresses 16#8540 - 16#855E.	1	"	none
EEM_8600_TCP	Function block for reading and standardizing min / max values for mean values of register addresses 16#8600 - 16#863E.	1	"	none
EEM_8700_TCP	Function block for reading and standardizing min / max values for instantaneous values of register addresses 16#8700 - 16#873E.	1	"	none
EEM_8800_TCP	Function block for reading and standardizing total harmonic distortion of register addresses 16#8800 - 16#8809.	2	"	none

EEM_9000_TCP	Function block for reading out voltage dips of register addresses 16#9000 - 16#9045.	1	EEM-SB370-C (1158951) EEM-SB371-C (1158947) EEM-MA370 (2907983) EEM-MA371 (2908307) EEM-MA370-R (2907980) EEM-MA371-R (29079850) EEM-MA770 (2907945) EEM-MA771 (2908286) EEM-MA770-R (2907944) EEM-MA771-R (2908285) EEM-MA770-PN (2907946) EEM-MA771-PN (2908301) EEM-MA770-EIP (2907953) EEM-MA771-EIP (2908302) EEM-MB370 (2907954) EEM-MB371 (2907955) EEM-MB370-PN (2907984) EEM-MB371-PN (2908308) EEM-MB370-EIP (2907971) EEM-MB371-EIP (2907976)	none
EEM_9046_TCP	Function block for reading out voltage dips of register addresses 16#9046 - 16#908B.	1	"	none
EEM_9100_TCP	Function block for reading out voltage swells of register addresses 16#9100 - 16#9145.	1	"	none
EEM_9146_TCP	Function block for reading out voltage swells of register addresses 16#9146 - 16#918B.	1	"	none
EEM_9300_TCP	Function block for reading energy meters (not resetable) of register addresses 16#9300 - 16#9356.	1	"	none
EEM_9380_TCP	Function block for reading energy meters (resetable) of register addresses 16#9380 - 16#93D6.	1	"	none
EEM_9400_TCP	Function block for reading energy meter tariff 1 of register addresses 16#9400 - 16#9456.	1	"	none

EEM_9480_TCP	Function block for reading energy meter tariff 2 of register addresses 16#9480 - 16#94D6.	1	"	none
EEM_9500_TCP	Function block for reading energy meter tariff 3 of register addresses 16#9500 - 16#9556.	1	"	none
EEM_9580_TCP	Function block for reading energy meter tariff 4 of register addresses 16#9580 - 16#95D6.	1	"	none
EEM_C550_TCP	Function block for reading and standardizing the data in Table 16#C550 (register addresses C550 – C58C) from EMPros 250, 400 and 600. The register addresses display the current main measured values based on current and voltage transducer (under consideration of transmission ratios).	1	EEM-MA600 (2901366)[*3], [*10] EEM-MA600-24DC (2902352)[*3], [*10]	none
EEM_C650_TCP	Function block for reading and standardizing the data in Table 16#C650 (register addresses C650 – C690) from EMPros 250, 400 and 600. The register addresses display energy values.	1	"	none
EEM_C750_TCP	Function block for reading and standardizing the data in Table 16#C750 (register addresses C750 – C795) from EMPros 250, 400 and 600. The register addresses display average values based on current and voltage transducer via the set measurement duration (under consideration of transmission ratios).	1	"	none
EEM_C850_TCP	Function block for reading and standardizing the data in Table 16#C850 (register addresses C850 – C872) from EMPros 250, 400 and 600. The register addresses display current main measured values not based on current and voltage transducer (without consideration of transmission ratios).	1	"	none
EEM_C900_TCP	Function block for reading and standardizing the data in Table 16#C900 (register addresses C900 – C907) from EMPros 250 and 600. The register addresses display temperature inputs.	1	"	none
EEM_C950_TCP	Function block for reading and standardizing the data in Table 16#C950 (register addresses C950 – C959) from EMPros 250, 400 and 600. The register addresses display current harmonic distortion and harmonics.	1	"	none
EEM_C95A_TCP	Function block for reading and standardizing the data in Table 16#C95A (register addresses C95A – CA92) from EMPros 600. The register addresses display individual harmonic content for currents.	1	"	none
EEM_CB00_TCP	Function block for reading and standardizing the data in Table 16#CB00 (register addresses CB00 – CB03) from EMPros 250, 400 and 600. The register addresses display the status of inputs/outputs.	1	"	none
EEM_0000_Values	Function block that converts the EEM_UDT_0000 input structure into individual output parameters.	1	-	none
EEM_0370_Values	Function block that converts the EEM_UDT_0370 input structure into individual output parameters.	1	-	none

EEM_0741_Values	Function block that converts the EEM_UDT_0741 input structure into individual output parameters.	1	-	none
EEM_1000_Values	Function block that converts the EEM_UDT_1000 input structure into individual output parameters.	1	-	none
EEM_1100_Values	Function block that converts the EEM_UDT_Tab1100 input structure into individual output parameters.	1	-	none
EEM_2300_Values	Function block that converts the EEM_UDT_Tab2300 input structure into individual output parameters.	1	-	none
EEM_2400_Values	Function block that converts the EEM_UDT_Tab2400 input structure into individual output parameters.	1	-	none
EEM_8000_Values	Function block that converts the EEM_UDT_8000 input structure into individual output parameters.	2	-	none
EEM_8100_Values	Function block that converts the EEM_UDT_8100 input structure into individual output parameters.	1	-	none
EEM_8200_Values	Function block that converts the EEM_UDT_8200 input structure into individual output parameters.	1	-	none
EEM_8300_Values	Function block that converts the EEM_UDT_8300 input structure into individual output parameters.	1	-	none
EEM_8340_Values	Function block that converts the EEM_UDT_8340 input structure into individual output parameters.	1	-	none
EEM_8500_Values	Function block that converts the EEM_UDT_8500 input structure into individual output parameters.	1	-	none
EEM_8540_Values	Function block that converts the EEM_UDT_8540 input structure into individual output parameters.	1	-	none
EEM_8600_Values	Function block that converts the EEM_UDT_8600 input structure into individual output parameters.	1	-	none
EEM_8700_Values	Function block that converts the EEM_UDT_8700 input structure into individual output parameters.	1	-	none
EEM_8800_Values	Function block that converts the EEM_UDT_8800 input structure into individual output parameters.	2	-	none
EEM_9000_Values	Function block that converts the EEM_UDT_9000 input structure into individual output parameters.	1	EEM-MA771-R (2908285)	none
EEM_9046_Values	Function block that converts the EEM_UDT_9046 input structure into individual output parameters.	1	EEM-MA771-R (2908285)	none
EEM_9100_Values	Function block that converts the EEM_UDT_9100 input structure into individual output parameters.	1	EEM-MA771-R (2908285)	none
EEM_9146_Values	Function block that converts the EEM_UDT_9146 input structure into individual output parameters.	1	EEM-MA771-R (2908285)	none
EEM_9300_Values	Function block that converts the EEM_UDT_9300 input structure into individual output parameters.	1	-	none
EEM_9380_Values	Function block that converts the EEM_UDT_9380 input structure into individual output parameters.	1	-	none
EEM_9400_Values	Function block that converts the EEM_UDT_9400 input structure into individual output parameters.	1	-	none
EEM_9480_Values	Function block that converts the EEM_UDT_9480 input structure into individual output parameters.	1	-	none
EEM_9500_Values	Function block that converts the EEM_UDT_9500 input structure into individual output parameters.	1	-	none

EEM_9580_Values	Function block that converts the EEM_UDT_9580 input structure into individual output parameters.	1	-	none
EEM_C550_Values	Function block that converts the EEM_UDT_C550 input structure into individual output parameters.	1	-	none
EEM_C650_Values	Function block that converts the EEM_UDT_C650 input structure into individual output parameters.	1	-	none
EEM_C750_Values	Function block that converts the EEM_UDT_C750 input structure into individual output parameters.	1	-	none
EEM_C850_Values	Function block that converts the EEM_UDT_C850 input structure into individual output parameters.	1	-	none
EEM_C900_Values	Function block that converts the EEM_UDT_C900 input structure into individual output parameters.	1	-	none
EEM_C950_Values	Function block that converts the EEM_UDT_C950 input structure into individual output parameters.	1	-	none
EEM_CB00_Values	Function block that converts the EEM_UDT_CB00 input structure into individual output parameters.	1	-	none

[*1] with additional module EEM-RS485-MA400 (2901365)

[*2] with additional module EEM-RS485-MA600 (2901367)

[*3] with additional modulel EEM-PB 12-MA600 (2901418)

[*10] Terminated

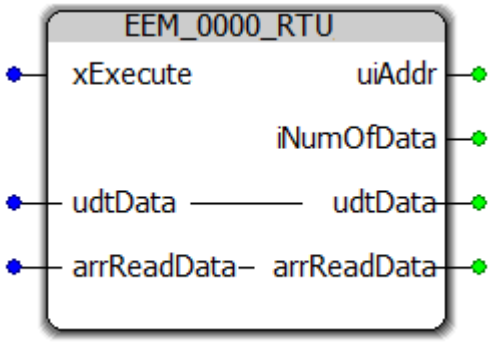
6 EEM_*_RTU

The EEM_*_RTU function blocks are used for communication with Electric Energy Meters (EEM). Communication takes place via Modbus RTU.

6.1 EEM_0000_RTU

Function block EEM_0000_RTU is used to read instantaneous values of register addresses 16#0000 – 16#0052. The values are written into a data structure in a standardized format.

6.1.1 Function block call



6.1.2 Input parameters

Name	Type	Description
xExecute	BOOL	By setting a positive edge, the data is loaded from the Modbus structure into the block, where it is transferred to the input and output structure of the block in a standardized format.

6.1.3 Output parameters

Name	Type	Description
uiAddr	UINT	This parameter transfers the start address of the Modbus register to the Modbus block.
iNumOfData	INT	This parameter transfers the amount of data to be read to the Modbus block.

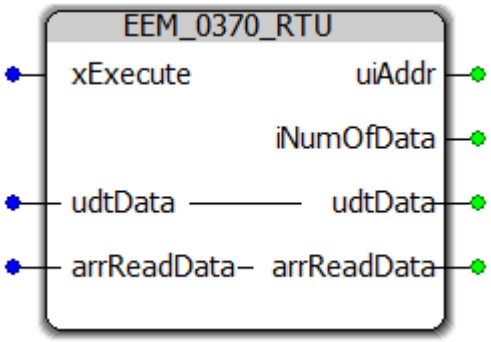
6.1.4 Inout parameters

Name	Type	Description
udtData	EEM_UDT_0000	Input structure for standardized values in the table.
arrReadData	arrModbus2_w_1_125	Structure of the Modbus data.

6.2 EEM_0370_RTU

Function block EEM_0370_RTU is used to read the current average current/voltage value based on current and voltage transducer (under consideration of transmission ratios) of register addresses 16#0370 – 16#0374. The values are written into a data structure in a standardized format.

6.2.1 Function block call



6.2.2 Input parameters

Name	Type	Description
xExecute	BOOL	By setting a positive edge, the data is loaded from the Modbus structure into the block, where it is transferred to the input and output structure of the block in a standardized format.

6.2.3 Output parameters

Name	Type	Description
uiAddr	UINT	This parameter transfers the start address of the Modbus register to the Modbus block.
iNumOfData	INT	This parameter transfers the amount of data to be read to the Modbus block.

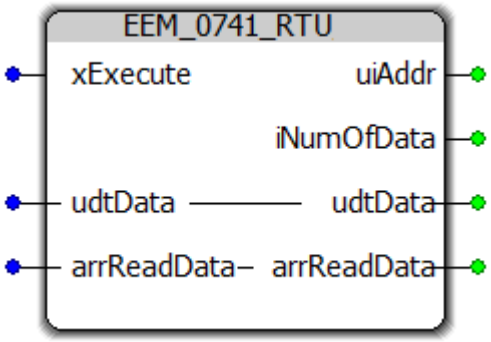
6.2.4 Inout parameters

Name	Type	Description
udtData	EEM_UDT_0370	Input structure for standardized values in the table.
arrReadData	arrModbus2_w_1_125	Structure of the Modbus data.

6.3 EEM_0741_RTU

Function block EEM_0741_RTU is used to read current average current/voltage value not based on current and voltage transducer (without consideration of transmission ratios) of register addresses 16#0741 – 16#0743. The values are written into a data structure in a standardized format.

6.3.1 Function block call



6.3.2 Input parameters

Name	Type	Description
xExecute	BOOL	By setting a positive edge, the data is loaded from the Modbus structure into the block, where it is transferred to the input and output structure of the block in a standardized format.

6.3.3 Output parameters

Name	Type	Description
uiAddr	UINT	This parameter transfers the start address of the Modbus register to the Modbus block.
iNumOfData	INT	This parameter transfers the amount of data to be read to the Modbus block.

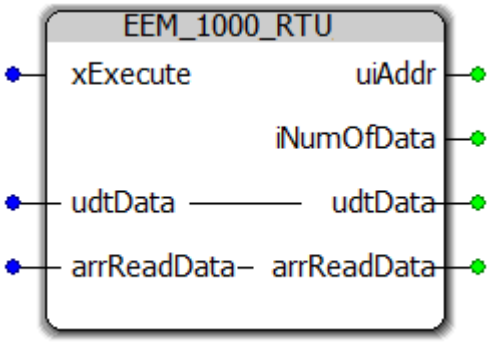
6.3.4 Inout parameters

Name	Type	Description
udtData	EEM_UDT_0741	Input structure for standardized values in the table.
arrReadData	arrModbus2_w_1_125	Structure of the Modbus data.

6.4 EEM_1000_RTU

Function block EEM_1000_RTU is used to read instantaneous values of register addresses 16#1000 – 16#103A. The values are written into a data structure in a standardized format.

6.4.1 Function block call



6.4.2 Input parameters

Name	Type	Description
xExecute	BOOL	By setting a positive edge, the data is loaded from the Modbus structure into the block, where it is transferred to the input and output structure of the block in a standardized format.

6.4.3 Output parameters

Name	Type	Description
uiAddr	UINT	This parameter transfers the start address of the Modbus register to the Modbus block.
iNumOfData	INT	This parameter transfers the amount of data to be read to the Modbus block.

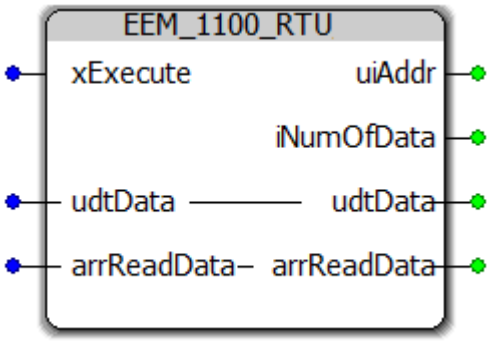
6.4.4 Inout parameters

Name	Type	Description
udtData	EEM_UDT_1000	Input structure for standardized values in the table.
arrReadData	arrModbus2_w_1_125	Structure of the Modbus data.

6.5 EEM_1100_RTU

Function block EEM_1100_RTU is used to read meter values of register addresses 16#1100 – 16#114E. The values are written into a data structure in a standardized format.

6.5.1 Function block call



6.5.2 Input parameters

Name	Type	Description
xExecute	BOOL	By setting a positive edge, the data is loaded from the Modbus structure into the block, where it is transferred to the input and output structure of the block in a standardized format.

6.5.3 Output parameters

Name	Type	Description
uiAddr	UINT	This parameter transfers the start address of the Modbus register to the Modbus block.
iNumOfData	INT	This parameter transfers the amount of data to be read to the Modbus block.

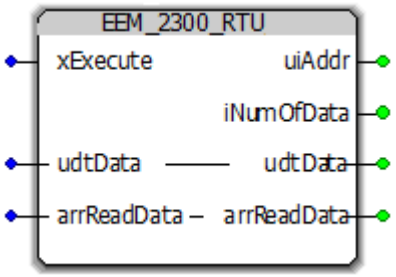
6.5.4 Inout parameters

Name	Type	Description
udtData	EEM_UDT_1000	Input structure for standardized values in the table.
arrReadData	arrModbus2_w_1_125	Structure of the Modbus data.

6.6 EEM_2300_RTU

Function block EEM_2300_RTU is used for extern modules/extern gateway Proficloud using register addresses 16#2300 – 16#2322. The values are written into a data structure in a standardized format.

6.6.1 Function block call



6.6.2 Input parameters

Name	Type	Description
xExecute	BOOL	By setting a positive edge, the data is loaded from the Modbus structure into the block, where it is transferred to the input and output structure of the block in a standardized format.

6.6.3 Output parameters

Name	Type	Description
uiAddr	UINT	This parameter transfers the start address of the Modbus register to the Modbus block.
iNumOfData	INT	This parameter transfers the amount of data to be read to the Modbus block.

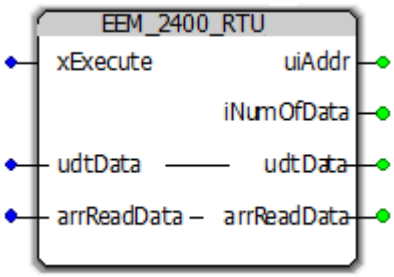
6.6.4 Inout parameters

Name	Type	Description
udtData	EEM_UDT_2300	Input structure for standardized values in the table.
arrReadData	arrModbus2_w_1_125	Structure of the Modbus data.

6.7 EEM_2400_RTU

Function block EEM_2400_RTU is used for extern modules / extern Gateway Profinet / Ethernet/IP using register addresses 16#2400 – 16#240B. The values are written into a data structure in a standardized format.

6.7.1 Function block call



6.7.2 Input parameters

Name	Type	Description
xExecute	BOOL	By setting a positive edge, the data is loaded from the Modbus structure into the block, where it is transferred to the input and output structure of the block in a standardized format.

6.7.3 Output parameters

Name	Type	Description
uiAddr	UINT	This parameter transfers the start address of the Modbus register to the Modbus block.
iNumOfData	INT	This parameter transfers the amount of data to be read to the Modbus block.

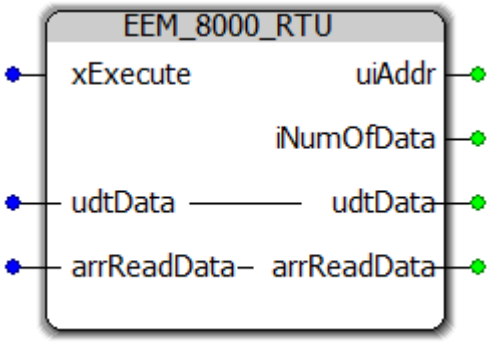
6.7.4 Inout parameters

Name	Type	Description
udtData	EEM_UDT_2400	Input structure for standardized values in the table.
arrReadData	arrModbus2_w_1_125	Structure of the Modbus data.

6.8 EEM_8000_RTU

Function block EEM_8000_RTU is used to read instantaneous values of register addresses 16#8000 – 16#8057. The values are written into a data structure in a standardized format.

6.8.1 Function block call



6.8.2 Input parameters

Name	Type	Description
xExecute	BOOL	By setting a positive edge, the data is loaded from the Modbus structure into the block, where it is transferred to the input and output structure of the block in a standardized format.

6.8.3 Output parameters

Name	Type	Description
uiAddr	UINT	This parameter transfers the start address of the Modbus register to the Modbus block.
iNumOfData	INT	This parameter transfers the amount of data to be read to the Modbus block.

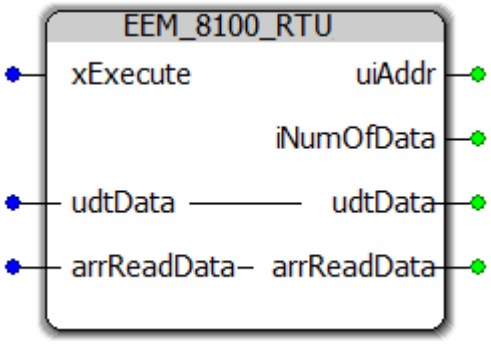
6.8.4 Inout parameters

Name	Type	Description
udtData	EEM_UDT_8000	Input structure for standardized values in the table.
arrReadData	arrModbus2_w_1_125	Structure of the Modbus data.

6.9 EEM_8100_RTU

Function block EEM_8100_RTU is used to read energy meter values of register addresses 16#8100 – 16#8116. The values are written into a data structure in a standardized format.

6.9.1 Function block call



6.9.2 Input parameters

Name	Type	Description
xExecute	BOOL	By setting a positive edge, the data is loaded from the Modbus structure into the block, where it is transferred to the input and output structure of the block in a standardized format.

6.9.3 Output parameters

Name	Type	Description
uiAddr	UINT	This parameter transfers the start address of the Modbus register to the Modbus block.
iNumOfData	INT	This parameter transfers the amount of data to be read to the Modbus block.

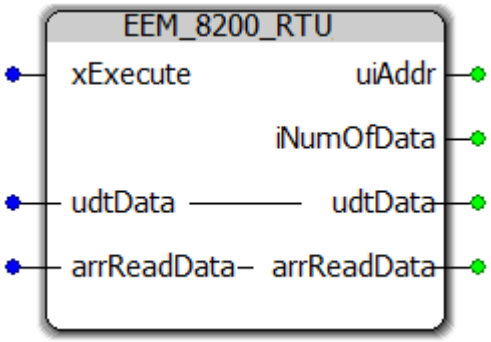
6.9.4 Inout parameters

Name	Type	Description
udtData	EEM_UDT_8100	Input structure for standardized values in the table.
arrReadData	arrModbus2_w_1_125	Structure of the Modbus data.

6.10 EEM_8200_RTU

Function block EEM_8200_RTU is used to read tariff meter values of register addresses 16#8200 – 16#822F. The values are written into a data structure in a standardized format.

6.10.1 Function block call



6.10.2 Input parameters

Name	Type	Description
xExecute	BOOL	By setting a positive edge, the data is loaded from the Modbus structure into the block, where it is transferred to the input and output structure of the block in a standardized format.

6.10.3 Output parameters

Name	Type	Description
uiAddr	UINT	This parameter transfers the start address of the Modbus register to the Modbus block.
iNumOfData	INT	This parameter transfers the amount of data to be read to the Modbus block.

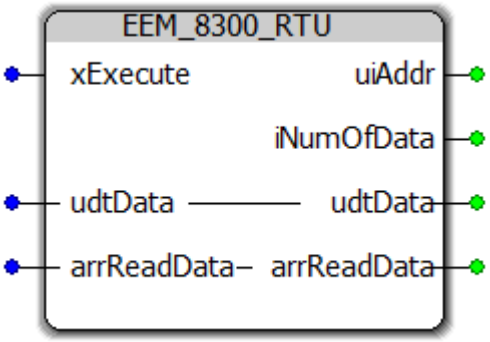
6.10.4 Inout parameters

Name	Type	Description
udtData	EEM_UDT_8200	Input structure for standardized values in the table.
arrReadData	arrModbus2_w_1_125	Structure of the Modbus data.

6.11 EEM_8300_RTU

Function block EEM_8300_RTU is used to read operating hours of register addresses 16#8300 – 16#8302. The values are written into a data structure in a standardized format.

6.11.1 Function block call



6.11.2 Input parameters

Name	Type	Description
xExecute	BOOL	By setting a positive edge, the data is loaded from the Modbus structure into the block, where it is transferred to the input and output structure of the block in a standardized format.

6.11.3 Output parameters

Name	Type	Description
uiAddr	UINT	This parameter transfers the start address of the Modbus register to the Modbus block.
iNumOfData	INT	This parameter transfers the amount of data to be read to the Modbus block.

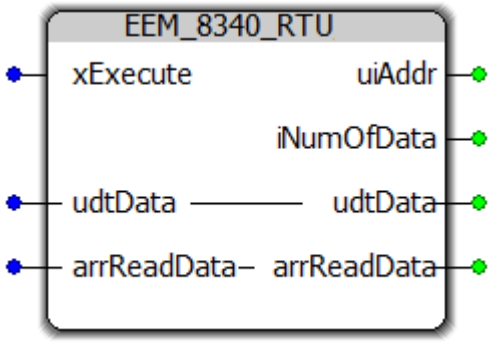
6.11.4 Inout parameters

Name	Type	Description
udtData	EEM_UDT_8300	Input structure for standardized values in the table.
arrReadData	arrModbus2_w_1_125	Structure of the Modbus data.

6.12 EEM_8340_RTU

Function block EEM_8340_RTU is used to read the state of impulse counter of digital input of register address 16#8340. The value is written into a data structure in a standardized format.

6.12.1 Function block call



6.12.2 Input parameters

Name	Type	Description
xExecute	BOOL	By setting a positive edge, the data is loaded from the Modbus structure into the block, where it is transferred to the input and output structure of the block in a standardized format.

6.12.3 Output parameters

Name	Type	Description
uiAddr	UINT	This parameter transfers the start address of the Modbus register to the Modbus block.
iNumOfData	INT	This parameter transfers the amount of data to be read to the Modbus block.

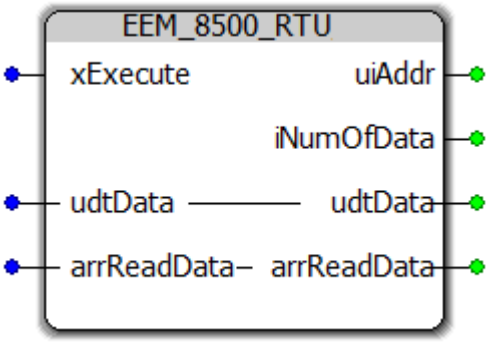
6.12.4 Inout parameters

Name	Type	Description
udtData	EEM_UDT_8340	Input structure for standardized values in the table.
arrReadData	arrModbus2_w_1_125	Structure of the Modbus data.

6.13 EEM_8500_RTU

Function block EEM_8500_RTU is used to read statistics (prediction) of register addresses 16#8500 - 16#850E. The values are written into a data structure in a standardized format.

6.13.1 Function block call



6.13.2 Input parameters

Name	Type	Description
xExecute	BOOL	By setting a positive edge, the data is loaded from the Modbus structure into the block, where it is transferred to the input and output structure of the block in a standardized format.

6.13.3 Output parameters

Name	Type	Description
uiAddr	UINT	This parameter transfers the start address of the Modbus register to the Modbus block.
iNumOfData	INT	This parameter transfers the amount of data to be read to the Modbus block.

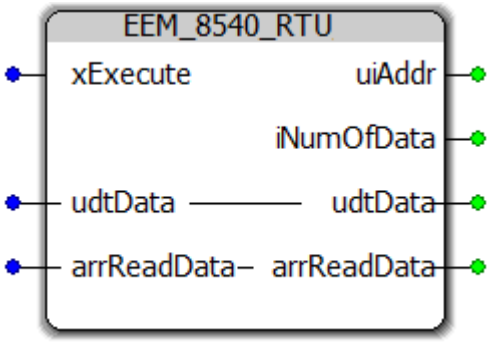
6.13.4 Inout parameters

Name	Type	Description
udtData	EEM_UDT_8500	Input structure for standardized values in the table.
arrReadData	arrModbus2_w_1_125	Structure of the Modbus data.

6.14 EEM_8540_RTU

Function block EEM_8540_RTU is used to read statistics (mean values) of register addresses 16#8540 - 16#855E. The values are written into a data structure in a standardized format.

6.14.1 Function block call



6.14.2 Input parameters

Name	Type	Description
xExecute	BOOL	By setting a positive edge, the data is loaded from the Modbus structure into the block, where it is transferred to the input and output structure of the block in a standardized format.

6.14.3 Output parameters

Name	Type	Description
uiAddr	UINT	This parameter transfers the start address of the Modbus register to the Modbus block.
iNumOfData	INT	This parameter transfers the amount of data to be read to the Modbus block.

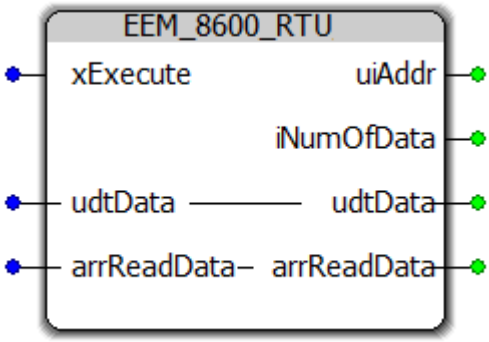
6.14.4 Inout parameters

Name	Type	Description
udtData	EEM_UDT_8540	Input structure for standardized values in the table.
arrReadData	arrModbus2_w_1_125	Structure of the Modbus data.

6.15 EEM_8600_RTU

Function block EEM_8600_RTU is used to read min / max values for mean values of register addresses 16#8600 - 16#863E. The values are written into a data structure in a standardized format.

6.15.1 Function block call



6.15.2 Input parameters

Name	Type	Description
xExecute	BOOL	By setting a positive edge, the data is loaded from the Modbus structure into the block, where it is transferred to the input and output structure of the block in a standardized format.

6.15.3 Output parameters

Name	Type	Description
uiAddr	UINT	This parameter transfers the start address of the Modbus register to the Modbus block.
iNumOfData	INT	This parameter transfers the amount of data to be read to the Modbus block.

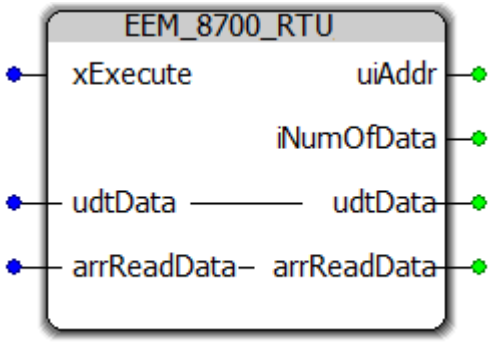
6.15.4 Inout parameters

Name	Type	Description
udtData	EEM_UDT_8600	Input structure for standardized values in the table.
arrReadData	arrModbus2_w_1_125	Structure of the Modbus data.

6.16 EEM_8700_RTU

Function block EEM_8700_RTU is used to read min / max values for instantaneous values of register addresses 16#8700 - 16#873E. The values are written into a data structure in a standardized format.

6.16.1 Function block call



6.16.2 Input parameters

Name	Type	Description
xExecute	BOOL	By setting a positive edge, the data is loaded from the Modbus structure into the block, where it is transferred to the input and output structure of the block in a standardized format.

6.16.3 Output parameters

Name	Type	Description
uiAddr	UINT	This parameter transfers the start address of the Modbus register to the Modbus block.
iNumOfData	INT	This parameter transfers the amount of data to be read to the Modbus block.

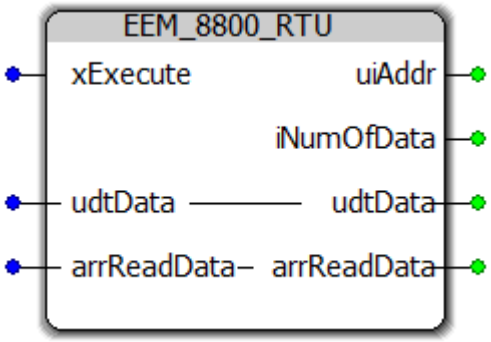
6.16.4 Inout parameters

Name	Type	Description
udtData	EEM_UDT_8700	Input structure for standardized values in the table.
arrReadData	arrModbus2_w_1_125	Structure of the Modbus data.

6.17 EEM_8800_RTU

Function block EEM_8800_RTU is used to read total harmonic distortion of register addresses 16#8800 - 16#8809. The values are written into a data structure in a standardized format.

6.17.1 Function block call



6.17.2 Input parameters

Name	Type	Description
xExecute	BOOL	By setting a positive edge, the data is loaded from the Modbus structure into the block, where it is transferred to the input and output structure of the block in a standardized format.

6.17.3 Output parameters

Name	Type	Description
uiAddr	UINT	This parameter transfers the start address of the Modbus register to the Modbus block.
iNumOfData	INT	This parameter transfers the amount of data to be read to the Modbus block.

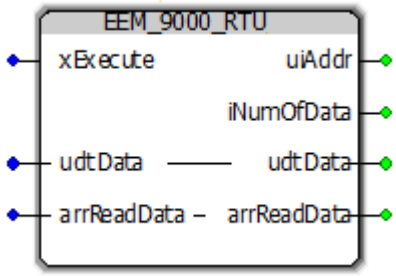
6.17.4 Inout parameters

Name	Type	Description
udtData	EEM_UDT_8800	Input structure for standardized values in the table.
arrReadData	arrModbus2_w_1_125	Structure of the Modbus data.

6.18 EEM_9000_RTU

Function block EEM_9000_RTU is used to read out voltage dips of register addresses 16#9000 - 16#9045. The values are written into a data structure in a standardized format.

6.18.1 Function block call



6.18.2 Input parameters

Name	Type	Description
xExecute	BOOL	By setting a positive edge, the data is loaded from the Modbus structure into the block, where it is transferred to the input and output structure of the block in a standardized format.

6.18.3 Output parameters

Name	Type	Description
uiAddr	UINT	This parameter transfers the start address of the Modbus register to the Modbus block.
iNumOfData	INT	This parameter transfers the amount of data to be read to the Modbus block.

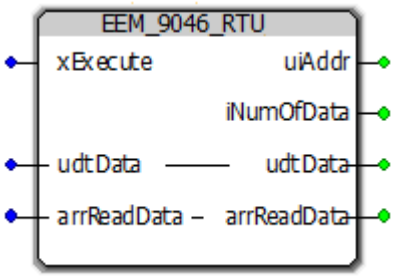
6.18.4 Inout parameters

Name	Type	Description
udtData	EEM_UDT_9000	Input structure for standardized values in the table.
arrReadData	arrModbus2_w_1_125	Structure of the Modbus data.

6.19 EEM_9046_RTU

Function block EEM_9046_RTU is used to read out voltage dips of register addresses 16#9046 - 16#908B. The values are written into a data structure in a standardized format.

6.19.1 Function block call



6.19.2 Input parameters

Name	Type	Description
xExecute	BOOL	By setting a positive edge, the data is loaded from the Modbus structure into the block, where it is transferred to the input and output structure of the block in a standardized format.

6.19.3 Output parameters

Name	Type	Description
uiAddr	UINT	This parameter transfers the start address of the Modbus register to the Modbus block.
iNumOfData	INT	This parameter transfers the amount of data to be read to the Modbus block.

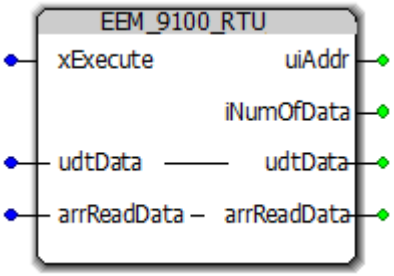
6.19.4 Inout parameters

Name	Type	Description
udtData	EEM_UDT_9046	Input structure for standardized values in the table.
arrReadData	arrModbus2_w_1_125	Structure of the Modbus data.

6.20 EEM_9100_RTU

Function block EEM_9100_RTU is used to read out voltage swells of register addresses 16#9100 - 16#9145. The values are written into a data structure in a standardized format.

6.20.1 Function block call



6.20.2 Input parameters

Name	Type	Description
xExecute	BOOL	By setting a positive edge, the data is loaded from the Modbus structure into the block, where it is transferred to the input and output structure of the block in a standardized format.

6.20.3 Output parameters

Name	Type	Description
uiAddr	UINT	This parameter transfers the start address of the Modbus register to the Modbus block.
iNumOfData	INT	This parameter transfers the amount of data to be read to the Modbus block.

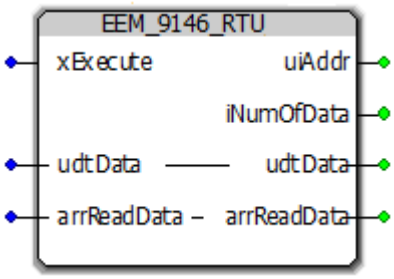
6.20.4 Inout parameters

Name	Type	Description
udtData	EEM_UDT_9100	Input structure for standardized values in the table.
arrReadData	arrModbus2_w_1_125	Structure of the Modbus data.

6.21 EEM_9146_RTU

Function block EEM_9146_RTU is used to read out voltage swells of register addresses 16#9146 - 16#918B. The values are written into a data structure in a standardized format.

6.21.1 Function block call



6.21.2 Input parameters

Name	Type	Description
xExecute	BOOL	By setting a positive edge, the data is loaded from the Modbus structure into the block, where it is transferred to the input and output structure of the block in a standardized format.

6.21.3 Output parameters

Name	Type	Description
uiAddr	UINT	This parameter transfers the start address of the Modbus register to the Modbus block.
iNumOfData	INT	This parameter transfers the amount of data to be read to the Modbus block.

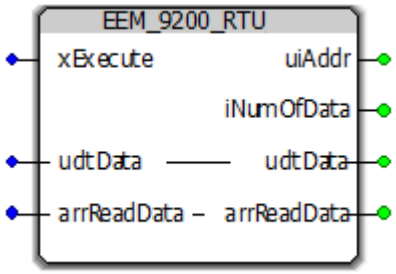
6.21.4 Inout parameters

Name	Type	Description
udtData	EEM_UDT_9146	Input structure for standardized values in the table.
arrReadData	arrModbus2_w_1_125	Structure of the Modbus data.

6.22 EEM_9200_RTU

Function block EEM_9200_RTU is used to read out voltage interrupts of register addresses 16#9200 - 16#923B. The values are written into a data structure in a standardized format.

6.22.1 Function block call



6.22.2 Input parameters

Name	Type	Description
xExecute	BOOL	By setting a positive edge, the data is loaded from the Modbus structure into the block, where it is transferred to the input and output structure of the block in a standardized format.

6.22.3 Output parameters

Name	Type	Description
uiAddr	UINT	This parameter transfers the start address of the Modbus register to the Modbus block.
iNumOfData	INT	This parameter transfers the amount of data to be read to the Modbus block.

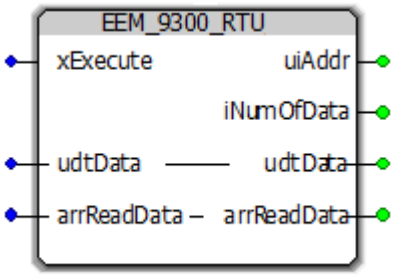
6.22.4 Inout parameters

Name	Type	Description
udtData	EEM_UDT_9200	Input structure for standardized values in the table.
arrReadData	arrModbus2_w_1_125	Structure of the Modbus data.

6.23 EEM_9300_RTU

Function block EEM_9300_RTU is used to read energy meters (not resetable) of register addresses 16#9300 - 16#9356. The values are written into a data structure in a standardized format.

6.23.1 Function block call



6.23.2 Input parameters

Name	Type	Description
xExecute	BOOL	By setting a positive edge, the data is loaded from the Modbus structure into the block, where it is transferred to the input and output structure of the block in a standardized format.

6.23.3 Output parameters

Name	Type	Description
uiAddr	UINT	This parameter transfers the start address of the Modbus register to the Modbus block.
iNumOfData	INT	This parameter transfers the amount of data to be read to the Modbus block.

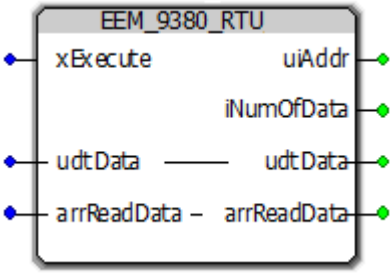
6.23.4 Inout parameters

Name	Type	Description
udtData	EEM_UDT_9300	Input structure for standardized values in the table.
arrReadData	arrModbus2_w_1_125	Structure of the Modbus data.

6.24 EEM_9380_RTU

Function block EEM_9380_RTU is used to read energy meters (resetable) of register addresses 16#9380 - 16#93D6. The values are written into a data structure in a standardized format.

6.24.1 Function block call



6.24.2 Input parameters

Name	Type	Description
xExecute	BOOL	By setting a positive edge, the data is loaded from the Modbus structure into the block, where it is transferred to the input and output structure of the block in a standardized format.

6.24.3 Output parameters

Name	Type	Description
uiAddr	UINT	This parameter transfers the start address of the Modbus register to the Modbus block.
iNumOfData	INT	This parameter transfers the amount of data to be read to the Modbus block.

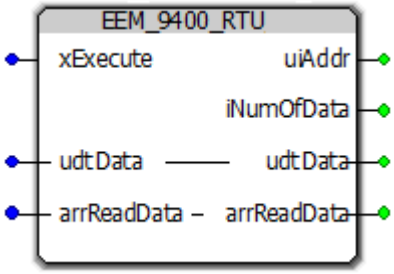
6.24.4 Inout parameters

Name	Type	Description
udtData	EEM_UDT_9380	Input structure for standardized values in the table.
arrReadData	arrModbus2_w_1_125	Structure of the Modbus data.

6.25 EEM_9400_RTU

Function block EEM_9400_RTU is used to read energy meter tariff 1 of register addresses 16#9400 - 16#9456. The values are written into a data structure in a standardized format.

6.25.1 Function block call



6.25.2 Input parameters

Name	Type	Description
xExecute	BOOL	By setting a positive edge, the data is loaded from the Modbus structure into the block, where it is transferred to the input and output structure of the block in a standardized format.

6.25.3 Output parameters

Name	Type	Description
uiAddr	UINT	This parameter transfers the start address of the Modbus register to the Modbus block.
iNumOfData	INT	This parameter transfers the amount of data to be read to the Modbus block.

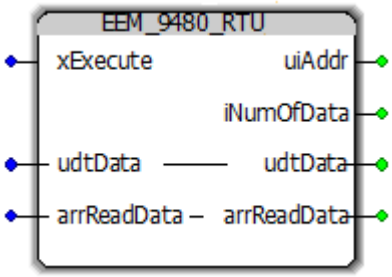
6.25.4 Inout parameters

Name	Type	Description
udtData	EEM_UDT_9400	Input structure for standardized values in the table.
arrReadData	arrModbus2_w_1_125	Structure of the Modbus data.

6.26 EEM_9400_RTU

Function block EEM_9480_RTU is used to read energy meter tariff 2 of register addresses 16#9480 - 16#94D6. The values are written into a data structure in a standardized format.

6.26.1 Function block call



6.26.2 Input parameters

Name	Type	Description
xExecute	BOOL	By setting a positive edge, the data is loaded from the Modbus structure into the block, where it is transferred to the input and output structure of the block in a standardized format.

6.26.3 Output parameters

Name	Type	Description
uiAddr	UINT	This parameter transfers the start address of the Modbus register to the Modbus block.
iNumOfData	INT	This parameter transfers the amount of data to be read to the Modbus block.

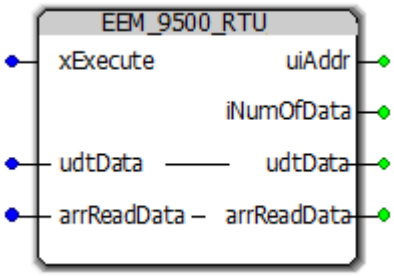
6.26.4 Inout parameters

Name	Type	Description
udtData	EEM_UDT_9480	Input structure for standardized values in the table.
arrReadData	arrModbus2_w_1_125	Structure of the Modbus data.

6.27 EEM_9500_RTU

Function block EEM_9500_RTU is used to read energy meter tariff 3 of register addresses 16#9500 - 16#9556. The values are written into a data structure in a standardized format.

6.27.1 Function block call



6.27.2 Input parameters

Name	Type	Description
xExecute	BOOL	By setting a positive edge, the data is loaded from the Modbus structure into the block, where it is transferred to the input and output structure of the block in a standardized format.

6.27.3 Output parameters

Name	Type	Description
uiAddr	UINT	This parameter transfers the start address of the Modbus register to the Modbus block.
iNumOfData	INT	This parameter transfers the amount of data to be read to the Modbus block.

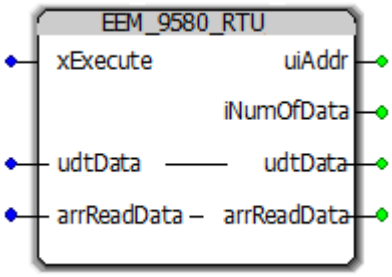
6.27.4 Inout parameters

Name	Type	Description
udtData	EEM_UDT_9500	Input structure for standardized values in the table.
arrReadData	arrModbus2_w_1_125	Structure of the Modbus data.

6.28 EEM_9580_RTU

Function block EEM_9580_RTU is used to read energy meter tariff 4 of register addresses 16#9580 - 16#95D6. The values are written into a data structure in a standardized format.

6.28.1 Function block call



6.28.2 Input parameters

Name	Type	Description
xExecute	BOOL	By setting a positive edge, the data is loaded from the Modbus structure into the block, where it is transferred to the input and output structure of the block in a standardized format.

6.28.3 Output parameters

Name	Type	Description
uiAddr	UINT	This parameter transfers the start address of the Modbus register to the Modbus block.
iNumOfData	INT	This parameter transfers the amount of data to be read to the Modbus block.

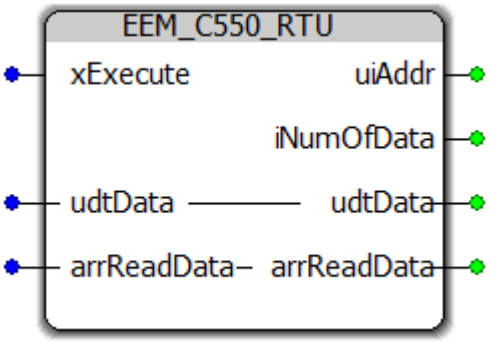
6.28.4 Inout parameters

Name	Type	Description
udtData	EEM_UDT_9580	Input structure for standardized values in the table.
arrReadData	arrModbus2_w_1_125	Structure of the Modbus data.

6.29 EEM_C550_RTU

The function block EEM_C550_RTU is used to read instanteous values of register addresses 16#C550 – 16#C58C. The values are written into a data structure in a standardized format.

6.29.1 Function block call



6.29.2 Input parameters

Name	Type	Description
xExecute	BOOL	By setting a positive edge, the data is loaded from the Modbus structure into the block, where it is transferred to the input and output structure of the block in a standardized format.

6.29.3 Output parameters

Name	Type	Description
uiAddr	UINT	This parameter transfers the start address of the Modbus register to the Modbus block.
iNumOfData	INT	This parameter transfers the amount of data to be read to the Modbus block.

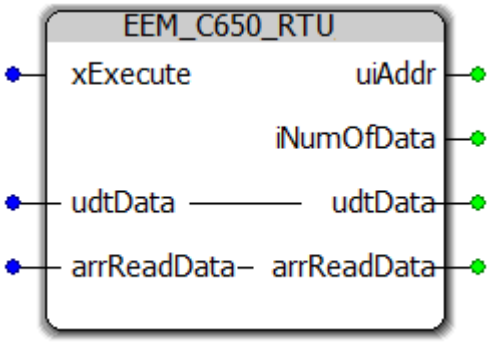
6.29.4 Inout parameters

Name	Type	Description
udtData	EEM_UDT_C550	Input structure for standardized values in the table.
arrReadData	arrModbus2_w_1_125	Structure of the Modbus data.

6.30 EEM_C650_RTU

Block EEM_C650_RTU is used to read energy values of register addresses 16#C650 – 16#C664. The values are written into a data structure in a standardized format.

6.30.1 Function block call



6.30.2 Input parameters

Name	Type	Description
xExecute	BOOL	By setting a positive edge, the data is loaded from the Modbus structure into the block, where it is transferred to the input and output structure of the block in a standardized format.

6.30.3 Output parameters

Name	Type	Description
uiAddr	UINT	This parameter transfers the start address of the Modbus register to the Modbus block.
iNumOfData	INT	This parameter transfers the amount of data to be read to the Modbus block.

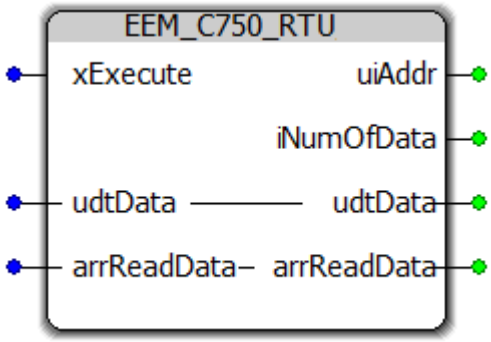
6.30.4 Inout parameters

Name	Type	Description
udtData	EEM_UDT_C650	Input structure for standardized values in the table.
arrReadData	arrModbus2_w_1_125	Structure of the Modbus data.

6.31 EEM_C750_RTU

The function block EEM_C750_RTU is used to read the average values based on current and voltage transducer via the set measurement duration (under consideration of transmission ratios) of register addresses 16#C750 – 16#C795. The values are written into a data structure in a standardized format.

6.31.1 Function block call



6.31.2 Input parameters

Name	Type	Description
xExecute	BOOL	By setting a positive edge, the data is loaded from the Modbus structure into the block, where it is transferred to the input and output structure of the block in a standardized format.

6.31.3 Output parameters

Name	Type	Description
uiAddr	UINT	This parameter transfers the start address of the Modbus register to the Modbus block.
iNumOfData	INT	This parameter transfers the amount of data to be read to the Modbus block.

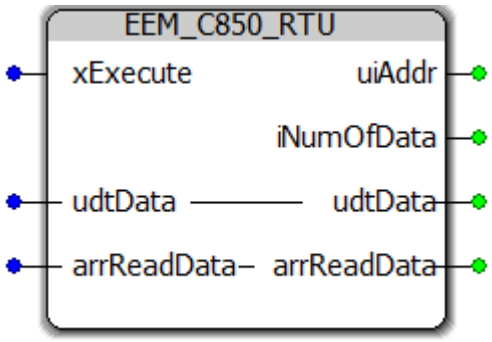
6.31.4 Inout parameters

Name	Type	Description
udtData	EEM_UDT_C750	Input structure for standardized values in the table.
arrReadData	arrModbus2_w_1_125	Structure of the Modbus data.

6.32 EEM_C850_RTU

The function block EEM_C850_RTU is used to read current main measured values not based on current transformer (without taking into consideration transformation ratios) of register addresses 16#C850 – 16#C872. The values are written into a data structure in a standardized format.

6.32.1 Function block call



6.32.2 Input parameters

Name	Type	Description
xExecute	BOOL	By setting a positive edge, the data is loaded from the Modbus structure into the block, where it is transferred to the input and output structure of the block in a standardized format.

6.32.3 Output parameters

Name	Type	Description
uiAddr	UINT	This parameter transfers the start address of the Modbus register to the Modbus block.
iNumOfData	INT	This parameter transfers the amount of data to be read to the Modbus block.

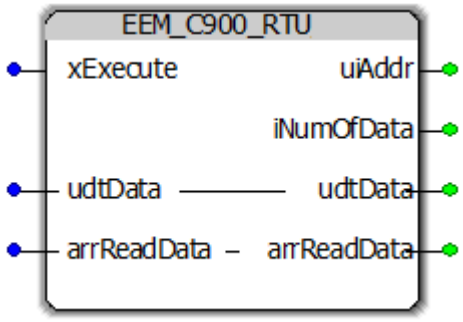
6.32.4 Inout parameters

Name	Type	Description
udtData	EEM_UDT_C850	Input structure for standardized values in the table.
arrReadData	arrModbus2_w_1_125	Structure of the Modbus data.

6.33 EEM_C900_RTU

Function block EEM_C900_RTU is used to read temperature inputs of register addresses 16#C900 – 16#C907. The values are written into a data structure in a standardized format.

6.33.1 Function block call



6.33.2 Input parameters

Name	Type	Description
xExecute	BOOL	By setting a positive edge, the data is loaded from the Modbus structure into the block, where it is transferred to the input and output structure of the block in a standardized format.

6.33.3 Output parameters

Name	Type	Description
uiAddr	UINT	This parameter transfers the start address of the Modbus register to the Modbus block.
iNumOfData	INT	This parameter transfers the amount of data to be read to the Modbus block.

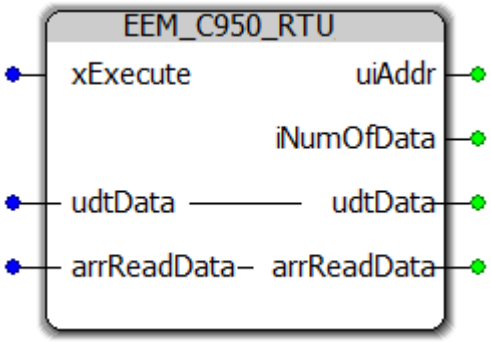
6.33.4 Inout parameters

Name	Type	Description
udtData	EEM_UDT_C900	Input structure for standardized values in the table.
arrReadData	arrModbus2_w_1_125	Structure of the Modbus data.

6.34 EEM_C950_RTU

Function block EEM_C950_RTU is used to read current harmonic distortion and harmonics of register addresses 16#C950 – 16#C959. The values are written into a data structure in a standardized format.

6.34.1 Function block call



6.34.2 Input parameters

Name	Type	Description
xExecute	BOOL	By setting a positive edge, the data is loaded from the Modbus structure into the block, where it is transferred to the input and output structure of the block in a standardized format.

6.34.3 Output parameters

Name	Type	Description
uiAddr	UINT	This parameter transfers the start address of the Modbus register to the Modbus block.
iNumOfData	INT	This parameter transfers the amount of data to be read to the Modbus block.

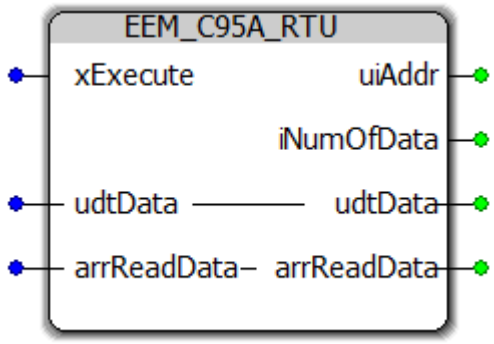
6.34.4 Inout parameters

Name	Type	Description
udtData	EEM_UDT_C950	Input structure for standardized values in the table.
arrReadData	arrModbus2_w_1_125	Structure of the Modbus data.

6.35 EEM_C95A_RTU

Function block EEM_C95A_RTU is used to read individual harmonic content for currents of register addresses 16#C95A – 16#CA92. The values are written into a data structure in a standardized format.

6.35.1 Function block call



6.35.2 Input parameters

Name	Type	Description
xExecute	BOOL	By setting a positive edge, the data is loaded from the Modbus structure into the block, where it is transferred to the input and output structure of the block in a standardized format.

6.35.3 Output parameters

Name	Type	Description
uiAddr	UINT	This parameter transfers the start address of the Modbus register to the Modbus block.
iNumOfData	INT	This parameter transfers the amount of data to be read to the Modbus block.

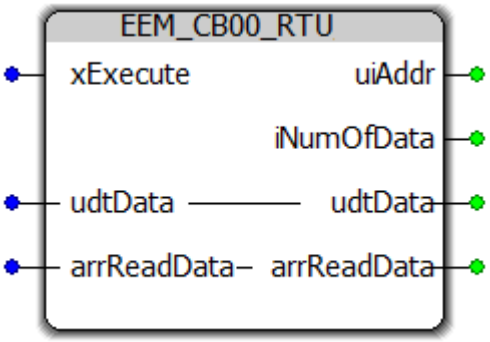
6.35.4 Inout parameters

Name	Type	Description
udtData	EEM_UDT_C95A	Input structure for standardized values in the table.
arrReadData	arrModbus2_w_1_125	Structure of the Modbus data.

6.36 EEM_CB00_RTU

Function block EEM_CB00_RTU is used to read the status of inputs/outputs for register addresses 16#CB00 – 16#CB03. The values are written into a data structure in a standardized format.

6.36.1 Function block call



6.36.2 Input parameters

Name	Type	Description
xExecute	BOOL	By setting a positive edge, the data is loaded from the Modbus structure into the block, where it is transferred to the input and output structure of the block in a standardized format.

6.36.3 Output parameters

Name	Type	Description
uiAddr	UINT	This parameter transfers the start address of the Modbus register to the Modbus block.
iNumOfData	INT	This parameter transfers the amount of data to be read to the Modbus block.

6.36.4 Inout parameters

Name	Type	Description
udtData	EEM_UDT_CB00	Input structure for standardized values in the table.
arrReadData	arrModbus2_w_1_125	Structure of the Modbus data.

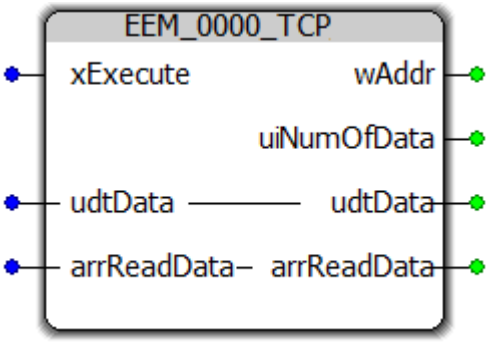
7 EEM_*_TCP

The EEM_*_TCP function blocks are used for communication with Electric Energy Meters (EEM). Communication takes place via Modbus TCP.

7.1 EEM_0000_TCP

Function block EEM_0000_TCP is used to read instantaneous values of register addresses 16#0000 – 16#0052. The values are written into a data structure in a standardized format.

7.1.1 Function block call



7.1.2 Input parameters

Name	Type	Description
xExecute	BOOL	By setting a positive edge, the data is loaded from the Modbus structure into the block, where it is transferred to the input and output structure of the block in a standardized format.

7.1.3 Output parameters

Name	Type	Description
uiAddr	UINT	This parameter transfers the start address of the Modbus register to the Modbus block.
iNumOfData	INT	This parameter transfers the amount of data to be read to the Modbus block.

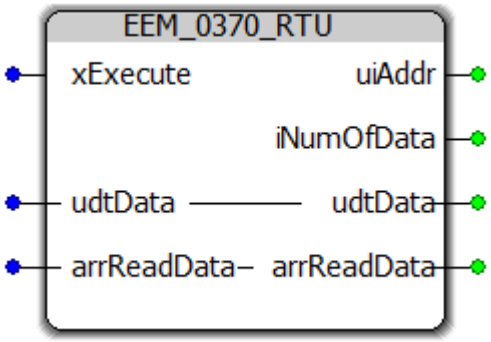
7.1.4 Inout parameters

Name	Type	Description
udtData	EEM_UDT_0000	Input structure for standardized values in the table.
arrReadData	MB_TCP_ARR_W_1_125	Structure of the Modbus data.

7.2 EEM_0370_TCP

Function block EEM_0370_TCP is used to read the current average current/voltage value based on current and voltage transducer (under consideration of transmission ratios) of register addresses 16#0370 – 16#0374. The values are written into a data structure in a standardized format.

7.2.1 Function block call



7.2.2 Input parameters

Name	Type	Description
xExecute	BOOL	By setting a positive edge, the data is loaded from the Modbus structure into the block, where it is transferred to the input and output structure of the block in a standardized format.

7.2.3 Output parameters

Name	Type	Description
uiAddr	UINT	This parameter transfers the start address of the Modbus register to the Modbus block.
iNumOfData	INT	This parameter transfers the amount of data to be read to the Modbus block.

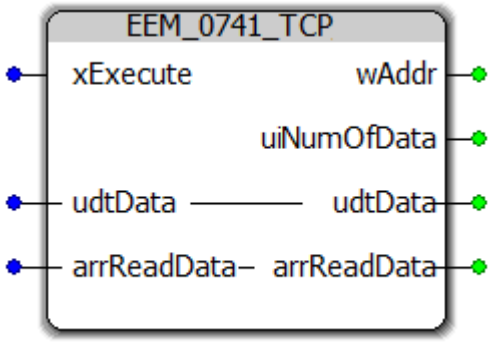
7.2.4 Inout parameters

Name	Type	Description
udtData	EEM_UDT_0370	Input structure for standardized values in the table.
arrReadData	MB_TCP_ARR_W_1_125	Structure of the Modbus data.

7.3 EEM_0741_TCP

Function block EEM_0741_TCP is used to read current average current/voltage value not based on current and voltage transducer (without consideration of transmission ratios) of register addresses 16#0741 – 16#0743. The values are written into a data structure in a standardized format.

7.3.1 Function block call



7.3.2 Input parameters

Name	Type	Description
xExecute	BOOL	By setting a positive edge, the data is loaded from the Modbus structure into the block, where it is transferred to the input and output structure of the block in a standardized format.

7.3.3 Output parameters

Name	Type	Description
uiAddr	UINT	This parameter transfers the start address of the Modbus register to the Modbus block.
iNumOfData	INT	This parameter transfers the amount of data to be read to the Modbus block.

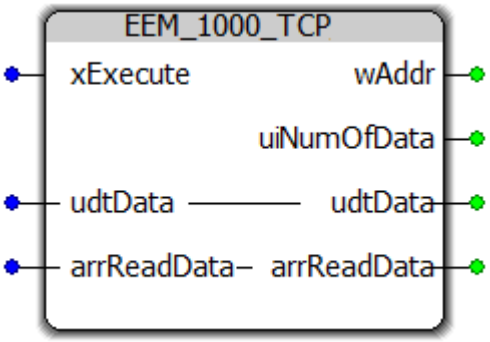
7.3.4 Inout parameters

Name	Type	Description
udtData	EEM_UDT_0741	Input structure for standardized values in the table.
arrReadData	MB_TCP_ARR_W_1_125	Structure of the Modbus data.

7.4 EEM_1000_TCP

Function block EEM_1000_TCP is used to read instantaneous values of register addresses 16#1000 – 16#103A. The values are written into a data structure in a standardized format.

7.4.1 Function block call



7.4.2 Input parameters

Name	Type	Description
xExecute	BOOL	By setting a positive edge, the data is loaded from the Modbus structure into the block, where it is transferred to the input and output structure of the block in a standardized format.

7.4.3 Output parameters

Name	Type	Description
uiAddr	UINT	This parameter transfers the start address of the Modbus register to the Modbus block.
iNumOfData	INT	This parameter transfers the amount of data to be read to the Modbus block.

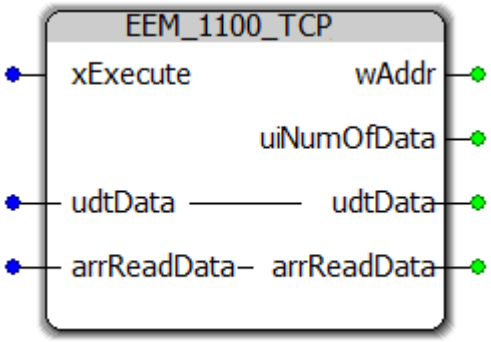
7.4.4 Inout parameters

Name	Type	Description
udtData	EEM_UDT_1000	Input structure for standardized values in the table.
arrReadData	MB_TCP_ARR_W_1_125	Structure of the Modbus data.

7.5 EEM_1100_TCP

Function block EEM_1100_TCP is used to read meter values of register addresses 16#1100 – 16#114E. The values are written into a data structure in a standardized format.

7.5.1 Function block call



7.5.2 Input parameters

Name	Type	Description
xExecute	BOOL	By setting a positive edge, the data is loaded from the Modbus structure into the block, where it is transferred to the input and output structure of the block in a standardized format.

7.5.3 Output parameters

Name	Type	Description
uiAddr	UINT	This parameter transfers the start address of the Modbus register to the Modbus block.
iNumOfData	INT	This parameter transfers the amount of data to be read to the Modbus block.

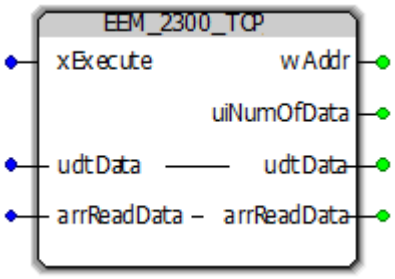
7.5.4 Inout parameters

Name	Type	Description
udtData	EEM_UDT_1000	Input structure for standardized values in the table.
arrReadData	MB_TCP_ARR_W_1_125	Structure of the Modbus data.

7.6 EEM_2300_TCP

Function block EEM_2300_TCP is used for extern modules/extern gateway Proficloud using register addresses 16#2300 – 16#2322. The values are written into a data structure in a standardized format.

7.6.1 Function block call



7.6.2 Input parameters

Name	Type	Description
xExecute	BOOL	By setting a positive edge, the data is loaded from the Modbus structure into the block, where it is transferred to the input and output structure of the block in a standardized format.

7.6.3 Output parameters

Name	Type	Description
uiAddr	UINT	This parameter transfers the start address of the Modbus register to the Modbus block.
iNumOfData	INT	This parameter transfers the amount of data to be read to the Modbus block.

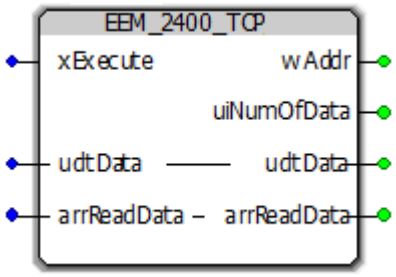
7.6.4 Inout parameters

Name	Type	Description
udtData	EEM_UDT_2300	Input structure for standardized values in the table.
arrReadData	MB_TCP_ARR_W_1_125	Structure of the Modbus data.

7.7 EEM_2400_TCP

Function block EEM_2400_TCP is used for extern modules / extern Gateway Profinet / Ethernet/IP using register addresses 16#2400 – 16#240B. The values are written into a data structure in a standardized format.

7.7.1 Function block call



7.7.2 Input parameters

Name	Type	Description
xExecute	BOOL	By setting a positive edge, the data is loaded from the Modbus structure into the block, where it is transferred to the input and output structure of the block in a standardized format.

7.7.3 Output parameters

Name	Type	Description
uiAddr	UINT	This parameter transfers the start address of the Modbus register to the Modbus block.
iNumOfData	INT	This parameter transfers the amount of data to be read to the Modbus block.

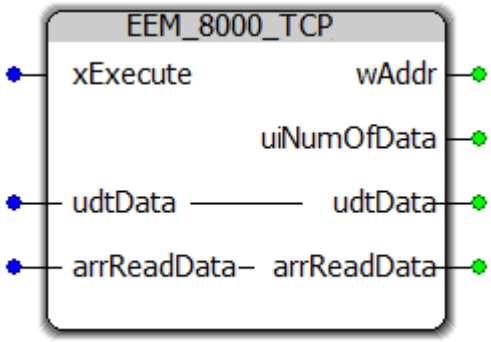
7.7.4 Inout parameters

Name	Type	Description
udtData	EEM_UDT_2400	Input structure for standardized values in the table.
arrReadData	MB_TCP_ARR_W_1_125	Structure of the Modbus data.

7.8 EEM_8000_TCP

Function block EEM_8000_TCP is used to read instantaneous values of register addresses 16#8000 – 16#8057. The values are written into a data structure in a standardized format.

7.8.1 Function block call



7.8.2 Input parameters

Name	Type	Description
xExecute	BOOL	By setting a positive edge, the data is loaded from the Modbus structure into the block, where it is transferred to the input and output structure of the block in a standardized format.

7.8.3 Output parameters

Name	Type	Description
uiAddr	UINT	This parameter transfers the start address of the Modbus register to the Modbus block.
iNumOfData	INT	This parameter transfers the amount of data to be read to the Modbus block.

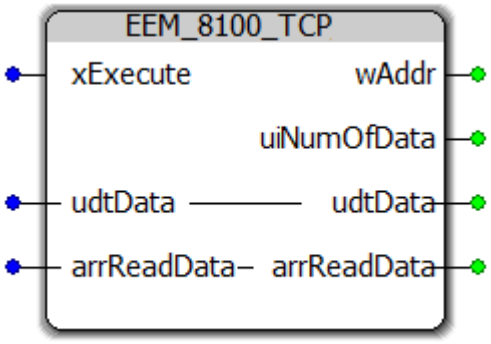
7.8.4 Inout parameters

Name	Type	Description
udtData	EEM_UDT_8000	Input structure for standardized values in the table.
arrReadData	MB_TCP_ARR_W_1_125	Structure of the Modbus data.

7.9 EEM_8100_TCP

Function block EEM_8100_TCP is used to read energy meter values of register addresses 16#8100 – 16#8116. The values are written into a data structure in a standardized format.

7.9.1 Function block call



7.9.2 Input parameters

Name	Type	Description
xExecute	BOOL	By setting a positive edge, the data is loaded from the Modbus structure into the block, where it is transferred to the input and output structure of the block in a standardized format.

7.9.3 Output parameters

Name	Type	Description
uiAddr	UINT	This parameter transfers the start address of the Modbus register to the Modbus block.
iNumOfData	INT	This parameter transfers the amount of data to be read to the Modbus block.

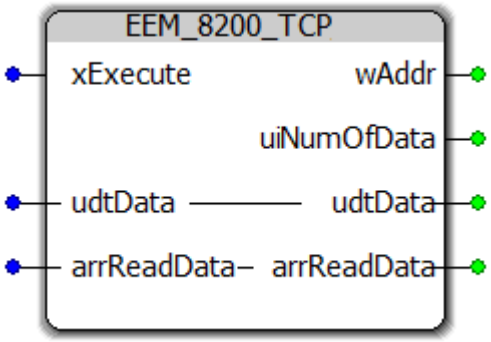
7.9.4 Inout parameters

Name	Type	Description
udtData	EEM_UDT_8100	Input structure for standardized values in the table.
arrReadData	MB_TCP_ARR_W_1_125	Structure of the Modbus data.

7.10 EEM_8200_TCP

Function block EEM_8200_TCP is used to read tariff meter values of register addresses 16#8200 – 16#822F. The values are written into a data structure in a standardized format.

7.10.1 Function block call



7.10.2 Input parameters

Name	Type	Description
xExecute	BOOL	By setting a positive edge, the data is loaded from the Modbus structure into the block, where it is transferred to the input and output structure of the block in a standardized format.

7.10.3 Output parameters

Name	Type	Description
uiAddr	UINT	This parameter transfers the start address of the Modbus register to the Modbus block.
iNumOfData	INT	This parameter transfers the amount of data to be read to the Modbus block.

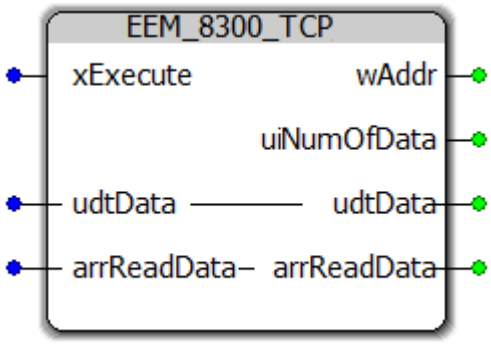
7.10.4 Inout parameters

Name	Type	Description
udtData	EEM_UDT_8200	Input structure for standardized values in the table.
arrReadData	MB_TCP_ARR_W_1_125	Structure of the Modbus data.

7.11 EEM_8300_TCP

Function block EEM_8300_TCP is used to read operating hours of register addresses 16#8300 – 16#8302. The values are written into a data structure in a standardized format.

7.11.1 Function block call



7.11.2 Input parameters

Name	Type	Description
xExecute	BOOL	By setting a positive edge, the data is loaded from the Modbus structure into the block, where it is transferred to the input and output structure of the block in a standardized format.

7.11.3 Output parameters

Name	Type	Description
uiAddr	UINT	This parameter transfers the start address of the Modbus register to the Modbus block.
iNumOfData	INT	This parameter transfers the amount of data to be read to the Modbus block.

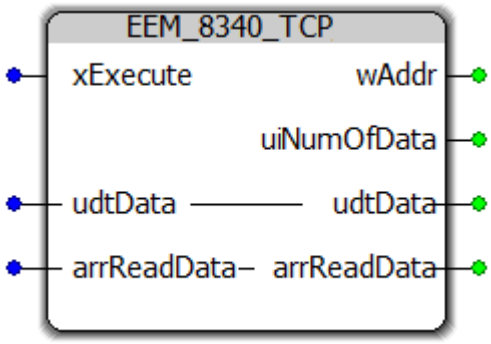
7.11.4 Inout parameters

Name	Type	Description
udtData	EEM_UDT_8300	Input structure for standardized values in the table.
arrReadData	MB_TCP_ARR_W_1_125	Structure of the Modbus data.

7.12 EEM_8340_TCP

Function block EEM_8340_TCP is used to read the state of impulse counter of digital input of register address 16#8340. The value is written into a data structure in a standardized format.

7.12.1 Function block call



7.12.2 Input parameters

Name	Type	Description
xExecute	BOOL	By setting a positive edge, the data is loaded from the Modbus structure into the block, where it is transferred to the input and output structure of the block in a standardized format.

7.12.3 Output parameters

Name	Type	Description
uiAddr	UINT	This parameter transfers the start address of the Modbus register to the Modbus block.
iNumOfData	INT	This parameter transfers the amount of data to be read to the Modbus block.

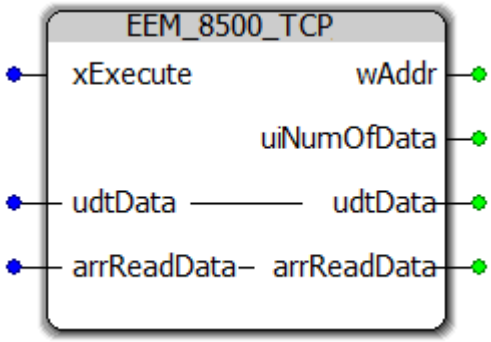
7.12.4 Inout parameters

Name	Type	Description
udtData	EEM_UDT_8340	Input structure for standardized values in the table.
arrReadData	MB_TCP_ARR_W_1_125	Structure of the Modbus data.

7.13 EEM_8500_TCP

Function block EEM_8500_TCP is used to read statistics (prediction) of register addresses 16#8500 - 16#850E. The values are written into a data structure in a standardized format.

7.13.1 Function block call



7.13.2 Input parameters

Name	Type	Description
xExecute	BOOL	By setting a positive edge, the data is loaded from the Modbus structure into the block, where it is transferred to the input and output structure of the block in a standardized format.

7.13.3 Output parameters

Name	Type	Description
uiAddr	UINT	This parameter transfers the start address of the Modbus register to the Modbus block.
iNumOfData	INT	This parameter transfers the amount of data to be read to the Modbus block.

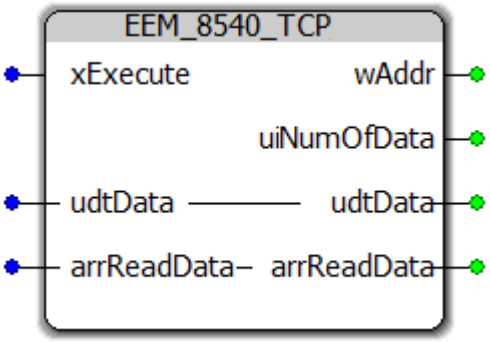
7.13.4 Inout parameters

Name	Type	Description
udtData	EEM_UDT_8500	Input structure for standardized values in the table.
arrReadData	MB_TCP_ARR_W_1_125	Structure of the Modbus data.

7.14 EEM_8540_TCP

Function block EEM_8540_TCP is used to read statistics (mean values) of register addresses 16#8540 - 16#855E. The values are written into a data structure in a standardized format.

7.14.1 Function block call



7.14.2 Input parameters

Name	Type	Description
xExecute	BOOL	By setting a positive edge, the data is loaded from the Modbus structure into the block, where it is transferred to the input and output structure of the block in a standardized format.

7.14.3 Output parameters

Name	Type	Description
uiAddr	UINT	This parameter transfers the start address of the Modbus register to the Modbus block.
iNumOfData	INT	This parameter transfers the amount of data to be read to the Modbus block.

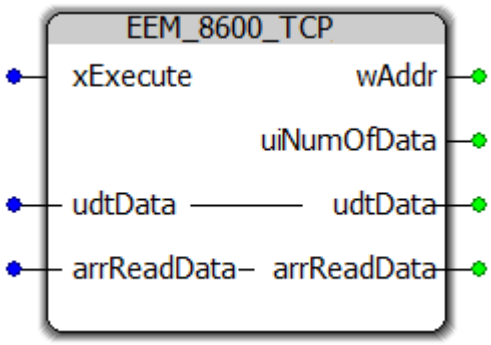
7.14.4 Inout parameters

Name	Type	Description
udtData	EEM_UDT_8540	Input structure for standardized values in the table.
arrReadData	MB_TCP_ARR_W_1_125	Structure of the Modbus data.

7.15 EEM_8600_TCP

Function block EEM_8600_TCP is used to read min / max values for mean values of register addresses 16#8600 - 16#863E. The values are written into a data structure in a standardized format.

7.15.1 Function block call



7.15.2 Input parameters

Name	Type	Description
xExecute	BOOL	By setting a positive edge, the data is loaded from the Modbus structure into the block, where it is transferred to the input and output structure of the block in a standardized format.

7.15.3 Output parameters

Name	Type	Description
uiAddr	UINT	This parameter transfers the start address of the Modbus register to the Modbus block.
iNumOfData	INT	This parameter transfers the amount of data to be read to the Modbus block.

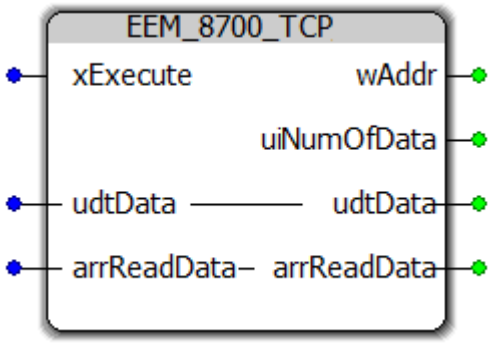
7.15.4 Inout parameters

Name	Type	Description
udtData	EEM_UDT_8600	Input structure for standardized values in the table.
arrReadData	MB_TCP_ARR_W_1_125	Structure of the Modbus data.

7.16 EEM_8700_TCP

Function block EEM_8700_TCP is used to read min / max values for instantaneous values of register addresses 16#8700 - 16#873E. The values are written into a data structure in a standardized format.

7.16.1 Function block call



7.16.2 Input parameters

Name	Type	Description
xExecute	BOOL	By setting a positive edge, the data is loaded from the Modbus structure into the block, where it is transferred to the input and output structure of the block in a standardized format.

7.16.3 Output parameters

Name	Type	Description
uiAddr	UINT	This parameter transfers the start address of the Modbus register to the Modbus block.
iNumOfData	INT	This parameter transfers the amount of data to be read to the Modbus block.

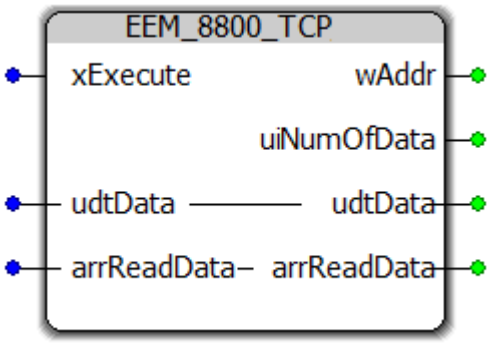
7.16.4 Inout parameters

Name	Type	Description
udtData	EEM_UDT_8700	Input structure for standardized values in the table.
arrReadData	MB_TCP_ARR_W_1_125	Structure of the Modbus data.

7.17 EEM_8800_TCP

Function block EEM_8800_TCP is used to read total harmonic distortion of register addresses 16#8800 - 16#8809. The values are written into a data structure in a standardized format.

7.17.1 Function block call



7.17.2 Input parameters

Name	Type	Description
xExecute	BOOL	By setting a positive edge, the data is loaded from the Modbus structure into the block, where it is transferred to the input and output structure of the block in a standardized format.

7.17.3 Output parameters

Name	Type	Description
uiAddr	UINT	This parameter transfers the start address of the Modbus register to the Modbus block.
iNumOfData	INT	This parameter transfers the amount of data to be read to the Modbus block.

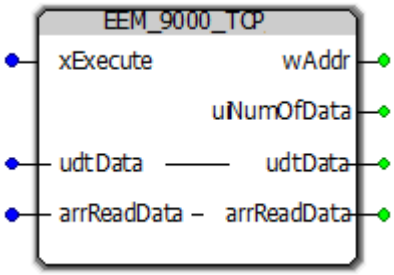
7.17.4 Inout parameters

Name	Type	Description
udtData	EEM_UDT_8800	Input structure for standardized values in the table.
arrReadData	MB_TCP_ARR_W_1_125	Structure of the Modbus data.

7.18 EEM_9000_TCP

Function block EEM_9000_TCP is used to read meter readings of register addresses 16#9000 - 16#9456. The values are written into a data structure in a standardized format.

7.18.1 Function block call



7.18.2 Input parameters

Name	Type	Description
xExecute	BOOL	By setting a positive edge, the data is loaded from the Modbus structure into the block, where it is transferred to the input and output structure of the block in a standardized format.

7.18.3 Output parameters

Name	Type	Description
uiAddr	UINT	This parameter transfers the start address of the Modbus register to the Modbus block.
iNumOfData	INT	This parameter transfers the amount of data to be read to the Modbus block.

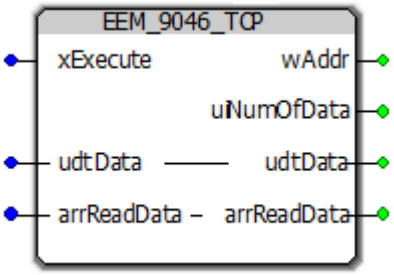
7.18.4 Inout parameters

Name	Type	Description
udtData	EEM_UDT_9000	Input structure for standardized values in the table.
arrReadData	arrModbus2_w_1_125	Structure of the Modbus data.

7.19 EEM_9046_TCP

Function block EEM_9046_TCP is used to read out voltage dips of register addresses 16#9046 - 16#908B. The values are written into a data structure in a standardized format.

7.19.1 Function block call



7.19.2 Input parameters

Name	Type	Description
xExecute	BOOL	By setting a positive edge, the data is loaded from the Modbus structure into the block, where it is transferred to the input and output structure of the block in a standardized format.

7.19.3 Output parameters

Name	Type	Description
uiAddr	UINT	This parameter transfers the start address of the Modbus register to the Modbus block.
iNumOfData	INT	This parameter transfers the amount of data to be read to the Modbus block.

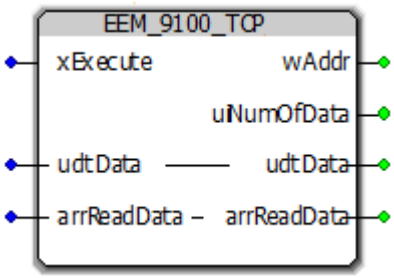
7.19.4 Inout parameters

Name	Type	Description
udtData	EEM_UDT_9046	Input structure for standardized values in the table.
arrReadData	arrModbus2_w_1_125	Structure of the Modbus data.

7.20 EEM_9100_TCP

Function block EEM_9100_TCP is used to read out voltage swells of register addresses 16#9100 - 16#9145. The values are written into a data structure in a standardized format.

7.20.1 Function block call



7.20.2 Input parameters

Name	Type	Description
xExecute	BOOL	By setting a positive edge, the data is loaded from the Modbus structure into the block, where it is transferred to the input and output structure of the block in a standardized format.

7.20.3 Output parameters

Name	Type	Description
uiAddr	UINT	This parameter transfers the start address of the Modbus register to the Modbus block.
iNumOfData	INT	This parameter transfers the amount of data to be read to the Modbus block.

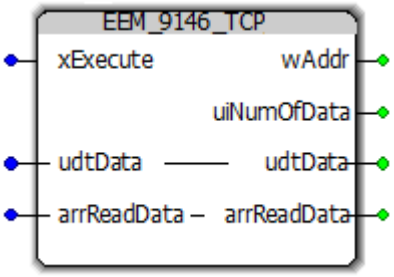
7.20.4 Inout parameters

Name	Type	Description
udtData	EEM_UDT_9100	Input structure for standardized values in the table.
arrReadData	arrModbus2_w_1_125	Structure of the Modbus data.

7.21 EEM_9146_TCP

Function block EEM_9146_TCP is used to read out voltage swells of register addresses 16#9146 - 16#918B. The values are written into a data structure in a standardized format.

7.21.1 Function block call



7.21.2 Input parameters

Name	Type	Description
xExecute	BOOL	By setting a positive edge, the data is loaded from the Modbus structure into the block, where it is transferred to the input and output structure of the block in a standardized format.

7.21.3 Output parameters

Name	Type	Description
uiAddr	UINT	This parameter transfers the start address of the Modbus register to the Modbus block.
iNumOfData	INT	This parameter transfers the amount of data to be read to the Modbus block.

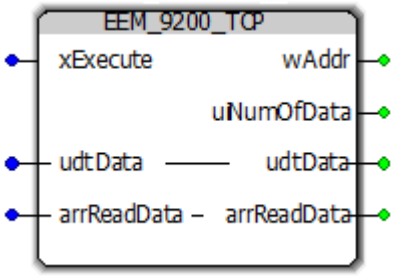
7.21.4 Inout parameters

Name	Type	Description
udtData	EEM_UDT_9146	Input structure for standardized values in the table.
arrReadData	arrModbus2_w_1_125	Structure of the Modbus data.

7.22 EEM_9200_TCP

Function block EEM_9200_TCP is used to read out voltage interrupts of register addresses 16#9200 - 16#923B. The values are written into a data structure in a standardized format.

7.22.1 Function block call



7.22.2 Input parameters

Name	Type	Description
xExecute	BOOL	By setting a positive edge, the data is loaded from the Modbus structure into the block, where it is transferred to the input and output structure of the block in a standardized format.

7.22.3 Output parameters

Name	Type	Description
uiAddr	UINT	This parameter transfers the start address of the Modbus register to the Modbus block.
iNumOfData	INT	This parameter transfers the amount of data to be read to the Modbus block.

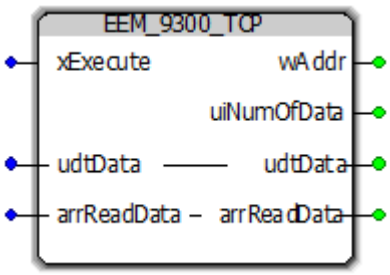
7.22.4 Inout parameters

Name	Type	Description
udtData	EEM_UDT_9200	Input structure for standardized values in the table.
arrReadData	arrModbus2_w_1_125	Structure of the Modbus data.

7.23 EEM_9300_TCP

Function block EEM_9300_TCP is used to read energy meters (not resetable) of register addresses 16#9300 - 16#9356. The values are written into a data structure in a standardized format.

7.23.1 Function block call



7.23.2 Input parameters

Name	Type	Description
xExecute	BOOL	By setting a positive edge, the data is loaded from the Modbus structure into the block, where it is transferred to the input and output structure of the block in a standardized format.

7.23.3 Output parameters

Name	Type	Description
uiAddr	UINT	This parameter transfers the start address of the Modbus register to the Modbus block.
iNumOfData	INT	This parameter transfers the amount of data to be read to the Modbus block.

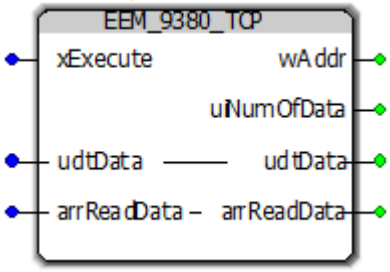
7.23.4 Inout parameters

Name	Type	Description
udtData	EEM_UDT_9300	Input structure for standardized values in the table.
arrReadData	MB_TCP_ARR_W_1_125	Structure of the Modbus data.

7.24 EEM_9380_TCP

Function block EEM_9380_TCP is used to read energy meters (not resetable) of register addresses 16#9380 - 16#93D6. The values are written into a data structure in a standardized format.

7.24.1 Function block call



7.24.2 Input parameters

Name	Type	Description
xExecute	BOOL	By setting a positive edge, the data is loaded from the Modbus structure into the block, where it is transferred to the input and output structure of the block in a standardized format.

7.24.3 Output parameters

Name	Type	Description
uiAddr	UINT	This parameter transfers the start address of the Modbus register to the Modbus block.
iNumOfData	INT	This parameter transfers the amount of data to be read to the Modbus block.

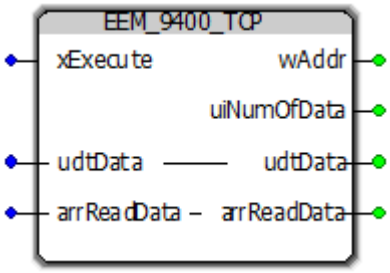
7.24.4 Inout parameters

Name	Type	Description
udtData	EEM_UDT_9380	Input structure for standardized values in the table.
arrReadData	MB_TCP_ARR_W_1_125	Structure of the Modbus data.

7.25 EEM_9400_TCP

Function block EEM_9400_TCP is used to read meter readings of register addresses 16#9400 - 16#9456. The values are written into a data structure in a standardized format.

7.25.1 Function block call



7.25.2 Input parameters

Name	Type	Description
xExecute	BOOL	By setting a positive edge, the data is loaded from the Modbus structure into the block, where it is transferred to the input and output structure of the block in a standardized format.

7.25.3 Output parameters

Name	Type	Description
uiAddr	UINT	This parameter transfers the start address of the Modbus register to the Modbus block.
iNumOfData	INT	This parameter transfers the amount of data to be read to the Modbus block.

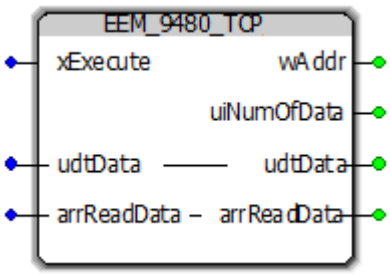
7.25.4 Inout parameters

Name	Type	Description
udtData	EEM_UDT_9400	Input structure for standardized values in the table.
arrReadData	arrModbus2_w_1_125	Structure of the Modbus data.

7.26 EEM_9480_TCP

Function block EEM_9480_TCP is used to read energy meter tariff 1 of register addresses 16#9480 - 16#94D6. The values are written into a data structure in a standardized format.

7.26.1 Function block call



7.26.2 Input parameters

Name	Type	Description
xExecute	BOOL	By setting a positive edge, the data is loaded from the Modbus structure into the block, where it is transferred to the input and output structure of the block in a standardized format.

7.26.3 Output parameters

Name	Type	Description
uiAddr	UINT	This parameter transfers the start address of the Modbus register to the Modbus block.
iNumOfData	INT	This parameter transfers the amount of data to be read to the Modbus block.

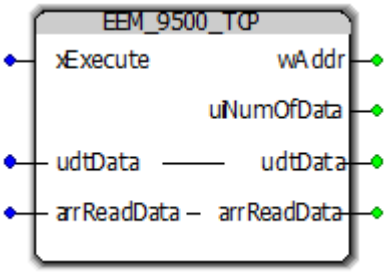
7.26.4 Inout parameters

Name	Type	Description
udtData	EEM_UDT_9480	Input structure for standardized values in the table.
arrReadData	arrModbus2_w_1_125	Structure of the Modbus data.

7.27 EEM_9500_TCP

Function block EEM_9500_TCP is used to read energy meter tariff 3 of register addresses 16#9500 - 16#9556. The values are written into a data structure in a standardized format.

7.27.1 Function block call



7.27.2 Input parameters

Name	Type	Description
xExecute	BOOL	By setting a positive edge, the data is loaded from the Modbus structure into the block, where it is transferred to the input and output structure of the block in a standardized format.

7.27.3 Output parameters

Name	Type	Description
uiAddr	UINT	This parameter transfers the start address of the Modbus register to the Modbus block.
iNumOfData	INT	This parameter transfers the amount of data to be read to the Modbus block.

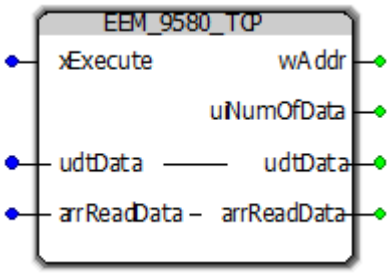
7.27.4 Inout parameters

Name	Type	Description
udtData	EEM_UDT_9500	Input structure for standardized values in the table.
arrReadData	arrModbus2_w_1_125	Structure of the Modbus data.

7.28 EEM_9580_TCP

Function block EEM_9580_TCP is used to read energy meter tariff 4 of register addresses 16#9580 - 16#95D6. The values are written into a data structure in a standardized format.

7.28.1 Function block call



7.28.2 Input parameters

Name	Type	Description
xExecute	BOOL	By setting a positive edge, the data is loaded from the Modbus structure into the block, where it is transferred to the input and output structure of the block in a standardized format.

7.28.3 Output parameters

Name	Type	Description
uiAddr	UINT	This parameter transfers the start address of the Modbus register to the Modbus block.
iNumOfData	INT	This parameter transfers the amount of data to be read to the Modbus block.

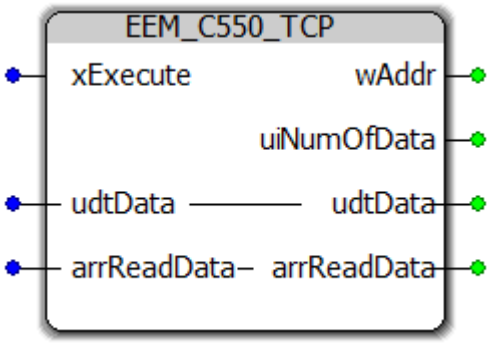
7.28.4 Inout parameters

Name	Type	Description
udtData	EEM_UDT_9580	Input structure for standardized values in the table.
arrReadData	arrModbus2_w_1_125	Structure of the Modbus data.

7.29 EEM_C550_TCP

The function block EEM_C550_RTU is used to read instanteous values of register addresses 16#C550 – 16#C58C. The values are written into a data structure in a standardized format.

7.29.1 Function block call



7.29.2 Input parameters

Name	Type	Description
xExecute	BOOL	By setting a positive edge, the data is loaded from the Modbus structure into the block, where it is transferred to the input and output structure of the block in a standardized format.

7.29.3 Output parameters

Name	Type	Description
uiAddr	UINT	This parameter transfers the start address of the Modbus register to the Modbus block.
iNumOfData	INT	This parameter transfers the amount of data to be read to the Modbus block.

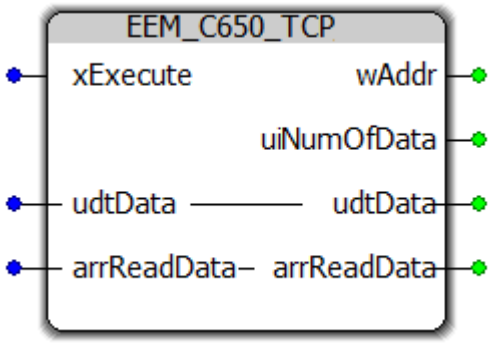
7.29.4 Inout parameters

Name	Type	Description
udtData	EEM_UDT_C550	Input structure for standardized values in the table.
arrReadData	MB_TCP_ARR_W_1_125	Structure of the Modbus data.

7.30 EEM_C650_TCP

Block EEM_C650_TCP is used to read energy values of register addresses 16#C650 – 16#C664. The values are written into a data structure in a standardized format.

7.30.1 Function block call



7.30.2 Input parameters

Name	Type	Description
xExecute	BOOL	By setting a positive edge, the data is loaded from the Modbus structure into the block, where it is transferred to the input and output structure of the block in a standardized format.

7.30.3 Output parameters

Name	Type	Description
uiAddr	UINT	This parameter transfers the start address of the Modbus register to the Modbus block.
iNumOfData	INT	This parameter transfers the amount of data to be read to the Modbus block.

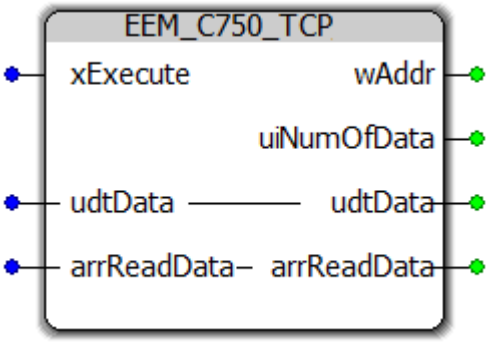
7.30.4 Inout parameters

Name	Type	Description
udtData	EEM_UDT_C650	Input structure for standardized values in the table.
arrReadData	MB_TCP_ARR_W_1_125	Structure of the Modbus data.

7.31 EEM_C750_TCP

The function block EEM_C750_TCP is used to read the average values based on current and voltage transducer via the set measurement duration (under consideration of transmission ratios) of register addresses 16#C750 – 16#C795. The values are written into a data structure in a standardized format.

7.31.1 Function block call



7.31.2 Input parameters

Name	Type	Description
xExecute	BOOL	By setting a positive edge, the data is loaded from the Modbus structure into the block, where it is transferred to the input and output structure of the block in a standardized format.

7.31.3 Output parameters

Name	Type	Description
uiAddr	UINT	This parameter transfers the start address of the Modbus register to the Modbus block.
iNumOfData	INT	This parameter transfers the amount of data to be read to the Modbus block.

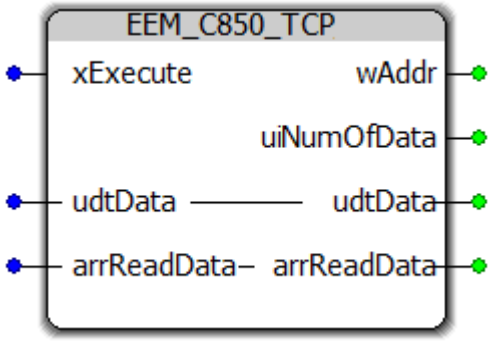
7.31.4 Inout parameters

Name	Type	Description
udtData	EEM_UDT_C750	Input structure for standardized values in the table.
arrReadData	MB_TCP_ARR_W_1_125	Structure of the Modbus data.

7.32 EEM_C850_TCP

The function block EEM_C850_TCP is used to read current main measured values not based on current transformer (without taking into consideration transformation ratios) of register addresses 16#C850 – 16#C872. The values are written into a data structure in a standardized format.

7.32.1 Function block call



7.32.2 Input parameters

Name	Type	Description
xExecute	BOOL	By setting a positive edge, the data is loaded from the Modbus structure into the block, where it is transferred to the input and output structure of the block in a standardized format.

7.32.3 Output parameters

Name	Type	Description
uiAddr	UINT	This parameter transfers the start address of the Modbus register to the Modbus block.
iNumOfData	INT	This parameter transfers the amount of data to be read to the Modbus block.

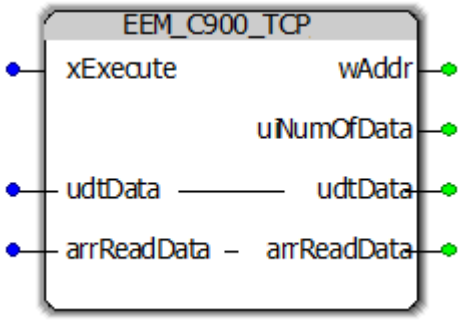
7.32.4 Inout parameters

Name	Type	Description
udtData	EEM_UDT_C850	Input structure for standardized values in the table.
arrReadData	MB_TCP_ARR_W_1_125	Structure of the Modbus data.

7.33 EEM_C900_TCP

Function block EEM_C900_TCP is used to read temperature inputs of register addresses 16#C900 – 16#C907. The values are written into a data structure in a standardized format.

7.33.1 Function block call



7.33.2 Input parameters

Name	Type	Description
xExecute	BOOL	By setting a positive edge, the data is loaded from the Modbus structure into the block, where it is transferred to the input and output structure of the block in a standardized format.

7.33.3 Output parameters

Name	Type	Description
uiAddr	UINT	This parameter transfers the start address of the Modbus register to the Modbus block.
iNumOfData	INT	This parameter transfers the amount of data to be read to the Modbus block.

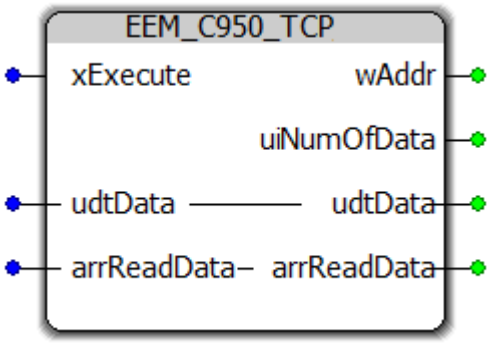
7.33.4 Inout parameters

Name	Type	Description
udtData	EEM_UDT_C900	Input structure for standardized values in the table.
arrReadData	MB_TCP_ARR_W_1_125	Structure of the Modbus data.

7.34 EEM_C950_TCP

Function block EEM_C950_TCP is used to read current harmonic distortion and harmonics of register addresses. The values are written into a data structure in a standardized format.

7.34.1 Function block call



7.34.2 Input parameters

Name	Type	Description
xExecute	BOOL	By setting a positive edge, the data is loaded from the Modbus structure into the block, where it is transferred to the input and output structure of the block in a standardized format.

7.34.3 Output parameters

Name	Type	Description
uiAddr	UINT	This parameter transfers the start address of the Modbus register to the Modbus block.
iNumOfData	INT	This parameter transfers the amount of data to be read to the Modbus block.

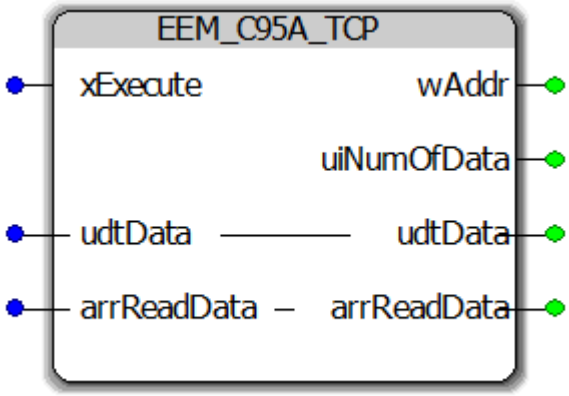
7.34.4 Inout parameters

Name	Type	Description
udtData	EEM_UDT_C950	Input structure for standardized values in the table.
arrReadData	MB_TCP_ARR_W_1_125	Structure of the Modbus data.

7.35 EEM_C95A_TCP

Function block EEM_C95A_TCP is used to read individual harmonic content for currents of register addresses 16#C95A – 16#CA92. The values are written into a data structure in a standardized format.

7.35.1 Function block call



7.35.2 Input parameters

Name	Type	Description
xExecute	BOOL	By setting a positive edge, the data is loaded from the Modbus structure into the block, where it is transferred to the input and output structure of the block in a standardized format.

7.35.3 Output parameters

Name	Type	Description
uiAddr	UINT	This parameter transfers the start address of the Modbus register to the Modbus block.
iNumOfData	INT	This parameter transfers the amount of data to be read to the Modbus block.

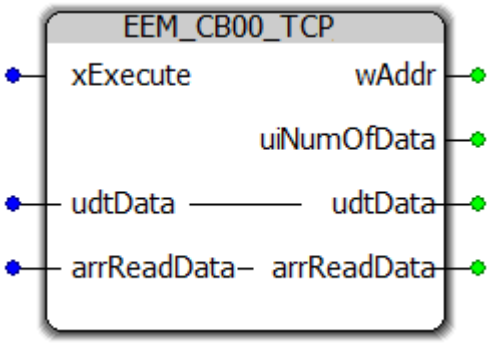
7.35.4 Inout parameters

Name	Type	Description
udtData	EEM_UDT_C95A	Input structure for standardized values in the table.
arrReadData	MB_TCP_ARR_W_1_125	Structure of the Modbus data.

7.36 EEM_CB00_TCP

Function block EEM_CB00_TCP is used to read the status of inputs/outputs for register addresses 16#CB00 – 16#CB03. The values are written into a data structure in a standardized format.

7.36.1 Function block call



7.36.2 Input parameters

Name	Type	Description
xExecute	BOOL	By setting a positive edge, the data is loaded from the Modbus structure into the block, where it is transferred to the input and output structure of the block in a standardized format.

7.36.3 Output parameters

Name	Type	Description
uiAddr	UINT	This parameter transfers the start address of the Modbus register to the Modbus block.
iNumOfData	INT	This parameter transfers the amount of data to be read to the Modbus block.

7.36.4 Inout parameters

Name	Type	Description
udtData	EEM_UDT_CB00	Input structure for standardized values in the table.
arrReadData	MB_TCP_ARR_W_1_125	Structure of the Modbus data.

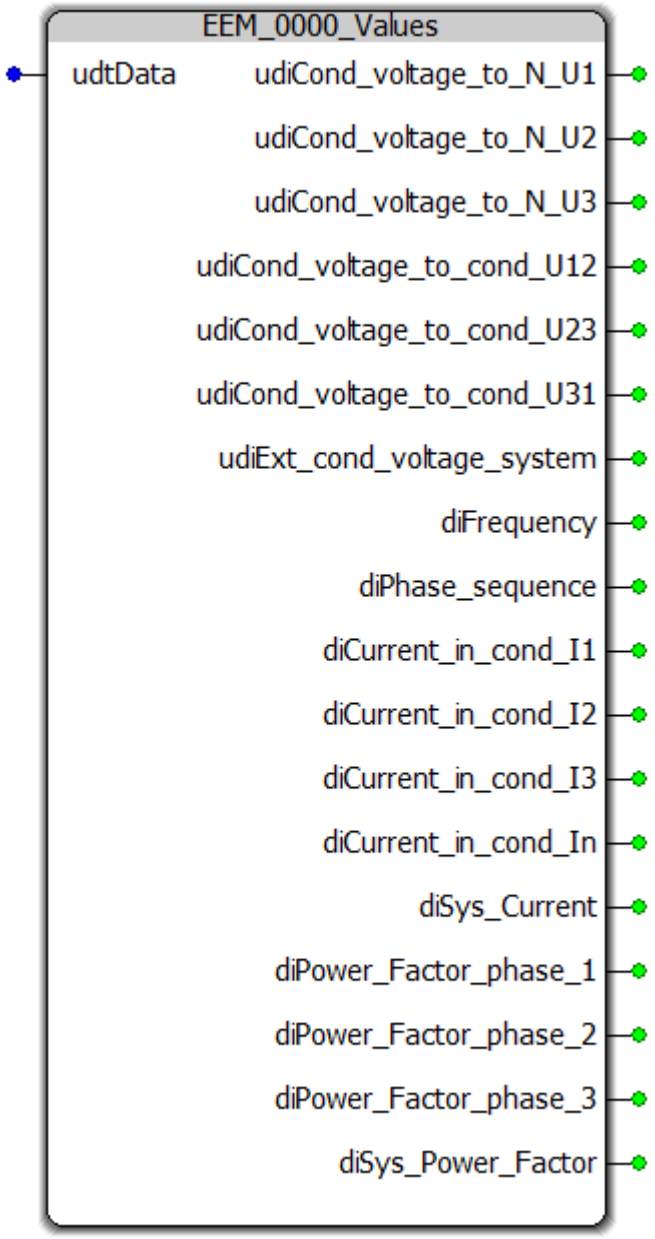
8 EEM_*_Values

These function blocks convert the EEM_UDT_* input structures into individual output parameters.

8.1 EEM_0000_Values

The function block EEM_0000_Values converts the EEM_UDT_0000 input structure into individual output parameters.

8.1.1 Function block call



8.1.2 Input parameters

Name	Type	Description
udtData	EEM_UDT_0000	Input structure for standardized values in the table.

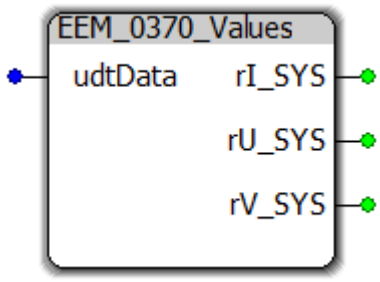
8.1.3 Output parameters

Name	Type	Description
udiCond_voltage_to_N_U1	UDINT	Conductor voltage to N U1.
udiCond_voltage_to_N_U2	UDINT	Conductor voltage to N U2.
udiCond_voltage_to_N_U3	UDINT	Conductor voltage to N U3.
udiCond_voltage_to_cond_U12	UDINT	Conductor voltage to conductor U12.
udiCond_voltage_to_cond_U23	UDINT	Conductor voltage to conductor U23.
udiCond_voltage_to_cond_U31	UDINT	Conductor voltage to conductor U31.
udiExt_cond_voltage_system	UDINT	External conductor voltage system.
diCurrent_in_cond_I1	DINT	Current in conductor I1.
diCurrent_in_cond_I2	DINT	Current in conductor I2.
diCurrent_in_cond_I3	DINT	Current in conductor I3.
diCurrent_in_cond_In	DINT	Current in conductor In.
diSys_Current	DINT	System current.
diPower_Factor_phase_1	DINT	Power factor phase 1 +/-.
diPower_Factor_phase_2	DINT	Power factor phase 2 +/-.
diPower_Factor_phase_3	DINT	Power factor phase 3 +/-.
diSys_Power_Factor	DINT	System power factor.
diFrequency	DINT	Frequency.
diPhase_sequence	DINT	Phase sequence: 0: 123 (CCW: counterclockwise) 1: 132 (CW: clockwise) 2: Not available

8.2 EEM_0370_Values

The function block EEM_0370_Values converts the EEM_UDT_0370 input structure into individual output parameters.

8.2.1 Function block call



8.2.2 Input parameters

Name	Type	Description
udtData	EEM_UDT_0370	Input structure for standardized values in the table.

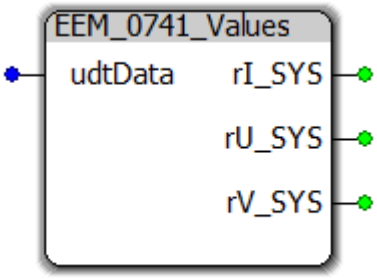
8.2.3 Output parameters

Name	Type	Description
rI_SYS	REAL	ISYS: current average value(I1+I2+I3)/3.
rU_SYS	REAL	USYS: line voltages (U12 + U23 + U31)/3 average value.
rV_SYS	REAL	VSYS: average value of line voltages against N (V1 + V2 + V3)/3.

8.3 EEM_0741_Values

The function block EEM_0741_Values converts the EEM_UDT_0741 input structure into individual output parameters.

8.3.1 Function block call



8.3.2 Input parameters

Name	Type	Description
udtData	EEM_UDT_0741	Input structure for standardized values in the table.

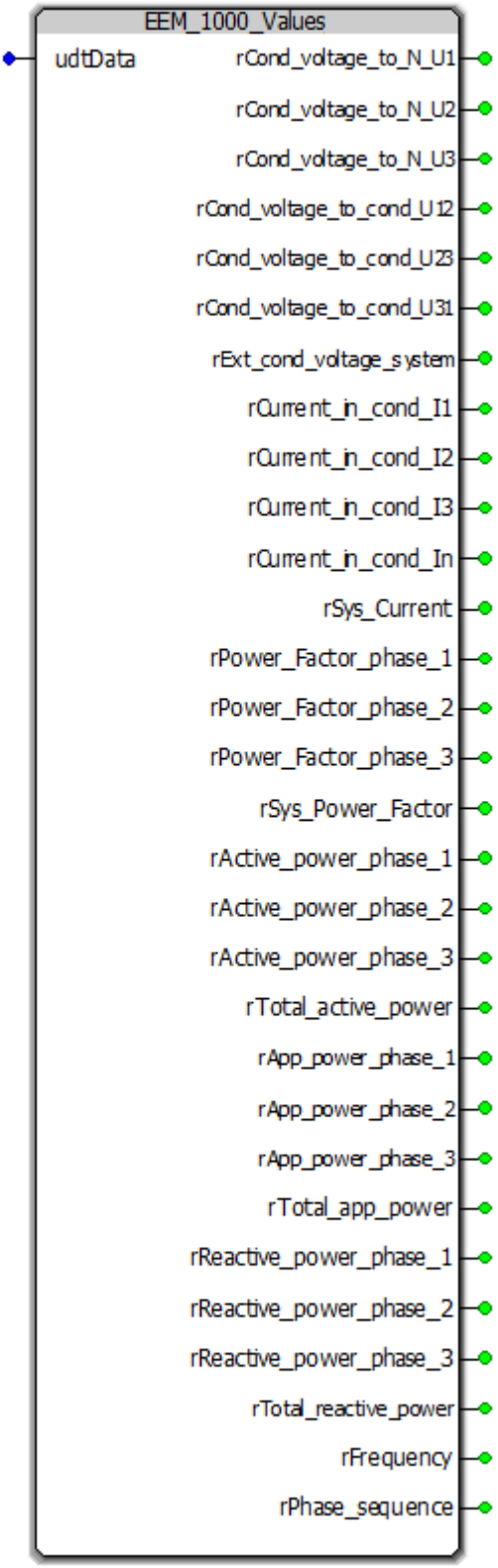
8.3.3 Output parameters

Name	Type	Description
rI_SYS	REAL	ISYS: current average value(I1+I2+I3)/3.
rU_SYS	REAL	USYS: line voltages (U12 + U23 + U31)/3 average value.
rV_SYS	REAL	VSYS: average value of line voltages against N (V1 + V2 + V3)/3.

8.4 EEM_1000_Values

The function block EEM_1000_Values converts the EEM_UDT_1000 input structure into individual output parameters.

8.4.1 Function block call



8.4.2 Input parameters

Name	Type	Description
udtData	EEM_UDT_1000	Input structure for standardized values in the table.

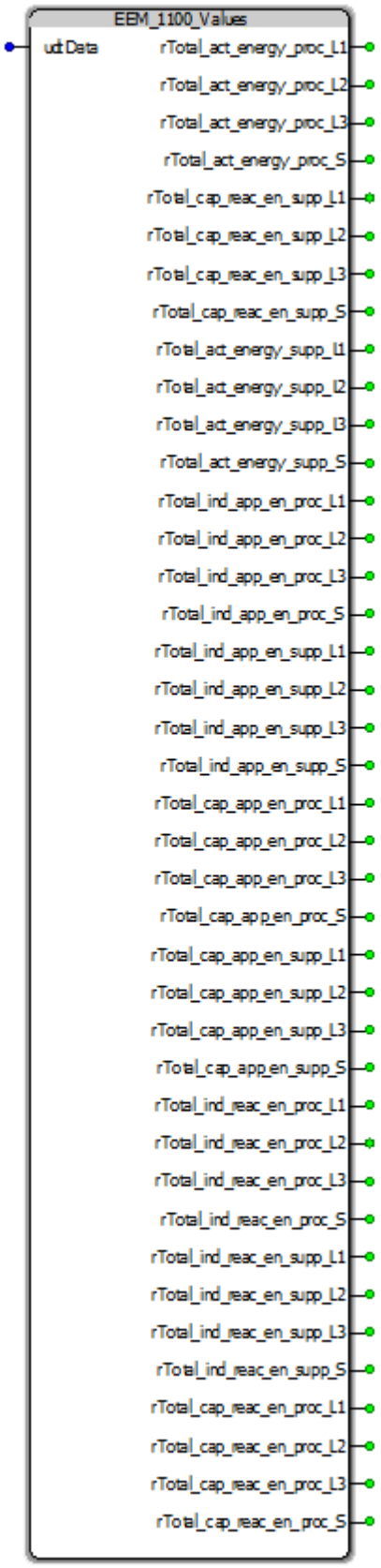
8.4.3 Output parameters

Name	Type	Description
rCond_voltage_to_N_U1	REAL	Conductor voltage to N U1.
rCond_voltage_to_N_U2	REAL	Conductor voltage to N U2.
rCond_voltage_to_N_U3	REAL	Conductor voltage to N U3.
rCond_voltage_to_cond_U12	REAL	Conductor voltage to conductor U12.
rCond_voltage_to_cond_U23	REAL	Conductor voltage to conductor U123.
rCond_voltage_to_cond_U31	REAL	Conductor voltage to conductor U31.
rExt_cond_voltage_system	REAL	External conductor voltage system.
rCurrent_in_cond_I1	REAL	Current in conductor I1.
rCurrent_in_cond_I2.	REAL	Current in conductor I2.
rCurrent_in_cond_I3	REAL	Current in conductor I3.
rCurrent_in_cond_In	REAL	Current in conductor In.
rSys_Current	REAL	System current.
rPower_factor_phase_1	REAL	Power factor phase 1 +/-.
rPower_factor_phase_2	REAL	Power factor phase 2 +/-.
rPower_factor_phase_3	REAL	Power factor phase 3 +/-.
rSys_power_factor	REAL	System power factor.
rActive_power_phase_1	REAL	Active power phase 1 +/-.
rActive_power_phase_2	REAL	Active power phase 2 +/-.
rActive_power_phase_3	REAL	Active power phase 3 +/-.
rTotal_active_power	REAL	Total active power +/-.
rApp_power_phase_1	REAL	Apparent power phase 1.
rApp_power_phase_2	REAL	Apparent power phase 2.
rApp_power_phase_3	REAL	Apparent power phase 3.
rTotal_app_power	REAL	Total apparent power +/-.
rReactive_power_phase_1	REAL	Reactive power phase 1 +/-.
rReactive_power_phase_2	REAL	Reactive power phase 2 +/-.
rReactive_power_phase_3	REAL	Reactive power phase 3 +/-.
rTotal_reactive_power	REAL	Total reactive power +/-.
rFrequency	REAL	Frequency.
rPhase_sequence	REAL	Phase sequence: 0.123 (CCW: counterclockwise) 0.132: (CW: clockwise) 0: Not available

8.5 EEM_1100_Values

The function block EEM_1100_Values converts the EEM_UDT_1100 input structure into individual output parameters.

8.5.1 Function block call



8.5.2 Input parameters

Name	Type	Description
udtData	EEM_UDT_1000	Input structure for standardized values in the table.

8.5.3 Output parameters

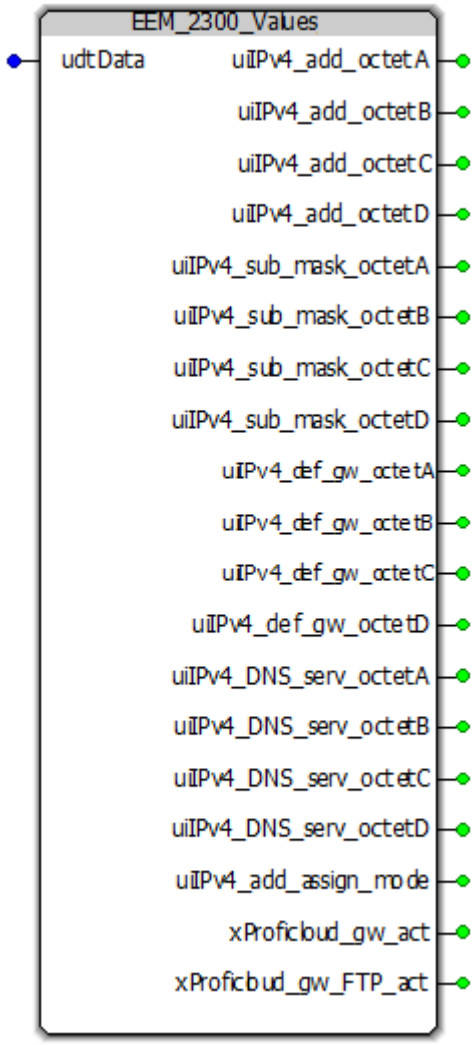
Name	Type	Description
rTotal_act_energy_proc_L1	REAL	Total active energy procurement L1 (cannot be reset).
rTotal_act_energy_proc_L2	REAL	Total active energy procurement L2 (cannot be reset).
rTotal_act_energy_proc_L3	REAL	Total active energy procurement L3 (cannot be reset).
rTotal_act_energy_proc_S	REAL	Total active energy procurement Σ (cannot be reset).
rTotal_act_energy_supp_L1	REAL	Total active energy supply L1 (cannot be reset).
rTotal_act_energy_supp_L2	REAL	Total active energy supply L2 (cannot be reset).
rTotal_act_energy_supp_L3	REAL	Total active energy supply L3 (cannot be reset).
rTotal_act_energy_supp_S	REAL	Total active energy supply Σ (cannot be reset).
rTotal_ind_app_en_proc_L1	REAL	Total inductive apparent energy procurement L1 (cannot be reset).
rTotal_ind_app_en_proc_L2	REAL	Total inductive apparent energy procurement L2 (cannot be reset).
rTotal_ind_app_en_proc_L3	REAL	Total inductive apparent energy procurement L3 (cannot be reset).
rTotal_ind_app_en_proc_S	REAL	Total inductive apparent energy procurement Σ (cannot be reset).
rTotal_ind_app_en_supp_L1	REAL	Total inductive apparent energy supply L1 (cannot be reset).
rTotal_ind_app_en_supp_L2	REAL	Total inductive apparent energy supply L2 (cannot be reset).
rTotal_ind_app_en_supp_L3	REAL	Total inductive apparent energy supply L3 (cannot be reset).
rTotal_ind_app_en_supp_S	REAL	Total inductive apparent energy supply Σ (cannot be reset).
rTotal_cap_app_en_proc_L1	REAL	Total capacitive apparent energy procurement L1 (cannot be reset).
rTotal_cap_app_en_proc_L2	REAL	Total capacitive apparent energy procurement L2 (cannot be reset).
rTotal_cap_app_en_proc_L3	REAL	Total capacitive apparent energy procurement L3 (cannot be reset).
rTotal_cap_app_en_proc_S	REAL	Total capacitive apparent energy procurement Σ (cannot be reset).
rTotal_cap_app_en_supp_L1	REAL	Total capacitive apparent energy supply L1 (cannot be reset).
rTotal_cap_app_en_supp_L2	REAL	Total capacitive apparent energy supply L2 (cannot be reset).
rTotal_cap_app_en_supp_L3	REAL	Total capacitive apparent energy supply L3 (cannot be reset).
rTotal_cap_app_en_supp_S	REAL	Total capacitive apparent energy supply Σ (cannot be reset).
rTotal_ind_reac_en_proc_L1	REAL	Total inductive reactive energy procurement L1 (cannot be reset).
rTotal_ind_reac_en_proc_L2	REAL	Total inductive reactive energy procurement L2 (cannot be reset).
rTotal_ind_reac_en_proc_L3	REAL	Total inductive reactive energy procurement L3 (cannot be reset).
rTotal_ind_reac_en_proc_S	REAL	Total inductive reactive energy procurement Σ (cannot be reset).
rTotal_ind_reac_en_supp_L1	REAL	Total inductive reactive energy supply L1 (cannot be reset).
rTotal_ind_reac_en_supp_L2	REAL	Total inductive reactive energy supply L2 (cannot be reset).
rTotal_ind_reac_en_supp_L3	REAL	Total inductive reactive energy supply L3 (cannot be reset).
rTotal_ind_reac_en_supp_S	REAL	Total inductive reactive energy supply Σ (cannot be reset).
rTotal_cap_reac_en_proc_L1	REAL	Total inductive reactive energy procurement L1 (cannot be reset).
rTotal_cap_reac_en_proc_L2	REAL	Total inductive reactive energy procurement L2 (cannot be reset).
rTotal_cap_reac_en_proc_L3	REAL	Total inductive reactive energy procurement L3 (cannot be reset).
rTotal_cap_reac_en_proc_S	REAL	Total inductive reactive energy procurement Σ (cannot be reset).

rTotal_cap_reac_en_supp_L1	REAL	Total capacitive reactive energy supply L1 (cannot be reset).
rTotal_cap_reac_en_supp_L2	REAL	Total capacitive reactive energy supply L2 (cannot be reset).
rTotal_cap_reac_en_supp_L3	REAL	Total capacitive reactive energy supply L3 (cannot be reset).
rTotal_cap_reac_en_supp_S	REAL	Total capacitive reactive energy supply Σ (cannot be reset).

8.6 EEM_2300_Values

The function block EEM_2300_Values converts the EEM_UDT_2300 input structure into individual output parameters.

8.6.1 Function block call



8.6.2 Input parameters

Name	Type	Description
udtData	EEM_UDT_2300	Input structure for standardized values in the table.

8.6.3 Output parameters

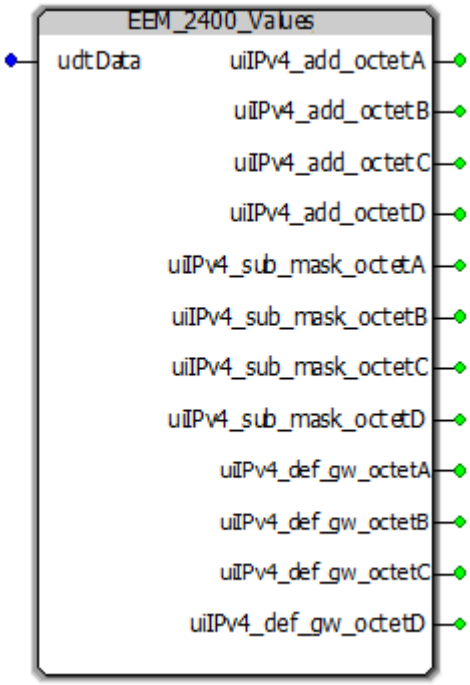
Name	Type	Description
uiIPv4_add_octetA	UINT	(only if register 8976 = 0) The IPv4 address is specified in the following format: A.B.C.D Octet A
uiIPv4_add_octetB	UINT	(only if register 8976 = 0) The IPv4 address is specified in the following format: A.B.C.D Octet B
uiIPv4_add_octetC	UINT	(only if register 8976 = 0) The IPv4 address is specified in the following format: A.B.C.D Octet C
uiIPv4_add_octetD	UINT	(only if register 8976 = 0) The IPv4 address is specified in the following format: A.B.C.D Octet D
uiIPv4_sub_mask_octetA	UINT	(only if register 8976 = 0) The IPv4 subnet mask is specified in the following format: A.B.C.D Octet A
uiIPv4_sub_mask_octetB	UINT	(only if register 8976 = 0) The IPv4 subnet mask is specified in the following format: A.B.C.D Octet B
uiIPv4_sub_mask_octetC	UINT	(only if register 8976 = 0) The IPv4 subnet mask is specified in the following format: A.B.C.D Octet C
uiIPv4_sub_mask_octetD	UINT	(only if register 8976 = 0) The IPv4 subnet mask is specified in the following format: A.B.C.D Octet D
uiIPv4_def_gw_octetA	UINT	(only if register 8976 = 0) The IPv4 default gateway is specified in the following format: A.B.C.D Octet A
uiIPv4_def_gw_octetB	UINT	(only if register 8976 = 0) The IPv4 default gateway is specified in the following format: A.B.C.D Octet B
uiIPv4_def_gw_octetC	UINT	(only if register 8976 = 0) The IPv4 default gateway is specified in the following format: A.B.C.D Octet C
uiIPv4_def_gw_octetD	UINT	(only if register 8976 = 0) The IPv4 default gateway is specified in the following format: A.B.C.D Octet D
uiIPv4_DNS_serv_octetA	UINT	(only if register 8976 = 0) The IPv4 DNS server is specified in the following format: A.B.C.D Octet A
uuiIPv4_DNS_serv_octetB	UINT	(only if register 8976 = 0) The IPv4 DNS server is specified in the following format: A.B.C.D Octet B

uiIPv4_DNS_serv_octetC	UINT	(only if register 8976 = 0) The IPv4 DNS server is specified in the following format: A.B.C.D Octet C
uiIPv4_DNS_serv_octetD	UINT	(only if register 8976 = 0) The IPv4 DNS server is specified in the following format: A.B.C.D Octet D
uiIPv4_add_assign_mode	UINT	0: Static 1: Dynamic (DCHP)
xProficloud_gw_act	BOOL	0: Proficloud gateway is deactivated 1: Proficloud gateway is activated
xProficloud_gw_FTP_act	BOOL	Proficloud gateway FTP service activated.

8.7 EEM_2400_Values

The function block EEM_2400_Values converts the EEM_UDT_2400 input structure into individual output parameters.

8.7.1 Function block call



8.7.2 Input parameters

Name	Type	Description
udtData	EEM_UDT_2400	Input structure for standardized values in the table.

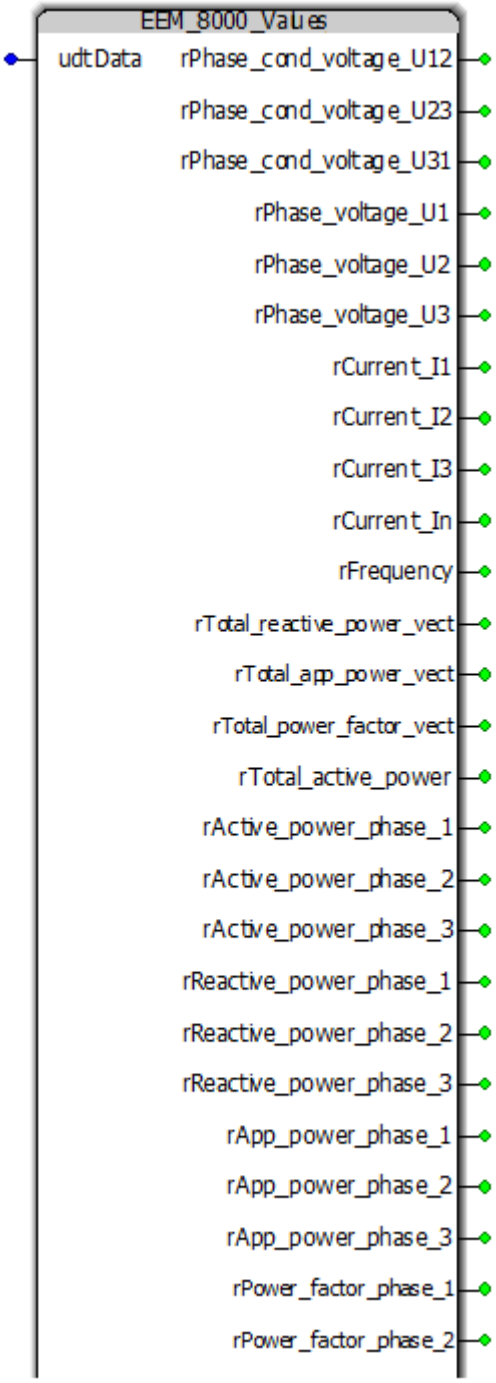
8.7.3 Output parameters

Name	Type	Description
uiIPv4_add_octetA	UINT	(only if register 8976 = 0) The IPv4 address is specified in the following format: A.B.C.D Octet A
uiIPv4_add_octetB	UINT	(only if register 8976 = 0) The IPv4 address is specified in the following format: A.B.C.D Octet B
uiIPv4_add_octetC	UINT	(only if register 8976 = 0) The IPv4 address is specified in the following format: A.B.C.D Octet C
uiIPv4_add_octetD	UINT	(only if register 8976 = 0) The IPv4 address is specified in the following format: A.B.C.D Octet D
uiIPv4_sub_mask_octetA	UINT	(only if register 8976 = 0) The IPv4 subnet mask is specified in the following format: A.B.C.D Octet A
uiIPv4_sub_mask_octetB	UINT	(only if register 8976 = 0) The IPv4 subnet mask is specified in the following format: A.B.C.D Octet B
uiIPv4_sub_mask_octetC	UINT	(only if register 8976 = 0) The IPv4 subnet mask is specified in the following format: A.B.C.D Octet C
uiIPv4_sub_mask_octetD	UINT	(only if register 8976 = 0) The IPv4 subnet mask is specified in the following format: A.B.C.D Octet D
uiIPv4_def_gw_octetA	UINT	(only if register 8976 = 0) The IPv4 default gateway is specified in the following format: A.B.C.D Octet A
uiIPv4_def_gw_octetB	UINT	(only if register 8976 = 0) The IPv4 default gateway is specified in the following format: A.B.C.D Octet B
uiIPv4_def_gw_octetC	UINT	(only if register 8976 = 0) The IPv4 default gateway is specified in the following format: A.B.C.D Octet C
uiIPv4_def_gw_octetD	UINT	(only if register 8976 = 0) The IPv4 default gateway is specified in the following format: A.B.C.D Octet D

8.8 EEM_8000_Values

The function block EEM_8000_Values converts the EEM_UDT_8000 input structure into individual outputs.

8.8.1 Function block call



8.8.2 Input parameters

Name	Type	Description
udtData	EEM_UDT_8000	Input structure for standardized values in the table.

8.8.3 Output parameters

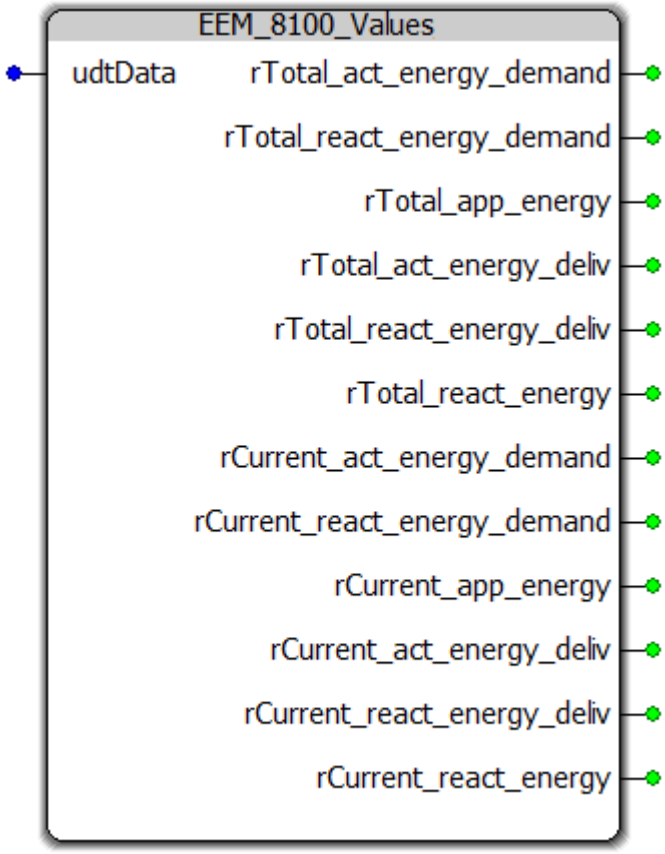
Name	Type	Description
rPhase_cond_voltage_U12	REAL	Effective value voltage U12.
rPhase_cond_voltage_U23	REAL	Effective value voltage U23.
rPhase_cond_voltage_U31	REAL	Effective value voltage U31.
rPhase_voltage_U1	REAL	Effective value voltage U1.
rPhase_voltage_U2	REAL	Effective value voltage U2.
rPhase_voltage_U3	REAL	Effective value voltage U3.
rFrequency	REAL	Frequency.
rCurrent_I1	REAL	Effective value current I1.
rCurrent_I2	REAL	Effective value current I2.
rCurrent_I3	REAL	Effective value current I3.
rCurrent_In	REAL	Effective value current IN, measured or calculated, depending on configuration.
rTotal_active_power	REAL	Sum of active power according to DIN EN 61557-12 with sign. >0: Demand <0: Delivery
rTotal_reactive_power_vect	REAL	Vectorial total reactive power according to DIN EN 61557-12 with sign as sum of the individual reactive powers. >0: Demand <0: Delivery
rTotal_app_power_vect	REAL	Vectorial total apparent power according to DIN EN 61557-12 with sign. >0: Demand <0: Delivery
rTotal_power_factor_vect	REAL	Vectorial total power factor with sign according to IEEE. >0: Inductive <0: Capacitive
rActive_power_phase_1	REAL	Active power with sign on phase 1. >0: Demand <0: Delivery
rActive_power_phase_2	REAL	Active power with sign on phase 2. >0: Demand <0: Delivery
rActive_power_phase_3	REAL	Active power with sign on phase 3. >0: Demand <0: Delivery
rReactive_power_phase_1	REAL	Reactive power on phase 1 (unsigned) >0: Demand <0: Delivery
rReactive_power_phase_2	REAL	Reactive power on phase 2 (unsigned) >0: Demand <0: Delivery
rReactive_power_phase_3	REAL	Reactive power on phase 3 (unsigned) >0: Demand <0: Delivery
rApp_power_phase_1	REAL	Apparent power on phase 1 (unsigned).
rApp_power_phase_2	REAL	Apparent power on phase 2 (unsigned).
rApp_power_phase_3	REAL	Apparent power on phase 3 (unsigned).

rPower_factor_phase_1	REAL	Power factor on phase 1 acc. to IEEE: with sign >0: Demand <0: Delivery acc. to IEC: unsigned
rPower_factor_phase_2	REAL	Power factor on phase 2 acc. to IEEE: with sign >0: Demand <0: Delivery acc. to IEC: unsigned
rPower_factor_phase_3	REAL	Power factor on phase 3 acc. to IEEE: with sign >0: Demand <0: Delivery acc. to IEC: unsigned
iPF_Ind_Cap_1	INT	Indicator of inductive or capacitive behavior on phase 1. -1: Capacitive +1: Inductive 0: Resistive
iPF_Ind_Cap_2	INT	Indicator of inductive or capacitive behavior on phase 2. -1: Capacitive +1: Inductive 0: Resistive
iPF_Ind_Cap_3	INT	Indicator of inductive or capacitive behavior on phase 3. -1: Capacitive +1: Inductive 0: Resistive
rSys_current	REAL	Mean value of effective values of all phase currents.
rExt_cond_voltage_system	REAL	Mean value of effective values of all phase conductor voltages.
rSys_cond_voltage_system	REAL	Mean value of effective values of all phase voltages.
rTotal_react_power_arith	REAL	Arithmetic total reactive power according to DIN EN 61557-12 with sign. >0: Demand <0: Delivery
rTotal_app_power_arith	REAL	Arithmetic total apparent power according to DIN EN 61557-12 with sign. >0: Demand <0: Delivery
rTotal_power_factor_arith	REAL	Arithmethic total power factor with sign acc. to IEEE. >0: Inductive <0: Capacitive
rPHI_UI1	REAL	Phase angle between voltage and current at phase 1.
rPHI_UI2	REAL	Phase angle between voltage and current at phase 2.
rPHI_UI3	REAL	Phase angle between voltage and current at phase 3.
rCos_PHI_1	REAL	Power factor of fundamental wave at phase 1.
rCos_PHI_2	REAL	Power factor of fundamental wave at phase 2.
rCos_PHI_3	REAL	Power factor of fundamental wave at phase 3.
rPHI_U1	REAL	Angle of voltage U1.
rPHI_U2	REAL	Angle of voltage U2.
rPHI_U3	REAL	Angle of voltage U13.
rRotating_field	REAL	0: No rotary field 1: Clockwise rotation (123) -1: Counterclockwise rotation (321)

8.9 EEM_8100_Values

The function block EEM_8100_Values converts the EEM_UDT_8100 input structure into individual outputs.

8.9.1 Function block call



8.9.2 Input parameters

Name	Type	Description
udtData	EEM_UDT_8100	Input structure for standardized values in the table.

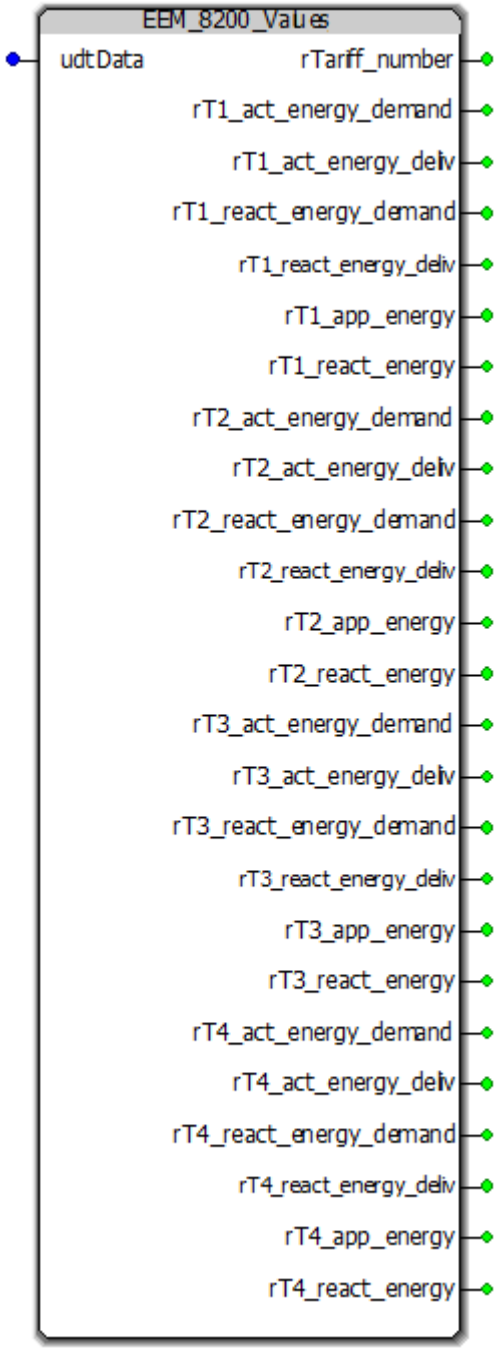
8.9.3 Output parameters

Name	Type	Description
rTotal_act_energy_demand	REAL	Total active energy demand (not resettable).
rTotal_react_energy_demand	REAL	Total reactive energy demand (not resettable).
rTotal_app_energy	REAL	Total apparent energy (not resettable).
rTotal_act_energy_deliv	REAL	Total active energy delivery (not resettable).
rTotal_react_energy_deliv	REAL	Total reactive energy delivery (cannot be reset).
rTotal_react_energy	REAL	Total unsigned reactive energy (not resettable).
rCurrent_act_energy_demand	REAL	Active energy demand since last reset.
rCurrent_react_energy_demand	REAL	Reactive energy demand since last reset.
rCurrent_app_energy	REAL	Apparent energy since last reset.
rCurrent_act_energy_deliv	REAL	Active energy delivery since last reset.
rCurrent_react_energy_deliv	REAL	Reactive energy delivery since last reset.
rCurrent_react_energy	REAL	Unsigned reactive energy since last reset.

8.10 EEM_8200_Values

The function block EEM_8200_Values converts the EEM_UDT_8200 input structure into individual outputs.

8.10.1 Function block call



8.10.2 Input parameters

Name	Type	Description
udtData	EEM_UDT_8100	Input structure for standardized values in the table.

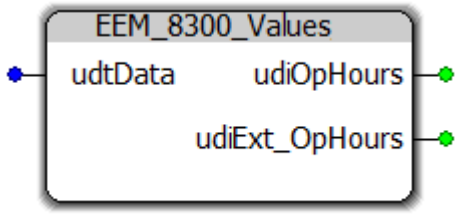
8.10.3 Output parameters

Name	Type	Description
rTariff_number	REAL	Copy of the tariff selection register (cf. register 16512).
rT1_act_energy_demand	REAL	Active energy demand since last reset in tariff 1.
rT1_act_energy_deliv	REAL	Active energy delivery since last reset in tariff 1.
rT1_react_energy_demand	REAL	Reactive energy demand since last reset in tariff 1.
rT1_react_energy_deliv	REAL	Reactive energy delivery since last reset in tariff 1.
rT1_app_energy	REAL	Apparent energy since last reset in tariff 1.
rT1_react_energy	REAL	Unsigned reactive energy since last reset in tariff 1
rT2_act_energy_demand	REAL	Active energy delivery since last reset in tariff 2.
rT2_act_energy_deliv	REAL	Active energy delviery since last reset in tariff 2.
rT2_react_energy_demand	REAL	Reactive energy demand since last reset in tariff 2.
rT2_react_energy_deliv	REAL	Reactive energy delivery since last reset in tariff 2
rT2_app_energy	REAL	Apparent energy since last reset in tariff 2.
rT2_react_energy	REAL	Unsigned reactive energy since last reset in tariff 2.
rT3_act_energy_demand	REAL	Active energy delivery since last reset in tariff 3.
rT3_act_energy_deliv	REAL	Active energy delviery since last reset in tariff 3.
rT3_react_energy_demand	REAL	Reactive energy demand since last reset in tariff 3.
rT3_react_energy_deliv	REAL	Reactive energy delivery since last reset in tariff 3
rT3_app_energy	REAL	Apparent energy since last reset in tariff 3.
rT3_react_energy	REAL	Unsigned reactive energy since last reset in tariff 3.
rT4_act_energy_demand	REAL	Active energy delivery since last reset in tariff 4.
rT4_act_energy_deliv	REAL	Active energy delviery since last reset in tariff 4.
rT4_react_energy_demand	REAL	Reactive energy demand since last reset in tariff 4.
rT4_react_energy_deliv	REAL	Reactive energy delivery since last reset in tariff 4
rT4_app_energy	REAL	Apparent energy since last reset in tariff 4.
rT4_react_energy	REAL	Unsigned reactive energy since last reset in tariff 4.

8.11 EEM_8300_Values

The function block EEM_8300_Values converts the EEM_UDT_8300 input structure into individual outputs.

8.11.1 Function block call



8.11.2 Input parameters

Name	Type	Description
udtData	EEM_UDT_8300	Input structure for standardized values in the table.

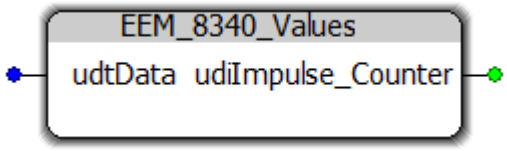
8.11.3 Output parameters

Name	Type	Description
udiOpHours	UDINT	Total operating time since commissioning. Not resettable.
udiExt_OpHours	UDINT	Operating hours counter for external consumers. The counter counts the operating time only from a defined measured power consumption of external loads. This counter can be reset by the user.

8.12 EEM_8340_Values

The function block EEM_8340_Values converts the EEM_UDT_8340 input structure into individual outputs.

8.12.1 Function block call



8.12.2 Input parameters

Name	Type	Description
udtData	EEM_UDT_8340	Input structure for standardized values in the table.

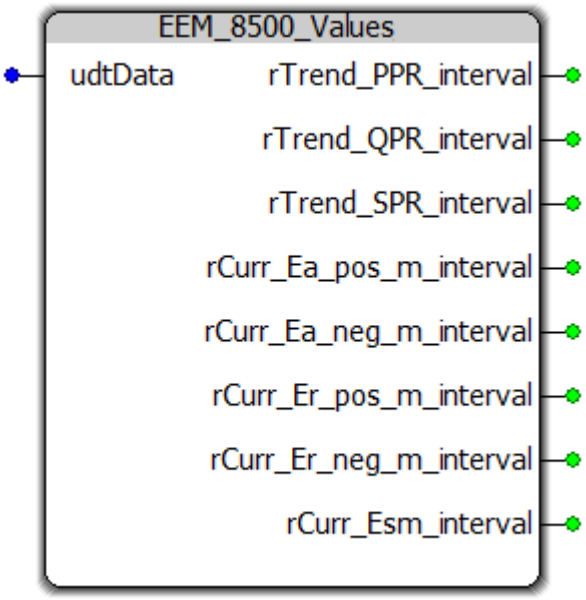
8.12.3 Output parameters

Name	Type	Description
udiImpulse_Counter	UDINT	State of impulse counter of digital input.

8.13 EEM_8500_Values

The function block EEM_8500_Values converts the EEM_UDT_8500 input structure into individual outputs.

8.13.1 Function block call



8.13.2 Input parameters

Name	Type	Description
udtData	EEM_UDT_8500	Input structure for standardized values in the table.

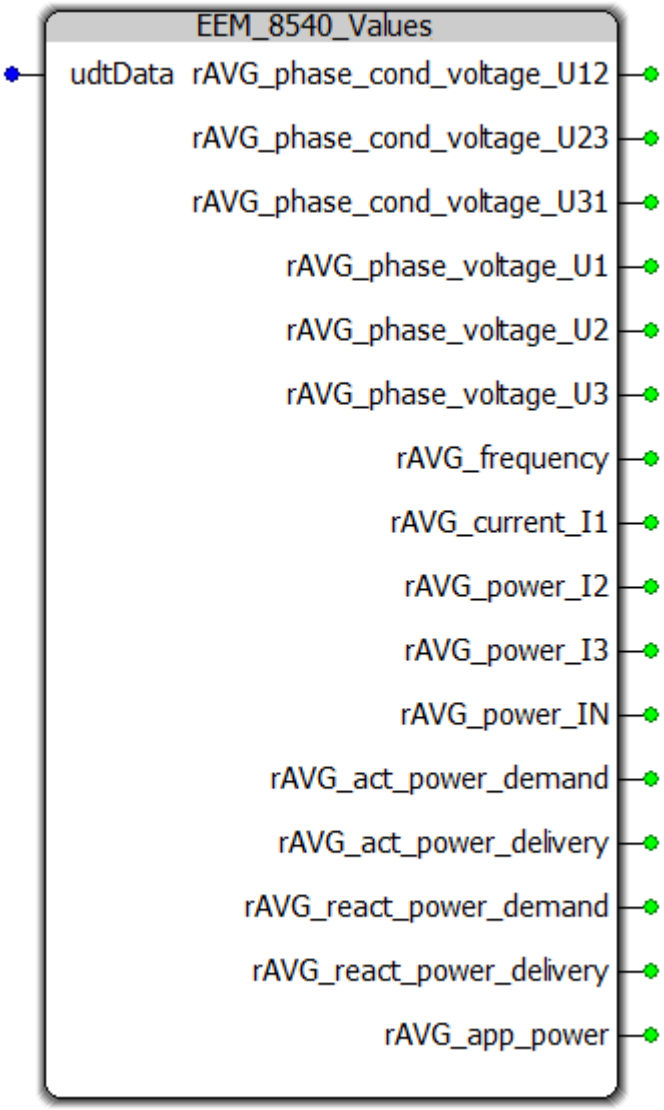
8.13.3 Output parameters

Name	Type	Description
rTrend_PPR_interval	REAL	Active power trend for peak load detection.
rTrend_QPR_interval	REAL	Reactive power trend for peak load detection.
rTrend_SPR_interval	REAL	Apparent power trend for peak load detection.
rCurr_Ea_pos_m_interval	REAL	Active energy demand of the last measuring interval.
rCurr_Ea_neg_m_interval	REAL	Active energy delivery of the last measuring interval
rCurr_Er_pos_m_interval	REAL	Reactive energy demand of the last measuring interval.
rCurr_Er_neg_m_interval	REAL	Reactive energy delivery of the last measuring interval.
rCurr_Esm_interval	REAL	Apparent energy of the last measuring interval.

8.14 EEM_8540_Values

The function block EEM_8540_Values converts the EEM_UDT_8540 input structure into individual outputs.

8.14.1 Function block call



8.14.2 Input parameters

Name	Type	Description
udtData	EEM_UDT_8540	Input structure for standardized values in the table.

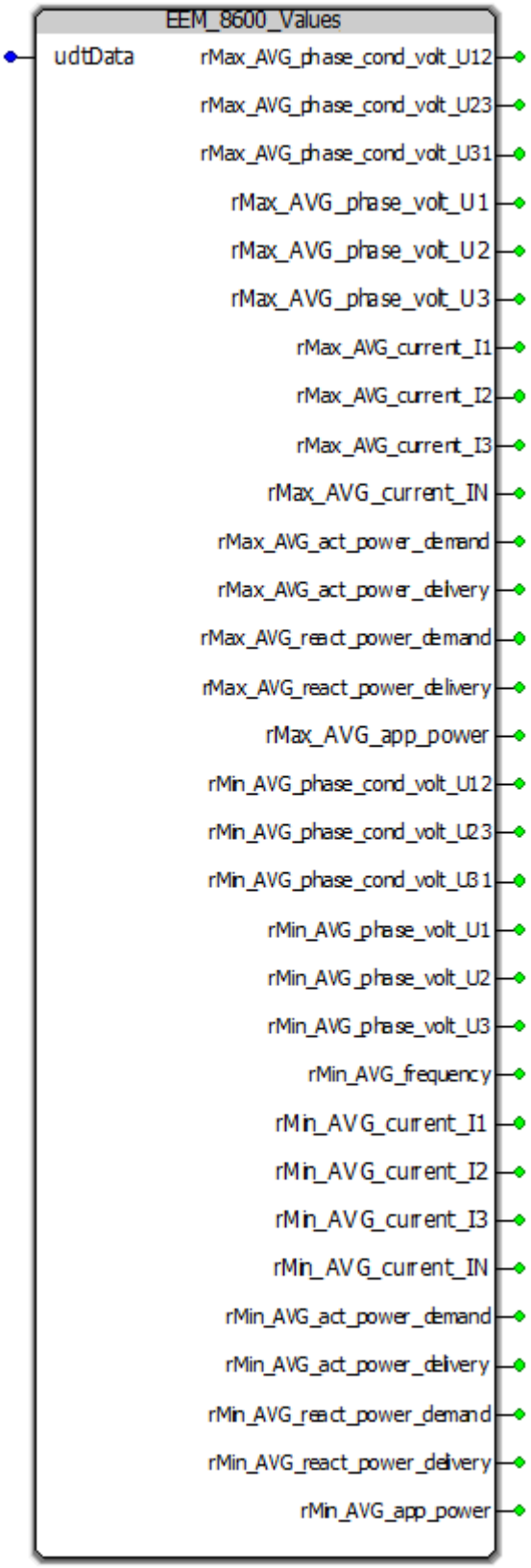
8.14.3 Output parameters

Name	Type	Description
rAVG_phase_cond_voltage_U12	REAL	AVG value of voltage U12.
rAVG_phase_cond_voltage_U23	REAL	AVG value of voltage U23.
rAVG_phase_cond_voltage_U31	REAL	AVG value of voltage U31.
rAVG_phase_voltage_U1	REAL	AVG value of voltage U1.
rAVG_phase_voltage_U2	REAL	AVG value of voltage U2.
rAVG_phase_voltage_U3	REAL	AVG value of voltage U3.
rAVG_frequency	REAL	AVG value of the frequency.
rAVG_current_I1	REAL	AVG value of current I1.
rAVG_power_I2	REAL	AVG value of current I2.
rAVG_power_I3	REAL	AVG value of current I3.
rAVG_power_IN	REAL	AVG value of current IN.
rAVG_act_power_demand	REAL	AVG value of the total active power demand.
rAVG_act_power_delivery	REAL	AVG value of the total active power delivery.
rAVG_react_power_demand	REAL	AVG value of the total reactive power demand.
rAVG_react_power_delivery	REAL	AVG value of the total reactive power delivery.
rAVG_app_power	REAL	AVG value of the total apparent power.

8.15 EEM_8600_Values

The function block EEM_8600_Values converts the EEM_UDT_8600 input structure into individual outputs.

8.15.1 Function block call



8.15.2 Input parameters

Name	Type	Description
udtData	EEM_UDT_8600	Input structure for standardized values in the table.

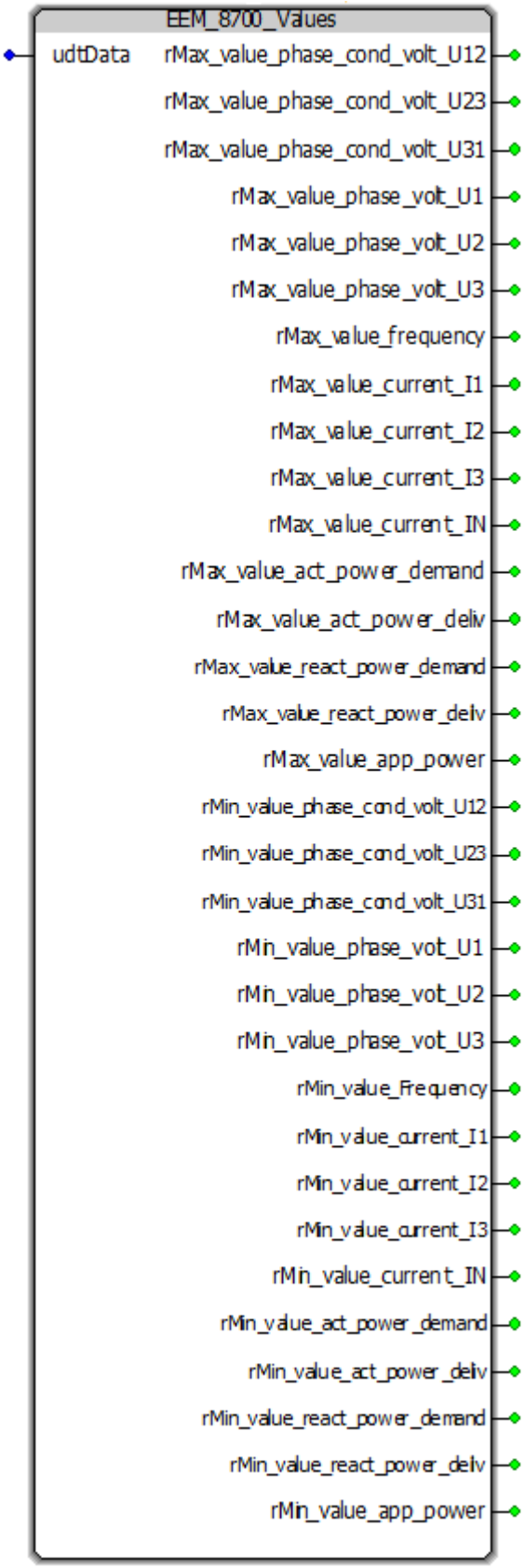
8.15.3 Output parameters

Name	Type	Description
rMax_AVG_phase_cond_volt_U12	REAL	Maximum AVG value of voltage U12.
rMax_AVG_phase_cond_volt_U23	REAL	Maximum AVG value of voltage U23.
rMax_AVG_phase_cond_volt_U31	REAL	Maximum AVG value of voltage U31.
rMax_AVG_phase_volt_U1	REAL	Maximum AVG value of voltage U1.
rMax_AVG_phase_volt_U2	REAL	Maximum AVG value of voltage U2.
rMax_AVG_phase_volt_U3	REAL	Maximum AVG value of voltage U23.
rMax_AVG_frequency	REAL	Maximum AVG value of the frequency.
rMax_AVG_current_I1	REAL	Maximum AVG value of current I1.
rMax_AVG_current_I2	REAL	Maximum AVG value of current I2.
rMax_AVG_current_I3	REAL	Maximum AVG value of current I3.
rMax_AVG_current_IN	REAL	Maximum AVG value of current IN.
rMax_AVG_act_power_demand	REAL	Maximum AVG value of the total active power demand.
rMax_AVG_act_power_delivery	REAL	Maximum AVG value of the total active power delivery.
rMax_AVG_react_power_demand	REAL	Maximum AVG value of the total reactive power demand.
rMax_AVG_react_power_delivery	REAL	Maximum AVG value of the total reactive power delivery.
rMax_AVG_app_power	REAL	Maximum AVG value of the total apparent power
rMin_AVG_phase_cond_volt_U12	REAL	Minimum AVG value of voltage U12.
rMin_AVG_phase_cond_volt_U23	REAL	Minimum AVG value of voltage U23.
rMin_AVG_phase_cond_volt_U31	REAL	Minimum AVG value of voltage U31.
rMin_AVG_phase_volt_U1	REAL	Minimum AVG value of voltage U1.
rMin_AVG_phase_volt_U2	REAL	Minimum AVG value of voltage U2.
rMin_AVG_phase_volt_U3	REAL	Minimum AVG value of voltage U23.
rMin_AVG_frequency	REAL	Minimum AVG value of the frequency.
rMin_AVG_current_I1	REAL	Minimum AVG value of current I1.
rMin_AVG_current_I2	REAL	Minimum AVG value of current I2.
rMin_AVG_current_I3	REAL	Minimum AVG value of current I3.
rMin_AVG_current_IN	REAL	Minimum AVG value of current IN.
rMin_AVG_act_power_demand	REAL	Minimum AVG value of the total active power demand.
rMin_AVG_act_power_delivery	REAL	Minimum AVG value of the total active power delivery.
rMin_AVG_react_power_demand	REAL	Minimum AVG value of the total reactive power demand.
rMin_AVG_react_power_delivery	REAL	Minimum AVG value of the total reactive power delivery.
rMin_AVG_app_power	REAL	Minimum AVG value of the total apparent power

8.16 EEM_8700_Values

The function block EEM_8700_Values converts the EEM_UDT_8700 input structure into individual outputs.

8.16.1 Function block call



8.16.2 Input parameters

Name	Type	Description
udtData	EEM_UDT_8700	Input structure for standardized values in the table.

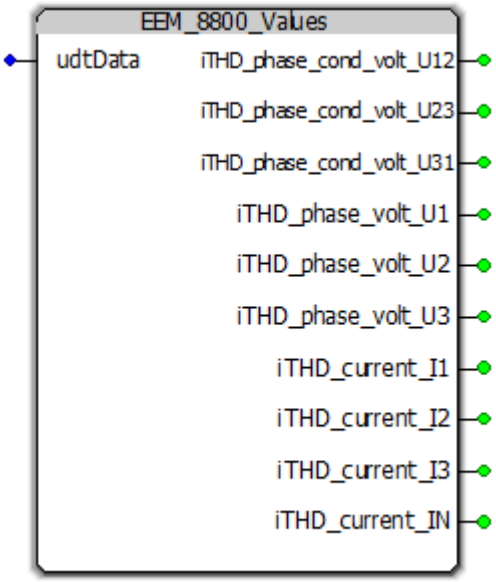
8.16.3 Output parameters

Name	Type	Description
rMax_value_phase_cond_volt_U12	REAL	Maximum value of voltage U12.
rMax_value_phase_cond_volt_U23	REAL	Maximum value of voltage U23.
rMax_value_phase_cond_volt_U31	REAL	Maximum value of voltage U31.
rMax_value_phase_volt_U1	REAL	Maximum value of voltage U1.
rMax_value_phase_volt_U2	REAL	Maximum value of voltage U2.
rMax_value_phase_volt_U3	REAL	Maximum value of voltage U3.
rMax_value_frequency	REAL	Maximum value of frequency.
rMax_value_current_I1	REAL	Maximum value of current I1.
rMax_value_current_I2	REAL	Maximum value of current I2.
rMax_value_current_I3	REAL	Maximum value of current I3.
rMax_value_current_IN	REAL	Maximum value of current IN.
rMax_value_act_power_demand	REAL	Maximum value of the total active power demand.
rMax_value_act_power_deliv	REAL	Maximum value of the total active power delivery.
rMax_value_react_power_demand	REAL	Maximum value of the total reactive power demand.
rMax_value_react_power_deliv	REAL	Maximum value of the total reactive power delivery.
rMax_value_app_power	REAL	Maximum value of the total apparent power.
rMin_value_phase_cond_volt_U12	REAL	Minimum value of voltage U12.
rMin_value_phase_cond_volt_U23	REAL	Minimum value of voltage U23.
rMin_value_phase_cond_volt_U31	REAL	Minimum value of voltage U31.
rMin_value_phase_volt_U1	REAL	Minimum value of voltage U1.
rMin_value_phase_volt_U2	REAL	Minimum value of voltage U2.
rMin_value_phase_volt_U3	REAL	Minimum value of voltage U3.
rMin_value_Frequency	REAL	Minimum value of frequency.
rMin_value_current_I1	REAL	Minimum value of current I1.
rMin_value_current_I2	REAL	Minimum value of current I2.
rMin_value_current_I3	REAL	Minimum value of current I3.
rMin_value_current_IN	REAL	Minimum value of current IN.
rMin_value_act_power_demand	REAL	Minimum value of the total active power demand.
rMin_value_act_power_deliv	REAL	Minimum value of the total active power delivery.
rMin_value_react_power_demand	REAL	Minimum value of the total reactive power demand.
rMin_value_react_power_deliv	REAL	Minimum value of the total reactive power delivery.
rMin_value_app_power	REAL	Minimum value of the total apparent power.

8.17 EEM_8800_Values

The function block EEM_8800_Values converts the EEM_UDT_8800 input structure into individual outputs.

8.17.1 Function block call



8.17.2 Input parameters

Name	Type	Description
udtData	EEM_UDT_8800	Input structure for standardized values in the table.

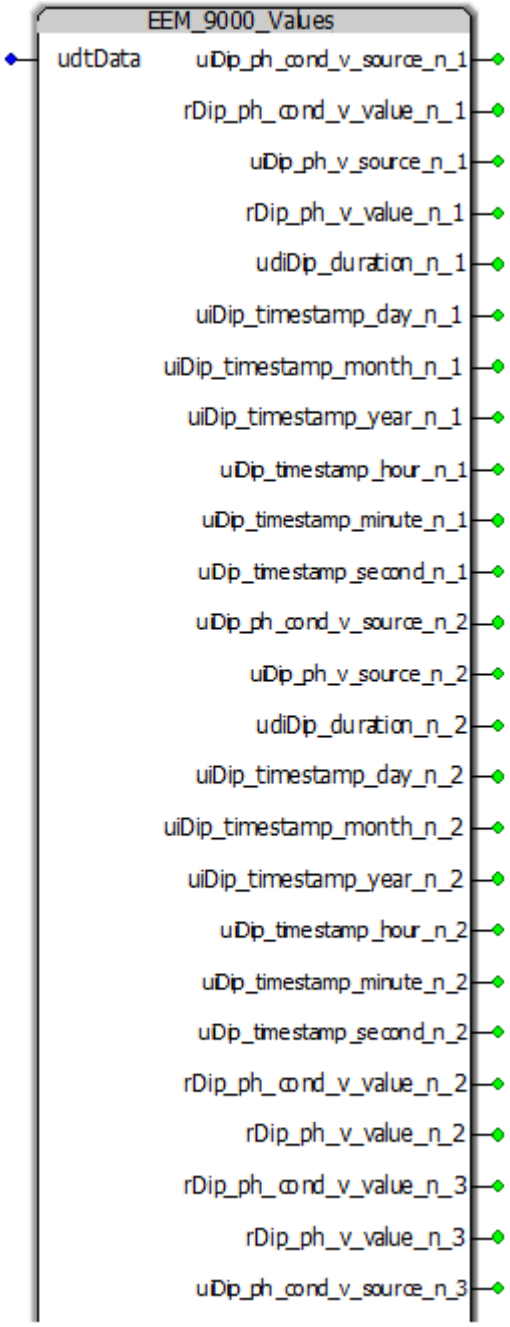
8.17.3 Output parameters

Name	Type	Description
iTHD_phase_cond_volt_U12	INT	Total harmonic distortion of the phase conductor voltage U12.
iTHD_phase_cond_volt_U23	INT	Total harmonic distortion of the phase conductor voltage U23.
iTHD_phase_cond_volt_U31	INT	Total harmonic distortion of the phase conductor voltage U31.
iTHD_phase_volt_U1	INT	Total harmonic distortion of the phase voltage U1.
iTHD_phase_volt_U2	INT	Total harmonic distortion of the phase voltage U2.
iTHD_phase_volt_U3	INT	Total harmonic distortion of the phase voltage U3.
iTHD_current_I1	INT	Total harmonic distortion of current I1.
iTHD_current_I2	INT	Total harmonic distortion of current I2.
iTHD_current_I3	INT	Total harmonic distortion of current I3.
iTHD_current_IN	INT	Total harmonic distortion of current IN.

8.18 EEM_9000_Values

The function block EEM_9000_Values converts the EEM_UDT_9000 input structure into individual outputs.

8.18.1 Function block call



8.18.2 Input parameters

Name	Type	Description
udtData	EEM_UDT_9000	Input structure for standardized values in the table.

8.18.3 Output parameters

Name	Type	Description
uiDip_ph_cond_v_source_n_1	UINT	Cause of voltage dips (Udip) at the phase voltages. Bit mask: Bit 0: U1 Bit 1: U2 Bit 2: U3
rDip_ph_cond_v_value_n_1	REAL	Lowest value of phase conductor voltages during voltage dip.
uiDip_ph_v_source_n_1	UINT	Cause of voltage dips (Udip) at the phase voltages. Bit mask: Bit 0: U1 Bit 1: U2 Bit 2: U3
rDip_ph_v_value_n_1	REAL	Lowest value of phase voltages during voltage dip.
udiDip_duration_n_1	UDINT	Duration of the voltage dip in milliseconds.
uiDip_timestamp_day_n_1	UINT	Day of occurrence of voltage dip (UTC).
uiDip_timestamp_month_n_1	UINT	Month of occurrence of voltage dip (UTC).
uiDip_timestamp_year_n_1	UINT	Year of occurrence of the voltage dip (UTC).
uiDip_timestamp_hour_n_1	UINT	Hour of occurrence of voltage dip (UTC).
uiDip_timestamp_minute_n_1	UINT	Minute of occurrence of voltage dip (UTC).
uiDip_timestamp_second_n_1	UINT	Second of occurrence of voltage dip (UTC).
uiDip_ph_cond_v_source_n_2	UINT	Cause of voltage dips (Udip) at the phase voltages. Bit mask: Bit 0: U1 Bit 1: U2 Bit 2: U3
rDip_ph_cond_v_value_n_2	REAL	Lowest value of phase conductor voltages during voltage dip.
uiDip_ph_v_source_n_2	UINT	Cause of voltage dips (Udip) at the phase voltages. Bit mask: Bit 0: U1 Bit 1: U2 Bit 2: U3
rDip_ph_v_value_n_2	REAL	Lowest value of phase voltages during voltage dip.
udiDip_duration_n_2	UDINT	Duration of the voltage dip in milliseconds.
uiDip_timestamp_day_n_2	UINT	Day of occurrence of voltage dip (UTC).
uiDip_timestamp_month_n_2	UINT	Month of occurrence of voltage dip (UTC).
uiDip_timestamp_year_n_2	UINT	Year of occurrence of the voltage dip (UTC).
uiDip_timestamp_hour_n_2	UINT	Hour of occurrence of voltage dip (UTC).
uiDip_timestamp_minute_n_2	UINT	Minute of occurrence of voltage dip (UTC).
uiDip_timestamp_second_n_2	UINT	Second of occurrence of voltage dip (UTC).

uiDip_ph_cond_v_source_n_3	UINT	Cause of voltage dips (Udip) at the phase voltages. Bit mask: Bit 0: U1 Bit 1: U2 Bit 2: U3
rDip_ph_cond_v_value_n_3	REAL	Lowest value of phase conductor voltages during voltage dip.
uiDip_ph_v_source_n_3	UINT	Cause of voltage dips (Udip) at the phase voltages. Bit mask: Bit 0: U1 Bit 1: U2 Bit 2: U3
rDip_ph_v_value_n_3	REAL	Lowest value of phase voltages during voltage dip.
udiDip_duration_n_3	UDINT	Duration of the voltage dip in milliseconds.
uiDip_timestamp_day_n_3	UINT	Day of occurrence of voltage dip (UTC).
uiDip_timestamp_month_n_3	UINT	Month of occurrence of voltage dip (UTC).
uiDip_timestamp_year_n_3	UINT	Year of occurrence of the voltage dip (UTC).
uiDip_timestamp_hour_n_3	UINT	Hour of occurrence of voltage dip (UTC).
uiDip_timestamp_minute_n_3	UINT	Minute of occurrence of voltage dip (UTC).
uiDip_timestamp_second_n_3	UINT	Second of occurrence of voltage dip (UTC).
uiDip_ph_cond_v_source_n_4	UINT	Cause of voltage dips (Udip) at the phase voltages. Bit mask: Bit 0: U1 Bit 1: U2 Bit 2: U3
rDip_ph_cond_v_value_n_4	REAL	Lowest value of phase conductor voltages during voltage dip.
uiDip_ph_v_source_n_4	UINT	Cause of voltage dips (Udip) at the phase voltages. Bit mask: Bit 0: U1 Bit 1: U2 Bit 2: U3
rDip_ph_v_value_n_4	REAL	Lowest value of phase voltages during voltage dip.
udiDip_duration_n_4	UDINT	Duration of the voltage dip in milliseconds.
uiDip_timestamp_day_n_4	UINT	Day of occurrence of voltage dip (UTC).
uiDip_timestamp_month_n_4	UINT	Month of occurrence of voltage dip (UTC).
uiDip_timestamp_year_n_4	UINT	Year of occurrence of the voltage dip (UTC).
uiDip_timestamp_hour_n_4	UINT	Hour of occurrence of voltage dip (UTC).
uiDip_timestamp_minute_n_4	UINT	Minute of occurrence of voltage dip (UTC).
uiDip_timestamp_second_n_4	UINT	Second of occurrence of voltage dip (UTC).
uiDip_ph_cond_v_source_n_5	UINT	Cause of voltage dips (Udip) at the phase voltages. Bit mask: Bit 0: U1 Bit 1: U2 Bit 2: U3
rDip_ph_cond_v_value_n_5	REAL	Lowest value of phase conductor voltages during voltage dip.
uiDip_ph_v_source_n_5	UINT	Cause of voltage dips (Udip) at the phase voltages. Bit mask: Bit 0: U1 Bit 1: U2 Bit 2: U3
rDip_ph_v_value_n_5	REAL	Lowest value of phase voltages during voltage dip.
udiDip_duration_n_5	UDINT	Duration of the voltage dip in milliseconds.

uiDip_timestamp_day_n_5	UINT	Day of occurrence of voltage dip (UTC).
uiDip_timestamp_month_n_5	UINT	Month of occurrence of voltage dip (UTC).
uiDip_timestamp_year_n_5	UINT	Year of occurrence of the voltage dip (UTC).
uiDip_timestamp_hour_n_5	UINT	Hour of occurrence of voltage dip (UTC).
uiDip_timestamp_minute_n_5	UINT	Minute of occurrence of voltage dip (UTC).
uiDip_timestamp_second_n_5	UINT	Second of occurrence of voltage dip (UTC).

8.19 EEM_9046_Values

The function block EEM_9046_Values converts the EEM_UDT_9046 input structure into individual outputs.

8.19.1 Function block call



8.19.2 Input parameters

Name	Type	Description
udtData	EEM_UDT_9046	Input structure for standardized values in the table.

8.19.3 Output parameters

Name	Type	Description
uiDip_ph_cond_v_source_n_6	UINT	Cause of voltage dips (Udip) at the phase voltages. Bit mask: Bit 0: U1 Bit 1: U2 Bit 2: U3
rDip_ph_cond_v_value_n_6	REAL	Lowest value of phase conductor voltages during voltage dip.
uiDip_ph_v_source_n_6	UINT	Cause of voltage dips (Udip) at the phase voltages. Bit mask: Bit 0: U1 Bit 1: U2 Bit 2: U3
rDip_ph_v_value_n_6	REAL	Lowest value of phase voltages during voltage dip.
udiDip_duration_n_6	UDINT	Duration of the voltage dip in milliseconds.
uiDip_timestamp_day_n_6	UINT	Day of occurrence of voltage dip (UTC).
uiDip_timestamp_month_n_6	UINT	Month of occurrence of voltage dip (UTC).
uiDip_timestamp_year_n_6	UINT	Year of occurrence of the voltage dip (UTC).
uiDip_timestamp_hour_n_6	UINT	Hour of occurrence of voltage dip (UTC).
uiDip_timestamp_minute_n_6	UINT	Minute of occurrence of voltage dip (UTC).
uiDip_timestamp_second_n_6	UINT	Second of occurrence of voltage dip (UTC).
uiDip_ph_cond_v_source_n_7	UINT	Cause of voltage dips (Udip) at the phase voltages. Bit mask: Bit 0: U1 Bit 1: U2 Bit 2: U3
rDip_ph_cond_v_value_n_7	REAL	Lowest value of phase conductor voltages during voltage dip.
uiDip_ph_v_source_n_7	UINT	Cause of voltage dips (Udip) at the phase voltages. Bit mask: Bit 0: U1 Bit 1: U2 Bit 2: U3
rDip_ph_v_value_n_7	REAL	Lowest value of phase voltages during voltage dip.
udiDip_duration_n_7	UDINT	Duration of the voltage dip in milliseconds.
uiDip_timestamp_day_n_7	UINT	Day of occurrence of voltage dip (UTC).
uiDip_timestamp_month_n_7	UINT	Month of occurrence of voltage dip (UTC).
uiDip_timestamp_year_n_7	UINT	Year of occurrence of the voltage dip (UTC).
uiDip_timestamp_hour_n_7	UINT	Hour of occurrence of voltage dip (UTC).
uiDip_timestamp_minute_n_7	UINT	Minute of occurrence of voltage dip (UTC).
uiDip_timestamp_second_n_7	UINT	Second of occurrence of voltage dip (UTC).

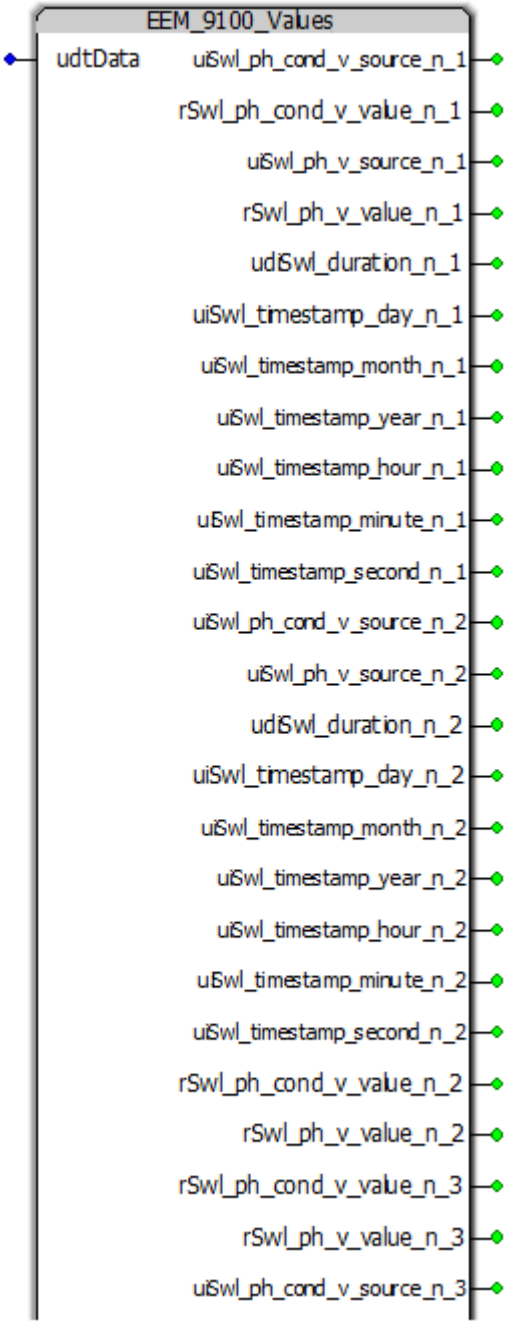
uiDip_ph_cond_v_source_n_8	UINT	Cause of voltage dips (Udip) at the phase voltages. Bit mask: Bit 0: U1 Bit 1: U2 Bit 2: U3
rDip_ph_cond_v_value_n_8	REAL	Lowest value of phase conductor voltages during voltage dip.
uiDip_ph_v_source_n_8	UINT	Cause of voltage dips (Udip) at the phase voltages. Bit mask: Bit 0: U1 Bit 1: U2 Bit 2: U3
rDip_ph_v_value_n_8	REAL	Lowest value of phase voltages during voltage dip.
udiDip_duration_n_8	UDINT	Duration of the voltage dip in milliseconds.
uiDip_timestamp_day_n_8	UINT	Day of occurrence of voltage dip (UTC).
uiDip_timestamp_month_n_8	UINT	Month of occurrence of voltage dip (UTC).
uiDip_timestamp_year_n_8	UINT	Year of occurrence of the voltage dip (UTC).
uiDip_timestamp_hour_n_8	UINT	Hour of occurrence of voltage dip (UTC).
uiDip_timestamp_minute_n_8	UINT	Minute of occurrence of voltage dip (UTC).
uiDip_timestamp_second_n_8	UINT	Second of occurrence of voltage dip (UTC).
uiDip_ph_cond_v_source_n_9	UINT	Cause of voltage dips (Udip) at the phase voltages. Bit mask: Bit 0: U1 Bit 1: U2 Bit 2: U3
rDip_ph_cond_v_value_n_9	REAL	Lowest value of phase conductor voltages during voltage dip.
uiDip_ph_v_source_n_9	UINT	Cause of voltage dips (Udip) at the phase voltages. Bit mask: Bit 0: U1 Bit 1: U2 Bit 2: U3
rDip_ph_v_value_n_9	REAL	Lowest value of phase voltages during voltage dip.
udiDip_duration_n_9	UDINT	Duration of the voltage dip in milliseconds.
uiDip_timestamp_day_n_9	UINT	Day of occurrence of voltage dip (UTC).
uiDip_timestamp_month_n_9	UINT	Month of occurrence of voltage dip (UTC).
uiDip_timestamp_year_n_9	UINT	Year of occurrence of the voltage dip (UTC).
uiDip_timestamp_hour_n_9	UINT	Hour of occurrence of voltage dip (UTC).
uiDip_timestamp_minute_n_9	UINT	Minute of occurrence of voltage dip (UTC).
uiDip_timestamp_second_n_9	UINT	Second of occurrence of voltage dip (UTC).
uiDip_ph_cond_v_source_n_10	UINT	Cause of voltage dips (Udip) at the phase voltages. Bit mask: Bit 0: U1 Bit 1: U2 Bit 2: U3
rDip_ph_cond_v_value_n_10	REAL	Lowest value of phase conductor voltages during voltage dip.
uiDip_ph_v_source_n_10	UINT	Cause of voltage dips (Udip) at the phase voltages. Bit mask: Bit 0: U1 Bit 1: U2 Bit 2: U3
rDip_ph_v_value_n_10	REAL	Lowest value of phase voltages during voltage dip.
udiDip_duration_n_10	UDINT	Duration of the voltage dip in milliseconds.

uiDip_timestamp_day_n_10	UINT	Day of occurrence of voltage dip (UTC).
uiDip_timestamp_month_n_10	UINT	Month of occurrence of voltage dip (UTC).
uiDip_timestamp_year_n_10	UINT	Year of occurrence of the voltage dip (UTC).
uiDip_timestamp_hour_n_10	UINT	Hour of occurrence of voltage dip (UTC).
uiDip_timestamp_minute_n_10	UINT	Minute of occurrence of voltage dip (UTC).
uiDip_timestamp_second_n_10	UINT	Second of occurrence of voltage dip (UTC).

8.20 EEM_9100_Values

The function block EEM_9100_Values converts the EEM_UDT_9100 input structure into individual outputs.

8.20.1 Function block call



8.20.2 Input parameters

Name	Type	Description
udtData	EEM_UDT_9100	Input structure for standardized values in the table.

8.20.3 Output parameters

Name	Type	Description
uiSwl_ph_cond_v_source_n_1	UINT	Cause of voltage swells (Uswl) at the phase voltages. Bit mask: Bit 0: U1 Bit 1: U2 Bit 2: U3
rSwl_ph_cond_v_value_n_1	REAL	Lowest value of phase conductor voltages during voltage swell.
uiSwl_ph_v_source_n_1	UINT	Cause of voltage swells (Uswl) at the phase voltages. Bit mask: Bit 0: U1 Bit 1: U2 Bit 2: U3
rSwl_ph_v_value_n_1	REAL	Lowest value of phase voltages during voltage swell.
udiSwl_duration_n_1	UDINT	Duration of the voltage swell in milliseconds.
uiSwl_timestamp_day_n_1	UINT	Day of occurrence of voltage swell (UTC).
uiSwl_timestamp_month_n_1	UINT	Month of occurrence of voltage swell (UTC).
uiSwl_timestamp_year_n_1	UINT	Year of occurrence of the voltage swell (UTC).
uiSwl_timestamp_hour_n_1	UINT	Hour of occurrence of voltage swell (UTC).
uiSwl_timestamp_minute_n_1	UINT	Minute of occurrence of voltage swell (UTC).
uiSwl_timestamp_second_n_1	UINT	Second of occurrence of voltage swell (UTC).
rSwl_ph_cond_v_value_n_2	REAL	Lowest value of phase conductor voltages during voltage swell.
uiSwl_ph_v_source_n_2	UINT	Cause of voltage swells (Uswl) at the phase voltages. Bit mask: Bit 0: U1 Bit 1: U2 Bit 2: U3
rSwl_ph_v_value_n_2	REAL	Lowest value of phase voltages during voltage swell.
uiSwl_ph_v_source_n_2	UINT	Cause of voltage swells (Uswl) at the phase voltages. Bit mask: Bit 0: U1 Bit 1: U2 Bit 2: U3
udiSwl_duration_n_2	UDINT	Duration of the voltage swell in milliseconds.
uiSwl_timestamp_day_n_2	UINT	Day of occurrence of voltage swell (UTC).
uiSwl_timestamp_month_n_2	UINT	Month of occurrence of voltage swell (UTC).
uiSwl_timestamp_year_n_2	UINT	Year of occurrence of the voltage swell (UTC).
uiSwl_timestamp_hour_n_2	UINT	Hour of occurrence of voltage swell (UTC).
uiSwl_timestamp_minute_n_2	UINT	Minute of occurrence of voltage swell (UTC).
uiSwl_timestamp_second_n_2	UINT	Second of occurrence of voltage swell (UTC).
rSwl_ph_cond_v_value_n_3	REAL	Lowest value of phase conductor voltages during voltage swell.

uiSwl_ph_v_source_n_3	UINT	Cause of voltage swells (Uswl) at the phase voltages. Bit mask: Bit 0: U1 Bit 1: U2 Bit 2: U3
rSwl_ph_v_value_n_3	REAL	Lowest value of phase voltages during voltage swell.
uiSwl_ph_v_source_n_3	UINT	Cause of voltage swells (Uswl) at the phase voltages. Bit mask: Bit 0: U1 Bit 1: U2 Bit 2: U3
udiSwl_duration_n_3	UDINT	Duration of the voltage swell in milliseconds.
uiSwl_timestamp_day_n_3	UINT	Day of occurrence of voltage swell (UTC).
uiSwl_timestamp_month_n_3	UINT	Month of occurrence of voltage swell (UTC).
uiSwl_timestamp_year_n_3	UINT	Year of occurrence of the voltage swell (UTC).
uiSwl_timestamp_hour_n_3	UINT	Hour of occurrence of voltage swell (UTC).
uiSwl_timestamp_minute_n_3	UINT	Minute of occurrence of voltage swell (UTC).
uiSwl_timestamp_second_n_3	UINT	Second of occurrence of voltage swell (UTC).
rSwl_ph_cond_v_value_n_4	REAL	Lowest value of phase conductor voltages during voltage swell.
uiSwl_ph_v_source_n_4	UINT	Cause of voltage swells (Uswl) at the phase voltages. Bit mask: Bit 0: U1 Bit 1: U2 Bit 2: U3
rSwl_ph_v_value_n_4	REAL	Lowest value of phase voltages during voltage swell.
uiSwl_ph_v_source_n_4	UINT	Cause of voltage swells (Uswl) at the phase voltages. Bit mask: Bit 0: U1 Bit 1: U2 Bit 2: U3
udiSwl_duration_n_4	UDINT	Duration of the voltage swell in milliseconds.
uiSwl_timestamp_day_n_4	UINT	Day of occurrence of voltage swell (UTC).
uiSwl_timestamp_month_n_4	UINT	Month of occurrence of voltage swell (UTC).
uiSwl_timestamp_year_n_4	UINT	Year of occurrence of the voltage swell (UTC).
uiSwl_timestamp_hour_n_4	UINT	Hour of occurrence of voltage swell (UTC).
uiSwl_timestamp_minute_n_4	UINT	Minute of occurrence of voltage swell (UTC).
uiSwl_timestamp_second_n_4	UINT	Second of occurrence of voltage swell (UTC).
rSwl_ph_cond_v_value_n_5	REAL	Lowest value of phase conductor voltages during voltage swell.
uiSwl_ph_v_source_n_5	UINT	Cause of voltage swells (Uswl) at the phase voltages. Bit mask: Bit 0: U1 Bit 1: U2 Bit 2: U3
rSwl_ph_v_value_n_5	REAL	Lowest value of phase voltages during voltage swell.
uiSwl_ph_v_source_n_5	UINT	Cause of voltage swells (Uswl) at the phase voltages. Bit mask: Bit 0: U1 Bit 1: U2 Bit 2: U3
udiSwl_duration_n_5	UDINT	Duration of the voltage swell in milliseconds.
uiSwl_timestamp_day_n_5	UINT	Day of occurrence of voltage swell (UTC).

uiSwl_timestamp_month_n_5	UINT	Month of occurrence of voltage swell (UTC).
uiSwl_timestamp_year_n_5	UINT	Year of occurrence of the voltage swell (UTC).
uiSwl_timestamp_hour_n_5	UINT	Hour of occurrence of voltage swell (UTC).
uiSwl_timestamp_minute_n_5	UINT	Minute of occurrence of voltage swell (UTC).
uiSwl_timestamp_second_n_5	UINT	Second of occurrence of voltage swell (UTC).

8.21 EEM_9146_Values

The function block EEM_9146_Values converts the EEM_UDT_9146 input structure into individual outputs.

8.21.1 Function block call



8.21.2 Input parameters

Name	Type	Description
udtData	EEM_UDT_9146	Input structure for standardized values in the table.

8.21.3 Output parameters

Name	Type	Description
uiDip_ph_cond_v_source_n_6	UINT	Cause of voltage dips (Udip) at the phase voltages. Bit mask: Bit 0: U1 Bit 1: U2 Bit 2: U3
rDip_ph_cond_v_value_n_6	REAL	Lowest value of phase conductor voltages during voltage dip.
uiDip_ph_v_source_n_6	UINT	Cause of voltage dips (Udip) at the phase voltages. Bit mask: Bit 0: U1 Bit 1: U2 Bit 2: U3
rDip_ph_v_value_n_6	REAL	Lowest value of phase voltages during voltage dip.
udiDip_duration_n_6	UDINT	Duration of the voltage dip in milliseconds.
uiDip_timestamp_day_n_6	UINT	Day of occurrence of voltage dip (UTC).
uiDip_timestamp_month_n_6	UINT	Month of occurrence of voltage dip (UTC).
uiDip_timestamp_year_n_6	UINT	Year of occurrence of the voltage dip (UTC).
uiDip_timestamp_hour_n_6	UINT	Hour of occurrence of voltage dip (UTC).
uiDip_timestamp_minute_n_6	UINT	Minute of occurrence of voltage dip (UTC).
uiDip_timestamp_second_n_6	UINT	Second of occurrence of voltage dip (UTC).
rDip_ph_cond_v_value_n_7	REAL	Lowest value of phase conductor voltages during voltage dip.
uiDip_ph_v_source_n_7	UINT	Cause of voltage dips (Udip) at the phase voltages. Bit mask: Bit 0: U1 Bit 1: U2 Bit 2: U3
rDip_ph_v_value_n_7	REAL	Lowest value of phase voltages during voltage dip.
uiDip_ph_v_source_n_7	UINT	Cause of voltage dips (Udip) at the phase voltages. Bit mask: Bit 0: U1 Bit 1: U2 Bit 2: U3
udiDip_duration_n_7	UDINT	Duration of the voltage dip in milliseconds.
uiDip_timestamp_day_n_7	UINT	Day of occurrence of voltage dip (UTC).
uiDip_timestamp_month_n_7	UINT	Month of occurrence of voltage dip (UTC).
uiDip_timestamp_year_n_7	UINT	Year of occurrence of the voltage dip (UTC).
uiDip_timestamp_hour_n_7	UINT	Hour of occurrence of voltage dip (UTC).
uiDip_timestamp_minute_n_7	UINT	Minute of occurrence of voltage dip (UTC).
uiDip_timestamp_second_n_7	UINT	Second of occurrence of voltage dip (UTC).
rDip_ph_cond_v_value_n_8	REAL	Lowest value of phase conductor voltages during voltage dip.

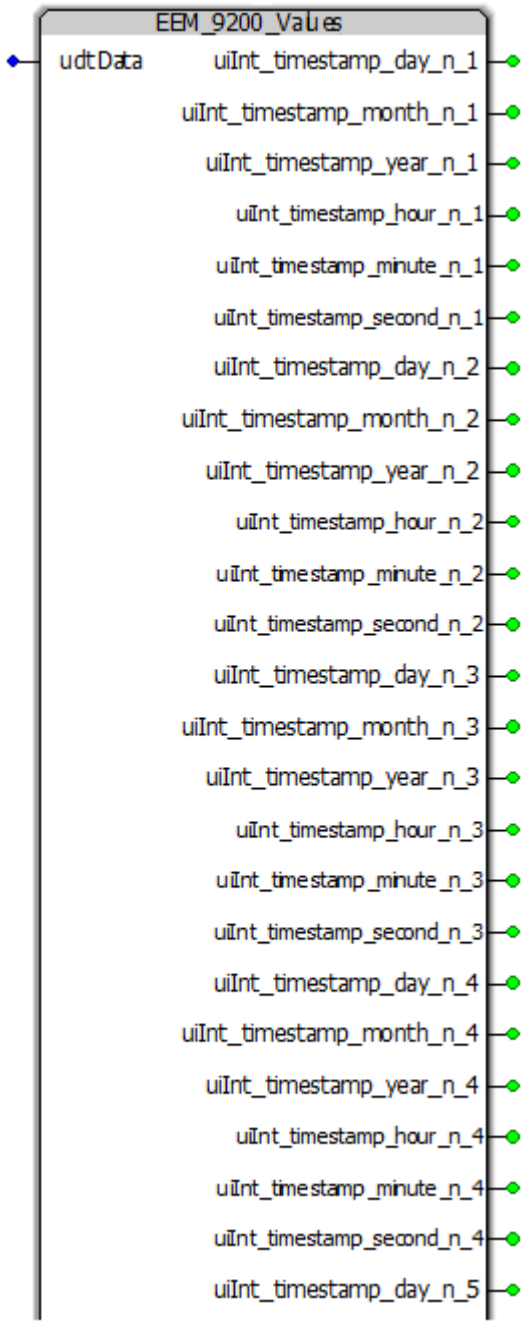
uiDip_ph_v_source_n_8	UINT	Cause of voltage dips (Udip) at the phase voltages. Bit mask: Bit 0: U1 Bit 1: U2 Bit 2: U3
rDip_ph_v_value_n_8	REAL	Lowest value of phase voltages during voltage dip.
uiDip_ph_v_source_n_8	UINT	Cause of voltage dips (Udip) at the phase voltages. Bit mask: Bit 0: U1 Bit 1: U2 Bit 2: U3
udiDip_duration_n_8	UDINT	Duration of the voltage dip in milliseconds.
uiDip_timestamp_day_n_8	UINT	Day of occurrence of voltage dip (UTC).
uiDip_timestamp_month_n_8	UINT	Month of occurrence of voltage dip (UTC).
uiDip_timestamp_year_n_8	UINT	Year of occurrence of the voltage dip (UTC).
uiDip_timestamp_hour_n_8	UINT	Hour of occurrence of voltage dip (UTC).
uiDip_timestamp_minute_n_8	UINT	Minute of occurrence of voltage dip (UTC).
uiDip_timestamp_second_n_8	UINT	Second of occurrence of voltage dip (UTC).
rDip_ph_cond_v_value_n_9	REAL	Lowest value of phase conductor voltages during voltage dip.
uiDip_ph_v_source_n_9	UINT	Cause of voltage dips (Udip) at the phase voltages. Bit mask: Bit 0: U1 Bit 1: U2 Bit 2: U3
rDip_ph_v_value_n_9	REAL	Lowest value of phase voltages during voltage dip.
uiDip_ph_v_source_n_9	UINT	Cause of voltage dips (Udip) at the phase voltages. Bit mask: Bit 0: U1 Bit 1: U2 Bit 2: U3
udiDip_duration_n_9	UDINT	Duration of the voltage dip in milliseconds.
uiDip_timestamp_day_n_9	UINT	Day of occurrence of voltage dip (UTC).
uiDip_timestamp_month_n_9	UINT	Month of occurrence of voltage dip (UTC).
uiDip_timestamp_year_n_9	UINT	Year of occurrence of the voltage dip (UTC).
uiDip_timestamp_hour_n_9	UINT	Hour of occurrence of voltage dip (UTC).
uiDip_timestamp_minute_n_9	UINT	Minute of occurrence of voltage dip (UTC).
uiDip_timestamp_second_n_9	UINT	Second of occurrence of voltage dip (UTC).
rDip_ph_cond_v_value_n_10	REAL	Lowest value of phase conductor voltages during voltage dip.
uiDip_ph_v_source_n_10	UINT	Cause of voltage dips (Udip) at the phase voltages. Bit mask: Bit 0: U1 Bit 1: U2 Bit 2: U3
rDip_ph_v_value_n_10	REAL	Lowest value of phase voltages during voltage dip.
uiDip_ph_v_source_n_10	UINT	Cause of voltage dips (Udip) at the phase voltages. Bit mask: Bit 0: U1 Bit 1: U2 Bit 2: U3
udiDip_duration_n_10	UDINT	Duration of the voltage dip in milliseconds.
uiDip_timestamp_day_n_10	UINT	Day of occurrence of voltage dip (UTC).

uiDip_timestamp_month_n_10	UINT	Month of occurrence of voltage dip (UTC).
uiDip_timestamp_year_n_10	UINT	Year of occurrence of the voltage dip (UTC).
uiDip_timestamp_hour_n_10	UINT	Hour of occurrence of voltage dip (UTC).
uiDip_timestamp_minute_n_10	UINT	Minute of occurrence of voltage dip (UTC).
uiDip_timestamp_second_n_10	UINT	Second of occurrence of voltage dip (UTC).

8.22 EEM_9200_Values

The function block EEM_9200_Values converts the EEM_UDT_9200 input structure into individual outputs.

8.22.1 Function block call



8.22.2 Input parameters

Name	Type	Description
udtData	EEM_UDT_9200	Input structure for standardized values in the table.

8.22.3 Output parameters

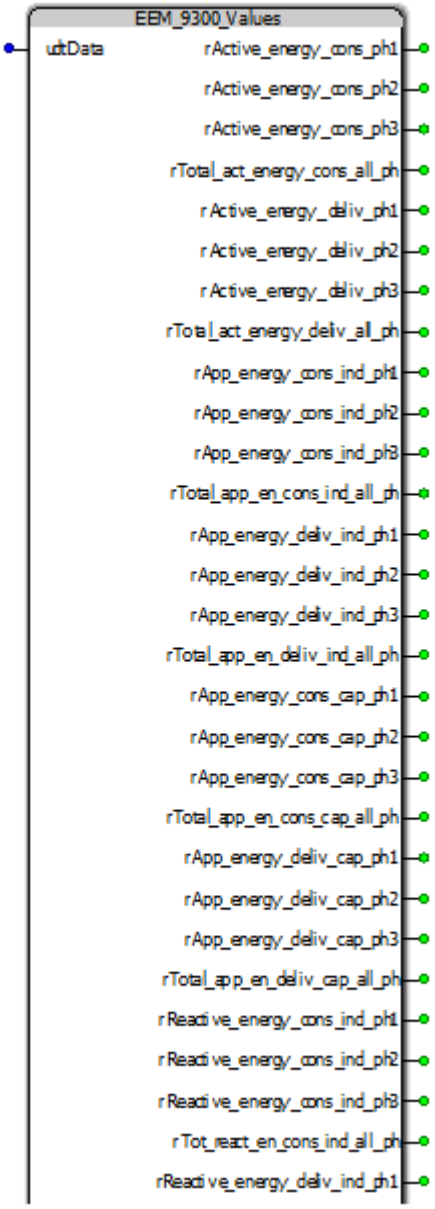
Name	Type	Description
uiInt_timestamp_day_n_1	UINT	Day of occurrence of voltage interruption (UTC).
uiInt_timestamp_month_n_1	UINT	Month of the occurrence of voltage interruption (UTC).
uiInt_timestamp_year_n_1	UINT	Year of occurrence of voltage interruption (UTC).
uiInt_timestamp_hour_n_1	UINT	Hour of occurrence of voltage interruption (UTC).
uiInt_timestamp_minute_n_1	UINT	Minute of occurrence of voltage interruption (UTC).
uiInt_timestamp_second_n_1	UINT	Second of the occurrence of voltage interruption (UTC).
uiInt_timestamp_day_n_2	UINT	Day of occurrence of voltage interruption (UTC).
uiInt_timestamp_month_n_2	UINT	Month of the occurrence of voltage interruption (UTC).
uiInt_timestamp_year_n_2	UINT	Year of occurrence of voltage interruption (UTC).
uiInt_timestamp_hour_n_2	UINT	Hour of occurrence of voltage interruption (UTC).
uiInt_timestamp_minute_n_2	UINT	Minute of occurrence of voltage interruption (UTC).
uiInt_timestamp_second_n_2	UINT	Second of the occurrence of voltage interruption (UTC).
uiInt_timestamp_day_n_3	UINT	Day of occurrence of voltage interruption (UTC).
uiInt_timestamp_month_n_3	UINT	Month of the occurrence of voltage interruption (UTC).
uiInt_timestamp_year_n_3	UINT	Year of occurrence of voltage interruption (UTC).
uiInt_timestamp_hour_n_3	UINT	Hour of occurrence of voltage interruption (UTC).
uiInt_timestamp_minute_n_3	UINT	Minute of occurrence of voltage interruption (UTC).
uiInt_timestamp_second_n_3	UINT	Second of the occurrence of voltage interruption (UTC).
uiInt_timestamp_day_n_4	UINT	Day of occurrence of voltage interruption (UTC).
uiInt_timestamp_month_n_4	UINT	Month of the occurrence of voltage interruption (UTC).
uiInt_timestamp_year_n_4	UINT	Year of occurrence of voltage interruption (UTC).
uiInt_timestamp_hour_n_4	UINT	Hour of occurrence of voltage interruption (UTC).
uiInt_timestamp_minute_n_4	UINT	Minute of occurrence of voltage interruption (UTC).
uiInt_timestamp_second_n_4	UINT	Second of the occurrence of voltage interruption (UTC).
uiInt_timestamp_day_n_5	UINT	Day of occurrence of voltage interruption (UTC).
uiInt_timestamp_month_n_5	UINT	Month of the occurrence of voltage interruption (UTC).
uiInt_timestamp_year_n_5	UINT	Year of occurrence of voltage interruption (UTC).
uiInt_timestamp_hour_n_5	UINT	Hour of occurrence of voltage interruption (UTC).
uiInt_timestamp_minute_n_5	UINT	Minute of occurrence of voltage interruption (UTC).
uiInt_timestamp_second_n_5	UINT	Second of the occurrence of voltage interruption (UTC).
uiInt_timestamp_day_n_6	UINT	Day of occurrence of voltage interruption (UTC).
uiInt_timestamp_month_n_6	UINT	Month of the occurrence of voltage interruption (UTC).
uiInt_timestamp_year_n_6	UINT	Year of occurrence of voltage interruption (UTC).
uiInt_timestamp_hour_n_6	UINT	Hour of occurrence of voltage interruption (UTC).
uiInt_timestamp_minute_n_6	UINT	Minute of occurrence of voltage interruption (UTC).
uiInt_timestamp_second_n_6	UINT	Second of the occurrence of voltage interruption (UTC).

uiInt_timestamp_day_n_7	UINT	Day of occurrence of voltage interruption (UTC).
uiInt_timestamp_month_n_7	UINT	Month of the occurrence of voltage interruption (UTC).
uiInt_timestamp_year_n_7	UINT	Year of occurrence of voltage interruption (UTC).
uiInt_timestamp_hour_n_7	UINT	Hour of occurrence of voltage interruption (UTC).
uiInt_timestamp_minute_n_7	UINT	Minute of occurrence of voltage interruption (UTC).
uiInt_timestamp_second_n_7	UINT	Second of the occurrence of voltage interruption (UTC).
uiInt_timestamp_day_n_8	UINT	Day of occurrence of voltage interruption (UTC).
uiInt_timestamp_month_n_8	UINT	Month of the occurrence of voltage interruption (UTC).
uiInt_timestamp_year_n_8	UINT	Year of occurrence of voltage interruption (UTC).
uiInt_timestamp_hour_n_8	UINT	Hour of occurrence of voltage interruption (UTC).
uiInt_timestamp_minute_n_8	UINT	Minute of occurrence of voltage interruption (UTC).
uiInt_timestamp_second_n_8	UINT	Second of the occurrence of voltage interruption (UTC).
uiInt_timestamp_day_n_9	UINT	Day of occurrence of voltage interruption (UTC).
uiInt_timestamp_month_n_9	UINT	Month of the occurrence of voltage interruption (UTC).
uiInt_timestamp_year_n_9	UINT	Year of occurrence of voltage interruption (UTC).
uiInt_timestamp_hour_n_9	UINT	Hour of occurrence of voltage interruption (UTC).
uiInt_timestamp_minute_n_9	UINT	Minute of occurrence of voltage interruption (UTC).
uiInt_timestamp_second_n_9	UINT	Second of the occurrence of voltage interruption (UTC).
uiInt_timestamp_day_n_10	UINT	Day of occurrence of voltage interruption (UTC).
uiInt_timestamp_month_n_10	UINT	Month of the occurrence of voltage interruption (UTC).
uiInt_timestamp_year_n_10	UINT	Year of occurrence of voltage interruption (UTC).
uiInt_timestamp_hour_n_10	UINT	Hour of occurrence of voltage interruption (UTC).
uiInt_timestamp_minute_n_10	UINT	Minute of occurrence of voltage interruption (UTC).
uiInt_timestamp_second_n_10	UINT	Second of the occurrence of voltage interruption (UTC).

8.23 EEM_9300_Values

The function block EEM_9300_Values converts the EEM_UDT_9300 input structure into individual outputs.

8.23.1 Function block call



8.23.2 Input parameters

Name	Type	Description
udtData	EEM_UDT_9300	Input structure for standardized values in the table.

8.23.3 Output parameters

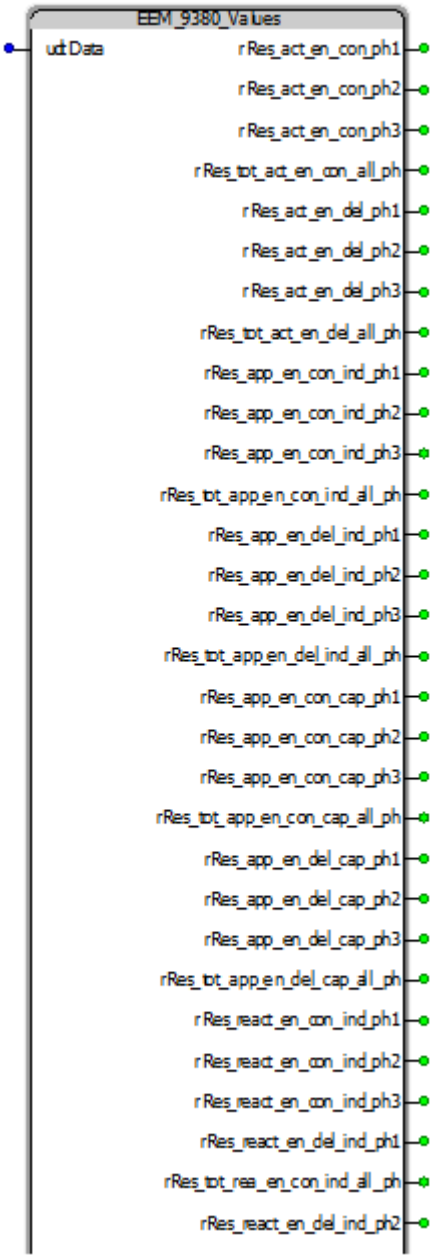
Name	Type	Description
rActive_energy_cons_ph1	REAL	Active energy consumed phase 1.
rActive_energy_cons_ph2	REAL	Active energy consumed phase 2.
rActive_energy_cons_ph3	REAL	Active energy consumed phase 3.
rTotal_act_energy_cons_all_ph	REAL	Total active energy consumed all phases.
rActive_energy_deliv_ph1	REAL	Active energy delivered phase 1.
rActive_energy_deliv_ph2	REAL	Active energy delivered phase 2.
rActive_energy_deliv_ph3	REAL	Active energy delivered phase 3.
rTotal_act_energy_deliv_all_ph	REAL	Total active energy delivered all phases.
rApp_energy_cons_ind_ph1	REAL	Apparent energy consumed inductive phase 1.
rApp_energy_cons_ind_ph2	REAL	Apparent energy consumed inductive phase 2.
rApp_energy_cons_ind_ph3	REAL	Apparent energy consumed inductive phase 3.
rTotal_app_en_cons_ind_all_ph	REAL	Total apparent energy consumed inductively all phases.
rApp_energy_deliv_ind_ph1	REAL	Apparent energy delivered inductive phase 1.
rApp_energy_deliv_ind_ph2	REAL	Apparent energy delivered inductive phase 2.
rApp_energy_deliv_ind_ph3	REAL	Apparent energy delivered inductive phase 3.
rTotal_app_en_deliv_ind_all_ph	REAL	Total apperent energy delivered inductively all phases.
rApp_energy_cons_cap_ph1	REAL	Apparent energy consumed capacitive phase 1.
rApp_energy_cons_cap_ph2	REAL	Apparent energy consumed capacitive phase 2.
rApp_energy_cons_cap_ph3	REAL	Apparent energy consumed capacitive phase 3.
rTotal_app_en_cons_cap_all_ph	REAL	Total apparent energy consumed capacitive all phases.
rApp_energy_deliv_cap_ph1	REAL	Apparent energy delivered capacitive phase 1.
rApp_energy_deliv_cap_ph2	REAL	Apparent energy delivered capacitive phase 2.
rApp_energy_deliv_cap_ph3	REAL	Apparent energy delivered capacitive phase 3.
rTotal_app_en_deliv_cap_all_ph	REAL	Total apparent energy delivered capacitive all phases.
rReactive_energy_cons_ind_ph1	REAL	Reactive energy consumed inductive phase 1.
rReactive_energy_cons_ind_ph2	REAL	Reactive energy consumed inductive phase 2.
rReactive_energy_cons_ind_ph3	REAL	Reactive energy consumed inductive phase 3.
rTot_react_en_cons_ind_all_ph	REAL	Total reactive energy consumed inductively all phases.
rReactive_energy_deliv_ind_ph1	REAL	Reactive energy delivered inductive phase 1.
rReactive_energy_deliv_ind_ph2	REAL	Reactive energy delivered inductive phase 2.
rReactive_energy_deliv_ind_ph3	REAL	Reactive energy delivered inductive phase 3.
rTot_react_en_deliv_ind_all_ph	REAL	Total reactive energy delivered inductively all phases.
rReactive_energy_cons_cap_ph1	REAL	Reactive energy consumed capacitive phase 1.
rReactive_energy_cons_cap_ph2	REAL	Reactive energy consumed capacitive phase 2.
rReactive_energy_cons_cap_ph3	REAL	Reactive energy consumed capacitive phase 3.
rTot_react_en_cons_cap_all_ph	REAL	Total reactive energy consumed capacitively all phases.
rReactive_energy_deliv_cap_ph1	REAL	Reactive energy delivered capacitive phase 1.

rReactive_energy_deliv_cap_ph2	REAL	Reactive energy delivered capacitive phase 2.
rReactive_energy_deliv_cap_ph3	REAL	Reactive energy delivered capacitive phase 3.
rTot_react_en_deliv_cap_all_ph	REAL	Total reactive energy delivered capacitive all phases.
rTotal_react_en_cons_all_ph	REAL	Total reactive energy consumed all phases.
rTotal_react_en_deliv_all_ph	REAL	Total reactive energy delivered all phases.
rrTotal_react_en_absolut_all_ph	REAL	Total reactive energy absolutely all phases.
rTotal_app_en_all_ph	REAL	Total apparent energy all phases.

8.24 EEM_9380_Values

The function block EEM_9380_Values converts the EEM_UDT_9380 input structure into individual outputs.

8.24.1 Function block call



8.24.2 Input parameters

Name	Type	Description
udtData	EEM_UDT_9380	Input structure for standardized values in the table.

8.24.3 Output parameters

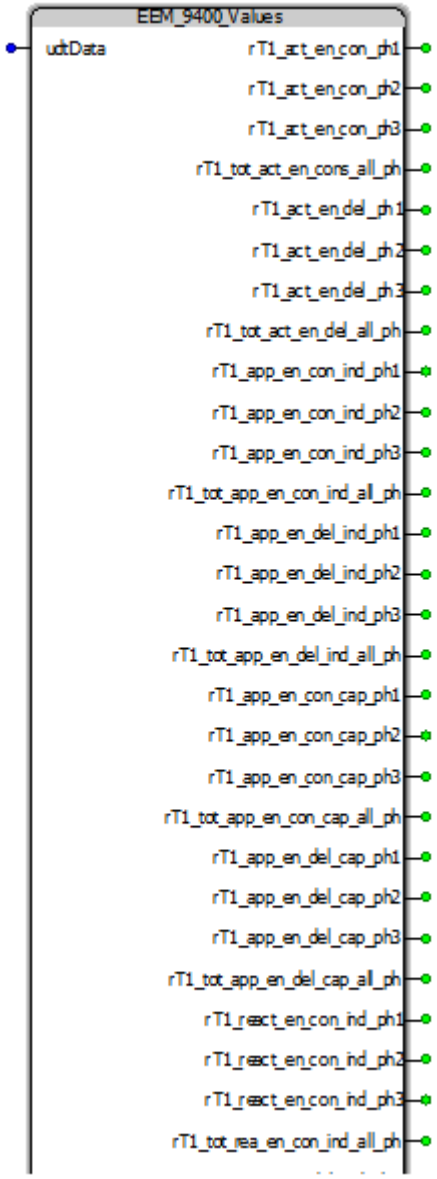
Name	Type	Description
rRes_act_en_con_ph1	REAL	Resetable active energy consumed phase 1.
rRes_act_en_con_ph2	REAL	Resetable active energy consumed phase 2.
rRes_act_en_con_ph3	REAL	Resetable active energy consumed phase 3.
rRes_tot_act_en_con_all_ph	REAL	Resetable total active energy consumed all phases
rRes_act_en_del_ph1	REAL	Resetable active energy delivered phase 1.
rRes_act_en_del_ph2	REAL	Resetable active energy delivered phase 2.
rRes_act_en_del_ph3	REAL	Resetable active energy delivered phase 3.
rRes_tot_act_en_del_all_ph	REAL	Resetable total active energy delivered all phases.
rRes_app_en_con_ind_ph1	REAL	Resetable apparent energy consumed inductive phase 1.
rRes_app_en_con_ind_ph2	REAL	Resetable apparent energy consumed inductive phase 2.
rRes_app_en_con_ind_ph3	REAL	Resetable apparent energy consumed inductive phase 3.
rRes_tot_app_en_con_ind_all_ph	REAL	Resetable total apparent energy consumed inductively all phases.
rRes_app_en_del_ind_ph1	REAL	Resetable apparent energy delivered inductive phase 1.
rRes_app_en_del_ind_ph2	REAL	Resetable apparent energy delivered inductive phase 2.
rRes_app_en_del_ind_ph3	REAL	Resetable apparent energy delivered inductive phase 3.
rRes_tot_app_en_del_ind_all_ph	REAL	Resetable total apparent energy delivered inductively all phases.
rRes_app_en_con_cap_ph1	REAL	Resetable apparent energy consumed capacitive phase 1.
rRes_app_en_con_cap_ph2	REAL	Resetable apparent energy consumed capacitive phase 2.
rRes_app_en_con_cap_ph3	REAL	Resetable apparent energy consumed capacitive phase 3.
rRes_tot_app_en_con_cap_all_ph	REAL	Resetable total apparent energy consumed capacitive all phases.
rRes_app_en_del_cap_ph1	REAL	Resetable apparent energy delivered capacitive phase 1.
rRes_app_en_del_cap_ph2	REAL	Resetable apparent energy delivered capacitive phase 2.
rRes_app_en_del_cap_ph3	REAL	Resetable apparent energy delivered capacitive phase 3.
rRes_tot_app_en_del_cap_all_ph	REAL	Resetable total apparent energy delivered capacitive all phases.
rRes_react_en_con_ind_ph1	REAL	Resetable reactive energy consumed inductive phase 1.
rRes_react_en_con_ind_ph2	REAL	Resetable reactive energy consumed inductive phase 2.
rRes_react_en_con_ind_ph3	REAL	Resetable reactive energy consumed inductive phase 3.
rRes_tot_rea_en_con_ind_all_ph	REAL	Resetable total reactive energy consumed inductively all phases.
rRes_react_en_del_ind_ph1	REAL	Resetable reactive energy delivered inductive phase 1.
rRes_react_en_del_ind_ph2	REAL	Resetable reactive energy delivered inductive phase 2.
rRes_react_en_del_ind_ph3	REAL	Resetable reactive energy delivered inductive phase 3.
rRes_tot_rea_en_del_ind_all_ph	REAL	Resetable total reactive energy delivered inductively all phases.
rRes_react_en_con_cap_ph1	REAL	Resetable reactive energy consumed capacitive phase 1.
rRes_react_en_con_cap_ph2	REAL	Resetable reactive energy consumed capacitive phase 2.
rRes_react_en_con_cap_ph3	REAL	Resetable reactive energy consumed capacitive phase 3.
rRes_tot_rea_en_con_cap_all_ph	REAL	Resetable total reactive energy consumed capacitively all phases.
rRes_react_en_del_cap_ph1	REAL	Resetable reactive energy delivered capacitive phase 1.

rRes_react_en_del_cap_ph2	REAL	Resetable reactive energy delivered capacitive phase 2.
rRes_react_en_del_cap_ph3	REAL	Resetable reactive energy delivered capacitive phase 3.
rRes_tot_rea_en_del_cap_all_ph	REAL	Resetable total reactive energy delivered capacitive all phases.
rRes_tot_rea_en_con_all_ph	REAL	Resetable total reactive energy consumed all phases.
rRes_tot_react_en_del_all_ph	REAL	Resetable total reactive energy delivered all phases.
rRes_tot_react_en_abs_all_ph	REAL	Resetable total reactive energy absolutely all phases.
rRes_tot_app_en_all_ph	REAL	Resetable total apperent energy all phases.

8.25 EEM_9400_Values

The function block EEM_9400_Values converts the EEM_UDT_9400 input structure into individual outputs.

8.25.1 Function block call



8.25.2 Input parameters

Name	Type	Description
udtData	EEM_UDT_9400	Input structure for standardized values in the table.

8.25.3 Output parameters

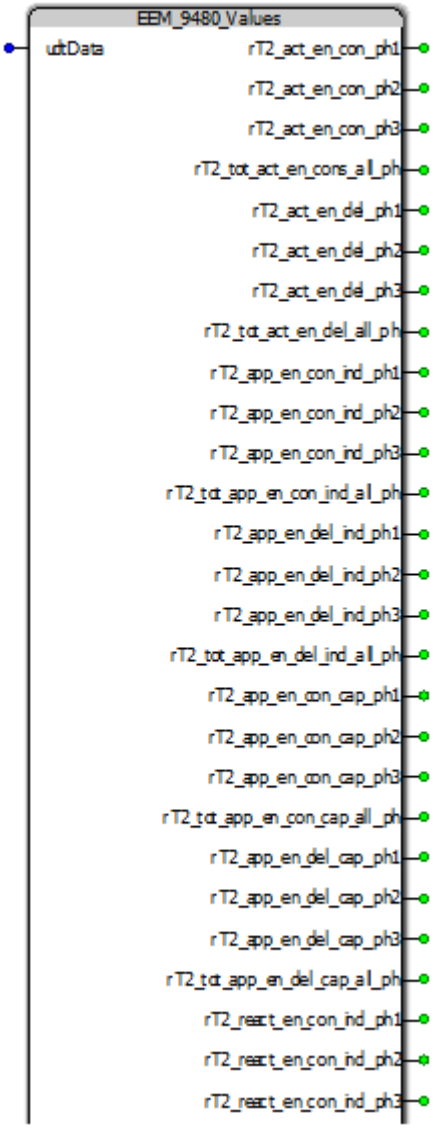
Name	Type	Description
rT1_act_en_con_ph1	REAL	Tariff 1 active energy consumed phase 1.
rT1_act_en_con_ph2	REAL	Tariff 1 active energy consumed phase 2.
rT1_act_en_con_ph3	REAL	Tariff 1 active energy consumed phase 3.
rT1_tot_act_en_cons_all_ph	REAL	Tariff 1 total active energy consumed all phases.
rT1_act_en_del_ph1	REAL	Tariff 1 active energy delivered phase 1.
rT1_act_en_del_ph2	REAL	Tariff 1 active energy delivered phase 2.
rT1_act_en_del_ph3	REAL	Tariff 1 active energy delivered phase 3.
rT1_tot_act_en_del_all_ph	REAL	Tariff 1 total active energy delivered all phases.
rT1_app_en_con_ind_ph1	REAL	Tariff 1 apparent energy consumed inductive phase 1.
rT1_app_en_con_ind_ph2	REAL	Tariff 1 apparent energy consumed inductive phase 2.
rT1_app_en_con_ind_ph3	REAL	Tariff 1 apparent energy consumed inductive phase 3.
rT1_tot_app_en_con_ind_all_ph	REAL	Tariff 1 total apperent energy consumed inductively all phases.
rT1_app_en_del_ind_ph1	REAL	Tariff 1 apparent energy delivered inductive phase 1.
rT1_app_en_del_ind_ph2	REAL	Tariff 1 apparent energy delivered inductive phase 2.
rT1_app_en_del_ind_ph3	REAL	Tariff 1 apparent energy delivered inductive phase 3.
rT1_tot_app_en_del_ind_all_ph	REAL	Tariff 1 total apparent energy delivered inductively all phases.
rT1_app_en_con_cap_ph1	REAL	Tariff 1 apparent energy consumed capacitive phase 1.
rT1_app_en_con_cap_ph2	REAL	Tariff 1 apparent energy consumed capacitive phase 2.
rT1_app_en_con_cap_ph3	REAL	Tariff 1 apparent energy consumed capacitive phase 3.
rT1_tot_app_en_con_cap_all_ph	REAL	Tariff 1 total apparent energy consumed capacitive all phases.
rT1_app_en_del_cap_ph1	REAL	Tariff 1 apparent energy delivered capacitive phase 1.
rT1_app_en_del_cap_ph2	REAL	Tariff 1 apparent energy delivered capacitive phase 2.
rT1_app_en_del_cap_ph3	REAL	Tariff 1 apparent energy delivered capacitive phase 3.
rT1_tot_app_en_del_cap_all_ph	REAL	Tariff 1 total apparent energy delivered capacitive all phases.
rT1_react_en_con_ind_ph1	REAL	Tariff 1 reactive energy consumed inductive phase 1.
rT1_react_en_con_ind_ph2	REAL	Tariff 1 reactive energy consumed inductive phase 2.
rT1_react_en_con_ind_ph3	REAL	Tariff 1 reactive energy consumed inductive phase 3.
rT1_tot_rea_en_con_ind_all_ph	REAL	Tariff 1 total reactive energy consumed inductively all phases.
rT1_react_en_del_ind_ph1	REAL	Tariff 1 reactive energy delivered inductive phase 1.
rT1_react_en_del_ind_ph2	REAL	Tariff 1 reactive energy delivered inductive phase 2.
rT1_react_en_del_ind_ph3	REAL	Tariff 1 reactive energy delivered inductive phase 3.
rT1_tot_rea_en_del_ind_all_ph	REAL	Tariff 1 total reactive energy delivered inductively all phases.
rT1_react_en_con_cap_ph1	REAL	Tariff 1 reactive energy consumed capacitive phase 1.
rT1_react_en_con_cap_ph2	REAL	Tariff 1 reactive energy consumed capacitive phase 2.
rT1_react_en_con_cap_ph3	REAL	Tariff 1 reactive energy consumed capacitive phase 3.
rT1_tot_rea_en_con_cap_all_ph	REAL	Tariff 1 total reactive energy consumed capacitively all phases.
rT1_react_en_del_cap_ph1	REAL	Tariff 1 reactive energy delivered capacitive phase 1.

rT1_react_en_del_cap_ph2	REAL	Tariff 1 reactive energy delivered capacitive phase 2.
rT1_react_en_del_cap_ph3	REAL	Tariff 1 reactive energy delivered capacitive phase 3.
rT1_tot_rea_en_del_cap_all_ph	REAL	Tariff 1 total reactive energy delivered capacitive all phases.
rT1_tot_react_en_con_all_ph	REAL	Tariff 1 total reactive energy consumed all phases.
rT1_tot_react_en_del_all_ph	REAL	Tariff 1 total reactive energy delivered all phases.
rT1_tot_react_en_abs_all_ph	REAL	Tariff 1 total reactive energy absolutely all phases.
rT1_tot_app_en_all_ph	REAL	Tariff 1 total apperent energy all phases.

8.26 EEM_9480_Values

The function block EEM_9480_Values converts the EEM_UDT_9480 input structure into individual outputs.

8.26.1 Function block call



8.26.2 Input parameters

Name	Type	Description
udtData	EEM_UDT_9480	Input structure for standardized values in the table.

8.26.3 Output parameters

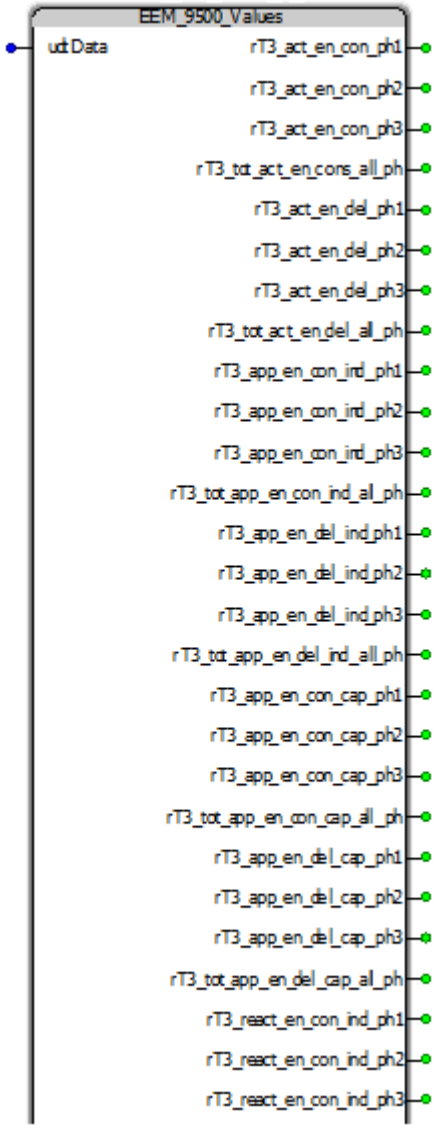
Name	Type	Description
rT2_act_en_con_ph1	REAL	Tariff 2 active energy consumed phase 1.
rT2_act_en_con_ph2	REAL	Tariff 2 active energy consumed phase 2.
rT2_act_en_con_ph3	REAL	Tariff 2 active energy consumed phase 3.
rT2_tot_act_en_cons_all_ph	REAL	Tariff 2 total active energy consumed all phases.
rT2_act_en_del_ph1	REAL	Tariff 2 active energy delivered phase 1.
rT2_act_en_del_ph2	REAL	Tariff 2 active energy delivered phase 2.
rT2_act_en_del_ph3	REAL	Tariff 2 active energy delivered phase 3.
rT2_tot_act_en_del_all_ph	REAL	Tariff 2 total active energy delivered all phases.
rT2_app_en_con_ind_ph1	REAL	Tariff 2 apparent energy consumed inductive phase 1.
rT2_app_en_con_ind_ph2	REAL	Tariff 2 apparent energy consumed inductive phase 2.
rT2_app_en_con_ind_ph3	REAL	Tariff 2 apparent energy consumed inductive phase 3.
rT2_tot_app_en_con_ind_all_ph	REAL	Tariff 2 total apperent energy consumed inductively all phases.
rT2_app_en_del_ind_ph1	REAL	Tariff 2 apparent energy delivered inductive phase 1.
rT2_app_en_del_ind_ph2	REAL	Tariff 2 apparent energy delivered inductive phase 2.
rT2_app_en_del_ind_ph3	REAL	Tariff 2 apparent energy delivered inductive phase 3.
rT2_tot_app_en_del_ind_all_ph	REAL	Tariff 2 total apparent energy delivered inductively all phases.
rT2_app_en_con_cap_ph1	REAL	Tariff 2 apparent energy consumed capacitive phase 1.
rT2_app_en_con_cap_ph2	REAL	Tariff 2 apparent energy consumed capacitive phase 2.
rT2_app_en_con_cap_ph3	REAL	Tariff 2 apparent energy consumed capacitive phase 3.
rT2_tot_app_en_con_cap_all_ph	REAL	Tariff 2 total apparent energy consumed capacitive all phases.
rT2_app_en_del_cap_ph1	REAL	Tariff 2 apparent energy delivered capacitive phase 1.
rT2_app_en_del_cap_ph2	REAL	Tariff 2 apparent energy delivered capacitive phase 2.
rT2_app_en_del_cap_ph3	REAL	Tariff 2 apparent energy delivered capacitive phase 3.
rT2_tot_app_en_del_cap_all_ph	REAL	Tariff 2 total apparent energy delivered capacitive all phases.
rT2_react_en_con_ind_ph1	REAL	Tariff 2 reactive energy consumed inductive phase 1.
rT2_react_en_con_ind_ph2	REAL	Tariff 2 reactive energy consumed inductive phase 2.
rT2_react_en_con_ind_ph3	REAL	Tariff 2 reactive energy consumed inductive phase 3.
rT2_tot_rea_en_con_ind_all_ph	REAL	Tariff 2 total reactive energy consumed inductively all phases.
rT2_react_en_del_ind_ph1	REAL	Tariff 2 reactive energy delivered inductive phase 1.
rT2_react_en_del_ind_ph2	REAL	Tariff 2 reactive energy delivered inductive phase 2.
rT2_react_en_del_ind_ph3	REAL	Tariff 2 reactive energy delivered inductive phase 3.
rT2_tot_rea_en_del_ind_all_ph	REAL	Tariff 2 total reactive energy delivered inductively all phases.
rT2_react_en_con_cap_ph1	REAL	Tariff 2 reactive energy consumed capacitive phase 1.
rT2_react_en_con_cap_ph2	REAL	Tariff 2 reactive energy consumed capacitive phase 2.
rT2_react_en_con_cap_ph3	REAL	Tariff 2 reactive energy consumed capacitive phase 3.
rT2_tot_rea_en_con_cap_all_ph	REAL	Tariff 2 total reactive energy consumed capacitively all phases.
rT2_react_en_del_cap_ph1	REAL	Tariff 2 reactive energy delivered capacitive phase 1.

rT2_react_en_del_cap_ph2	REAL	Tariff 2 reactive energy delivered capacitive phase 2.
rT2_react_en_del_cap_ph3	REAL	Tariff 2 reactive energy delivered capacitive phase 3.
rT2_tot_rea_en_del_cap_all_ph	REAL	Tariff 2 total reactive energy delivered capacitive all phases.
rT2_tot_react_en_con_all_ph	REAL	Tariff 2 total reactive energy consumed all phases.
rT2_tot_react_en_del_all_ph	REAL	Tariff 2 total reactive energy delivered all phases.
rT2_tot_react_en_abs_all_ph	REAL	Tariff 2 total reactive energy absolutely all phases.
rT2_tot_app_en_all_ph	REAL	Tariff 2 total apperent energy all phases.

8.27 EEM_9500_Values

The function block EEM_9500_Values converts the EEM_UDT_9500 input structure into individual outputs.

8.27.1 Function block call



8.27.2 Input parameters

Name	Type	Description
udtData	EEM_UDT_9500	Input structure for standardized values in the table.

8.27.3 Output parameters

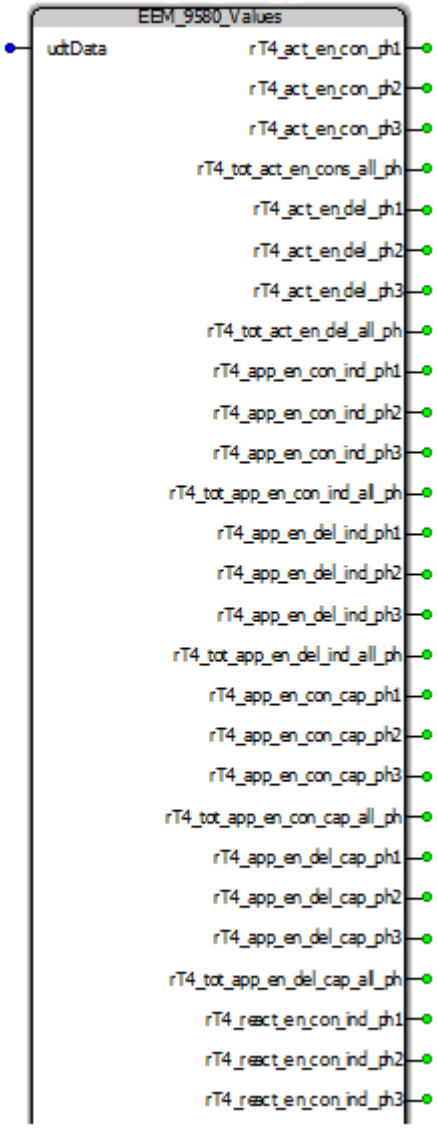
Name	Type	Description
rT3_act_en_con_ph1	REAL	Tariff 3 active energy consumed phase 1.
rT3_act_en_con_ph2	REAL	Tariff 3 active energy consumed phase 2.
rT3_act_en_con_ph3	REAL	Tariff 3 active energy consumed phase 3.
rT3_tot_act_en_cons_all_ph	REAL	Tariff 3 total active energy consumed all phases.
rT3_act_en_del_ph1	REAL	Tariff 3 active energy delivered phase 1.
rT3_act_en_del_ph2	REAL	Tariff 3 active energy delivered phase 2.
rT3_act_en_del_ph3	REAL	Tariff 3 active energy delivered phase 3.
rT3_tot_act_en_del_all_ph	REAL	Tariff 3 total active energy delivered all phases.
rT3_app_en_con_ind_ph1	REAL	Tariff 3 apparent energy consumed inductive phase 1.
rT3_app_en_con_ind_ph2	REAL	Tariff 3 apparent energy consumed inductive phase 2.
rT3_app_en_con_ind_ph3	REAL	Tariff 3 apparent energy consumed inductive phase 3.
rT3_tot_app_en_con_ind_all_ph	REAL	Tariff 3 total apperent energy consumed inductively all phases.
rT3_app_en_del_ind_ph1	REAL	Tariff 3 apparent energy delivered inductive phase 1.
rT3_app_en_del_ind_ph2	REAL	Tariff 3 apparent energy delivered inductive phase 2.
rT3_app_en_del_ind_ph3	REAL	Tariff 3 apparent energy delivered inductive phase 3.
rT3_tot_app_en_del_ind_all_ph	REAL	Tariff 3 total apparent energy delivered inductively all phases.
rT3_app_en_con_cap_ph1	REAL	Tariff 3 apparent energy consumed capacitive phase 1.
rT3_app_en_con_cap_ph2	REAL	Tariff 3 apparent energy consumed capacitive phase 2.
rT3_app_en_con_cap_ph3	REAL	Tariff 3 apparent energy consumed capacitive phase 3.
rT3_tot_app_en_con_cap_all_ph	REAL	Tariff 3 total apparent energy consumed capacitive all phases.
rT3_app_en_del_cap_ph1	REAL	Tariff 3 apparent energy delivered capacitive phase 1.
rT3_app_en_del_cap_ph2	REAL	Tariff 3 apparent energy delivered capacitive phase 2.
rT3_app_en_del_cap_ph3	REAL	Tariff 3 apparent energy delivered capacitive phase 3.
rT3_tot_app_en_del_cap_all_ph	REAL	Tariff 3 total apparent energy delivered capacitive all phases.
rT3_react_en_con_ind_ph1	REAL	Tariff 3 reactive energy consumed inductive phase 1.
rT3_react_en_con_ind_ph2	REAL	Tariff 3 reactive energy consumed inductive phase 2.
rT3_react_en_con_ind_ph3	REAL	Tariff 3 reactive energy consumed inductive phase 3.
rT3_tot_rea_en_con_ind_all_ph	REAL	Tariff 3 total reactive energy consumed inductively all phases.
rT3_react_en_del_ind_ph1	REAL	Tariff 3 reactive energy delivered inductive phase 1.
rT3_react_en_del_ind_ph2	REAL	Tariff 3 reactive energy delivered inductive phase 2.
rT3_react_en_del_ind_ph3	REAL	Tariff 3 reactive energy delivered inductive phase 3.
rT3_tot_rea_en_del_ind_all_ph	REAL	Tariff 3 total reactive energy delivered inductively all phases.
rT3_react_en_con_cap_ph1	REAL	Tariff 3 reactive energy consumed capacitive phase 1.
rT3_react_en_con_cap_ph2	REAL	Tariff 3 reactive energy consumed capacitive phase 2.
rT3_react_en_con_cap_ph3	REAL	Tariff 3 reactive energy consumed capacitive phase 3.
rT3_tot_rea_en_con_cap_all_ph	REAL	Tariff 3 total reactive energy consumed capacitively all phases.
rT3_react_en_del_cap_ph1	REAL	Tariff 3 reactive energy delivered capacitive phase 1.

rT3_react_en_del_cap_ph2	REAL	Tariff 3 reactive energy delivered capacitive phase 2.
rT3_react_en_del_cap_ph3	REAL	Tariff 3 reactive energy delivered capacitive phase 3.
rT3_tot_rea_en_del_cap_all_ph	REAL	Tariff 3 total reactive energy delivered capacitive all phases.
rT3_tot_react_en_con_all_ph	REAL	Tariff 3 total reactive energy consumed all phases.
rT3_tot_react_en_del_all_ph	REAL	Tariff 3 total reactive energy delivered all phases.
rT3_tot_react_en_abs_all_ph	REAL	Tariff 3 total reactive energy absolutely all phases.
rT3_tot_app_en_all_ph	REAL	Tariff 3 total apperent energy all phases.

8.28 EEM_9580_Values

The function block EEM_9580_Values converts the EEM_UDT_9580 input structure into individual outputs.

8.28.1 Function block call



8.28.2 Input parameters

Name	Type	Description
udtData	EEM_UDT_9580	Input structure for standardized values in the table.

8.28.3 Output parameters

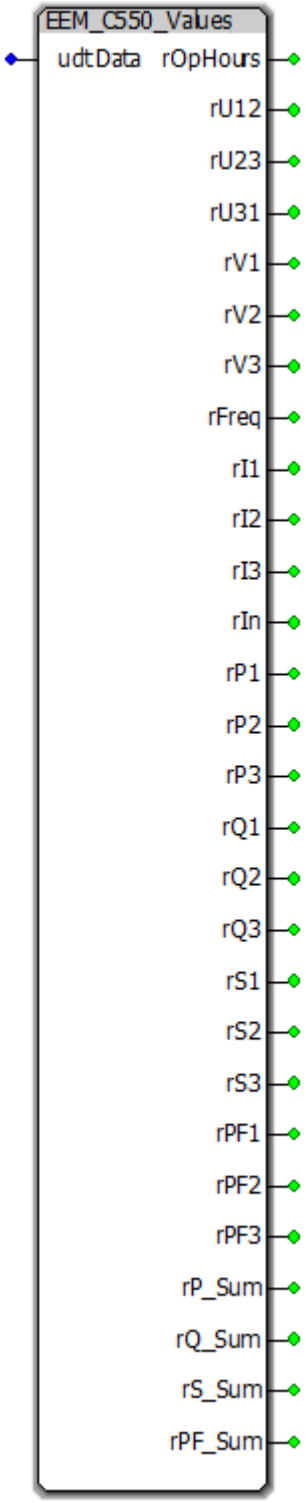
Name	Type	Description
rT4_act_en_con_ph1	REAL	Tariff 4 active energy consumed phase 1.
rT4_act_en_con_ph2	REAL	Tariff 4 active energy consumed phase 2.
rT4_act_en_con_ph3	REAL	Tariff 4 active energy consumed phase 3.
rT4_tot_act_en_cons_all_ph	REAL	Tariff 4 total active energy consumed all phases.
rT4_act_en_del_ph1	REAL	Tariff 4 active energy delivered phase 1.
rT4_act_en_del_ph2	REAL	Tariff 4 active energy delivered phase 2.
rT4_act_en_del_ph3	REAL	Tariff 4 active energy delivered phase 3.
rT4_tot_act_en_del_all_ph	REAL	Tariff 4 total active energy delivered all phases.
rT4_app_en_con_ind_ph1	REAL	Tariff 4 apparent energy consumed inductive phase 1.
rT4_app_en_con_ind_ph2	REAL	Tariff 4 apparent energy consumed inductive phase 2.
rT4_app_en_con_ind_ph3	REAL	Tariff 4 apparent energy consumed inductive phase 3.
rT4_tot_app_en_con_ind_all_ph	REAL	Tariff 4 total apperent energy consumed inductively all phases.
rT4_app_en_del_ind_ph1	REAL	Tariff 4 apparent energy delivered inductive phase 1.
rT4_app_en_del_ind_ph2	REAL	Tariff 4 apparent energy delivered inductive phase 2.
rT4_app_en_del_ind_ph3	REAL	Tariff 4 apparent energy delivered inductive phase 3.
rT4_tot_app_en_del_ind_all_ph	REAL	Tariff 4 total apparent energy delivered inductively all phases.
rT4_app_en_con_cap_ph1	REAL	Tariff 4 apparent energy consumed capacitive phase 1.
rT4_app_en_con_cap_ph2	REAL	Tariff 4 apparent energy consumed capacitive phase 2.
rT4_app_en_con_cap_ph3	REAL	Tariff 4 apparent energy consumed capacitive phase 3.
rT4_tot_app_en_con_cap_all_ph	REAL	Tariff 4 total apparent energy consumed capacitive all phases.
rT4_app_en_del_cap_ph1	REAL	Tariff 4 apparent energy delivered capacitive phase 1.
rT4_app_en_del_cap_ph2	REAL	Tariff 4 apparent energy delivered capacitive phase 2.
rT4_app_en_del_cap_ph3	REAL	Tariff 4 apparent energy delivered capacitive phase 3.
rT4_tot_app_en_del_cap_all_ph	REAL	Tariff 4 total apparent energy delivered capacitive all phases.
rT4_react_en_con_ind_ph1	REAL	Tariff 4 reactive energy consumed inductive phase 1.
rT4_react_en_con_ind_ph2	REAL	Tariff 4 reactive energy consumed inductive phase 2.
rT4_react_en_con_ind_ph3	REAL	Tariff 4 reactive energy consumed inductive phase 3.
rT4_tot_rea_en_con_ind_all_ph	REAL	Tariff 4 total reactive energy consumed inductively all phases.
rT4_react_en_del_ind_ph1	REAL	Tariff 4 reactive energy delivered inductive phase 1.
rT4_react_en_del_ind_ph2	REAL	Tariff 4 reactive energy delivered inductive phase 2.
rT4_react_en_del_ind_ph3	REAL	Tariff 4 reactive energy delivered inductive phase 3.
rT4_tot_rea_en_del_ind_all_ph	REAL	Tariff 4 total reactive energy delivered inductively all phases.
rT4_react_en_con_cap_ph1	REAL	Tariff 4 reactive energy consumed capacitive phase 1.
rT4_react_en_con_cap_ph2	REAL	Tariff 4 reactive energy consumed capacitive phase 2.
rT4_react_en_con_cap_ph3	REAL	Tariff 4 reactive energy consumed capacitive phase 3.
rT4_tot_rea_en_con_cap_all_ph	REAL	Tariff 4 total reactive energy consumed capacitively all phases.
rT4_react_en_del_cap_ph1	REAL	Tariff 4 reactive energy delivered capacitive phase 1.

rT4_react_en_del_cap_ph2	REAL	Tariff 4 reactive energy delivered capacitive phase 2.
rT4_react_en_del_cap_ph3	REAL	Tariff 4 reactive energy delivered capacitive phase 3.
rT4_tot_rea_en_del_cap_all_ph	REAL	Tariff 4 total reactive energy delivered capacitive all phases.
rT4_tot_react_en_con_all_ph	REAL	Tariff 4 total reactive energy consumed all phases.
rT4_tot_react_en_del_all_ph	REAL	Tariff 4 total reactive energy delivered all phases.
rT4_tot_react_en_abs_all_ph	REAL	Tariff 4 total reactive energy absolutely all phases.
rT4_tot_app_en_all_ph	REAL	Tariff 4 total apperent energy all phases.

8.29 EEM_C550_Values

Function block EEM_C550_Values converts the EEM_UDT_C550 input structure iinto individual output parameters.

8.29.1 Function block call



8.29.2 Input parameters

Name	Type	Description
udtData	EEM_UDT_C550	Input structure for standardized values in the table.

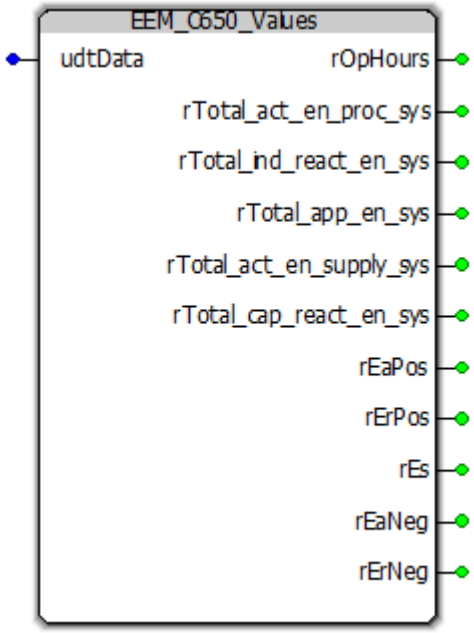
8.29.3 Output parameters

Name	Type	Description
OpHours	REAL	Operating hours counter.
rU12	REAL	Conductor voltage (L1-L2).
rU23	REAL	Conductor voltage (L2-L3).
rU31	REAL	Conductor voltage (L3-L1).
rV1	REAL	Conductor voltage (L1-N).
rV2	REAL	Conductor voltage (L2-N).
rV3	REAL	Conductor voltage (L3-N).
rFreq	REAL	Frequency.
rI1	REAL	Conductor current (L1).
rI2	REAL	Conductor current (L2).
rI3	REAL	Conductor current (L3).
rIn	REAL	Current in neutral conductor.
rP_Sum	REAL	Total real power +/-.
rQ_Sum	REAL	Total reactive power +/-.
rS_Sum	REAL	Total apparent power +/-.
rPF_Sum	REAL	Total power factor +/-.
rP1	REAL	Real power phase 1 +/-.
rP2	REAL	Real power phase 2 +/-.
rP3	REAL	Real power phase 3 +/-.
rQ1	REAL	Reactive power phase 1 +/-.
rQ2	REAL	Reactive power phase 2 +/-.
rQ3	REAL	Reactive power phase 3 +/-.
rS1	REAL	Apparent power phase 1 +/-.
rS2	REAL	Apparent power phase 2 +/-.
rS3	REAL	Apparent power phase 3 +/-.
rPF1	REAL	Power factor phase 1 +/-.
rPF2	REAL	Power factor phase 2 +/-.
rPF3	REAL	Power factor phase 3 +/-.

8.30 EEM_C650_Values

Block EEM_C650_Values converts the EEM_UDT_C650 input structure into individual output parameters.

8.30.1 Function block call



8.30.2 Input parameters

Name	Type	Description
udtData	EEM_UDT_C650	Input structure for standardized values in the table.

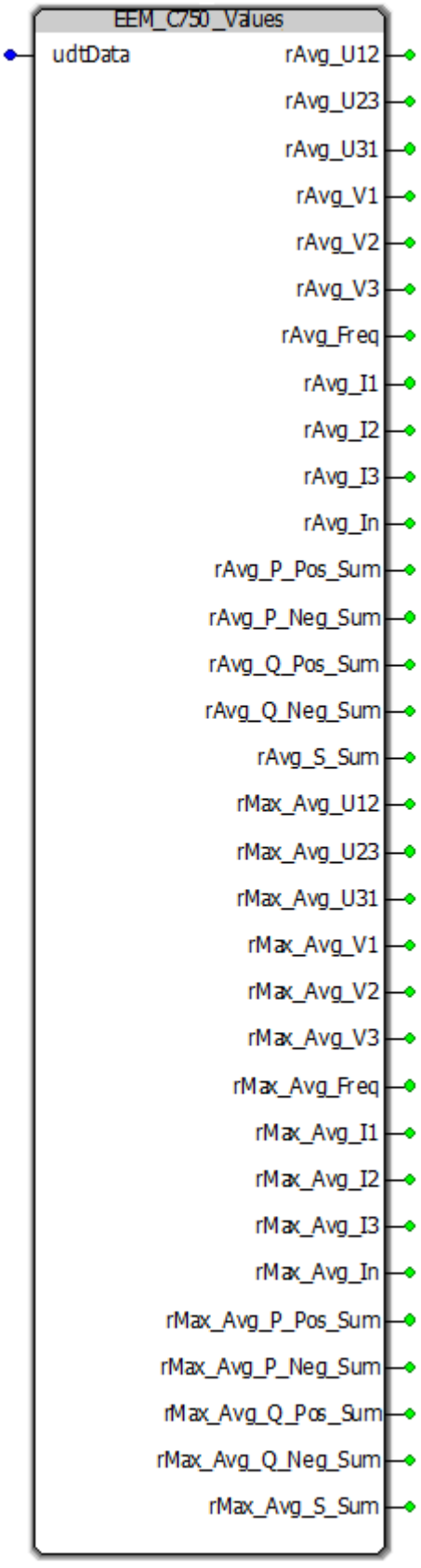
8.30.3 Output parameters

Name	Type	Description
OpHours	REAL	Operating hours counter.
rTotal_act_en_proc_sys	REAL	Total active energy procurement system.
rTotal_ind_react_en_sys	REAL	Total inductive reactive energy system.
rTotal_app_en_sys	REAL	Total apparent energy system.
rTotal_act_en_supply_sys	REAL	Total active energy supply system.
rTotal_cap_react_en_sys	REAL	Total capacitive reactive energy system.
rEaPos	REAL	Current positive real energy.
rErPos	REAL	Current positive reactive energy.
rEs	REAL	Current apparent energy.
rEaNeg	REAL	Current negative real energy.
rErNeg	REAL	Current negative reactive energy.

8.31 EEM_C750_Values

The function block EEM_C750_Values converts the EEM_UDT_C750 input structure into individual output parameters.

8.31.1 Function block call



8.31.2 Input parameters

Name	Type	Description
udtData	EEM_UDT_C750	Input structure for standardized values in the table.

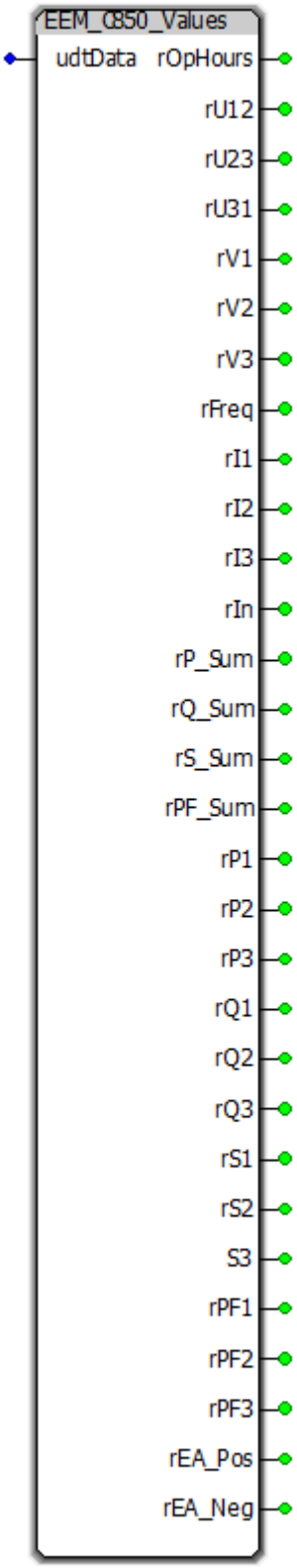
8.31.3 Output parameters

Name	Type	Description
rAvgU12	REAL	Average conductor voltage (L1-L2).
rAvgU23	REAL	Average conductor voltage (L2-L3).
rAvgU31	REAL	Average conductor voltage (L3-L1).
rAvgV1	REAL	Average conductor voltage (L1-N).
rAvgV2	REAL	Average conductor voltage (L2-N).
rAvgV3	REAL	Average conductor voltage (L3-N).
rAvgF	REAL	Average frequency.
rAvgI1	REAL	Average current L1.
rAvgI2	REAL	Average current L2.
rAvgI3	REAL	Average current L3.
rAvgIn	REAL	Average neutral conductor current.
rAvg_P_Pos_Sum	REAL	Average total positive real power.
rAvg_P_Neg_Sum	REAL	Average total negative real power.
rAvg_Q_Pos_Sum	REAL	Average total positive ractive power.
rAvg_Q_Neg_Sum	REAL	Average total negative ractive power.
rAvg_S_Sum	REAL	Average total apparent power.
rMax_Avg_U12	REAL	Maximum conductor voltage (L1-L2).
rMax_Avg_U23	REAL	Maximum conductor voltage (L2-L3).
rMax_Avg_U31	REAL	Maximum conductor voltage (L3-L1).
rMax_Avg_V1	REAL	Maximum conductor voltage (L1-N).
rMax_Avg_V2	REAL	Maximum conductor voltage (L2-N).
rMax_Avg_V3	REAL	Maximum conductor voltage (L3-N).
rMax_Avg_Freq	REAL	Maximum frequency.
rMax_Avg_I1	REAL	Maximum current L1.
rMax_Avg_I2	REAL	Maximum current L2.
rMax_Avg_I3	REAL	Maximum current L3.
rMax_Avg_In	REAL	Maximum current in neutral conductor.
rMax_Avg_P_Pos_Sum	REAL	Maximum total positive real power.
rMax_Avg_P_Neg_Sum	REAL	Maximum total negative real power.
rMax_Avg_Q_Pos_Sum	REAL	Maximum total positive reactive power.
rMax_Avg_Q_Neg_Sum	REAL	Maximum total negative reactive power.
rMax_Avg_S_Sum	REAL	Avg ΣS: average total apparent power value.

8.32 EEM_C850_Values

The function block EEM_C850_Values converts the EEM_UDT_C850 input structure into individual output parameters.

8.32.1 Function block call



8.32.2 Input parameters

Name	Type	Description
udtData	EEM_UDT_C850	Input structure for standardized values in the table.

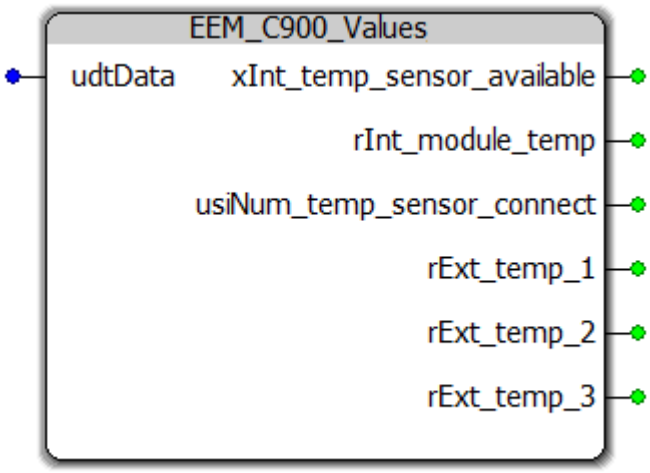
8.32.3 Output parameters

Name	Type	Description
OpHours	REAL	Operating hours counter.
rU12	REAL	Conductor voltage (L1-L2).
rU23	REAL	Conductor voltage (L2-L3).
rU31	REAL	Conductor voltage (L3-L1).
rV1	REAL	Conductor voltage (L1-N).
rV2	REAL	Conductor voltage (L2-N).
rV3	REAL	Conductor voltage (L3-N).
rFreq	REAL	Frequency.
rI1	REAL	Conductor current (L1).
rI2	REAL	Conductor current (L2).
rI3	REAL	Conductor current (L3).
rIn	REAL	Current in neutral conductor.
rP_Sum	REAL	Total real power +/-.
rQ_Sum	REAL	Total reactive power +/-.
rS_Sum	REAL	Total apparent power +/-.
rPF_Sum	REAL	Total power factor +/-.
rP1	REAL	Real power phase 1 +/-.
rP2	REAL	Real power phase 2 +/-.
rP3	REAL	Real power phase 3 +/-.
rQ1	REAL	Reactive power phase 1 +/-.
rQ2	REAL	Reactive power phase 2 +/-.
rQ3	REAL	Reactive power phase 3 +/-.
rS1	REAL	Apparent power phase 1 +/-.
rS2	REAL	Apparent power phase 2 +/-.
rS3	REAL	Apparent power phase 3 +/-.
rPF1	REAL	Power factor phase 1 +/-.
rPF2	REAL	Power factor phase 2 +/-.
rPF3	REAL	Power factor phase 3 +/-.
rEA_Pos	REAL	Total positive real energy since startup.
rEA_Neg	REAL	Total negative real energy since startup.

8.33 EEM_C900_Values

The function block EEM_C900_Values converts the EEM_UDT_C900 input structure into individual output parameters.

8.33.1 Function block call



8.33.2 Input parameters

Name	Type	Description
udtData	EEM_UDT_C900	Input structure for standardized values in the table.

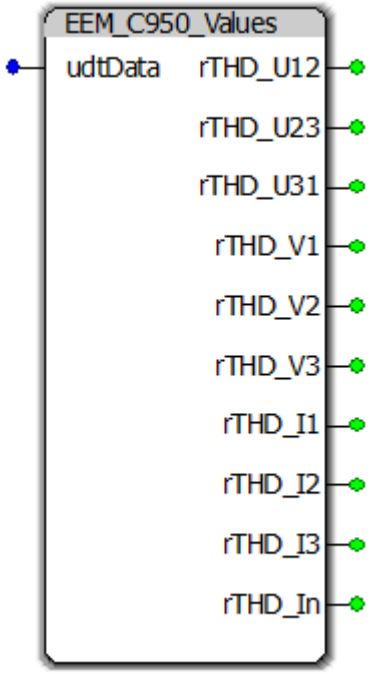
8.33.3 Output parameters

Name	Type	Description
xInt_temp_sensor_available	BOOL	Internal temperature sensor available.
rInt_module_temp	REAL	Internal module temperature.
usiNum_temp_sensor_connect	USINT	Number of temperature sensors connected.
rExtTemp1	REAL	External temperature 1
rExtTemp2	REAL	External temperature 2
rExtTemp3	REAL	External temperature 3

8.34 EEM_C950_Values

The function block EEM_C950_Values converts the EEM_UDT_C950 input structure into individual output parameters.

8.34.1 Function block call



8.34.2 Input parameters

Name	Type	Description
udtData	EEM_UDT_C950	Input structure for standardized values in the table.

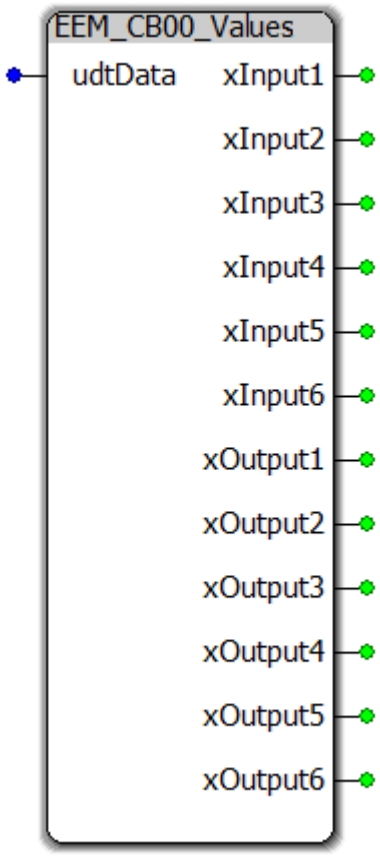
8.34.3 Output parameters

Name	Type	Description
rTHD_U12	REAL	Total harmonic distortion (L1-L2).
rTHD_U23	REAL	Total harmonic distortion (L2-L3).
rTHD_U31	REAL	Total harmonic distortion (L3-L1).
rTHD_V1	REAL	Total harmonic distortion (L1-N).
rTHD_V2	REAL	Total harmonic distortion (L2-N).
rTHD_V3	REAL	Total harmonic distortion (L3-N).
rTHD_I1	REAL	Total harmonic distortion (I1).
rTHD_I2	REAL	Total harmonic distortion (I2).
rTHD_I3	REAL	Total harmonic distortion (I3).

8.35 EEM_CB00_Values

The function block EEM_CB00_Values converts the EEM_UDT_CB00 input structure into individual output parameters.

8.35.1 Function block call



8.35.2 Input parameters

Name	Type	Description
udtData	EEM_UDT_CB00	Input structure for standardized values in the table.

8.35.3 Output parameters

Name	Type	Description
xInput1	BOOL	Digital input 1.
xInput2	BOOL	Digital input 2.
xInput3	BOOL	Digital input 3.
xInput4	BOOL	Digital input 4.
xInput5	BOOL	Digital input 5.
xInput6	BOOL	Digital input 6.
xOutput1	BOOL	Digital output 1.
xOutput2	BOOL	Digital output 2.
xOutput3	BOOL	Digital output 3.
xOutput4	BOOL	Digital output 4.
xOutput5	BOOL	Digital output 5.
xOutput6	BOOL	Digital output 6.

9 Startup examples

For the startup instruction of the EEM* function blocks please find the following examples:

- EEM_*_EXA_AXL_RSUNI.zwt
- EEM_*_EXA_TCP.zwt
- EEM_*_EXA_AXL_RSUNI_PN.zwt

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

They illustrate how to read out the current, voltage, power and energy values recorded by the Electric Energy Meters as well as electrical parameters from the various Modbus registers.

9.1 Example 1: EEM_*_EXA_AXL_RSUNI

In this example the tables 8000, 8300, 8600 and 9300 are read out via Modbus RTU with an EEM-MA771-R (2908285) device.

9.1.1 Bus structure

For this example, the following hardware is used:

- AXC 1050 (2700988)
- AXL F RS UNI 1H (2688666)
- EEM-MA771-R (2908285)

The hardware structure of the example is wired without electrical consumers, therefore not all tables are filled with values.

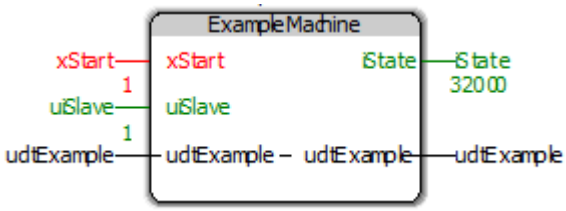
The EEM-MA771-R (2908285) device has to be parameterized like the AXL F RS UNI 1H (2688666) terminal.

9.1.2 Example description

Before starting the example, please specify the slave address at the uiSlave input to be communicated with.

9.1.2.1 Example machine

For starting ExampleMachine function block, xStart input has set to TRUE.



9.1.2.2 Example

```
CASE udtExample.iState OF
    0: (* Init *)
        IF xStart   = TRUE THEN
            (* Modbus master settings *)
            (* Transparent mode *)
            udtExample.udtMB_AXL_F_RSUNI_Master.xMode           := FALSE;
            (* Timeout for process data communication between PLC and module *)
            (* Default *)
            udtExample.udtMB_AXL_F_RSUNI_Master.tPD_Timeout     := TIME#2s;
            (* Defines the max time until Modbus response from
            slave must arrive *)
            (* Default *)
            udtExample.udtMB_AXL_F_RSUNI_Master.tModbus_Timeout := TIME#5s;
            (* Activation of MB_AXL_F_RSUNI_Master function block *)
            udtExample.udtMB_AXL_F_RSUNI_Master.xActivate       := TRUE;

            (* Slave address *)
            FOR iTab    := 1 TO 4 DO
                udtExample.arrMB_RTU_FC3[iTab].uiSlaveAddress   := UINT#1;
            END_FOR;
```

```
        (* Activation of MB_RTU_FC3 function blocks *)
        FOR iTab    := 1 TO 4 DO
            udtExample.arrMB_RTU_FC3[iTab].xActivate    :=TRUE;
        END_FOR;

        (* Checking for xActive of MB_AXL_F_RSUNI_Master *)
        IF udtExample.udtMB_AXL_F_RSUNI_Master.xActive  = TRUE THEN
            udtExample.iState   := 10;
        END_IF;
    END_IF;

10: (* Checking for xError *)
    IF udtExample.arrMB_RTU_FC3[1].xError   = TRUE THEN
        udtExample.iState   := 9000;
    (* Read values *)
    ELSIF udtExample.arrMB_RTU_FC3[1].xActive   = TRUE THEN
        udtExample.arrMB_RTU_FC3[1].xSendRequest    := TRUE;
        (* Checking for xDone *)
        IF udtExample.arrMB_RTU_FC3[1].xDone    = TRUE THEN
            (* Start reading data out of the Modbus structure *)
            udtExample.udtEEM_Tab8000_RTU.xExecute  := TRUE;

            udtExample.iState   := 20;
        END_IF;
    END_IF;

20: (* Checking for xError *)
    IF udtExample.arrMB_RTU_FC3[2].xError   = TRUE THEN
        udtExample.iState   := 9000;
    (* Read values *)
    ELSIF udtExample.arrMB_RTU_FC3[2].xActive   = TRUE THEN
        udtExample.arrMB_RTU_FC3[2].xSendRequest    := TRUE;
        (* Checking for xDone *)
        IF udtExample.arrMB_RTU_FC3[2].xDone    = TRUE THEN
            (* Start reading data out of the Modbus structure *)
            udtExample.udtEEM_Tab8300_RTU.xExecute  := TRUE;

            udtExample.iState   := 30;
        END_IF;
    END_IF;

30: (* Checking for xError *)
    IF udtExample.arrMB_RTU_FC3[3].xError   = TRUE THEN
        udtExample.iState   := 9000;
    (* Read values *)
    ELSIF udtExample.arrMB_RTU_FC3[3].xActive   = TRUE THEN
        udtExample.arrMB_RTU_FC3[3].xSendRequest    := TRUE;
        (* Checking for xDone *)
        IF udtExample.arrMB_RTU_FC3[3].xDone    = TRUE THEN
            (* Start reading data out of the Modbus structure *)
            udtExample.udtEEM_Tab8600_RTU.xExecute  := TRUE;

            udtExample.iState   := 40;
        END_IF;
    END_IF;

40: (* Checking for xError *)
    IF udtExample.arrMB_RTU_FC3[4].xError   = TRUE THEN
        udtExample.iState   := 9000;
    (* Read values *)
    ELSIF udtExample.arrMB_RTU_FC3[4].xActive   = TRUE THEN
        udtExample.arrMB_RTU_FC3[4].xSendRequest    := TRUE;
        (* Checking for xDone *)
        IF udtExample.arrMB_RTU_FC3[4].xDone    = TRUE THEN
```

```
                (* Start reading data out of the Modbus structure *)
                udtExample.udtEEM_Tab9300_RTU.xExecute  := TRUE;

                udtExample.iState   := 32000;
            END_IF;
        END_IF;

    9000: (* Error handling *)
        (* Insert your error handling here *)

        IF xStart   = FALSE THEN
            udtExample.iState   := 32000;
        END_IF;

    32000: (* Example is finished *)
        (* Deactivation of function blocks here *)
        udtExample.udtMB_AXL_F_RSUNI_Master.xActivate       := FALSE;
        udtExample.udtMB_AXL_F_RSUNI_Master.xMode           := FALSE;
        udtExample.udtMB_AXL_F_RSUNI_Master.tPD_Timeout     := TIME#0s;
        udtExample.udtMB_AXL_F_RSUNI_Master.tModbus_Timeout := TIME#0s;

        (* Tab8000 - Tab9300 *)
        FOR iTab    := 1 TO 4 DO
            udtExample.arrMB_RTU_FC3[iTab].xActivate        := FALSE;
            udtExample.arrMB_RTU_FC3[iTab].xSendRequest     := FALSE;
            udtExample.arrMB_RTU_FC3[iTab].uiSlaveAddress   := UINT#0;
        END_FOR;

        udtExample.udtEEM_Tab8000_RTU.xExecute  := FALSE;
        udtExample.udtEEM_Tab8300_RTU.xExecute  := FALSE;
        udtExample.udtEEM_Tab8600_RTU.xExecute  := FALSE;
        udtExample.udtEEM_Tab9300_RTU.xExecute  := FALSE;

        IF xStart   = FALSE THEN
            udtExample.iState   := 0;
        END_IF;
END_CASE;

iState  := udtExample.iState;
```

9.2 Example 2: EEM_*_EXA_TCP

In this example the tables 8000, 8300, 8600 and 9300 are read out via Modbus TCP with an EEM-MA771-R (2908285) device.

9.2.1 Bus structure

For this example, the following hardware is used:

- AXC 1050 (2700988)
- EEM-MA771-R (2908285)

The hardware structure of the example is wired without electrical consumers, therefore not all tables are filled with values.

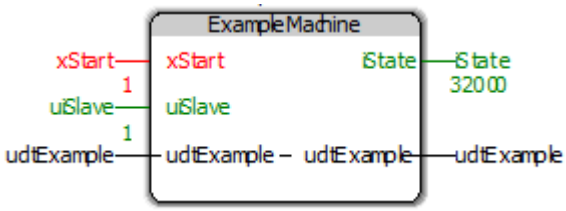
9.2.2 Example description

Before starting the example, please specify the slave address at the uiSlave input to be communicated with. This is the slave address of your EEM device. Default = 255. Please refer to the EEM datasheet for details.

For communication with the Modbus client you also need to specify the IP address of the EEM device at strIPAddress input.

9.2.2.1 Example machine

For starting ExampleMachine function block, xStart input has set to TRUE.



9.2.2.2 Example

```
CASE udtExample.iState OF

    0: (* Init *)
        IF xStart   = TRUE THEN
            (* Modbus client settings *)
            (* Modbus timeouts are acknowledged *)
            udtExample.udtMB_TCP_Client.xAcknowledge    := FALSE;
            (* Automatic acknowledgment of modbus timeouts *)
            udtExample.udtMB_TCP_Client.xAutoAck        := FALSE;
            (* IP address of the Modbus server *)
            udtExample.udtMB_TCP_Client.strServerIP     := '192.168.1.2';
            (* Port of the Modbus server *)
            udtExample.udtMB_TCP_Client.iPort           := INT#502;
            (* Modbus-TCP is used *)
            udtExample.udtMB_TCP_Client.xUDP_Mode       := FALSE;
            (* Delay between two IP connect executions *)
            udtExample.udtMB_TCP_Client.tReconnectDelay := TIME#0s;
            (* Timeout for communication monitoring *)
            udtExample.udtMB_TCP_Client.tTimeout        := TIME#10s;
            (* Activation of client function block *)
```

```
        udtExample.udtMB_TCP_Client.xActivate       := TRUE;
        (* MB_TCP_FC3 settings *)

        FOR iTab    := 1 TO 4 DO
            (* Parameter for addressing of a Modbus TCP device *)
            udtExample.arrMB_TCP_FC3[iTab].uiUnitIdentifier := UINT#1;
            (* Waiting time in ms between two read tasks *)
            udtExample.arrMB_TCP_FC3[iTab].tUpdateTime      := TIME#5s;
            (* Represents the connection between the function blocks *)
            udtExample.arrMB_TCP_FC3[iTab].iMT_ID           := iTab;
        END_FOR;
        (* Checking sucessful connection *)
        IF udtExample.udtMB_TCP_Client.xReady   = TRUE THEN
            FOR iTab    := 1 TO 4 DO
                udtExample.arrMB_TCP_FC3[iTab].xActivate    := TRUE;
                IF udtExample.arrMB_TCP_FC3[iTab].xActive   = TRUE THEN
                    udtExample.iState   := 100;
                END_IF;
            END_FOR;
        END_IF;
    END_IF;

100: (* Tab8000: checking of xError *)
    IF udtExample.arrMB_TCP_FC3[1].xError   = TRUE THEN
        (* Read values *)
        udtExample.iState   := 9000;
    ELSIF
        udtExample.arrMB_TCP_FC3[1].xNDR        = TRUE AND
        udtExample.arrMB_TCP_FC3[1].uiByteCount >UINT#0
    THEN
        udtExample.udtEEM_Tab8000_TCP.xExecute  := TRUE;
        udtExample.iState   := 200;
    END_IF;

200: (* Tab8300: checking of xError *)
    IF udtExample.arrMB_TCP_FC3[2].xError   = TRUE THEN
        (* Read values *)
        udtExample.iState   := 9000;
    ELSIF
        udtExample.arrMB_TCP_FC3[2].xNDR        = TRUE AND
        udtExample.arrMB_TCP_FC3[2].uiByteCount >UINT#0
    THEN
        udtExample.udtEEM_Tab8300_TCP.xExecute  := TRUE;
        udtExample.iState   := 300;
    END_IF;

300: (* Tab8600: checking of xError *)
    IF udtExample.arrMB_TCP_FC3[3].xError   = TRUE THEN
        (* Read values *)
        udtExample.iState   := 9000;
    ELSIF
        udtExample.arrMB_TCP_FC3[3].xNDR        = TRUE AND
        udtExample.arrMB_TCP_FC3[3].uiByteCount >UINT#0
    THEN
        udtExample.udtEEM_Tab8600_TCP.xExecute  := TRUE;
        udtExample.iState   := 400;
    END_IF;

400: (* Tab9300: checking of xError *)
    IF udtExample.arrMB_TCP_FC3[4].xError   = TRUE THEN
        (* Read values *)
        udtExample.iState   := 9000;
    ELSIF
        udtExample.arrMB_TCP_FC3[4].xNDR        = TRUE AND
```

```
            udtExample.arrMB_TCP_FC3[4].uiByteCount >UINT#0
        THEN
            udtExample.udtEEM_Tab9300_TCP.xExecute  := TRUE;
            udtExample.iState   := 32000;
        END_IF;

    9000: (* Error handling *)
        (* Insert your error handling here *)

        IF xStart   = FALSE THEN
            udtExample.iState   := 32000;
        END_IF;

    32000: (* Example is finished *)
        (* Deactivation of function blocks *)
        udtExample.udtMB_TCP_Client.xActivate       := FALSE;
        udtExample.udtMB_TCP_Client.xAcknowledge    := FALSE;
        udtExample.udtMB_TCP_Client.xAutoAck        := FALSE;
        udtExample.udtMB_TCP_Client.strServerIP     := '';
        udtExample.udtMB_TCP_Client.iPort           := INT#0;
        udtExample.udtMB_TCP_Client.xUDP_Mode       := FALSE;
        udtExample.udtMB_TCP_Client.tReconnectDelay := TIME#0s;
        udtExample.udtMB_TCP_Client.tTimeout        := TIME#0s;
        udtExample.udtMB_TCP_Client.xActivate       := FALSE;

        FOR iTab    := 1 TO 4 DO
            udtExample.arrMB_TCP_FC3[iTab].xActivate    := FALSE;
        END_FOR;

        FOR iTab    := 1 TO 4 DO
            udtExample.arrMB_TCP_FC3[iTab].uiUnitIdentifier := UINT#0;
            udtExample.arrMB_TCP_FC3[iTab].tUpdateTime      := TIME#0s;
            udtExample.arrMB_TCP_FC3[iTab].iMT_ID           := INT#0;
        END_FOR;

        udtExample.udtEEM_Tab8000_TCP.xExecute  := FALSE;
        udtExample.udtEEM_Tab8300_TCP.xExecute  := FALSE;
        udtExample.udtEEM_Tab8600_TCP.xExecute  := FALSE;
        udtExample.udtEEM_Tab9300_TCP.xExecute  := FALSE;

        IF xStart   = FALSE THEN
            udtExample.iState   := 0;
        END_IF;

END_CASE;

iState  := udtExample.iState;
```

9.3 Example 3: EEM_*_EXA_AXL_RSUNI_PN

9.3.1 Bus structure

For this example, the following hardware is used:

- AXC 1050 (2700988)
- AXL F BK PN (2701815)
- AXL F RS UNI 1H (2688666)
- EEM-MA771-R (2908285)

The hardware structure of the example is wired without electrical consumers, therefore not all tables are filled with values.

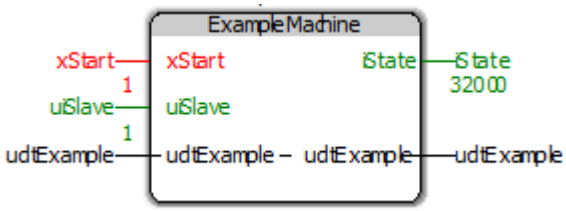
The EEM-MA771-R (2908285) device has to be parameterized like the AXL F RS UNI 1H (2688666) terminal.

9.3.2 Example description

Before starting the example, please specify the slave address at the uiSlave input to be communicated with.

9.3.2.1 Example machine

For starting ExampleMachine function block, xStart input has set to TRUE.



9.3.2.2 Example

Refer to Example 1: EEM_*_EXA_AXL_RSUNI

10 Appendix

10.1 Data types

```
TYPE

    EEM_UDT_0000 : STRUCT
    (* Instantaneous values (integer) *)
        udiCond_voltage_to_N_U1     : UDINT;
        udiCond_voltage_to_N_U2     : UDINT;
        udiCond_voltage_to_N_U3     : UDINT;
        udiCond_voltage_to_cond_U12 : UDINT;
        udiCond_voltage_to_cond_U23 : UDINT;
        udiCond_voltage_to_cond_U31 : UDINT;
        udiExt_cond_voltage_system  : UDINT;
        diCurrent_in_cond_I1        : DINT;
        diCurrent_in_cond_I2        : DINT;
        diCurrent_in_cond_I3        : DINT;
        diCurrent_in_cond_In        : DINT;
        diSys_Current               : DINT;
        diPower_Factor_phase_1      : DINT;
        diPower_Factor_phase_2      : DINT;
        diPower_Factor_phase_3      : DINT;
        diSys_Power_Factor          : DINT;
        diFrequency                 : DINT;
        diPhase_sequence            : DINT;
    END_STRUCT;

    EEM_UDT_0100 : STRUCT
    (* Meter (integer) *)
        udiCond_voltage_to_N_U1     : UDINT;
    END_STRUCT;

    EEM_UDT_0370 : STRUCT
    (* Display of current average current/voltage.
    value based on current and voltage transducer
    (under consideration of transmission ratios *)
        rI_SYS      : REAL;
        rU_SYS      : REAL;
        rV_SYS      : REAL;
    END_STRUCT;

    EEM_UDT_0741 : STRUCT
    (* Display of current average current/voltage
    value not based on current and voltage transducer
    (without consideration of transmission ratios) *)
        rI_SYS      : REAL;
        rU_SYS      : REAL;
        rV_SYS      : REAL;
    END_STRUCT;

    EEM_UDT_1000 : STRUCT
    (* Instantaneous values (float) *)
        rCond_voltage_to_N_U1       : REAL;
        rCond_voltage_to_N_U2       : REAL;
        rCond_voltage_to_N_U3       : REAL;
        rCond_voltage_to_cond_U12   : REAL;
        rCond_voltage_to_cond_U23   : REAL;
        rCond_voltage_to_cond_U31   : REAL;
        rExt_cond_voltage_system    : REAL;
        rCurrent_in_cond_I1         : REAL;
        rCurrent_in_cond_I2         : REAL;
```

```
    rCurrent_in_cond_I3         : REAL;
    rCurrent_in_cond_In         : REAL;
    rSys_Current                : REAL;
    rPower_factor_phase_1       : REAL;
    rPower_factor_phase_2       : REAL;
    rPower_factor_phase_3       : REAL;
    rSys_power_factor           : REAL;
    rActive_power_phase_1       : REAL;
    rActive_power_phase_2       : REAL;
    rActive_power_phase_3       : REAL;
    rTotal_active_power         : REAL;
    rApp_power_phase_1          : REAL;
    rApp_power_phase_2          : REAL;
    rApp_power_phase_3          : REAL;
    rTotal_app_power            : REAL;
    rReactive_power_phase_1     : REAL;
    rReactive_power_phase_2     : REAL;
    rReactive_power_phase_3     : REAL;
    rTotal_reactive_power       : REAL;
    rFrequency                  : REAL;
    rPhase_sequence             : REAL;
END_STRUCT;

EEM_UDT_1100 : STRUCT
(* Meter (float) *)
    rTotal_act_energy_proc_L1   : REAL;
    rTotal_act_energy_proc_L2   : REAL;
    rTotal_act_energy_proc_L3   : REAL;
    rTotal_act_energy_proc_S    : REAL;
    rTotal_act_energy_supp_L1   : REAL;
    rTotal_act_energy_supp_L2   : REAL;
    rTotal_act_energy_supp_L3   : REAL;
    rTotal_act_energy_supp_S    : REAL;
    rTotal_ind_app_en_proc_L1   : REAL;
    rTotal_ind_app_en_proc_L2   : REAL;
    rTotal_ind_app_en_proc_L3   : REAL;
    rTotal_ind_app_en_proc_S    : REAL;
    rTotal_ind_app_en_supp_L1   : REAL;
    rTotal_ind_app_en_supp_L2   : REAL;
    rTotal_ind_app_en_supp_L3   : REAL;
    rTotal_ind_app_en_supp_S    : REAL;
    rTotal_cap_app_en_proc_L1   : REAL;
    rTotal_cap_app_en_proc_L2   : REAL;
    rTotal_cap_app_en_proc_L3   : REAL;
    rTotal_cap_app_en_proc_S    : REAL;
    rTotal_cap_app_en_supp_L1   : REAL;
    rTotal_cap_app_en_supp_L2   : REAL;
    rTotal_cap_app_en_supp_L3   : REAL;
    rTotal_cap_app_en_supp_S    : REAL;
    rTotal_ind_reac_en_proc_L1  : REAL;
    rTotal_ind_reac_en_proc_L2  : REAL;
    rTotal_ind_reac_en_proc_L3  : REAL;
    rTotal_ind_reac_en_proc_S   : REAL;
    rTotal_ind_reac_en_supp_L1  : REAL;
    rTotal_ind_reac_en_supp_L2  : REAL;
    rTotal_ind_reac_en_supp_L3  : REAL;
    rTotal_ind_reac_en_supp_S   : REAL;
    rTotal_cap_reac_en_proc_L1  : REAL;
    rTotal_cap_reac_en_proc_L2  : REAL;
    rTotal_cap_reac_en_proc_L3  : REAL;
    rTotal_cap_reac_en_proc_S   : REAL;
    rTotal_cap_reac_en_supp_L1  : REAL;
    rTotal_cap_reac_en_supp_L2  : REAL;
    rTotal_cap_reac_en_supp_L3  : REAL;
```

```
    rTotal_cap_reac_en_supp_S   : REAL;
END_STRUCT;

EEM_UDT_2300 : STRUCT
(* Extension module //Extern Gateway
    Proficloud *)
    uiIPv4_add_octetA       : UINT;
    uiIPv4_add_octetB       : UINT;
    uiIPv4_add_octetC       : UINT;
    uiIPv4_add_octetD       : UINT;
    uiIPv4_sub_mask_octetA  : UINT;
    uiIPv4_sub_mask_octetB  : UINT;
    uiIPv4_sub_mask_octetC  : UINT;
    uiIPv4_sub_mask_octetD  : UINT;
    uiIPv4_def_gw_octetA    : UINT;
    uiIPv4_def_gw_octetB    : UINT;
    uiIPv4_def_gw_octetC    : UINT;
    uiIPv4_def_gw_octetD    : UINT;
    uiIPv4_DNS_serv_octetA  : UINT;
    uiIPv4_DNS_serv_octetB  : UINT;
    uiIPv4_DNS_serv_octetC  : UINT;
    uiIPv4_DNS_serv_octetD  : UINT;
    uiIPv4_add_assign_mode  : UINT;
    xProficloud_gw_act      : BOOL;
    xProficloud_gw_FTP_act  : BOOL;
END_STRUCT;

EEM_UDT_2400 : STRUCT
(* Extension module //Extern Gateway Profinet /
    Ethernet/IP *)
    uiIPv4_add_octetA       : UINT;
    uiIPv4_add_octetB       : UINT;
    uiIPv4_add_octetC       : UINT;
    uiIPv4_add_octetD       : UINT;
    uiIPv4_sub_mask_octetA  : UINT;
    uiIPv4_sub_mask_octetB  : UINT;
    uiIPv4_sub_mask_octetC  : UINT;
    uiIPv4_sub_mask_octetD  : UINT;
    uiIPv4_def_gw_octetA    : UINT;
    uiIPv4_def_gw_octetB    : UINT;
    uiIPv4_def_gw_octetC    : UINT;
    uiIPv4_def_gw_octetD    : UINT;
END_STRUCT;

EEM_UDT_8000 : STRUCT
(* Instantaneous values (float) *)
    rPhase_cond_voltage_U12     : REAL;
    rPhase_cond_voltage_U23     : REAL;
    rPhase_cond_voltage_U31     : REAL;
    rPhase_voltage_U1           : REAL;
    rPhase_voltage_U2           : REAL;
    rPhase_voltage_U3           : REAL;
    rFrequency                  : REAL;
    rCurrent_I1                 : REAL;
    rCurrent_I2                 : REAL;
    rCurrent_I3                 : REAL;
    rCurrent_In                 : REAL;
    rTotal_active_power         : REAL;
    rTotal_reactive_power_vect  : REAL;
    rTotal_app_power_vect       : REAL;
    rTotal_power_factor_vect    : REAL;
    rActive_power_phase_1       : REAL;
    rActive_power_phase_2       : REAL;
    rActive_power_phase_3       : REAL;
```

```
    rReactive_power_phase_1     : REAL;
    rReactive_power_phase_2     : REAL;
    rReactive_power_phase_3     : REAL;
    rApp_power_phase_1          : REAL;
    rApp_power_phase_2          : REAL;
    rApp_power_phase_3          : REAL;
    rPower_factor_phase_1       : REAL;
    rPower_factor_phase_2       : REAL;
    rPower_factor_phase_3       : REAL;
    iPF_Ind_Cap_1               : INT;
    iPF_Ind_Cap_2               : INT;
    iPF_Ind_Cap_3               : INT;
    rSys_Current                : REAL;
    rExt_cond_voltage_system    : REAL;
    rSys_cond_voltage_system    : REAL;
    rTotal_react_power_arith    : REAL;
    rTotal_app_power_arith      : REAL;
    rTotal_power_factor_arith   : REAL;
    rPHI_UI1                    : REAL;
    rPHI_UI2                    : REAL;
    rPHI_UI3                    : REAL;
    rCos_PHI_1                  : REAL;
    rCos_PHI_2                  : REAL;
    rCos_PHI_3                  : REAL;
    rPHI_U1                     : REAL;
    rPHI_U2                     : REAL;
    rPHI_U3                     : REAL;
    rRotating_field             : REAL;
END_STRUCT;

EEM_UDT_8100 : STRUCT
(* Energy meter values *)
    rTotal_act_energy_demand    : REAL;
    rTotal_react_energy_demand  : REAL;
    rTotal_app_energy           : REAL;
    rTotal_act_energy_deliv     : REAL;
    rTotal_react_energy_deliv   : REAL;
    rTotal_react_energy         : REAL;
    rCurrent_act_energy_demand  : REAL;
    rCurrent_react_energy_demand: REAL;
    rCurrent_app_energy         : REAL;
    rCurrent_act_energy_deliv   : REAL;
    rCurrent_react_energy_deliv : REAL;
    rCurrent_react_energy       : REAL;
END_STRUCT;

EEM_UDT_8200 : STRUCT
(* rTariff meter values *)
    rTariff_number              : REAL;
    rT1_act_energy_demand       : REAL;
    rT1_act_energy_deliv        : REAL;
    rT1_react_energy_demand     : REAL;
    rT1_react_energy_deliv      : REAL;
    rT1_app_energy              : REAL;
    rT1_react_energy            : REAL;
    rT2_act_energy_demand       : REAL;
    rT2_act_energy_deliv        : REAL;
    rT2_react_energy_demand     : REAL;
    rT2_react_energy_deliv      : REAL;
    rT2_app_energy              : REAL;
    rT2_react_energy            : REAL;
    rT3_act_energy_demand       : REAL;
    rT3_act_energy_deliv        : REAL;
    rT3_react_energy_demand     : REAL;
```

```
    rT3_react_energy_deliv      : REAL;
    rT3_app_energy              : REAL;
    rT3_react_energy            : REAL;
    rT4_act_energy_demand       : REAL;
    rT4_act_energy_deliv        : REAL;
    rT4_react_energy_demand     : REAL;
    rT4_react_energy_deliv      : REAL;
    rT4_app_energy              : REAL;
    rT4_react_energy            : REAL;
END_STRUCT;

EEM_UDT_8300 : STRUCT
(* Operating hours *)
    udiOpHours      : UDINT;
    udiExt_OpHours  : UDINT;
END_STRUCT;

EEM_UDT_8340 : STRUCT
(* Impulse counter of digital input *)
    udiImpulse_Counter  : UDINT;
END_STRUCT;

EEM_UDT_8500 : STRUCT
(* Statistics (prediction) *)
    rTrend_PPR_interval         : REAL;
    rTrend_QPR_interval         : REAL;
    rTrend_SPR_interval         : REAL;
    rCurr_Ea_pos_m_interval     : REAL;
    rCurr_Ea_neg_m_interval     : REAL;
    rCurr_Er_pos_m_interval     : REAL;
    rCurr_Er_neg_m_interval     : REAL;
    rCurr_Esm_interval          : REAL;
END_STRUCT;

EEM_UDT_8540 : STRUCT
(* Statistics (mean values) *)
    rAVG_phase_cond_voltage_U12 : REAL;
    rAVG_phase_cond_voltage_U23 : REAL;
    rAVG_phase_cond_voltage_U31 : REAL;
    rAVG_phase_voltage_U1       : REAL;
    rAVG_phase_voltage_U2       : REAL;
    rAVG_phase_voltage_U3       : REAL;
    rAVG_frequency              : REAL;
    rAVG_current_I1             : REAL;
    rAVG_power_I2               : REAL;
    rAVG_power_I3               : REAL;
    rAVG_power_IN               : REAL;
    rAVG_act_power_demand       : REAL;
    rAVG_act_power_delivery     : REAL;
    rAVG_react_power_demand     : REAL;
    rAVG_react_power_delivery   : REAL;
    rAVG_app_power              : REAL;
END_STRUCT;

EEM_UDT_8600 : STRUCT
(* Min / max values for mean values *)
    rMax_AVG_phase_cond_volt_U12    : REAL;
    rMax_AVG_phase_cond_volt_U23    : REAL;
    rMax_AVG_phase_cond_volt_U31    : REAL;
    rMax_AVG_phase_volt_U1          : REAL;
    rMax_AVG_phase_volt_U2          : REAL;
    rMax_AVG_phase_volt_U3          : REAL;
    rMax_AVG_frequency              : REAL;
    rMax_AVG_current_I1             : REAL;
```

```
    rMax_AVG_current_I2             : REAL;
    rMax_AVG_current_I3             : REAL;
    rMax_AVG_current_IN             : REAL;
    rMax_AVG_act_power_demand       : REAL;
    rMax_AVG_act_power_delivery     : REAL;
    rMax_AVG_react_power_demand     : REAL;
    rMax_AVG_react_power_delivery   : REAL;
    rMax_AVG_app_power              : REAL;
    rMin_AVG_phase_cond_volt_U12    : REAL;
    rMin_AVG_phase_cond_volt_U23    : REAL;
    rMin_AVG_phase_cond_volt_U31    : REAL;
    rMin_AVG_phase_volt_U1          : REAL;
    rMin_AVG_phase_volt_U2          : REAL;
    rMin_AVG_phase_volt_U3          : REAL;
    rMin_AVG_frequency              : REAL;
    rMin_AVG_current_I1             : REAL;
    rMin_AVG_current_I2             : REAL;
    rMin_AVG_current_I3             : REAL;
    rMin_AVG_current_IN             : REAL;
    rMin_AVG_act_power_demand       : REAL;
    rMin_AVG_act_power_delivery     : REAL;
    rMin_AVG_react_power_demand     : REAL;
    rMin_AVG_react_power_delivery   : REAL;
    rMin_AVG_app_power              : REAL;
END_STRUCT;

EEM_UDT_8700 : STRUCT
(* Min / max values for instantaneous values *)
    rMax_value_phase_cond_volt_U12  : REAL;
    rMax_value_phase_cond_volt_U23  : REAL;
    rMax_value_phase_cond_volt_U31  : REAL;
    rMax_value_phase_volt_U1        : REAL;
    rMax_value_phase_volt_U2        : REAL;
    rMax_value_phase_volt_U3        : REAL;
    rMax_value_frequency            : REAL;
    rMax_value_current_I1           : REAL;
    rMax_value_current_I2           : REAL;
    rMax_value_current_I3           : REAL;
    rMax_value_current_IN           : REAL;
    rMax_value_act_power_demand     : REAL;
    rMax_value_act_power_deliv      : REAL;
    rMax_value_react_power_demand   : REAL;
    rMax_value_react_power_deliv    : REAL;
    rMax_value_app_power            : REAL;
    rMin_value_phase_cond_volt_U12  : REAL;
    rMin_value_phase_cond_volt_U23  : REAL;
    rMin_value_phase_cond_volt_U31  : REAL;
    rMin_value_phase_volt_U1        : REAL;
    rMin_value_phase_volt_U2        : REAL;
    rMin_value_phase_volt_U3        : REAL;
    rMin_value_Frequency            : REAL;
    rMin_value_current_I1           : REAL;
    rMin_value_current_I2           : REAL;
    rMin_value_current_I3           : REAL;
    rMin_value_current_IN           : REAL;
    rMin_value_act_power_demand     : REAL;
    rMin_value_act_power_deliv      : REAL;
    rMin_value_react_power_demand   : REAL;
    rMin_value_react_power_deliv    : REAL;
    rMin_value_app_power            : REAL;
END_STRUCT;

EEM_UDT_8800 : STRUCT
(* Total harmonic distortion *)
```

```
    iTHD_phase_cond_volt_U12    : INT;
    iTHD_phase_cond_volt_U23    : INT;
    iTHD_phase_cond_volt_U31    : INT;
    iTHD_phase_volt_U1          : INT;
    iTHD_phase_volt_U2          : INT;
    iTHD_phase_volt_U3          : INT;
    iTHD_current_I1             : INT;
    iTHD_current_I2             : INT;
    iTHD_current_I3             : INT;
    iTHD_current_IN             : INT;
END_STRUCT;

EEM_UDT_9000 : STRUCT
(* Voltage equality // Voltage dips *)
    uiDip_ph_cond_v_source_n_1  : UINT;
    rDip_ph_cond_v_value_n_1    : REAL;
    uiDip_ph_v_source_n_1       : UINT;
    rDip_ph_v_value_n_1         : REAL;
    udiDip_duration_n_1         : UDINT;
    uiDip_timestamp_day_n_1     : UINT;
    uiDip_timestamp_month_n_1   : UINT;
    uiDip_timestamp_year_n_1    : UINT;
    uiDip_timestamp_hour_n_1    : UINT;
    uiDip_timestamp_minute_n_1  : UINT;
    uiDip_timestamp_second_n_1  : UINT;
    uiDip_ph_cond_v_source_n_2  : UINT;
    rDip_ph_cond_v_value_n_2    : REAL;
    uiDip_ph_v_source_n_2       : UINT;
    rDip_ph_v_value_n_2         : REAL;
    udiDip_duration_n_2         : UDINT;
    uiDip_timestamp_day_n_2     : UINT;
    uiDip_timestamp_month_n_2   : UINT;
    uiDip_timestamp_year_n_2    : UINT;
    uiDip_timestamp_hour_n_2    : UINT;
    uiDip_timestamp_minute_n_2  : UINT;
    uiDip_timestamp_second_n_2  : UINT;
    uiDip_ph_cond_v_source_n_3  : UINT;
    rDip_ph_cond_v_value_n_3    : REAL;
    uiDip_ph_v_source_n_3       : UINT;
    rDip_ph_v_value_n_3         : REAL;
    udiDip_duration_n_3         : UDINT;
    uiDip_timestamp_day_n_3     : UINT;
    uiDip_timestamp_month_n_3   : UINT;
    uiDip_timestamp_year_n_3    : UINT;
    uiDip_timestamp_hour_n_3    : UINT;
    uiDip_timestamp_minute_n_3  : UINT;
    uiDip_timestamp_second_n_3  : UINT;
    uiDip_ph_cond_v_source_n_4  : UINT;
    rDip_ph_cond_v_value_n_4    : REAL;
    uiDip_ph_v_source_n_4       : UINT;
    rDip_ph_v_value_n_4         : REAL;
    udiDip_duration_n_4         : UDINT;
    uiDip_timestamp_day_n_4     : UINT;
    uiDip_timestamp_month_n_4   : UINT;
    uiDip_timestamp_year_n_4    : UINT;
    uiDip_timestamp_hour_n_4    : UINT;
    uiDip_timestamp_minute_n_4  : UINT;
    uiDip_timestamp_second_n_4  : UINT;
    uiDip_ph_cond_v_source_n_5  : UINT;
    rDip_ph_cond_v_value_n_5    : REAL;
    uiDip_ph_v_source_n_5       : UINT;
    rDip_ph_v_value_n_5         : REAL;
    udiDip_duration_n_5         : UDINT;
    uiDip_timestamp_day_n_5     : UINT;
```

```
    uiDip_timestamp_month_n_5   : UINT;
    uiDip_timestamp_year_n_5    : UINT;
    uiDip_timestamp_hour_n_5    : UINT;
    uiDip_timestamp_minute_n_5  : UINT;
    uiDip_timestamp_second_n_5  : UINT;
END_STRUCT;

EEM_UDT_9046 : STRUCT
(* Voltage equality // Voltage dips *)
    uiDip_ph_cond_v_source_n_6  : UINT;
    rDip_ph_cond_v_value_n_6    : REAL;
    uiDip_ph_v_source_n_6       : UINT;
    rDip_ph_v_value_n_6         : REAL;
    udiDip_duration_n_6         : UDINT;
    uiDip_timestamp_day_n_6     : UINT;
    uiDip_timestamp_month_n_6   : UINT;
    uiDip_timestamp_year_n_6    : UINT;
    uiDip_timestamp_hour_n_6    : UINT;
    uiDip_timestamp_minute_n_6  : UINT;
    uiDip_timestamp_second_n_6  : UINT;
    uiDip_ph_cond_v_source_n_7  : UINT;
    rDip_ph_cond_v_value_n_7    : REAL;
    uiDip_ph_v_source_n_7       : UINT;
    rDip_ph_v_value_n_7         : REAL;
    udiDip_duration_n_7         : UDINT;
    uiDip_timestamp_day_n_7     : UINT;
    uiDip_timestamp_month_n_7   : UINT;
    uiDip_timestamp_year_n_7    : UINT;
    uiDip_timestamp_hour_n_7    : UINT;
    uiDip_timestamp_minute_n_7  : UINT;
    uiDip_timestamp_second_n_7  : UINT;
    uiDip_ph_cond_v_source_n_8  : UINT;
    rDip_ph_cond_v_value_n_8    : REAL;
    uiDip_ph_v_source_n_8       : UINT;
    rDip_ph_v_value_n_8         : REAL;
    udiDip_duration_n_8         : UDINT;
    uiDip_timestamp_day_n_8     : UINT;
    uiDip_timestamp_month_n_8   : UINT;
    uiDip_timestamp_year_n_8    : UINT;
    uiDip_timestamp_hour_n_8    : UINT;
    uiDip_timestamp_minute_n_8  : UINT;
    uiDip_timestamp_second_n_8  : UINT;
    uiDip_ph_cond_v_source_n_9  : UINT;
    rDip_ph_cond_v_value_n_9    : REAL;
    uiDip_ph_v_source_n_9       : UINT;
    rDip_ph_v_value_n_9         : REAL;
    udiDip_duration_n_9         : UDINT;
    uiDip_timestamp_day_n_9     : UINT;
    uiDip_timestamp_month_n_9   : UINT;
    uiDip_timestamp_year_n_9    : UINT;
    uiDip_timestamp_hour_n_9    : UINT;
    uiDip_timestamp_minute_n_9  : UINT;
    uiDip_timestamp_second_n_9  : UINT;
    uiDip_ph_cond_v_source_n_10 : UINT;
    rDip_ph_cond_v_value_n_10   : REAL;
    uiDip_ph_v_source_n_10      : UINT;
    rDip_ph_v_value_n_10        : REAL;
    udiDip_duration_n_10        : UDINT;
    uiDip_timestamp_day_n_10    : UINT;
    uiDip_timestamp_month_n_10  : UINT;
    uiDip_timestamp_year_n_10   : UINT;
    uiDip_timestamp_hour_n_10   : UINT;
    uiDip_timestamp_minute_n_10 : UINT;
    uiDip_timestamp_second_n_10 : UINT;
```

```
END_STRUCT;

EEM_UDT_9100 : STRUCT
(* Voltage swells *)
    uiSwl_ph_cond_v_source_n_1  : UINT;
    rSwl_ph_cond_v_value_n_1    : REAL;
    uiSwl_ph_v_source_n_1       : UINT;
    rSwl_ph_v_value_n_1         : REAL;
    udiSwl_duration_n_1         : UDINT;
    uiSwl_timestamp_day_n_1     : UINT;
    uiSwl_timestamp_month_n_1   : UINT;
    uiSwl_timestamp_year_n_1    : UINT;
    uiSwl_timestamp_hour_n_1    : UINT;
    uiSwl_timestamp_minute_n_1  : UINT;
    uiSwl_timestamp_second_n_1  : UINT;
    uiSwl_ph_cond_v_source_n_2  : UINT;
    rSwl_ph_cond_v_value_n_2    : REAL;
    uiSwl_ph_v_source_n_2       : UINT;
    rSwl_ph_v_value_n_2         : REAL;
    udiSwl_duration_n_2         : UDINT;
    uiSwl_timestamp_day_n_2     : UINT;
    uiSwl_timestamp_month_n_2   : UINT;
    uiSwl_timestamp_year_n_2    : UINT;
    uiSwl_timestamp_hour_n_2    : UINT;
    uiSwl_timestamp_minute_n_2  : UINT;
    uiSwl_timestamp_second_n_2  : UINT;
    uiSwl_ph_cond_v_source_n_3  : UINT;
    rSwl_ph_cond_v_value_n_3    : REAL;
    uiSwl_ph_v_source_n_3       : UINT;
    rSwl_ph_v_value_n_3         : REAL;
    udiSwl_duration_n_3         : UDINT;
    uiSwl_timestamp_day_n_3     : UINT;
    uiSwl_timestamp_month_n_3   : UINT;
    uiSwl_timestamp_year_n_3    : UINT;
    uiSwl_timestamp_hour_n_3    : UINT;
    uiSwl_timestamp_minute_n_3  : UINT;
    uiSwl_timestamp_second_n_3  : UINT;
    uiSwl_ph_cond_v_source_n_4  : UINT;
    rSwl_ph_cond_v_value_n_4    : REAL;
    uiSwl_ph_v_source_n_4       : UINT;
    rSwl_ph_v_value_n_4         : REAL;
    udiSwl_duration_n_4         : UDINT;
    uiSwl_timestamp_day_n_4     : UINT;
    uiSwl_timestamp_month_n_4   : UINT;
    uiSwl_timestamp_year_n_4    : UINT;
    uiSwl_timestamp_hour_n_4    : UINT;
    uiSwl_timestamp_minute_n_4  : UINT;
    uiSwl_timestamp_second_n_4  : UINT;
    uiSwl_ph_cond_v_source_n_5  : UINT;
    rSwl_ph_cond_v_value_n_5    : REAL;
    uiSwl_ph_v_source_n_5       : UINT;
    rSwl_ph_v_value_n_5         : REAL;
    udiSwl_duration_n_5         : UDINT;
    uiSwl_timestamp_day_n_5     : UINT;
    uiSwl_timestamp_month_n_5   : UINT;
    uiSwl_timestamp_year_n_5    : UINT;
    uiSwl_timestamp_hour_n_5    : UINT;
    uiSwl_timestamp_minute_n_5  : UINT;
    uiSwl_timestamp_second_n_5  : UINT;
END_STRUCT;

EEM_UDT_9146 : STRUCT
(* Voltage swells *)
    uiSwl_ph_cond_v_source_n_6  : UINT;
```

```
    rSwl_ph_cond_v_value_n_6    : REAL;
    uiSwl_ph_v_source_n_6       : UINT;
    rSwl_ph_v_value_n_6         : REAL;
    udiSwl_duration_n_6         : UDINT;
    uiSwl_timestamp_day_n_6     : UINT;
    uiSwl_timestamp_month_n_6   : UINT;
    uiSwl_timestamp_year_n_6    : UINT;
    uiSwl_timestamp_hour_n_6    : UINT;
    uiSwl_timestamp_minute_n_6  : UINT;
    uiSwl_timestamp_second_n_6  : UINT;
    uiSwl_ph_cond_v_source_n_7  : UINT;
    rSwl_ph_cond_v_value_n_7    : REAL;
    uiSwl_ph_v_source_n_7       : UINT;
    rSwl_ph_v_value_n_7         : REAL;
    udiSwl_duration_n_7         : UDINT;
    uiSwl_timestamp_day_n_7     : UINT;
    uiSwl_timestamp_month_n_7   : UINT;
    uiSwl_timestamp_year_n_7    : UINT;
    uiSwl_timestamp_hour_n_7    : UINT;
    uiSwl_timestamp_minute_n_7  : UINT;
    uiSwl_timestamp_second_n_7  : UINT;
    uiSwl_ph_cond_v_source_n_8  : UINT;
    rSwl_ph_cond_v_value_n_8    : REAL;
    uiSwl_ph_v_source_n_8       : UINT;
    rSwl_ph_v_value_n_8         : REAL;
    udiSwl_duration_n_8         : UDINT;
    uiSwl_timestamp_day_n_8     : UINT;
    uiSwl_timestamp_month_n_8   : UINT;
    uiSwl_timestamp_year_n_8    : UINT;
    uiSwl_timestamp_hour_n_8    : UINT;
    uiSwl_timestamp_minute_n_8  : UINT;
    uiSwl_timestamp_second_n_8  : UINT;
    uiSwl_ph_cond_v_source_n_9  : UINT;
    rSwl_ph_cond_v_value_n_9    : REAL;
    uiSwl_ph_v_source_n_9       : UINT;
    rSwl_ph_v_value_n_9         : REAL;
    udiSwl_duration_n_9         : UDINT;
    uiSwl_timestamp_day_n_9     : UINT;
    uiSwl_timestamp_month_n_9   : UINT;
    uiSwl_timestamp_year_n_9    : UINT;
    uiSwl_timestamp_hour_n_9    : UINT;
    uiSwl_timestamp_minute_n_9  : UINT;
    uiSwl_timestamp_second_n_9  : UINT;
    uiSwl_ph_cond_v_source_n_10 : UINT;
    rSwl_ph_cond_v_value_n_10   : REAL;
    uiSwl_ph_v_source_n_10      : UINT;
    rSwl_ph_v_value_n_10        : REAL;
    udiSwl_duration_n_10        : UDINT;
    uiSwl_timestamp_day_n_10    : UINT;
    uiSwl_timestamp_month_n_10  : UINT;
    uiSwl_timestamp_year_n_10   : UINT;
    uiSwl_timestamp_hour_n_10   : UINT;
    uiSwl_timestamp_minute_n_10 : UINT;
    uiSwl_timestamp_second_n_10 : UINT;
END_STRUCT;

EEM_UDT_9200 : STRUCT
    (* Voltage interrupts *)
    uiInt_timestamp_day_n_1     : UINT;
    uiInt_timestamp_month_n_1   : UINT;
    uiInt_timestamp_year_n_1    : UINT;
    uiInt_timestamp_hour_n_1    : UINT;
    uiInt_timestamp_minute_n_1  : UINT;
    uiInt_timestamp_second_n_1  : UINT;
```

```
    uiInt_timestamp_day_n_2     : UINT;
    uiInt_timestamp_month_n_2   : UINT;
    uiInt_timestamp_year_n_2    : UINT;
    uiInt_timestamp_hour_n_2    : UINT;
    uiInt_timestamp_minute_n_2  : UINT;
    uiInt_timestamp_second_n_2  : UINT;
    uiInt_timestamp_day_n_3     : UINT;
    uiInt_timestamp_month_n_3   : UINT;
    uiInt_timestamp_year_n_3    : UINT;
    uiInt_timestamp_hour_n_3    : UINT;
    uiInt_timestamp_minute_n_3  : UINT;
    uiInt_timestamp_second_n_3  : UINT;
    uiInt_timestamp_day_n_4     : UINT;
    uiInt_timestamp_month_n_4   : UINT;
    uiInt_timestamp_year_n_4    : UINT;
    uiInt_timestamp_hour_n_4    : UINT;
    uiInt_timestamp_minute_n_4  : UINT;
    uiInt_timestamp_second_n_4  : UINT;
    uiInt_timestamp_day_n_5     : UINT;
    uiInt_timestamp_month_n_5   : UINT;
    uiInt_timestamp_year_n_5    : UINT;
    uiInt_timestamp_hour_n_5    : UINT;
    uiInt_timestamp_minute_n_5  : UINT;
    uiInt_timestamp_second_n_5  : UINT;
    uiInt_timestamp_day_n_6     : UINT;
    uiInt_timestamp_month_n_6   : UINT;
    uiInt_timestamp_year_n_6    : UINT;
    uiInt_timestamp_hour_n_6    : UINT;
    uiInt_timestamp_minute_n_6  : UINT;
    uiInt_timestamp_second_n_6  : UINT;
    uiInt_timestamp_day_n_7     : UINT;
    uiInt_timestamp_month_n_7   : UINT;
    uiInt_timestamp_year_n_7    : UINT;
    uiInt_timestamp_hour_n_7    : UINT;
    uiInt_timestamp_minute_n_7  : UINT;
    uiInt_timestamp_second_n_7  : UINT;
    uiInt_timestamp_day_n_8     : UINT;
    uiInt_timestamp_month_n_8   : UINT;
    uiInt_timestamp_year_n_8    : UINT;
    uiInt_timestamp_hour_n_8    : UINT;
    uiInt_timestamp_minute_n_8  : UINT;
    uiInt_timestamp_second_n_8  : UINT;
    uiInt_timestamp_day_n_9     : UINT;
    uiInt_timestamp_month_n_9   : UINT;
    uiInt_timestamp_year_n_9    : UINT;
    uiInt_timestamp_hour_n_9    : UINT;
    uiInt_timestamp_minute_n_9  : UINT;
    uiInt_timestamp_second_n_9  : UINT;
    uiInt_timestamp_day_n_10    : UINT;
    uiInt_timestamp_month_n_10  : UINT;
    uiInt_timestamp_year_n_10   : UINT;
    uiInt_timestamp_hour_n_10   : UINT;
    uiInt_timestamp_minute_n_10 : UINT;
    uiInt_timestamp_second_n_10 : UINT;
END_STRUCT;


EEM_UDT_9300 : STRUCT
(* Meter readings // Energy meter not resetable *)
    rActive_energy_cons_ph1         : REAL;
    rActive_energy_cons_ph2         : REAL;
    rActive_energy_cons_ph3         : REAL;
    rTotal_act_energy_cons_all_ph   : REAL;
    rActive_energy_deliv_ph1        : REAL;
```

```
    rActive_energy_deliv_ph2        : REAL;
    rActive_energy_deliv_ph3        : REAL;
    rTotal_act_energy_deliv_all_ph  : REAL;
    rApp_energy_cons_ind_ph1        : REAL;
    rApp_energy_cons_ind_ph2        : REAL;
    rApp_energy_cons_ind_ph3        : REAL;
    rTotal_app_en_cons_ind_all_ph   : REAL;
    rApp_energy_deliv_ind_ph1       : REAL;
    rApp_energy_deliv_ind_ph2       : REAL;
    rApp_energy_deliv_ind_ph3       : REAL;
    rTotal_app_en_deliv_ind_all_ph  : REAL;
    rApp_energy_cons_cap_ph1        : REAL;
    rApp_energy_cons_cap_ph2        : REAL;
    rApp_energy_cons_cap_ph3        : REAL;
    rTotal_app_en_cons_cap_all_ph   : REAL;
    rApp_energy_deliv_cap_ph1       : REAL;
    rApp_energy_deliv_cap_ph2       : REAL;
    rApp_energy_deliv_cap_ph3       : REAL;
    rTotal_app_en_deliv_cap_all_ph  : REAL;
    rReactive_energy_cons_ind_ph1   : REAL;
    rReactive_energy_cons_ind_ph2   : REAL;
    rReactive_energy_cons_ind_ph3   : REAL;
    rTot_react_en_cons_ind_all_ph   : REAL;
    rReactive_energy_deliv_ind_ph1  : REAL;
    rReactive_energy_deliv_ind_ph2  : REAL;
    rReactive_energy_deliv_ind_ph3  : REAL;
    rTot_react_en_deliv_ind_all_ph  : REAL;
    rReactive_energy_cons_cap_ph1   : REAL;
    rReactive_energy_cons_cap_ph2   : REAL;
    rReactive_energy_cons_cap_ph3   : REAL;
    rTot_react_en_cons_cap_all_ph   : REAL;
    rReactive_energy_deliv_cap_ph1  : REAL;
    rReactive_energy_deliv_cap_ph2  : REAL;
    rReactive_energy_deliv_cap_ph3  : REAL;
    rTot_react_en_deliv_cap_all_ph  : REAL;
    rTotal_react_en_cons_all_ph     : REAL;
    rTotal_react_en_deliv_all_ph    : REAL;
    rTotal_react_en_absolut_all_ph  : REAL;
    rTotal_app_en_all_ph            : REAL;
END_STRUCT;

EEM_UDT_9380 : STRUCT
(* Meter readings // Energy meter resetable *)
    rRes_act_en_con_ph1             : REAL;
    rRes_act_en_con_ph2             : REAL;
    rRes_act_en_con_ph3             : REAL;
    rRes_tot_act_en_con_all_ph      : REAL;
    rRes_act_en_del_ph1             : REAL;
    rRes_act_en_del_ph2             : REAL;
    rRes_act_en_del_ph3             : REAL;
    rRes_tot_act_en_del_all_ph      : REAL;
    rRes_app_en_con_ind_ph1         : REAL;
    rRes_app_en_con_ind_ph2         : REAL;
    rRes_app_en_con_ind_ph3         : REAL;
    rRes_tot_app_en_con_ind_all_ph  : REAL;
    rRes_app_en_del_ind_ph1         : REAL;
    rRes_app_en_del_ind_ph2         : REAL;
    rRes_app_en_del_ind_ph3         : REAL;
    rRes_tot_app_en_del_ind_all_ph  : REAL;
    rRes_app_en_con_cap_ph1         : REAL;
    rRes_app_en_con_cap_ph2         : REAL;
    rRes_app_en_con_cap_ph3         : REAL;
    rRes_tot_app_en_con_cap_all_ph  : REAL;
    rRes_app_en_del_cap_ph1         : REAL;
```

```
    rRes_app_en_del_cap_ph2          : REAL;
    rRes_app_en_del_cap_ph3          : REAL;
    rRes_tot_app_en_del_cap_all_ph   : REAL;
    rRes_react_en_con_ind_ph1        : REAL;
    rRes_react_en_con_ind_ph2        : REAL;
    rRes_react_en_con_ind_ph3        : REAL;
    rRes_tot_rea_en_con_ind_all_ph   : REAL;
    rRes_react_en_del_ind_ph1        : REAL;
    rRes_react_en_del_ind_ph2        : REAL;
    rRes_react_en_del_ind_ph3        : REAL;
    rRes_tot_rea_en_del_ind_all_ph   : REAL;
    rRes_react_en_con_cap_ph1        : REAL;
    rRes_react_en_con_cap_ph2        : REAL;
    rRes_react_en_con_cap_ph3        : REAL;
    rRes_tot_rea_en_con_cap_all_ph   : REAL;
    rRes_react_en_del_cap_ph1        : REAL;
    rRes_react_en_del_cap_ph2        : REAL;
    rRes_react_en_del_cap_ph3        : REAL;
    rRes_tot_rea_en_con_all_ph       : REAL;
    rRes_tot_rea_en_del_cap_all_ph   : REAL;
    rRes_tot_react_en_del_all_ph     : REAL;
    rRes_tot_react_en_abs_all_ph     : REAL;
    rRes_tot_app_en_all_ph           : REAL;
END_STRUCT;

EEM_UDT_9400 : STRUCT
(* Energy meter tariff 1 *)
    rT1_act_en_con_ph1               : REAL;
    rT1_act_en_con_ph2               : REAL;
    rT1_act_en_con_ph3               : REAL;
    rT1_tot_act_en_cons_all_ph       : REAL;
    rT1_act_en_del_ph1               : REAL;
    rT1_act_en_del_ph2               : REAL;
    rT1_act_en_del_ph3               : REAL;
    rT1_tot_act_en_del_all_ph        : REAL;
    rT1_app_en_con_ind_ph1           : REAL;
    rT1_app_en_con_ind_ph2           : REAL;
    rT1_app_en_con_ind_ph3           : REAL;
    rT1_tot_app_en_con_ind_all_ph    : REAL;
    rT1_app_en_del_ind_ph1           : REAL;
    rT1_app_en_del_ind_ph2           : REAL;
    rT1_app_en_del_ind_ph3           : REAL;
    rT1_tot_app_en_del_ind_all_ph    : REAL;
    rT1_app_en_con_cap_ph1           : REAL;
    rT1_app_en_con_cap_ph2           : REAL;
    rT1_app_en_con_cap_ph3           : REAL;
    rT1_tot_app_en_con_cap_all_ph    : REAL;
    rT1_app_en_del_cap_ph1           : REAL;
    rT1_app_en_del_cap_ph2           : REAL;
    rT1_app_en_del_cap_ph3           : REAL;
    rT1_tot_app_en_del_cap_all_ph    : REAL;
    rT1_react_en_con_ind_ph1         : REAL;
    rT1_react_en_con_ind_ph2         : REAL;
    rT1_react_en_con_ind_ph3         : REAL;
    rT1_tot_rea_en_con_ind_all_ph    : REAL;
    rT1_react_en_del_ind_ph1         : REAL;
    rT1_react_en_del_ind_ph2         : REAL;
    rT1_react_en_del_ind_ph3         : REAL;
    rT1_tot_rea_en_del_ind_all_ph    : REAL;
    rT1_react_en_con_cap_ph1         : REAL;
    rT1_react_en_con_cap_ph2         : REAL;
    rT1_react_en_con_cap_ph3         : REAL;
    rT1_tot_rea_en_con_cap_all_ph    : REAL;
    rT1_react_en_del_cap_ph1         : REAL;
```

```
    rT1_react_en_del_cap_ph2        : REAL;
    rT1_react_en_del_cap_ph3        : REAL;
    rT1_tot_rea_en_del_cap_all_ph   : REAL;
    rT1_tot_react_en_con_all_ph     : REAL;
    rT1_tot_react_en_del_all_ph     : REAL;
    rT1_tot_react_en_abs_all_ph     : REAL;
    rT1_tot_app_en_all_ph           : REAL;
END_STRUCT;

EEM_UDT_9480 : STRUCT
(* Energy meter tariff 2 *)
    rT2_act_en_con_ph1              : REAL;
    rT2_act_en_con_ph2              : REAL;
    rT2_act_en_con_ph3              : REAL;
    rT2_tot_act_en_cons_all_ph      : REAL;
    rT2_act_en_del_ph1              : REAL;
    rT2_act_en_del_ph2              : REAL;
    rT2_act_en_del_ph3              : REAL;
    rT2_tot_act_en_del_all_ph       : REAL;
    rT2_app_en_con_ind_ph1          : REAL;
    rT2_app_en_con_ind_ph2          : REAL;
    rT2_app_en_con_ind_ph3          : REAL;
    rT2_tot_app_en_con_ind_all_ph   : REAL;
    rT2_app_en_del_ind_ph1          : REAL;
    rT2_app_en_del_ind_ph2          : REAL;
    rT2_app_en_del_ind_ph3          : REAL;
    rT2_tot_app_en_del_ind_all_ph   : REAL;
    rT2_app_en_con_cap_ph1          : REAL;
    rT2_app_en_con_cap_ph2          : REAL;
    rT2_app_en_con_cap_ph3          : REAL;
    rT2_tot_app_en_con_cap_all_ph   : REAL;
    rT2_app_en_del_cap_ph1          : REAL;
    rT2_app_en_del_cap_ph2          : REAL;
    rT2_app_en_del_cap_ph3          : REAL;
    rT2_tot_app_en_del_cap_all_ph   : REAL;
    rT2_react_en_con_ind_ph1        : REAL;
    rT2_react_en_con_ind_ph2        : REAL;
    rT2_react_en_con_ind_ph3        : REAL;
    rT2_tot_rea_en_con_ind_all_ph   : REAL;
    rT2_react_en_del_ind_ph1        : REAL;
    rT2_react_en_del_ind_ph2        : REAL;
    rT2_react_en_del_ind_ph3        : REAL;
    rT2_tot_rea_en_del_ind_all_ph   : REAL;
    rT2_react_en_con_cap_ph1        : REAL;
    rT2_react_en_con_cap_ph2        : REAL;
    rT2_react_en_con_cap_ph3        : REAL;
    rT2_tot_rea_en_con_cap_all_ph   : REAL;
    rT2_react_en_del_cap_ph1        : REAL;
    rT2_react_en_del_cap_ph2        : REAL;
    rT2_react_en_del_cap_ph3        : REAL;
    rT2_tot_rea_en_del_cap_all_ph   : REAL;
    rT2_tot_react_en_con_all_ph     : REAL;
    rT2_tot_react_en_del_all_ph     : REAL;
    rT2_tot_react_en_abs_all_ph     : REAL;
    rT2_tot_app_en_all_ph           : REAL;
END_STRUCT;

EEM_UDT_9500 : STRUCT
(* Energy meter tariff 3 *)
    rT3_act_en_con_ph1              : REAL;
    rT3_act_en_con_ph2              : REAL;
    rT3_act_en_con_ph3              : REAL;
    rT3_tot_act_en_cons_all_ph      : REAL;
    rT3_act_en_del_ph1              : REAL;
```

```
    rT3_act_en_del_ph2              : REAL;
    rT3_act_en_del_ph3              : REAL;
    rT3_tot_act_en_del_all_ph       : REAL;
    rT3_app_en_con_ind_ph1          : REAL;
    rT3_app_en_con_ind_ph2          : REAL;
    rT3_app_en_con_ind_ph3          : REAL;
    rT3_tot_app_en_con_ind_all_ph   : REAL;
    rT3_app_en_del_ind_ph1          : REAL;
    rT3_app_en_del_ind_ph2          : REAL;
    rT3_app_en_del_ind_ph3          : REAL;
    rT3_tot_app_en_del_ind_all_ph   : REAL;
    rT3_app_en_con_cap_ph1          : REAL;
    rT3_app_en_con_cap_ph2          : REAL;
    rT3_app_en_con_cap_ph3          : REAL;
    rT3_tot_app_en_con_cap_all_ph   : REAL;
    rT3_app_en_del_cap_ph1          : REAL;
    rT3_app_en_del_cap_ph2          : REAL;
    rT3_app_en_del_cap_ph3          : REAL;
    rT3_tot_app_en_del_cap_all_ph   : REAL;
    rT3_react_en_con_ind_ph1        : REAL;
    rT3_react_en_con_ind_ph2        : REAL;
    rT3_react_en_con_ind_ph3        : REAL;
    rT3_tot_rea_en_con_ind_all_ph   : REAL;
    rT3_react_en_del_ind_ph1        : REAL;
    rT3_react_en_del_ind_ph2        : REAL;
    rT3_react_en_del_ind_ph3        : REAL;
    rT3_tot_rea_en_del_ind_all_ph   : REAL;
    rT3_react_en_con_cap_ph1        : REAL;
    rT3_react_en_con_cap_ph2        : REAL;
    rT3_react_en_con_cap_ph3        : REAL;
    rT3_tot_rea_en_con_cap_all_ph   : REAL;
    rT3_react_en_del_cap_ph1        : REAL;
    rT3_react_en_del_cap_ph2        : REAL;
    rT3_react_en_del_cap_ph3        : REAL;
    rT3_tot_rea_en_del_cap_all_ph   : REAL;
    rT3_tot_react_en_con_all_ph     : REAL;
    rT3_tot_react_en_del_all_ph     : REAL;
    rT3_tot_react_en_abs_all_ph     : REAL;
    rT3_tot_app_en_all_ph           : REAL;
END_STRUCT;

EEM_UDT_9580 : STRUCT
(* Energy meter tariff 4 *)
    rT4_act_en_con_ph1              : REAL;
    rT4_act_en_con_ph2              : REAL;
    rT4_act_en_con_ph3              : REAL;
    rT4_tot_act_en_cons_all_ph      : REAL;
    rT4_act_en_del_ph1              : REAL;
    rT4_act_en_del_ph2              : REAL;
    rT4_act_en_del_ph3              : REAL;
    rT4_tot_act_en_del_all_ph       : REAL;
    rT4_app_en_con_ind_ph1          : REAL;
    rT4_app_en_con_ind_ph2          : REAL;
    rT4_app_en_con_ind_ph3          : REAL;
    rT4_tot_app_en_con_ind_all_ph   : REAL;
    rT4_app_en_del_ind_ph1          : REAL;
    rT4_app_en_del_ind_ph2          : REAL;
    rT4_app_en_del_ind_ph3          : REAL;
    rT4_tot_app_en_del_ind_all_ph   : REAL;
    rT4_app_en_con_cap_ph1          : REAL;
    rT4_app_en_con_cap_ph2          : REAL;
    rT4_app_en_con_cap_ph3          : REAL;
    rT4_tot_app_en_con_cap_all_ph   : REAL;
    rT4_app_en_del_cap_ph1          : REAL;
```

```
    rT4_app_en_del_cap_ph2          : REAL;
    rT4_app_en_del_cap_ph3          : REAL;
    rT4_tot_app_en_del_cap_all_ph   : REAL;
    rT4_react_en_con_ind_ph1        : REAL;
    rT4_react_en_con_ind_ph2        : REAL;
    rT4_react_en_con_ind_ph3        : REAL;
    rT4_tot_rea_en_con_ind_all_ph   : REAL;
    rT4_react_en_del_ind_ph1        : REAL;
    rT4_react_en_del_ind_ph2        : REAL;
    rT4_react_en_del_ind_ph3        : REAL;
    rT4_tot_rea_en_del_ind_all_ph   : REAL;
    rT4_react_en_con_cap_ph1        : REAL;
    rT4_react_en_con_cap_ph2        : REAL;
    rT4_react_en_con_cap_ph3        : REAL;
    rT4_tot_rea_en_con_cap_all_ph   : REAL;
    rT4_react_en_del_cap_ph1        : REAL;
    rT4_react_en_del_cap_ph2        : REAL;
    rT4_react_en_del_cap_ph3        : REAL;
    rT4_tot_rea_en_del_cap_all_ph   : REAL;
    rT4_tot_react_en_con_all_ph     : REAL;
    rT4_tot_react_en_del_all_ph     : REAL;
    rT4_tot_react_en_abs_all_ph     : REAL;
    rT4_tot_app_en_all_ph           : REAL;
END_STRUCT;

EEM_UDT_C550 : STRUCT
(* Tab 1: Display current main measured values based
on current transformer (taking into consideration
transformation ratios) *)
    rOpHours        : REAL;
    rU12            : REAL;
    rU23            : REAL;
    rU31            : REAL;
    rV1             : REAL;
    rV2             : REAL;
    rV3             : REAL;
    rFreq           : REAL;
    rI1             : REAL;
    rI2             : REAL;
    rI3             : REAL;
    rIn             : REAL;
    rP_Sum          : REAL;
    rQ_Sum          : REAL;
    rS_Sum          : REAL;
    rPF_Sum         : REAL;
    rP1             : REAL;
    rP2             : REAL;
    rP3             : REAL;
    rQ1             : REAL;
    rQ2             : REAL;
    rQ3             : REAL;
    rS1             : REAL;
    rS2             : REAL;
    rS3             : REAL;
    rPF1            : REAL;
    rPF2            : REAL;
    rPF3            : REAL;
END_STRUCT;

EEM_UDT_C650 : STRUCT
(* Tab 2 : Display energy values *)
    rOpHours                        : REAL;
    rTotal_act_en_proc_sys          : REAL;
    rTotal_ind_react_en_sys         : REAL;
```

```
    rTotal_app_en_sys               : REAL;
    rTotal_act_en_supply_sys        : REAL;
    rTotal_cap_react_en_sys         : REAL;
    rEaPos                          : REAL;
    rErPos                          : REAL;
    rEs                             : REAL;
    rEaNeg                          : REAL;
    rErNeg                          : REAL;
END_STRUCT;

EEM_UDT_C750 : STRUCT
(* Tab 3 : Display average values based on current
transformer over the set measurement duration
(taking into consideration transformation ratios) *)
    rAvg_U12            : REAL;
    rAvg_U23            : REAL;
    rAvg_U31            : REAL;
    rAvg_V1             : REAL;
    rAvg_V2             : REAL;
    rAvg_V3             : REAL;
    rAvg_Freq           : REAL;
    rAvg_I1             : REAL;
    rAvg_I2             : REAL;
    rAvg_I3             : REAL;
    rAvg_In             : REAL;
    rAvg_P_Pos_Sum      : REAL;
    rAvg_P_Neg_Sum      : REAL;
    rAvg_Q_Pos_Sum      : REAL;
    rAvg_Q_Neg_Sum      : REAL;
    rAvg_S_Sum          : REAL;
    rMax_Avg_U12        : REAL;
    rMax_Avg_U23        : REAL;
    rMax_Avg_U31        : REAL;
    rMax_Avg_V1         : REAL;
    rMax_Avg_V2         : REAL;
    rMax_Avg_V3         : REAL;
    rMax_Avg_Freq       : REAL;
    rMax_Avg_I1         : REAL;
    rMax_Avg_I2         : REAL;
    rMax_Avg_I3         : REAL;
    rMax_Avg_In         : REAL;
    rMax_Avg_P_Pos_Sum  : REAL;
    rMax_Avg_P_Neg_Sum  : REAL;
    rMax_Avg_Q_Pos_Sum  : REAL;
    rMax_Avg_Q_Neg_Sum  : REAL;
    rMax_Avg_S_Sum      : REAL;
END_STRUCT;

EEM_UDT_C850 : STRUCT
(* Tab 4 : Display current main measured values
not based on current transformer (without taking
into consideration transformation ratios) *)
    rOpHours    : REAL;
    rU12        : REAL;
    rU23        : REAL;
    rU31        : REAL;
    rV1         : REAL;
    rV2         : REAL;
    rV3         : REAL;
    rFreq       : REAL;
    rI1         : REAL;
    rI2         : REAL;
    rI3         : REAL;
    rIn         : REAL;
```

```
    rP_Sum      : REAL;
    rQ_Sum      : REAL;
    rS_Sum      : REAL;
    rPF_Sum     : REAL;
    rP1         : REAL;
    rP2         : REAL;
    rP3         : REAL;
    rQ1         : REAL;
    rQ2         : REAL;
    rQ3         : REAL;
    rS1         : REAL;
    rS2         : REAL;
    rS3         : REAL;
    rPF1        : REAL;
    rPF2        : REAL;
    rPF3        : REAL;
    rEA_Pos     : REAL;
    rEA_Neg     : REAL;
END_STRUCT;

EEM_UDT_C900 : STRUCT
(* Tab 7 : Display temperature input *)
    xInt_temp_sensor_available  : BOOL;
    rInt_module_temp            : REAL;
    usiNum_temp_sensor_connect  : USINT;
    rExt_temp_1                 : REAL;
    rExt_temp_2                 : REAL;
    rExt_temp_3                 : REAL;
END_STRUCT;

EEM_UDT_C950 : STRUCT
(* Tab 8a : Display harmonic distortion and
harmonics *)
    rTHD_U12    : REAL;
    rTHD_U23    : REAL;
    rTHD_U31    : REAL;
    rTHD_V1     : REAL;
    rTHD_V2     : REAL;
    rTHD_V3     : REAL;
    rTHD_I1     : REAL;
    rTHD_I2     : REAL;
    rTHD_I3     : REAL;
    rTHD_In     : REAL;
END_STRUCT;

EEM_UDT_HARMONICS   : STRUCT
    U12         : REAL;
    U23         : REAL;
    U31         : REAL;
    V1          : REAL;
    V2          : REAL;
    V3          : REAL;
    I1          : REAL;
    I2          : REAL;
    I3          : REAL;
    In          : REAL;
END_STRUCT;

EEM_UDT_C95A    : STRUCT
(* Tab 8b: Display harmonic distortion and
harmonics *)
    rMax_Harm   : USINT;
    rHarm_3rd   : EEM_UDT_HARMONICS;
    rHarm_5th   : EEM_UDT_HARMONICS;
```

```
        rHarm_7th   : EEM_UDT_HARMONICS;
        rHarm_9th   : EEM_UDT_HARMONICS;
        rHarm_11th  : EEM_UDT_HARMONICS;
        rHarm_13th  : EEM_UDT_HARMONICS;
        rHarm_15th  : EEM_UDT_HARMONICS;
        rHarm_17th  : EEM_UDT_HARMONICS;
        rHarm_19th  : EEM_UDT_HARMONICS;
        rHarm_21st  : EEM_UDT_HARMONICS;
        rHarm_23rd  : EEM_UDT_HARMONICS;
        rHarm_25th  : EEM_UDT_HARMONICS;
        rHarm_27th  : EEM_UDT_HARMONICS;
        rHarm_29th  : EEM_UDT_HARMONICS;
        rHarm_31st  : EEM_UDT_HARMONICS;
        rHarm_33rd  : EEM_UDT_HARMONICS;
        rHarm_35th  : EEM_UDT_HARMONICS;
        rHarm_37th  : EEM_UDT_HARMONICS;
        rHarm_39th  : EEM_UDT_HARMONICS;
        rHarm_41st  : EEM_UDT_HARMONICS;
        rHarm_43rd  : EEM_UDT_HARMONICS;
        rHarm_45th  : EEM_UDT_HARMONICS;
        rHarm_47th  : EEM_UDT_HARMONICS;
        rHarm_49th  : EEM_UDT_HARMONICS;
        rHarm_51st  : EEM_UDT_HARMONICS;
        rHarm_53rd  : EEM_UDT_HARMONICS;
        rHarm_55th  : EEM_UDT_HARMONICS;
        rHarm_57th  : EEM_UDT_HARMONICS;
        rHarm_59th  : EEM_UDT_HARMONICS;
        rHarm_61st  : EEM_UDT_HARMONICS;
        rHarm_63rd  : EEM_UDT_HARMONICS;
    END_STRUCT;

    EEM_UDT_CB00 : STRUCT
    (* Tab9 : Digital inputs and outputs *)
        xInput1     : BOOL;
        xInput2     : BOOL;
        xInput3     : BOOL;
        xInput4     : BOOL;
        xInput5     : BOOL;
        xInput6     : BOOL;
        xOutput1    : BOOL;
        xOutput2    : BOOL;
        xOutput3    : BOOL;
        xOutput4    : BOOL;
        xOutput5    : BOOL;
        xOutput6    : BOOL;
    END_STRUCT;

END_TYPE
```

11 In case of problems

In case of problems you should check in general:

1. Is the latest version of the library used? You can download the latest version from the Phoenix Contact website.
2. Is the library released for the PC Worx 6 version you are using? Please refer to the chapter “Change notes” of this documentation.
3. Is the library released for the PLC / PLC firmware version you are using? Please refer to the chapter “Change notes” of this documentation.
4. Does every module in your system has the required firmware version? In some cases functionalities are supported for higher firmware versions only.
5. Is your system running? Check all LEDs of all modules. Check the status in the web based management of your PLC.

12 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

In case of a support request we need:

- Development system with
  - Name (e.g. PC Worx, PLCnext Engineer)
  - Version (e.g. PLCnext Engineer 2022.0.1 LTS)
- Bus structure / plant including all articles with
  - Name
  - Order number
  - Firmware version
- External components
- Used libraries with
  - Name
  - Version (e.g. IOL_Basic_7)
- Detailed problem description with
  - Diag codes of all function blocks in error state