Application note

Translate MODBUS TCP to MODBUS RTU e-ILPH

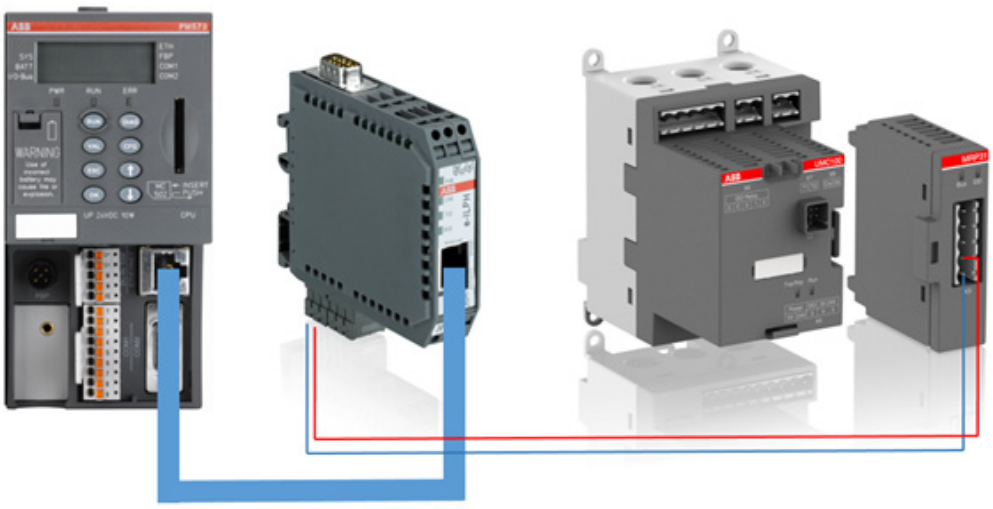
The following example describes how the e-ILPH converts the Modbus TCP telegram provided by an ABB AC500 (PM573-ETH) to a Modbus RTU telegram to communicate with an UMC100.3 using a MRP31.0 communication interface. The target is to read and write commands and monitor data, although two different communication protocols are used.

2CDC 281 003 S0013

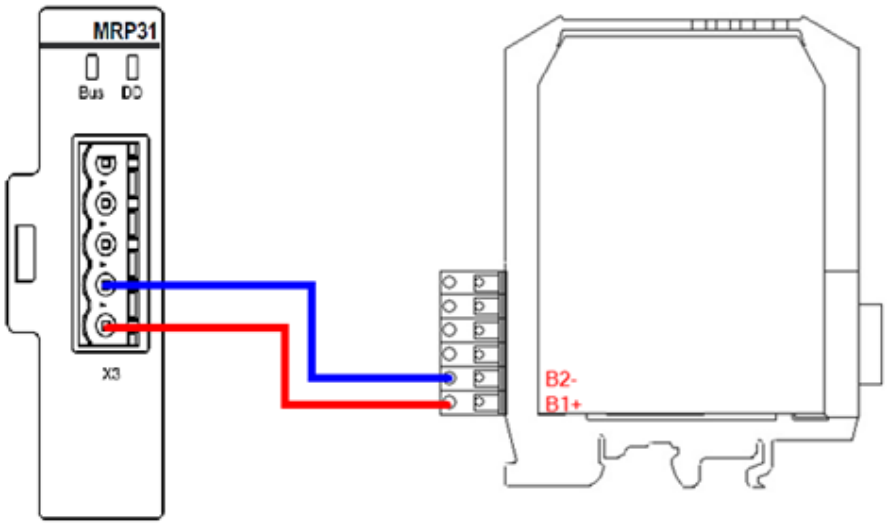
Power and productivity
for a better world™

The setup displayed below shows following connections:

1. AC500 (Ethernet port) to e-ILPH via Ethernet wire (CAT 5E)
2. e-ILPH (RS485) to MRP31.0 via two wires (B1+ to pin 5 of MRP31.0 and B2- to pin 4 of MRP31.0)
3. MRP31.0 directly plugged on UMC100.3



General setup

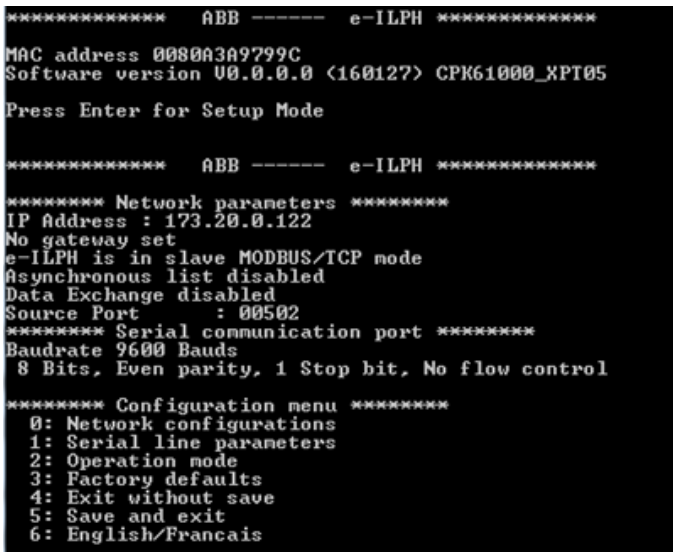


Wiring between MRP31.0 and e-ILPH

Configuration of e-ILPH

For the configuration of e-ILPH the device has to be connected directly to the PC and the function “telnet” has to be enabled in Windows.

1. Open Command Prompt
2. Type in “telnet xxx.xxx.xxx.xxx” (IP-address of e-ILPH)

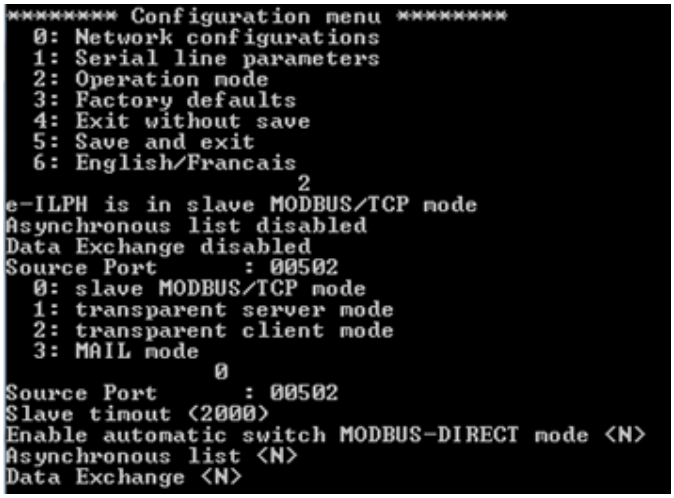


Main window after entering the e-ILPH

3. Open “1: Serial line parameters”
   - Set Baud rate
   - Set 8 Bits
   - Set Odd or Even parity
   - Set 1 Stop bit
   - Set no flow control

Serial line parameters

4. Open “2: Operation Mode”
5. Choose “0: slave MODBUS/TCP mode”
   - “Source port”: 502 (MODBUS/TCP port)
   - “Slave time out”: 2000
   - “Enable automatic switch MODBUS-DIRECT mode”: disabled
   - “Asynchronous list”: disabled
   - “Data Exchange”: disabled



Operation mode

6. Open “5: Save and Exit” to confirm and save the changes.

Configuration of UMC100.3

For the communication via MODBUS RTU some settings of the UMC100.3 are necessary:

1. Set the bus address via panel (Menu > Communication > Busaddress)
2. Set the Baudrate equal to e-ILPH (Menu > Communication > MODBUS Baudrate)
3. Set the Timeout time big enough (Menu > Communication > MODBUS bus timeout)

Supported MODBUS Function Codes

Following Function Codes (FC) are supported from MRP31.0/UMC100.3:

Commands	MODBUS Function Codes	Starting address
Read binary input values	FC 1 Read Coils	0000 Hex
	FC 2 Read Discrete Inputs	
Write binary output values	FC 15 Write Multiple Coils	0100 Hex
Read analog input values	FC 3 Read Holding Registers	0200 Hex
	FC 4 Read Input Registers	
Write analog output values	FC 16 Write Multiple Registers	0300 Hex
Read diagnostic data	FC 3 Read Holding Registers	2000 Hex
	FC 4 Read Input Registers	

Register addresses of UMC100.3

Register addresses of UMC100.3 acc. to the UMC100.3 manual section A1 Parameters and Data Structures on a Fieldbus.

Access to monitoring data:

Register	Bit 7 Bit 15	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0 Bit 8
0x0000	Summary Warning	Summary Fault	Local Control	Reverse Lockout Time	Overload warning	Run Forward / Opening	Off	Run Reverse / Closing
	UMC100 DI5	UMC100 DI4	UMC100 DI3	UMC100 DI2	UMC100 DI1	UMC100 DI0	Run Fast Forward	.
Register	**Data**							
0x0200	Motor Current in % of I_e (0-800%)							
0x0201	Analog Word (Thermal Load 0-100%)							
0x0202	Analog Word (Time to trip in seconds)							
0x0203	Analog Word (Time to restart in seconds)							
0x0204	Analog Word (Active power in selected scale)							
0x0205	DX1xx DI7	DX1xx DI6	DX1xx DI5	DX1xx DI4	DX1xx DI3	DX1xx DI2	DX1xx DI1	DX1xx DI0
	-	-	Run Time Exceeded	Out of Position	Torque Open	Torque Closed	End Pos Open	End Pos Closed
0x0206	U Imbal. warn	U Imbal. trip	Under voltage warn	Under voltage trip	Underpower warn	Underpower trip	Overpower warn	Overpower trip
	Earth fault warning	Earth fault trip	Cooling time running	-	THD warning	No start possible	1 start left	More than 1 start left
Register	**Access to command data**							
	Bit 7, Bit 15, Bit 23, Bit 31	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0, Bit 8, Bit 16, Bit 24**
0x0100	-	Fault Reset	Auto Mode	Prepare Emergency Start	-	Run Forward / Opening	Off	Run Reverse / Closing
	UMC100 DO2	UMC100 DO1	UMC100 DO0	UMC100 24 V DC OUT	-	-	Run Fast Forward	-
0x0101	VI15x DO0	-	-	-	DX1xx DO3	DX1xx DO2	DX1xx DO1	DX1xx DO0
	-	-	-	-	-	-	-	-
Register	**Data**							
0x0300	Analog Word							
0x0301	Analog Word							
0x0302	Analog Word							
0x0303	Analog Word							

Register	Access to diagnosis data							
	Bit 7, Bit 15	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0, Bit 8
0x2000	Checkback missing	PTC wiring failure	PTC hot	Pre-warning thermal model	Locked rotor during start-up (stall)	Phase imbalance	Phase loss	Thermal overload trip
	Actuator problem	UMC self-test error	Earth fault pre-warning	Earth fault trip (internal or externally triggered)	I above high current warning threshold	I above high current trip threshold	I below low current warning threshold	I below low current trip threshold
0x2001	Trip/Warning from Aux-Fault function block Input 5	Trip/Warning from Aux-Fault function block Input 4	Trip/Warning from Aux-Fault function block Input 3	Trip/Warning from Aux-Fault function block Input 2	Trip/Warning from Aux-Fault function block Input 1	HW fault on IO module	Custom application error	IO module missing
	-	-	-	-	Trip triggered from Multifunction DI2	Trip triggered from Multifunction DI1	Trip triggered from Multifunction DI0	Trip / Warning from Aux-Fault function block Input 6
0x2002	-	-	THD Warning	Voltage out of spec	Overload power	Underload power	-	-
	-	-	Cooling Time Running	Just one start left	Num Starts Overrun	-	-	-
0x2003	Extended diagnosis is available	Parameter out of range	-	-	-	-	-	-
	Fault code. See section "Error Handling, Maintenance and Service -> Fault messages for a description of the code.							

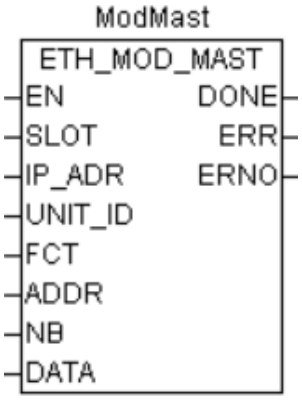
Read status of UMCs Digital Inputs

In this example is described how to read the status of the DIs of UMC. The first step is to choose a MODBUS function code, in this case FC1 Read Coils. According the MRP31.0 manual are the statuses saved in registers starting at address 0x0000.

Register	Bit 7 Bit 15	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0 Bit 8
0x0000	Summary Warning	Summary Fault	Local Control	Reverse Lockout Time	Overload warning	Run Forward / Opening	Off	Run Reverse / Closing
	UMC100 DI5	UMC100 DI4	UMC100 DI3	UMC100 DI2	UMC100 DI1	UMC100 DI0	Run Fast Forward	.

First monitoring word of UMC100.3

After opening ABB Automation Builder and configuring the hardware, the application has to be implemented. For communication via MODBUS/TCP the function block “ETH_MOD_MAST” has to be used.



ETH_MOD_MAST function block

The parameters are defined as following:

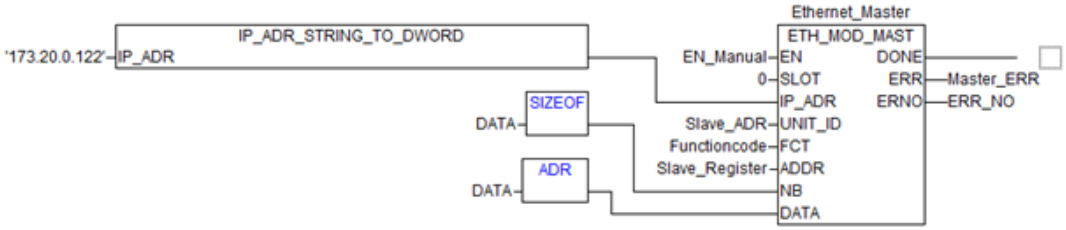
EN	Input	Bool	Enabling of the Function Block processing
SLOT	Input	Byte	Slot (module number) of the Communication Module
IP_ADR	Input	DWord	IP address of the server
UNIT_ID	Input	Byte	Slave sub address
FCT	Input	Byte	MODBUS function code
ADDR	Input	Word	Operand/register address in the server
NB	Input	Word	Number of data to be read/written
DATA	Input	DWord	Address of the first operand in the client from which data shall be written to the server or where the data read from the server shall be stored
DONE	Output	Bool	Ready message of the Function Block
ERR	Output	Bool	Error message of the Function Block
ERNO	Output	Word	Error number

After inserting the function block all variables have to be declared

```
0001 PROGRAM PLC_PRG
0002 VAR
0003     EN_Manual: BOOL;                (*Enables the function block and initialize the sending/receiving of data*)
0004
0005     Ethernet_Master: ETH_MOD_MAST;  (*Instance of the ETH_MOD_MAST function block*)
0006     DATA: ARRAY [0..15] OF BOOL;    (*Array to save the received Data; data type depends on Modbus function code*)
0007     Slave_ADR: BYTE := 3;           (*FBP-address of UMC connected to e-ILPH*)
0008     Functioncode: BYTE := 1;        (*Modbus function code*)
0009     Slave_Register: INT := 0;       (*Starting address of sending/reading data of UMC; refer to MRP31.0 manual*)
0010
0011     Master_ERR: BOOL;               (*Indicates if function block returns error*)
0012     ERR_NO: WORD;                   (*Error ID in case of error while sending/receiving*)
0013 END_VAR
```

Declaration of used variables

All parameters need to be connected to the corresponding in- / output of the function block.



Complete application to send/receive MODBUS telegrams

The input EN has to be enabled manually to send each command.

The input SLOT needs the information which communication interface shall send the telegram. Each interface connected to the CPU has its own slot and starts with “1” on the left side of the CPU. The CPU itself is on slot “0” and has an integrated Ethernet port which is used in this case.

Regarding the table above, the input IP_ADR expects a value from data type DWORD. Therefore it is necessary to convert the IP-address of the e-ILPH to DWORD with a function called “IP_ADR_STRING_TO_DWORD”.

The fieldbus address of the connected UMC100.3 has to be set at UNIT_ID input.

With FCT the MODBUS function code will be set (see chapter “Supported MODBUS Function Codes“).

Depending on the selected function code and the register in UMC100.3, this address has to be set. The registers can be found in chapter “Register addresses of UMC100.3“ or in MRP31.0 manual.

The input NB defines the size of data which shall be sent / received (IMPORTANT: for coils the array has to be Boolean and for registers Bytes!) with the “SIZEOF” function the actual size will be calculated automatically.

With the pointer on the first address of the array (function “ADR”) all data with a length defined by NB will be sent / recorded.

In case of an error ERR will be true and the error ID will be shown in ERNO.

Write command to start the motor

In this example the motor should be started by sending the signal from the PLC via e-ILPH to the UMC.

For this, it's only necessary to change some parameters:

- The data which should be sent have to be saved in the "data"-array
- The function code is FC15 (Write Multiple Coils) instead of FC1
- The register address has to be changed according the tables of chapter "Register addresses of UMC100.3"

The declaration of variables looks like the following:

```
0001 PROGRAM Start_Motor
0002 VAR
0003     EN_Manual: BOOL;                 (*Enables the function block and initialize the sending/receiving of data*)
0004
0005     Ethernet_Master: ETH_MOD_MAST;   (*Instance of the ETH_MOD_MAST function block*)
0006     DATA: ARRAY [0..15] OF BOOL;     (*Array to save the received Data; data type depends on Modbus function code*)
0007     Slave_ADR: BYTE := 3;            (*FBP-address of UMC connected to e-ILPH*)
0008     Functioncode: BYTE := 15;        (*Modbus function code*)
0009     Slave_Register: INT := 256;      (*Starting address of sending/reading data of UMC; refer to MRP31.0 manual*)
0010
0011     Master_ERR: BOOL;                (*Indicates if function block returns error*)
0012     ERR_NO: WORD;                    (*Error ID in case of error while sending/receiving*)
0013 END_VAR
```

The differences in comparison to the first example are the function code (now FC 15) and the slave register (register address 0x0100).

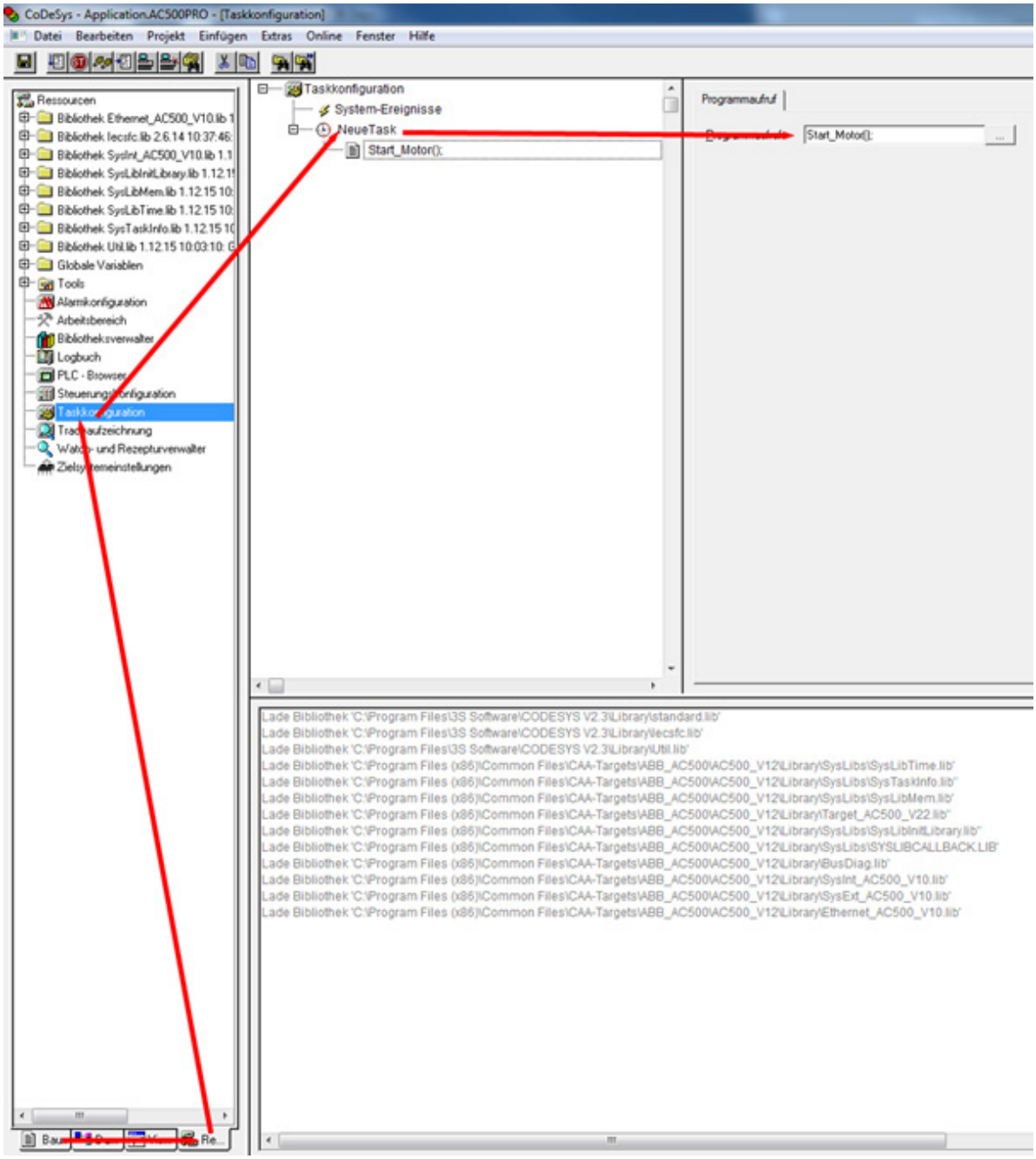
After going online the data array has to be filled:

- To start the motor data bits 0 and 5 have to be true (bit 0 is the start command and bit 5 enables the remote mode to control via PLC)
- To stop the motor data bits 1 and 5 have to be true (bit 1 is the stop command)
- In case of an error this can be reset by setting bit 5 and 6 (bit 6 is fault reset command)

The following picture shows the case of starting the motor:

```
0001     EN_Manual = FALSE <:= TRUE>
0002   ⊞ Ethernet_Master
0003   ⊟ DATA
0004       DATA[0] = TRUE
0005       DATA[1] = FALSE
0006       DATA[2] = FALSE
0007       DATA[3] = FALSE
0008       DATA[4] = FALSE
0009       DATA[5] = TRUE
0010       DATA[6] = FALSE
0011       DATA[7] = FALSE
0012       DATA[8] = FALSE
0013       DATA[9] = FALSE
0014       DATA[10] = FALSE
0015       DATA[11] = FALSE
0016       DATA[12] = FALSE
0017       DATA[13] = FALSE
0018       DATA[14] = FALSE
0019       DATA[15] = FALSE
0020     Slave_ADR = 16#03
0021     Functioncode = 16#0F
0022     Slave_Register = 16#0100
0023     Master_ERR = FALSE
0024     ERR_NO = 16#0000
```

Both examples can be found in the attached Automation Builder project (Automation Builder V1.2) and have to be enabled separately in the task configuration:



Related Documents:

UMC100.3 manual	2CDC135032D0203 (07.2015)
MRP31.0 manual	2CDC194005D0201 (09.2014)
e-ILPH manual	1SNB002323R2100 (12.2006)

Contact us

ABB STOTZ-KONTAKT GmbH
P. O. Box 10 16 80
69006 Heidelberg, Germany
Phone: +49 (0) 6221 7 01-0
Fax: +49 (0) 6221 7 01-13 25
E-mail: info.desto@de.abb.com

You can find the address of your
local sales organisation on the
ABB home page
http://www.abb.com/contacts
-> Low Voltage Products and Systems

Document number 2CDC 116 003 L0201 (02.2016)

Note:
We reserve the right to make technical changes or modify the contents of this document without prior notice. With regard to purchase orders, the agreed particulars shall prevail. ABB AG does not accept any responsibility whatsoever for potential errors or possible lack of information in this document.