

# Modular PLC

## XC-CPU121-2C256K

**Hardware, Engineering and  
Functional Description**

08/05 AWB2724-1578GB

**MOELLER** 

Think future. Switch to green.

All brand and product names are trademarks or registered trademarks of the owner concerned.

1<sup>st</sup> published 2005, edition date 0805

© Moeller GmbH, 53105 Bonn

Authors: Peter Roersch  
Editor: Thomas Kracht  
Translator: David Long

All rights reserved, including those of the translation.

No part of this manual may be reproduced in any form (printed, photocopy, microfilm or any other process) or processed, duplicated or distributed by means of electronic systems without written permission of Moeller GmbH, Bonn.

Subject to alteration without notice.



## Warning! Dangerous electrical voltage!

---

### Before commencing the installation

- Disconnect the power supply of the device.
- Ensure that devices cannot be accidentally restarted.
- Verify isolation from the supply.
- Earth and short circuit.
- Cover or enclose neighbouring units that are live.
- Follow the engineering instructions (AWA) of the device concerned.
- Only suitably qualified personnel in accordance with EN 50110-1/-2 (VDE 0105 Part 100) may work on this device/system.
- Before installation and before touching the device ensure that you are free of electrostatic charge.
- The functional earth (FE) must be connected to the protective earth (PE) or to the potential equalisation. The system installer is responsible for implementing this connection.
- Connecting cables and signal lines should be installed so that inductive or capacitive interference does not impair the automation functions.
- Install automation devices and related operating elements in such a way that they are well protected against unintentional operation.
- Suitable safety hardware and software measures should be implemented for the I/O interface so that a line or wire breakage on the signal side does not result in undefined states in the automation devices.
- Ensure a reliable electrical isolation of the low voltage for the 24 volt supply. Only use power supply units complying with IEC 60364-4-41 (VDE 0100 Part 410) or HD 384.4.41 S2.
- Deviations of the mains voltage from the rated value must not exceed the tolerance limits given in the specifications, otherwise this may cause malfunction and dangerous operation.
- Emergency stop devices complying with IEC/EN 60204-1 must be effective in all operating modes of the automation devices. Unlatching the emergency-stop devices must not cause restart.
- Devices that are designed for mounting in housings or control cabinets must only be operated and controlled after they have been installed with the housing closed. Desktop or portable units must only be operated and controlled in enclosed housings.
- Measures should be taken to ensure the proper restart of programs interrupted after a voltage dip or failure. This should not cause dangerous operating states even for a short time. If necessary, emergency-stop devices should be implemented.
- Wherever faults in the automation system may cause damage to persons or property, external measures must be implemented to ensure a safe operating state in the event of a fault or malfunction (for example, by means of separate limit switches, mechanical interlocks etc.).



## Contents

|  |   |    |
|--|---|----|
| <b>About this manual</b>                   |   | 5  |
|  | Additional documentation                          | 5  |
|  | Reading conventions                               | 5  |
| <b>1 Design of the XC121</b>               |   | 7  |
| <b>2 XC-CPU121 functions</b>               |   | 9  |
|  | Operating mode switch (S1)                        | 9  |
|  | SET button (S5)                                   | 9  |
|  | APPLICATION switch (S2)                           | 10 |
|  | DIP switch mode                                   | 10 |
|  | Memory card MCC                                   | 11 |
|  | – Data access on the MMC                          | 11 |
|  | – Erasing functions                               | 11 |
|  | LED status indication RUN/STOP and SF             | 11 |
|  | Real-time clock                                   | 11 |
|  | Limit values for memory usage                     | 12 |
|  | Serial interface COM1/COM2                        | 12 |
|  | CANopen interface CAN1/CAN2                       | 13 |
|  | – XC121 as a CAN Device (CAN1 and/or CAN2)        | 13 |
|  | – Setting of the XC121 as a CAN Master/CAN Device | 14 |
|  | – CAN Direct                                      | 14 |
|  | – Bus termination resistors                       | 14 |
|  | – Properties of the CANopen cable                 | 14 |
| <b>3 Mounting</b>                          |   | 15 |
|  | Mounting the XC121 on a top-hat rail              | 15 |
|  | Mounting the XIO-EXT121-1                         | 15 |
|  | APPLICATION switch setting                        | 16 |
|  | Input/output wiring                               | 16 |
| <b>4 Engineering</b>                       |   | 17 |
|  | Control panel layout                              | 17 |
|  | – Ventilation                                     | 17 |
|  | – Layout of units                                 | 17 |
|  | Preventing interference                           | 17 |
|  | – Suppressor circuitry for interference sources   | 17 |
|  | – Shielding                                       | 17 |
|  | Lighting protection                               | 18 |
|  | Connections                                       | 18 |
|  | – Connecting the power supply                     | 18 |
|  | – Connecting sensors and actuators                | 19 |
| <b>5 Configuration of the XIO-EXT121-1</b> |   | 21 |

|   |   |    |
|---|---|----|
| <b>6 Operation</b>                                |   | 23 |
|   | Switch-on behaviour   | 23 |
|   | – Switch on behaviour with boot project                         | 23 |
|   | Configuring the start-up behaviour with XSoft                   | 24 |
|   | Program START/STOP  | 24 |
|   | – Program start (STOP → RUN)                                    | 24 |
|   | – Behaviour after power off or power interruption               | 24 |
|   | – Program stop (RUN → STOP)                                     | 25 |
|   | Program processing and system time                              | 25 |
|   | Cycle time, monitoring  | 25 |
|   | Reset   | 25 |
|   | – Reset (warm)  | 25 |
|   | – Cold reset  | 25 |
|   | – Full reset  | 25 |
|   | – Reset for restoring the factory defaults.                     | 25 |
|   | – Behaviour of the variables after a Reset                      | 25 |
|   | Test and commissioning  | 26 |
|   | – Breakpoint/single-step mode                                   | 26 |
|   | – Single-cycle mode   | 26 |
|   | – Forcing variables and I/Os                                    | 26 |
|   | – XSoft status indication                                       | 26 |
|   | System events   | 27 |
|   | Interrupt processing  | 27 |
|   | – Parametric programming of the inputs                          | 27 |
|   | – Example for interrupt processing                              | 28 |
|   | – Timer interrupt   | 29 |
|   | Direct I/O access   | 30 |
|   | – ReadBitDirect   | 30 |
|   | – “Error code with direct peripheral access”                    | 31 |
|   | Creating and transferring boot project                          | 31 |
|   | – Saving boot project on MMC                                    | 31 |
|   | – Erase boot project  | 31 |
|   | Operating system, download/update                               | 32 |
|   | – Transferring the operating system from<br>the PC into the PLC | 32 |
|   | – Transferring the OS from the PC into the MMC                  | 33 |
|   | – Transferring the OS from the MMC into the PLC                 | 33 |
|   | User program source code  | 33 |
| <b>7 Browser commands</b>                         |   | 35 |
|   | – reflect   | 36 |
|   | – canload   | 36 |
|   | – setrtc  | 36 |
| <b>8 Libraries, function blocks and functions</b> |   | 37 |
|   | Using libraries   | 37 |
|   | Installing additional system libraries                          | 37 |
|   | XC121 specific functions  | 38 |
|   | – “Library XC121_Util.lib”                                      | 38 |
|   | – Function CAN_BUSLOAD  | 38 |
|   | – Function GETAPPLICATIONSWITCH                                 | 38 |

|  |   |    |
|--|---|----|
| <b>9 Connection set-up PC – XC121</b>                        |   | 39 |
|  | Communication settings of the PC                                      | 39 |
|  | Communication settings (baud rate) of the CPU                         | 39 |
| <b>10 Set the system parameters via the STARTUP.INI file</b> |   | 41 |
|  | Parameter overview  | 41 |
|  | Structure of the INI file   | 41 |
|  | INI file generation   | 41 |
|  | Entries of the INI file   | 42 |
|  | Start behaviour of the XC121 with inserted<br>MMC containing INI file | 42 |
|  | Changing settings   | 42 |
|  | Delete INI file   | 42 |
| <b>11 Programming via CANopen network (Routing)</b>          |   | 43 |
|  | Prerequisites   | 43 |
|  | Routing properties of the XC121                                       | 43 |
|  | – Setting via the PLC Configurator                                    | 43 |
|  | – Setting via the APPLICATION switch                                  | 44 |
|  | Routing through XC200   | 44 |
|  | Notes   | 44 |
|  | Addressing  | 45 |
|  | Procedure   | 45 |
|  | PLC combinations for routing  | 46 |
| <b>12 RS232 interface in transparent mode</b>                |   | 47 |
| <b>Appendix</b>  |   | 49 |
|  | Dimensions  | 49 |
|  | Technical data  | 50 |
|  | – XC-CPU121/XIO-EXT121-1  | 50 |
|  | – XC-CPU121   | 51 |
|  | – XIO-EXT121-1  | 53 |
|  | – 24 V DC line filter XT-FIL-1  | 55 |
| <b>Index</b>   |   | 57 |



## About this manual

---

### Additional documentation

In some places this manual contains references to more detailed descriptions in other manuals, which are described with their title and documentation number (e.g. AWB2786-1452GB).

Concrete information regarding communication with CAN stations and their configuration can be found in the following listed documentation:

- AN27K18GB: Interfacing of an XI/ON Station to the XC100/XC200 via CANopen Fieldbus
- AN27K19GB: Communication between two controls using network variables via CANopen
- AN27K20GB: Coupling multiple autonomous controls (CAN-Device) via CANopen
- AN27K27GB: Engineering of CAN stations
- AWB2786-1554: Library description CANUser.lib, CANUser\_Master.lib.

All manuals are available in PDF format. If you cannot find a particular manual on the product CD, you can download it as a PDF file from our website: Go to <http://www.moeller.net/support> and enter the document number in the Quick Search field.

---

### Reading conventions

Select «File → New» means: activate the instruction New in the File menu.



#### Important

Indicates a risk of material damage.



#### Caution!

Indicates a risk of serious material damage or slight injury.



#### Warning!

Indicates the risk of major damage to property, or serious or fatal injury.

For clarity of layout, we adhere to the following conventions in this manual: at the top of left-hand pages you will find the Chapter heading, at the top of right-hand pages the current Section heading; exceptions are the first pages of Chapters and empty pages at the end of Chapters.



# 1 Design of the XC121

The PLC XC121 is designed for application in machine and system control units. XSoft from Version 2.3.5 is required for programming.

The CPU XC-CPU121-2C256K can be applied autonomously and connected to the input/output devices via the CANopen interface. The I/O module XIO-EXT121-1 of the same construction design which features analogue and digital I/Os serves as the local expansion module for the CPU with inputs/outputs (I/O). All further X/IOC signal modules can be plugged into the module with the exception of the PROFIBUS-DP modules.

The XC121 PLC consists of the:

- XC-CPU121 with power supply unit
- I/O module XIO-EXT121-1 with power supply unit and digital and analogue I/Os, which are referred to as local I/Os in the following.
- Central X/IOC signal modules.

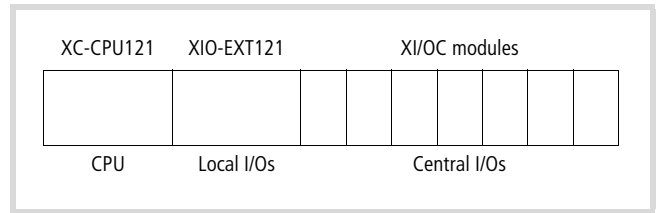


Figure 1: Design architecture

The CPU and the CPU-I/O module have the backplane with three slots as their base platform. Connect the XIO-EXT121 module with the CPU in order to expand the CPU with I/Os. You can then connect the backplanes XIOC-BP-3 and XIOC-BP-2 into which the X/IOC signal modules are inserted. An expansion within the X/IOC backplane is implemented using the backplane XIOC-BP-EXT. In the basic version with the X/IOC backplane a maximum of seven slots are available, with a maximum of 15 slots for the X/IOC signal modules when full expansion is implemented.

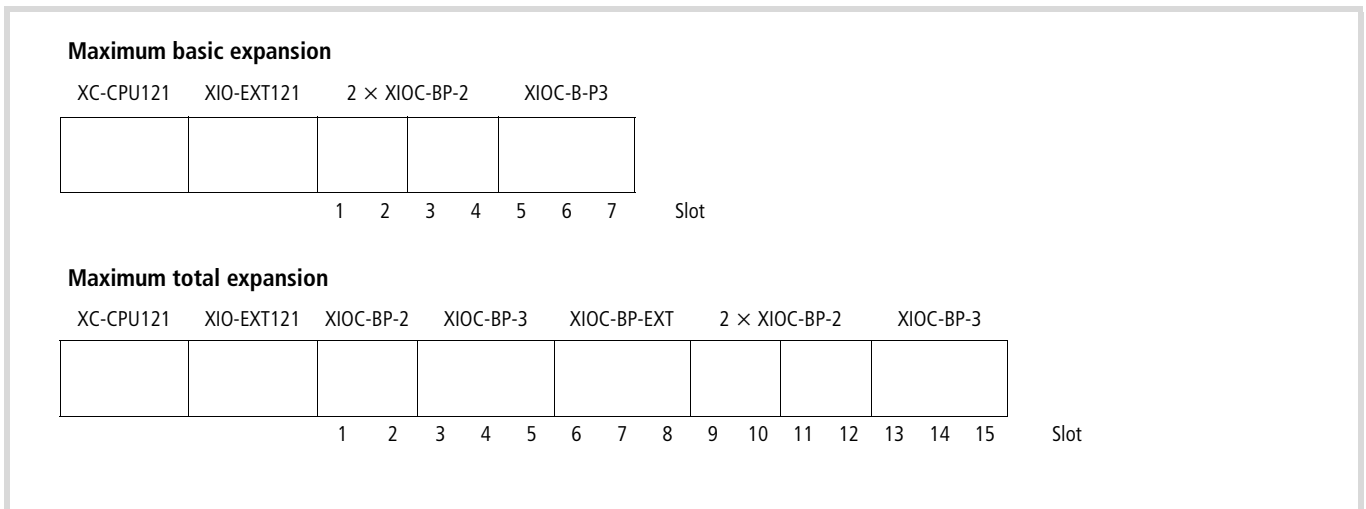


Figure 2: Expansion with XIOC signal modules

Detailed information concerning the backplanes and the X/IOC signal modules can be found in the "X/IOC Signal Modules" manual (AWB2715-1452GB).



## 2 XC-CPU121 functions

The functions are described in detail in the following.

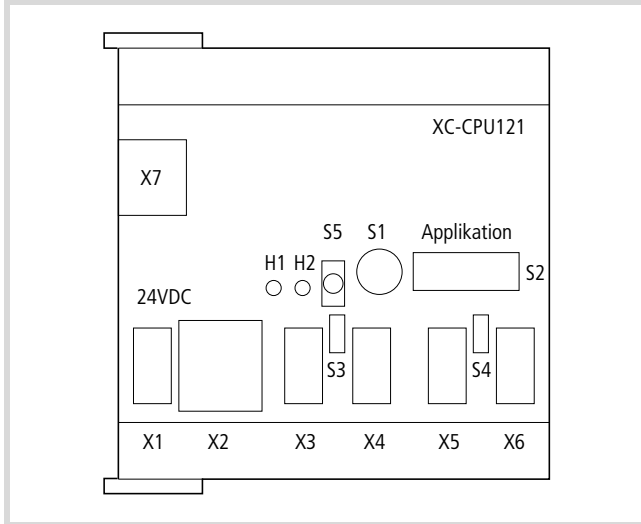


Figure 3: Overview XC-CPU121

Legends to figure 3:

|    |   |
|----|---|
| H1 | LED display RUN/STOP  |
| H2 | LED display SF  |
| S1 | Operating mode switch   |
| S2 | APPLICATION switch  |
| S3 | Switch for bus termination resistor for CAN1 interface        |
| S4 | Switch for bus termination resistor for CAN2 interface        |
| S5 | SET button  |
| X1 | Connection for power supply                                   |
| X2 | COM1 interface (RS232) for connection of a programming device |
| X3 | COM2 interface (RS232/RS485)                                  |
| X4 | CANopen interface CAN1  |
| X5 | CANopen interface CAN2  |
| X6 | CANopen interface CAN2  |
| X7 | Slot for MMC (Multimedia Card)                                |

### Operating mode switch (S1)

With the operating mode switch, you can set the functions shown in Table 1.

Table 1: Operating mode switch functions

| Switch position | Function   |
|-----------------|--|
| 0               | STOP   |
| 1               | RUN<br>Set the operating mode switch to position "1" and then press the set button in order to start the CPU.  |
| 2, 3, ..., 7    | STOP   |
| 8               | Restore factory default state:<br>When you press the SET button for at least three seconds, the values are read.<br>CPU → STOP<br>Important! Remove the Multimedia card beforehand otherwise the program will be erased. |
| 9               | Perform "Cold reset":<br>When you press the SET button for at least three seconds, a Reset is performed. CPU → STOP  |

For further information → chapter "Program START/STOP" from page 24.

### SET button (S5)

The SET button is enabled only in connection with setting 1, 8 and 9 of the operating mode switch. If you press the SET button while the switch is in position 8 or 9, the "Cold reset" or "Restore factory default state" are implemented (RUN/STOP LED flashes quickly).

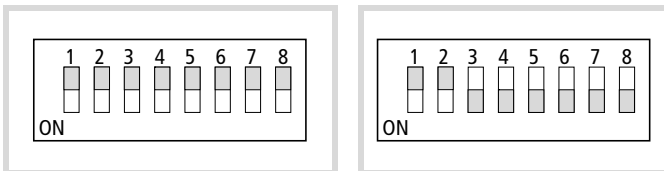
**APPLICATION switch (S2)**

Using the 8-pole DIP switch, values can be set which are to be evaluated differently to suit the set DIP switch mode. It is possible for example, with mode "Node ID CAN1" that you are dealing with the set node address (Node ID) for channel 1.

The DIP switch mode is set in the "PLC configuration" window (see following section). You can query the set value on the switch in the user program with the "GetApplicationSwitch" function. You can find this function in the library "XC121\_Util.lib". It interprets the switch position as a binary value:

- Switch 1 = least significant bit;
- Switch 7 = most significant bit;
- Switch 8 = selection of the CAN interface with Node-ID routing: OFF = CAN1; ON = CAN2.

e.g. provides the function with the switch position value 3 as indicated on the right.



Setting in the default state

Setting for address 3

Figure 4: DIP switch application switch

**DIP switch mode**

The mode is set in the PLC configuration. Activate the "Other Parameters" tab and select one of the following modes in the "DIP-Switch-Mode" field:

- Application
- Node Id CAN1
- Node Id CAN2
- Node Id Routing.

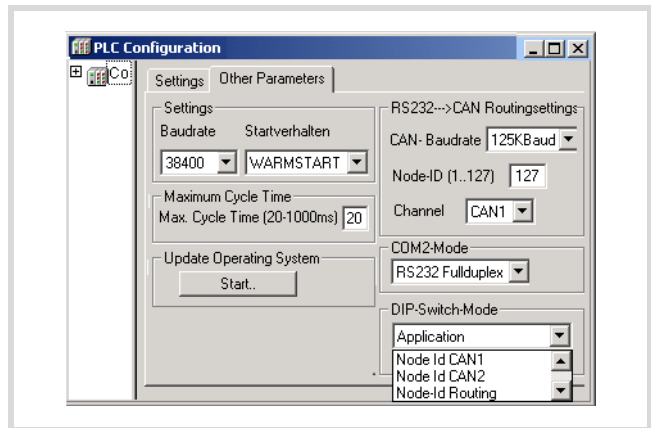


Figure 5: Mode of the application switch

**Application:** This setting has the effect that the DIP switch modes "Node Id CAN1/CAN2" and Node-Id Routing are not active.

**Node Id CAN1/CAN2:** The XC121 can operate for the CAN1 and/or CAN2 channel (interfaces) as a device. A Node-ID can be set on one of both "Node Id CAN1" or "Node Id CAN2" channels on the APPLICATION switch. For example, Node-Id 3 has been set in the representation on the right of figure 4.

Further information can be found at Section "CANopen interface CAN1/CAN2" on page 13.

**Node-Id Routing:** The XC121 can be used as the target control with routing. Access is provided via the CAN1 or CAN2 channel (interface). Set a Routing-Id for the channel on the APPLICATION switch.

In figure 4 Routing-Id "3" has been set on the switch indicated on the right.

Further information can be found at Section "CANopen interface CAN1/CAN2" on page 13.

### Memory card MCC

The MMC serves as mass memory. The operating system (OS) supports memory types with the FAT16 file system. The OS can also be transferred onto the MMC in order to transfer it to another XC121 from there.

→ section “Operating system, download/update” on page 32.



**Important**

The file system of the memory card is not transaction-safe. Ensure that all program files are closed before you insert or remove the MMC or switch off the voltage.

### Data access on the MMC

Using browser commands and functions you can download and transfer general files as well as the boot project or the source code of the project onto the MMC. Use the “copyprojtoMMC” browser command for example, to copy the boot project on the MMC.

A short description of the browser commands can be found from page 35.

The files on the MMC can be accessed with the “FileOpen” or “FileRead” functions from the user program. These functions are described in the library “XC121\_File.lib” and in the manual “Function Blocks for XSoft” (AWB2786-1456GB).

### Erasing functions

If a full reset command is executed in online mode in the PLC Configuration, the operating system and the project on the MMC are deleted. The parameters of the STARTUP.INI file are retained.

You can use the following browser commands:

- “format”: deletes the entire content on the MMC
- „removeprojfrommmc”: deletes the project and the INI file on the MMC. The data on the CPU is retained, → section “Delete INI file” on page 42.

### LED status indication RUN/STOP and SF

Table 2: LED status indicator

| LED   | Meaning  |
|---|--|
| RUN/STP + SF  |  |
| Off + Red   | System test being run (up to 6 seconds after start; after 6 seconds if no user program is present).<br>CPU in NOT READY! |
| Green + Red   | System update in progress  |
| Both flashing   | System test found a fault  |
| Green flashing + Off  | Load user program<br>CPU in STOP!  |
| Green + Off   | Load user program<br>CPU in RUN!   |
| Green flashing + Red  | Group error/diagnosis message issued.  |
| Green (flashing quickly during Reset) + red (dependent on group error Off/On) | Reset through operating mode switch (switch position 9)  |

### Real-time clock

The XC121 features a real-time clock, which can be referenced in the user program via the functions from the “SysLibRTC” library. The functions are described in the SysLibRTC PDF file. The file can be found in the directory “XsoftV2.3.4\Manuals\English\Software\XsoftProfessional\XsoftSyslibs\pdf”.

You can read and set with the browser commands “getrtc” and “setrtc” respectively. Further information can be found at Section “setrtc” on page 36.

During a voltage loss the clock is backed up for at least 72 hours.

**Limit values for memory usage**

The data memory of the XC121 is divided into memory segments. The memory size of the individual segments can be found in figure 6. The global data avails of multiple segments. can be specified to suit the size of the loaded program.

To view the specified segment size for a PLC type, select the Resources tab in the object organizer and double-click Target Settings. In the dialog select the Memory Layout tab:

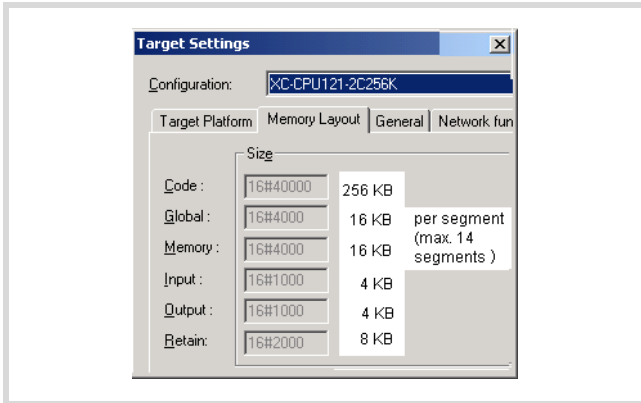


Figure 6: Memory layout

To allow maximum usage of the entire available memory range for global data, set the number of global data segments to 14 when you create a new project. The number of segments is set to 1 by default.

**Changing segment allocation**

Select in the directory "«Project Options → Compilation options»" the "Number of data segments" field and enter "14" for the number of segments with the selected PLC type XC121.

**Serial interface COM1/COM2**

The CPU features two serial interfaces COM1 and COM2.

**COM1: programming interface/transparent mode**

Through the programming interface **COM1** (RS 232) communication between CPU and the programming device takes place. The interface is initialised with the following default parameters when the PLC is started.

|              |            |
|--------------|------------|
| Data length: | 8 bit      |
| Parity:      | none       |
| Stop bits:   | 1          |
| Baud rate:   | 38400 Baud |

Control lines are not supported!

The interface is implemented physically as an RJ-45 socket. It is not electrically isolated.

→ More detailed information about transparent mode can be found from page 47.

Table 3: Assignment of the programming interface

|                                      |   | Signal |
|--------------------------------------|---|--------|
| 8<br>7<br>6<br>5<br>4<br>3<br>2<br>1 | 8 | RxD    |
|                                      | 7 | GND    |
|                                      | 6 | —      |
|                                      | 5 | TxD    |
|                                      | 4 | GND    |
|                                      | 3 | —      |
|                                      | 2 | —      |
|                                      | 1 | —      |

**COM2: transparent mode**

More detailed information about transparent mode can be found from page 47.

The COM2 interface can be switched between RS 232 (full duplex) and RS 485 (half duplex). The setting is performed in the "PLC Configuration".

It is not galvanically isolated and can only be accessed via the function blocks of the user program. You cannot use this interface as a programming interface. It is initialised with the following default parameters:

|              |            |
|--------------|------------|
| Data length: | 8 bit      |
| Parity:      | none       |
| Stop bits:   | 1          |
| Baud rate:   | 38400 Baud |

Further communication parameters can be found at Section "Technical data" on page 50.

Control lines of the RS 232 are not supported.

| COM2                       | Signal RS 232 | Signal RS 485 |
|----------------------------|---------------|---------------|
| 6<br>5<br>4<br>3<br>2<br>1 | 6             | Tx-/Rx-       |
|                            | 5             | Tx+/Rx+       |
|                            | 4             | Vcc           |
|                            | 3             | GND           |
|                            | 2             | RxD           |
|                            | 1             | TxD           |

A 6-pole plug-in springloaded terminal block is used as the connector type.

**CANopen interface CAN1/CAN2**

The PLC features two CANopen interfaces. They are designated as CAN1 and CAN2. The CAN2 interface has assignment designs. It is available on X5 as well as X6.

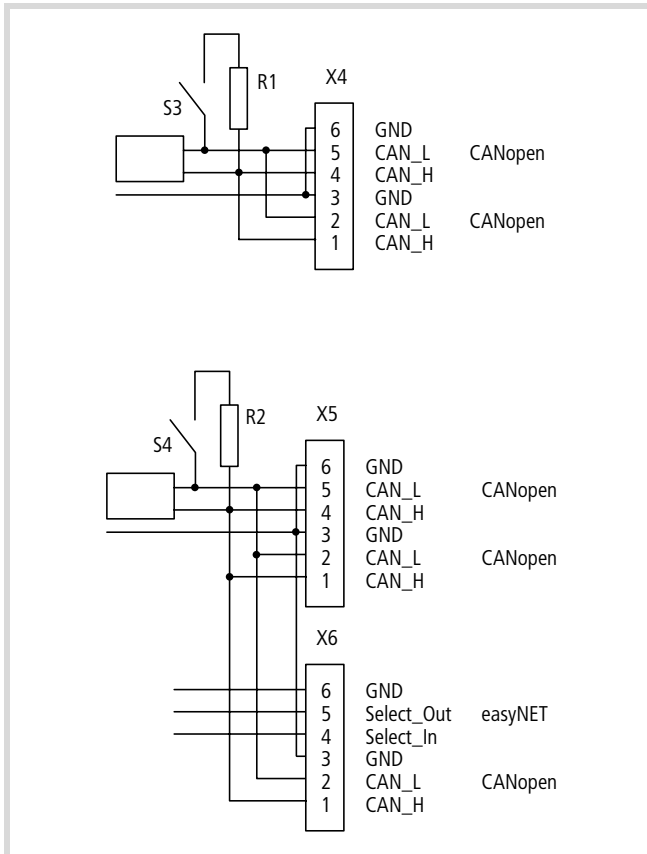


Figure 7: CANopen interface overview

- X 4, 5, 6      6-pole, plug-in springloaded terminal block
- S 3, 4        Switch for bus termination resistor
- R 1, 2        Bus termination resistor 120 Ω

Both CANopen interfaces are designed conform to the CIA specification DS 301 V4.0. The can be operated independently of each other both as a CAN Master as well as a CAN Device.

easy-NET: The interface is planned, it does not have a function.

**XC121 as a CAN Device (CAN1 and/or CAN2)**

In figure 8 an XC121 has been configured with two CAN Device channels. Each channel used must be assigned with a Node-Id which serves as an address. You can choose between two different methods to set the Node-Id.

- Setting via the PLC Configurator
- Setting via the APPLICATION switch.

→ The setting on the APPLICATION switch (for Node-Id and Routing-Id) have priority over the configurator setting.

**Setting via the PLC Configurator**

- ▶ Open the Other Parameters tab in the PLC Configurator. In the "DIP-Switch-Mode" field (→ figure 5) the operating mode Node ID CAN1 or Node ID CAN2 may not be selected!
- ▶ Click on the first folder CANDevice[VAR]. The "Base settings" tab is displayed.

→ The first folder CANDevice[VAR] is fundamentally assigned to the channel CAN1. The following folder contains channel CAN2.

- ▶ Enter any bus name instead of CAN1 (CAN1 has no significance).
- ▶ Change over to the "CAN settings" tab and set the Node-Id and the baud rate.
- ▶ Set the parameters on the CAN2 channel using the same method.

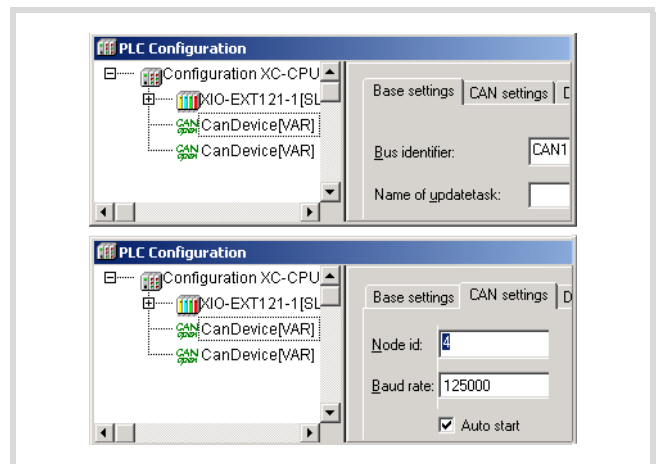


Figure 8: Node-Id setting via configurator (CAN1 = Device)

**Setting via the APPLICATION switch**

The Node-Id of an interface (CAN1 or CAN2) can be set via the APPLICATION switch, the Node-Id of further interfaces must be set in the PLC Configurator. For this purpose the "Node Id CAN..." must be set in the "DIP-Switch-Mode" (→ figure 5 on page 10). Now set the Node-Id in the APPLICATION switch on switches 1 to 7; switch 8 has no function.

| Settings     |  |
|--------------|--|
| Switch 1 – 7 | Node Id 1 –127 (with invalid address 0 the default node Id 127 is used!) |
| Switch 8     | No function  |

How to set the baud rate:

- ▶ Click on the folder "CAN Device [Var]", open the "CAN settings" and enter the baud rate.

**Setting of the XC121 as a CAN Master/CAN Device**

The figure shows an example for the XC121 as a CAN Master and as a CAN Device.

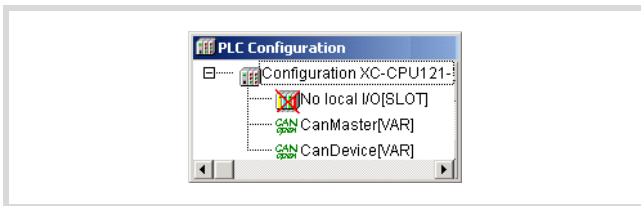


Figure 9: XC121 as CAN Master and as a CAN Device

The first folder CanMaster[VAR] is fundamentally assigned to the channel CAN 1. The second folder "CanDevice[VAR]" is assigned to channel 2. The parameters for channel 2 are set in the PLC Configuration or via the application switch.

→ Transfer rates higher than 500 kbit/s are not possible, even though they are selectable in the PLC configuration.

**CAN Direct**

You can directly access CAN objects via the function blocks of the SysLibCan library (→ Application Note AN27K27). Four independent ports are set up for each of both CAN interfaces:

- CAN1 – CAN4 Access to the CAN line connected to CAN1
- SECOND\_CAN1 – SECOND\_CAN4 Access to the CAN line connected to CAN2

**Bus termination resistors**

For each of both CAN interfaces you can switch in or out the bus termination resistor. The changeover switches S3 and S4 are located beside the plug connectors (→ figure 3).



Switched off bus termination resistor

Switched on bus termination resistor

Figure 10: Switching the bus termination resistors

**Properties of the CANopen cable**

Use only cable approved for CANopen applications and with the following characteristics:

- Characteristic impedance 100 to 120 Ω
- Capacitance < 60 pF/m

The demands placed on the cable, connectors and bus termination resistors are specified in ISO 11898. Following you will find some demands and stipulations listed for the CANopen network.

Table 4 lists default parameters for CANopen networks with fewer than 64 CANopen stations.

Table 4: Standard parameters for CANopen network cable according to the ISO 11898

| Bus length<br>[m] | Loop resistance<br>[mΩ/m] | Conductor cross-section<br>[mm <sup>2</sup> ] | Bus termination resistor<br>[Ω] | Transfer rate at cable length<br>[kbit/s] |
|-------------------|---------------------------|---|---------------------------------|---|
| 0 – 40            | 70                        | 0,25 – 0,34                                   | 124                             | 1000 at 40 m                              |
| 40 – 300          | < 60                      | 0,34 – 0,6                                    | 150 – 300                       | > 500 at 100 m                            |
| 300 – 600         | < 40                      | 0,5 – 0,6                                     | 150 – 300                       | > 100 at 500 m                            |
| 600 – 1000        | < 26                      | 0,75 – 0,8                                    | 150 – 300                       | > 50 at 1000 m                            |

### 3 Mounting

#### Mounting the XC121 on a top-hat rail

- ▶ Hook the XC-CPU121/XIO-EXT121-1 onto the mounting rail from above.
- ▶ Pull the locking slider downwards. **1**
- ▶ Press the bottom of the module against the mounting rail. **2**
- ▶ Push the locking slider up again. **3**
- ▶ Make sure that the module is securely attached to the top-hat rail.

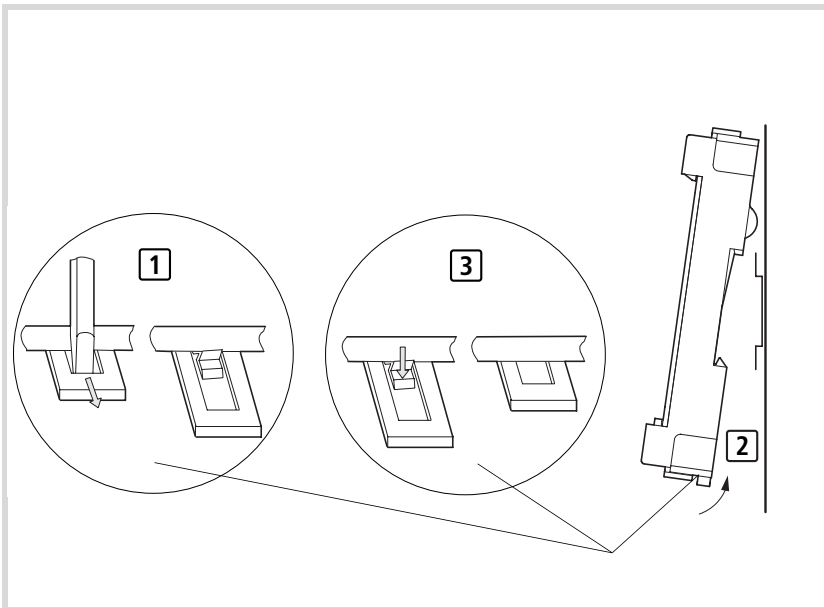


Figure 11: Mounting of the XC121

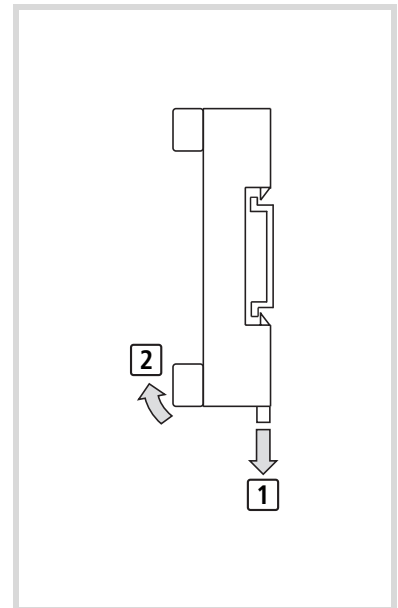


Figure 12: Removing the XC121

#### Mounting the XIO-EXT121-1

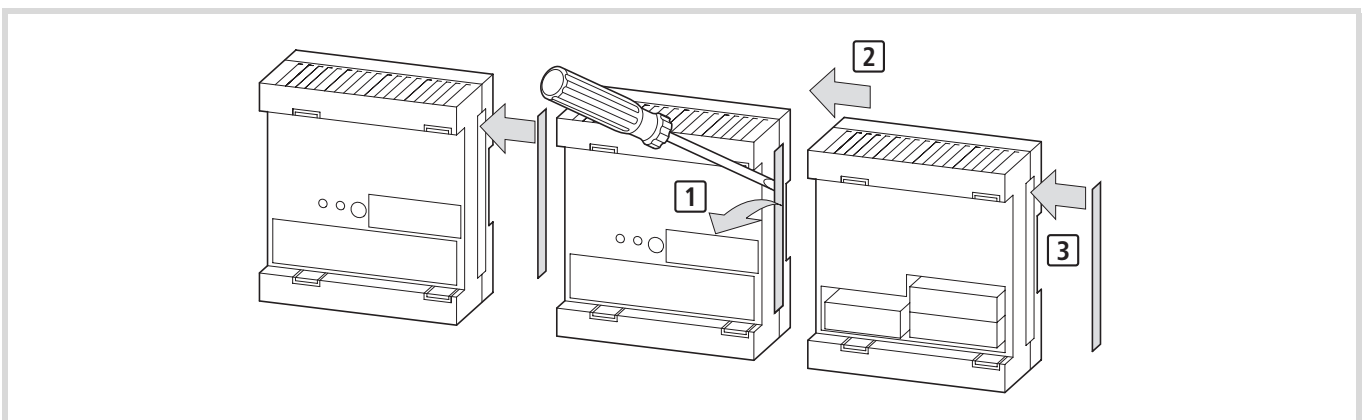


Figure 13: Mounting the XIO-EXT121-1 on the XC121

### APPLICATION switch setting

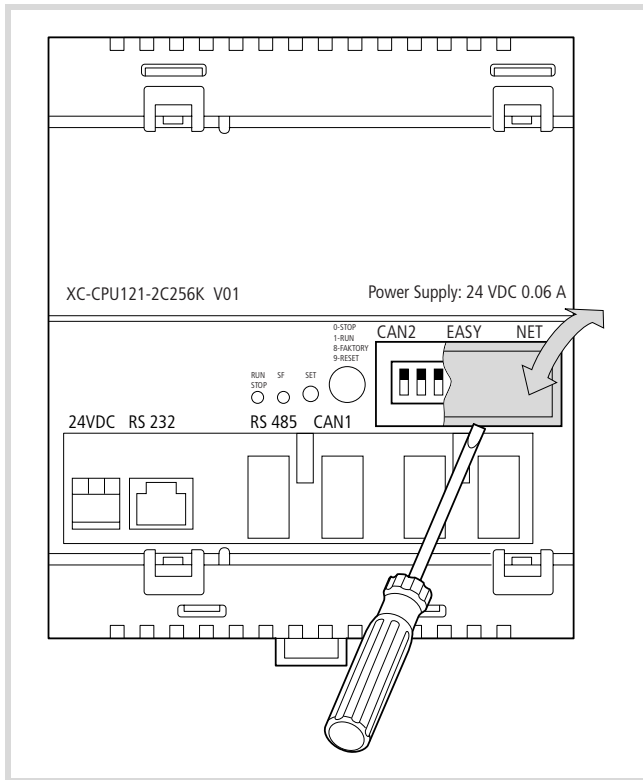


Figure 14: APPLICATION switch setting

### Input/output wiring

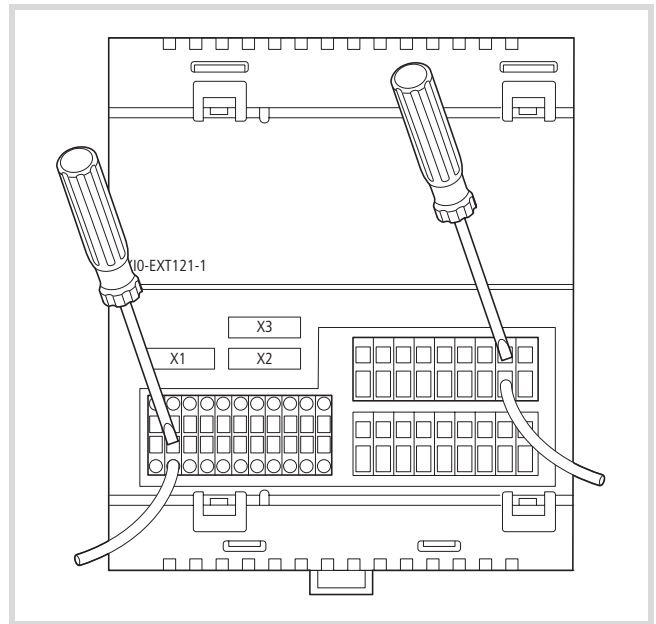


Figure 15: Input/output wiring

## 4 Engineering

### Control panel layout

The layout of the components inside the control panel is a major factor for achieving interference-free functioning of the plant or machinery. During the project planning and design phase, as well as its implementation, care must be taken that the power and control sections are separated. The power section includes:

- Contactors
- Coupling/interfacing components
- Transformers
- Frequency inverters
- Converters

In order to effectively exclude any electromagnetic contamination, it is a good idea to divide the system into sections, according to their power and interference levels. In small switchgear cabinets it is often enough to provide a sheet steel dividing wall, to reduce interference factors.

### Ventilation

A clear space of at least 50 mm must be kept between passive components, to ensure adequate ventilation. If the neighbouring components are active elements, such as power supplies or transformers, then the minimum spacing should be 75 mm. The values given in the technical specifications must be observed.

### Layout of units

Mount the PLC horizontally in a control panel:

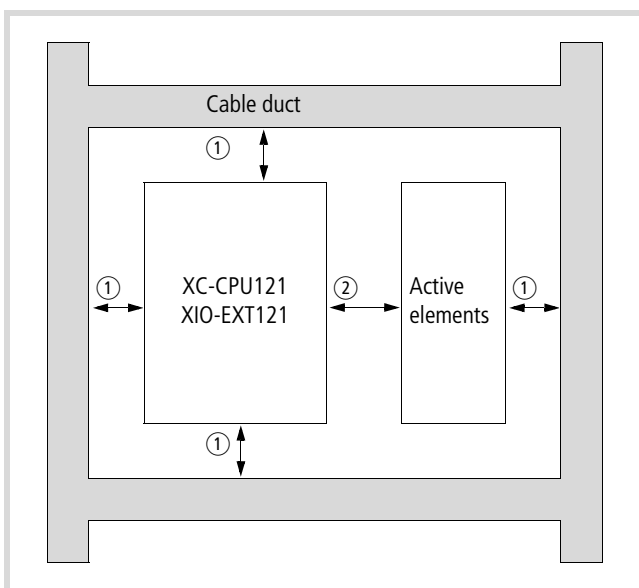


Figure 16: Cabinet layout

- ① Spacing > 50 mm
- ② Spacing > 75 mm

### Preventing interference

#### Cable routing and wiring

Cables are divided into the following categories:

- Power cables (e.g. cables that carry high currents, or cables to converters, contactors or solenoids)
- Control and signal cables (e.g. for digital inputs)
- Measurement and signal cables (e.g. for fieldbus connections)

→ Always route power cables and control cables as far apart as possible. This avoids capacitive and inductive coupling. If separate routing is not possible, then the first priority must be to shield the cable responsible for the interference.

Take care to implement proper cable routing both inside and outside the control panel, to keep interference as low as possible:

- ▶ Avoid parallel routing of sections of cable in different power categories.
- ▶ As a basis rule, keep AC cable separated from DC cables.
- ▶ Keep to the following minimum spacing:
  - at least 10 cm between power cables and signal cables;
  - at least 30 cm between power cables and data or analog cables.
  - When routing cables, make sure that the outgoing and return leads of a circuit pair are routed together: The currents flowing in opposite directions thus cancel each other out as a summation, and the electromagnetic fields cancel each other out.

#### Suppressor circuitry for interference sources

- ▶ Connect all suppressor circuits as close to the source of interference (contactors, relays, solenoids) as possible.

→ Switched inductors should always have suppressor circuitry fitted.

#### Shielding

- ▶ Use shielded cables for the connections to the data interfaces. The general rule is: the lower the coupling impedance, the better the shielding effect.

## Lighting protection

### External lightning protection

All cables that go outside buildings must be shielded. Metal conduit is best for this purpose. Fit signal cables with overvoltage protection, such as varistors or other surge voltage protectors. Where possible, protective elements should be fitted at the point of entry of the cable into the building, but no further away than the control panel.

### Internal lightning protection

Internal lightning protection covers all those measures taken to reduce the effects of a lightning strike and the resulting electrical and magnetic fields on metallic installation and electrical plant. These measures are:

- Equipotential bonding/earthing
- Shielding
- using overvoltage protection devices.

Please consult the following manuals for advice on cable routing and shielding measures:

- AWB27-1287 "EMC Engineering Guidelines for Automation Systems".
- TB27-001-GB "Electromagnetic Compatibility (EMC) for Automation systems".
- TB02-022-GB "Electromagnetic Compatibility (EMC) for Machinery and Plants".

## Connections

### Connecting the power supply

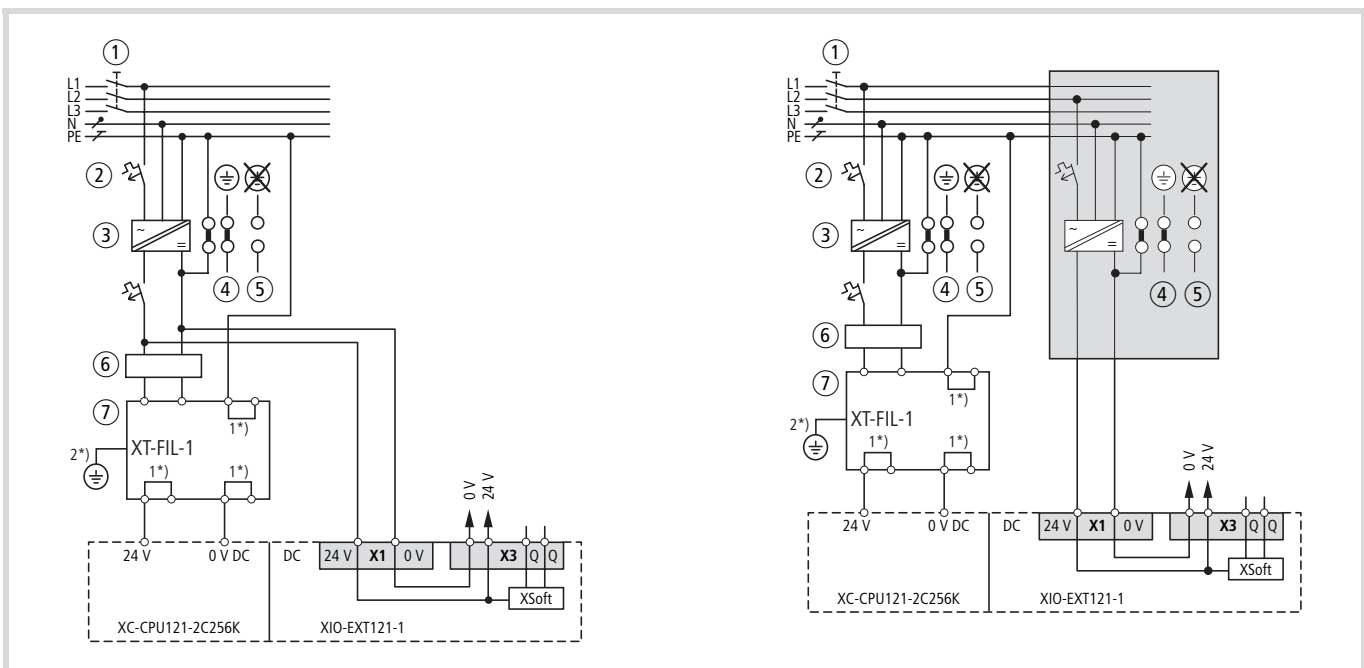


Figure 17: Wiring example: XC-CPU121 and XIO-EXT121-1 (left: common power supply, right separate supplies)

- ① Main switch
  - ② Protective cut-out
  - ③ 24 V DC supply voltage
  - ④ Earthed operation
  - ⑤ In floating (i.e. unearthed) operation, an isolation monitor must be used (IEC 204-1, EN 60204-1, DIN EN 60204-1)
  - ⑥ Ferrite ring
  - ⑦ 24 V DC line filter; ensures that a current of up to 2.2 A (maximum) is available at a rated voltage of 24 V DC. Ensures that the EMC stipulations for devices are fulfilled when the filter is used. The filter is not a component of the CPU and must therefore be ordered separately:  
Type: XT-FIL-1, Article no.: 285316 (Supplier: Moeller GmbH)  
→ Dimensions on page 50  
→ Technical data on page 55
- 1\*) Internal jumper  
2\*) Additional PE connection via contact spring on rear

→ When the XC-CPU121 is switched on, connected with the XIO-EXT121:

Each device features a separate supply voltage connection. The voltage on both devices must be switched on to start the CPU. If only one of the devices is switched on the CPU will not run a program and the LEDs will remain off.

**Connecting the XC-CPU121 supply voltage**

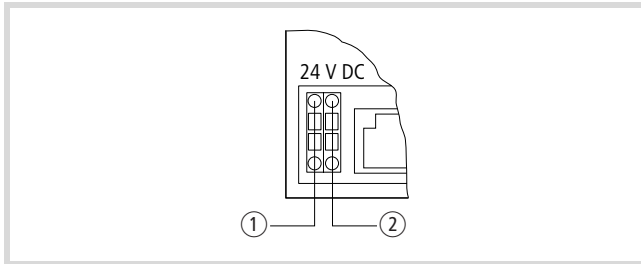


Figure 18: Voltage connection of the XC-CPU121

- ① 24 V DC
- ② 0 V

Refer to figure 21 for the voltage supply of the XIO-ETX121-1, plug connector X1

**Connecting sensors and actuators**

The digital sensors and actuators can be connected directly to plug connectors X2 and X3:

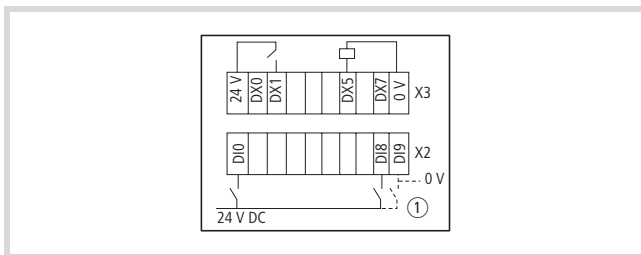


Figure 19: Connecting inputs/outputs to X2/X3

- ① If you use the connector type with LED display BLI/O3.5/10F on plug connector X2, you have to connect the terminal "D19" to 0 V.

The sensors and/or actuators can be connected at the terminals DX0 to DX7 of connector X3. Each terminal (connection) can be set (QX0.0 to QX0.7) or read (IX0.0 to IX0.7) from the user program.

The 24 V DC/0 V on connector X3 is used to provide a power supply to the sensors/actuators.

Two connector devices are available:

- without LED
- with LED

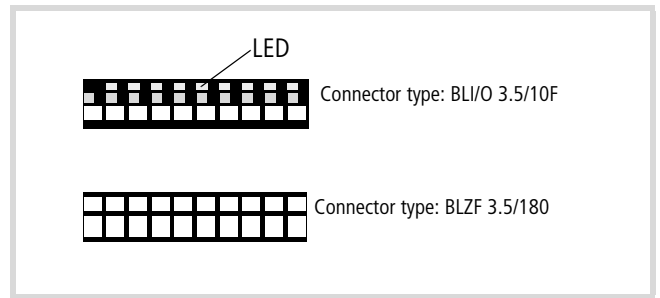


Figure 20: Connector types

You can use both connector types for the adapter extension X2 and/or X3.

The analog sensors/actuators are connected to plug connector X1:

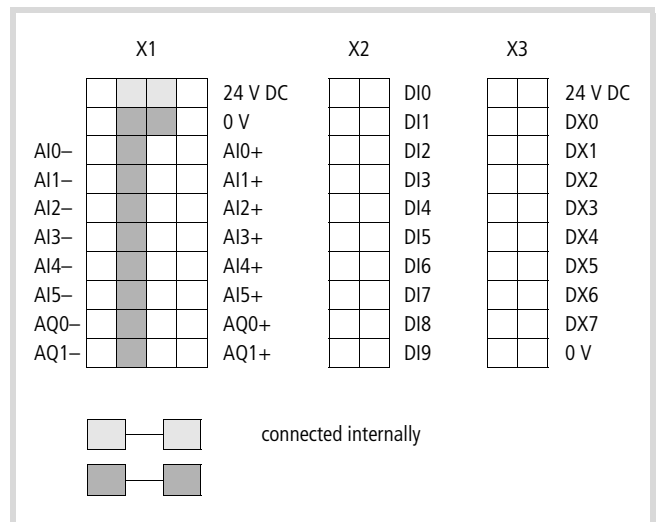


Figure 21: Designation of the plug connector and the inputs/outputs

Table 5: Analog input/output properties

| Analog type           | Resolution | Operand | Terminal (X1) |
|-----------------------|------------|---------|---------------|
| <b>Analog inputs</b>  |            |         |               |
| 0 – 10 V              | 10 Bit     | IW4     | AI0 -/+       |
| 0 – 10 V              | 10 Bit     | IW6     | AI1 -/+       |
| 0 – 20 mA             | 10 Bit     | IW8     | AI2 -/+       |
| 0 – 20 mA             | 10 Bit     | IW10    | AI3 -/+       |
| Pt100                 | 10 Bit     | IW12    | AI4 -/+       |
| Pt100                 | 10 Bit     | IW14    | AI5 -/+       |
| <b>Analog outputs</b> |            |         |               |
| 0 – 10 V              | 12 Bit     | QW2     | AQ0 -/+       |
| 0 – 10 V              | 12 Bit     | QW4     | AQ1 -/+       |

Table 6: Analog values

| 0 – 10 V       | 0 – 20 mA | dec  | hex |
|----------------|-----------|------|-----|
| Analog inputs  |           |      |     |
| 0              | 0         | 0    | 000 |
| 5              | 10        | 511  | 1FF |
| 10             | 20        | 1023 | 3FF |
| Analog outputs |           |      |     |
| 0              | –         | 0    | 000 |
| 5              | –         | 2047 | 7FF |
| 10             | –         | 4095 | FFF |

## 5 Configuration of the XIO-EXT121-1

In order to expand the XC121 with the I/O module XIO-EXT-121-1 the folder "No local I/O" must be replaced in the PLC Configurator by the element XIO-EXT121-1.

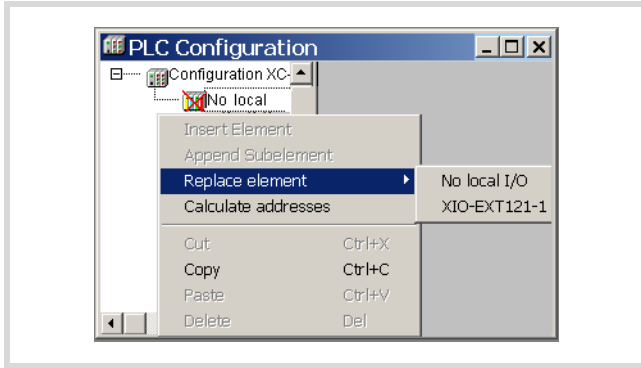


Figure 22: Configuration of XIO-EXT121-1

The new window displays all inputs and outputs of the I/O module.

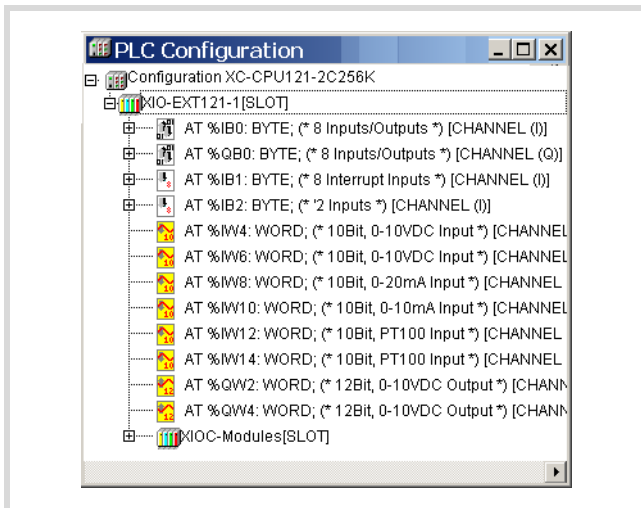


Figure 23: Configuring the inputs and outputs

The following overview indicates the assignment of the input / output syntax as indicated in the configurator to the connections on the plug connector.

Table 7: Overview of the I/O signals (analog)

| Plug connector type | Connector I/Q | Operand I Word | Operand Q Word | Type I/Q analog |
|---------------------|---------------|----------------|----------------|-----------------|
| X1                  | AI0           | IW4            |                | I 0 – 10 V DC   |
|                     | AI1           | IW6            |                | I 0 – 10 V DC   |
|                     | AI2           | IW8            |                | I 0 – 20 mA     |
|                     | AI3           | IW10           |                | I 0 – 20 mA     |
|                     | AI4           | IW12           |                | I Pt100         |
|                     | AI5           | IW14           |                | I Pt100         |
|                     | AQ0           |                | QW2            | Q 0 – 10 V DC   |
|                     | AQ1           |                | QW4            | Q 0 – 10 V DC   |

Table 8: Overview of the I/O signals (digital)

| Plug connector type | Connector I/Q | Operand I |      | Operand Q |      | Type I/Q digital |
|---------------------|---------------|-----------|------|-----------|------|------------------|
|                     |               | Bit       | Byte | Bit       | Byte |                  |
| X2                  | DI0           | IX1.0     | IB1  |           |      | I                |
|                     | ...           | ...       |      |           |      |                  |
|                     | DI7           | IX1.7     |      |           |      |                  |
|                     | DI8           | IX2.0     | IB2  |           |      |                  |
| X3                  | DI9           | IX2.1     |      |           |      | I/Q              |
|                     | DX0           | IX0.0     | IB0  | QX0.0     | QB0  |                  |
|                     | ...           | ...       | ...  | ...       | ...  |                  |
|                     | DX7           | IX0.7     |      | QX0.7     |      |                  |

See also „Designation of the plug connector and the inputs/outputs“, figure 21 on page 19.



## 6 Operation

### Switch-on behaviour

After switching on the supply voltage the CPU performs a system self-test. If the test has been completed successfully, the runtime system starts. The red and green LED flashes if there is a malfunction.

After the start of the runtime system the CPU checks if an operating system update is available on the inserted MMC and also if this must be loaded. As the PLC does not feature a battery for backup of the main (SRAM) memory, after the start of the runtime system it checks if a boot project is available. If this is the case, it is loaded in the main (SRAM) memory and starts

dependent on the position of the operating mode selection switch and the set start behaviour. If the flash memory contains no boot project, the PLC remains in NOT READY state.

### Switch on behaviour with boot project

When the PLC is started a boot project available on the MMC has priority over a project stored in the system memory (Flash). If the boot projects are different, the boot project from the MMC is copied into the system memory (Flash) and then run. Due to the copy process the PLC start-up phase will be extended by a few seconds.

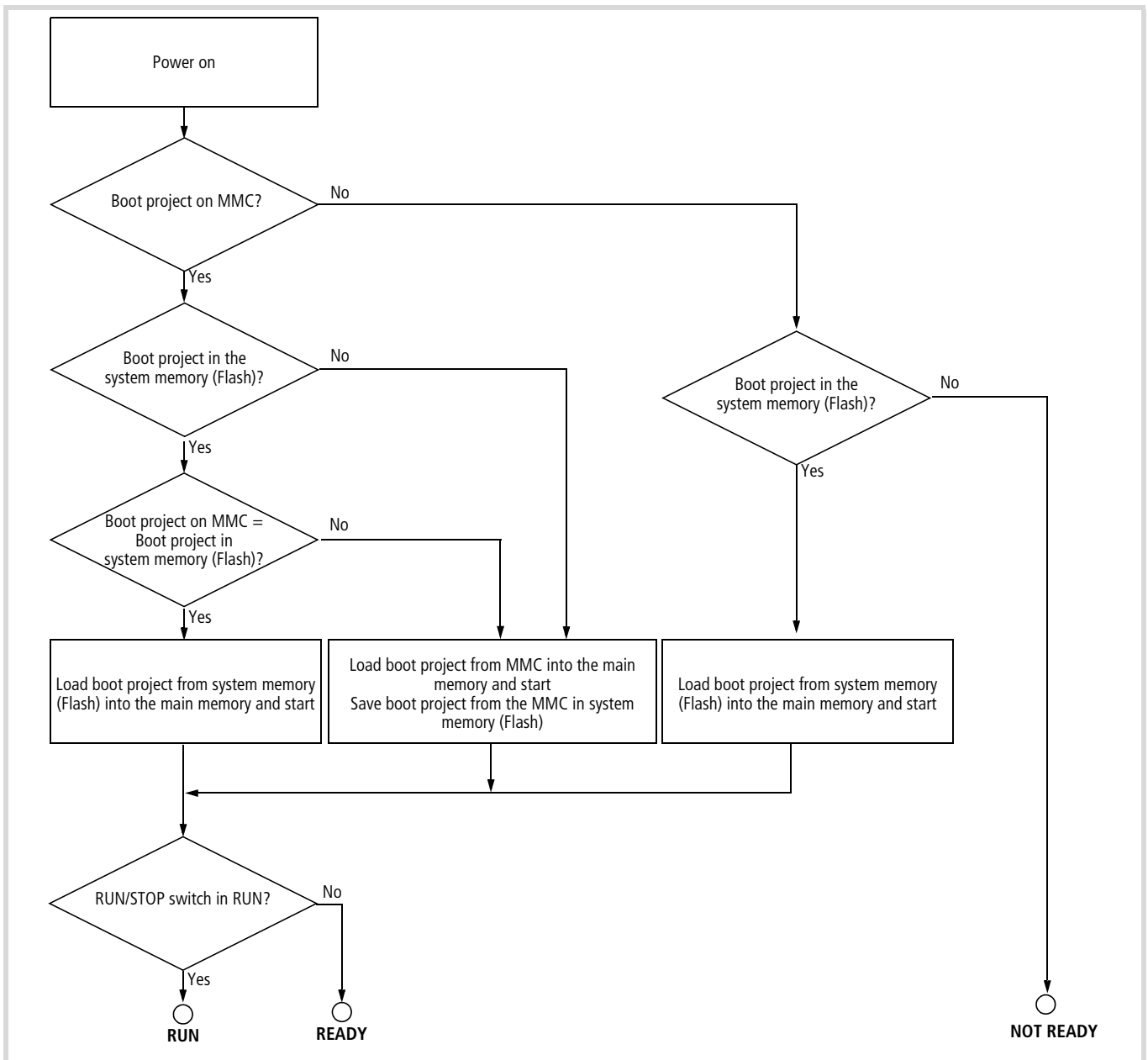


Figure 24: Switch on behaviour with boot project

### Configuring the start-up behaviour with XSoft

With the setting of the start-up behaviour you determine the start behaviour of the PLC when the supply voltage is switched on.

You can change the settings under PLC Configuration: Activate the "Other Parameters" tab there and select the desired start condition from the dropdown list.

- HALT
- WARMSTART
- COLDSTART.

Refer to Table 9 for the behaviour of the variables to suit the set start conditions.

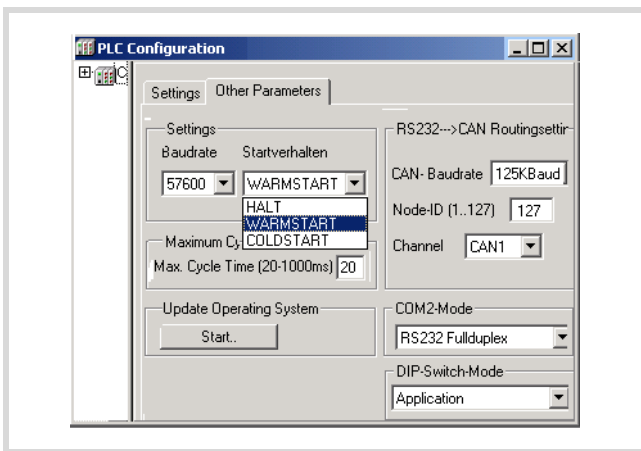


Figure 25: Definition of start behaviour

### Program START/STOP

#### Program start (STOP | RUN)

You can start the program in one of two ways:

- In online operation, issue the START command, for example after loading a program. The CPU must be in STOP state and the operating mode switch in the RUN position.
- Set the operating mode switch to its RUN position.

➔ The position of the operating mode switch has priority over the online command.

Table 9: Behaviour of the variables at PLC start

| Start-up condition                             | Variable type                    |                         |
|--|----------------------------------|-------------------------|
|  | Non-retentive                    | Retentive               |
| COLDSTART                                      | Activation of the initial values |                         |
| WARMSTART                                      | Activation of the initial values | Values remain in memory |
| Program loaded and started in online operation | Activation of the initial values |                         |
| Start/Stop/Start...                            | Values remain in memory          |                         |

#### Behaviour after power off or power interruption

If you switch off or interrupt the (CPU) voltage, the program cycle or task is interrupted immediately. The retentive data integrity is no longer given. All outputs are set to 0 or switched off.

The behaviour of retentive variables is shown in Table 9.

The remaining program cycle will not be completed when power is reconnected!

If inconsistent data is not acceptable in your application, you can, for example, use an uninterruptible power supply (UPS) with back-up.

The PLC restarts as defined by the settings in the PLC Configuration window, ➔ figure 25.

**Program stop (RUN | STOP)**

When you set the operating mode switch to STOP, the CPU changes to STOP state, as soon as the program cycle is completed. The outputs are set to 0.

You can stop the program in one of two ways:

- In online operation, issue the STOP command.
- Set the operating mode switch to its STOP position.

→ The position of the operating mode switch has priority over the online setting.

**Program processing and system time**

The user program is processed cyclically. The states of the inputs are read before the start of each program cycle, and the output states are written to the outputs at the end of the cycle.

As a result of the software architecture of the run-time system, timing divergence's may occur between individual processing cycles.

You can also program application routines that are started by the occurrence of system events; → section "System events" on page 27.

**Cycle time, monitoring**

A hardware timer monitors the cycles of the user program and the individual event tasks. If the cycle time exceeds a user-defined value, the PLC goes into STOP state and the outputs are switched off.

You can specify the timeout value on the "Other Parameters" tab in the "PLC Configuration" window between 20 ms (default value) and 1 000 ms.

**Reset**

There are four different Reset commands:

- Reset (warm)
- Cold reset
- Full reset
- Reset for restoring the factory defaults.

**Reset (warm)**

- The program is stopped.
- The non-retentive variables are initialised, the "Retain" variables are retained.
- The program can be restarted.

**Cold reset**

- The program is stopped.
- All variables are initialised.
- The program can be restarted.

**Full reset**

- The program in the PLC and the boot project are deleted.
- With inserted MMC:
  - All project-specific files and the boot project are erased
  - All user specific files and the startup.ini file remain unchanged
- The PLC is set into the NOT READY state.

**Reset for restoring the factory defaults.**

A prerequisite for the reset is that the operating mode switch is in position 8. If you then press the SET button, all interfaces are initialised with their default parameters. A loaded user program, all variables and the boot project are erased in the system memory (Flash) and on the MMC.

**Behaviour of the variables after a Reset**

| Reset                    | Variable type                    |                         |
|--------------------------|----------------------------------|-------------------------|
|                          | Non-retentive                    | Retain                  |
| Warm reset               | Activation of the initial values | Values remain in memory |
| Cold reset               | Activation of the initial values |                         |
| Full reset <sup>1)</sup> | Activation of the initial values |                         |

1) After a full reset, the program must be reloaded. In online operation, you can then restart the PLC.

## Test and commissioning

The PLC supports the following test and commissioning features:

- Breakpoint/single-step mode
- Single-cycle mode
- Forcing
- Online modification
- Progression display (Power Flow).

### Breakpoint/single-step mode

You can set breakpoints within the user program. If an instruction has a breakpoint attached, then the program will halt before execution of the program line. The following program instructions can be executed in single-step mode. Cycle-time monitoring is disabled.



#### Caution!

Any outputs already set when the program reaches the breakpoint remain set!



Use breakpoint/single-step mode and single-cycle mode only in the application's actual main program. Do **not** use them in den event routines, for example for start, stop and interrupt events, as this can cause problems in the control sequence.

XSoft does not prevent the use of breakpoints in the event routines.

### Single-cycle mode

In single-cycle operation, one program cycle is performed in real time. The outputs are enabled during the cycle. The cycle-time monitoring is active.



#### Caution!

Any outputs already set when the program reaches the breakpoint remain set!

### Forcing variables and I/Os

All variables of a user program can be forced into fixed values. Forced local outputs of the XI/ON modules are only switched through to the I/O in the RUN state.



The I/O connected through the CANopen field bus can not be forced.

### XSoft status indication

- The signal states of the physical, Boolean inputs are displayed in both the CPU's RUN state and in STOP.
- The signal states of the physical, Boolean inputs are only displayed in RUN state; in the STOP state they are designated with FALSE.
- All other variables are displayed with the current variable value.

## System events

You can respond to PLC system events with a user application routine (POU) that runs once when a particular event occurs. Its execution is time-monitored. The time base is the configured longest permissible cycle time.

Events are:

|           |   |
|-----------|---|
| STOP      | User program stop<br>(does not apply to cycle time timeout or hardware watchdogs) |
| START     | START: User program start<br>(cold and warm start)                                |
| COLDSTART | Cold start of the user program  |
| WARMSTART | Warm start of the user program  |

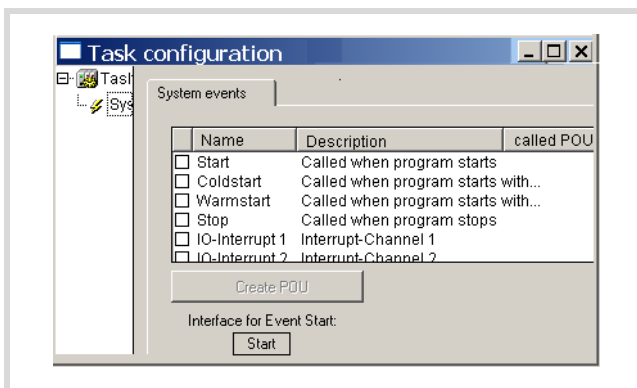


Figure 26: System events

→ Single step mode can not be used for system event program blocks.

## Interrupt processing

If an interrupt occurs the operating system runs the POU which is linked to the interrupt source.



### Caution!

The execution of the interrupt POU is not time monitored. Inadvertently programmed endless loops cant be exited.

The POU initiated by the interrupt is always run to completion and cannot be interrupted by a new interrupt. A new interrupt is only carried out after the current interrupt has ended.



### Important

All the outputs controlled (H signals) up to this point remain active and cant be switched off.

The interrupts are enabled in the RUN state of the CPU and inhibited in the STOP state. Interrupt sources which are not enabled in the configuration do not initiate an interrupt. If a POU is not assigned to an enabled interrupt source, the interrupt is recognised and executed but without running a POU.

If interrupts occur too frequently during operation of the program the programmed task time may be exceeded and the Watchdog will initiate a RESET.

You can inhibit and release interrupts from the program. The "DisableInterrupt" and "EnableInterrupt" functions are provided for this purpose. A call parameter in the XSoft determines if an individual interrupt or all interrupts are enabled or inhibited. Enabling of an inhibited interrupt must be performed with the same parameter used to inhibit it.

Both the "DisableInterrupt" and "EnableInterrupt" functions are components of the "XC121\_Util.lib" library. This library must – if not already done so – be integrated into the library manager of the XSoft.

**DisableInterrupt:** With this function, you disable (deactivate) a parameterized physical interrupt by accessing it from the user program.

**EnableInterrupt:** With this function, the physical interrupt which was deactivated beforehand can now be re-enabled as an active interrupt.

## Parametric programming of the inputs

The inputs I 1.0 (DI0) to I 1.7 (DI7) on plug connector X2 can be used as standard or as interrupt inputs. The inputs I 2.0 (DI8) and I 2.1 (DI9) are standard inputs. The individual interrupt inputs are assigned with their interrupt function only after the inputs are allocated with their edge type which triggers the interrupt.

The allocation is implemented in the PLC Configurator in the "Other Parameters" tab under "Local Interrupts".

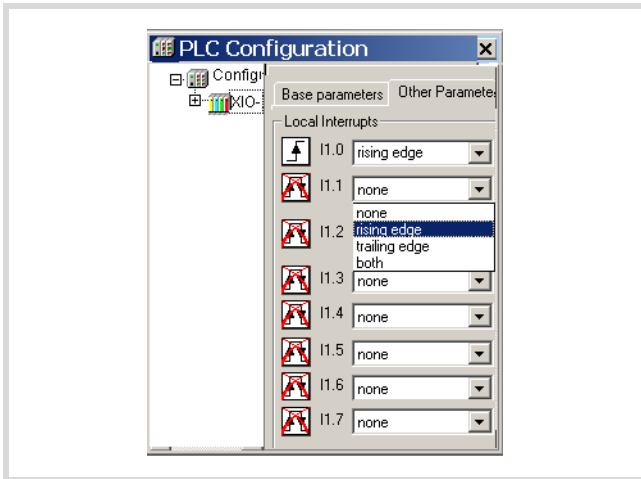


Figure 27: Defining the edge type on the local interrupt

The link between the IO interrupt and POU is made in the task configurator.

The interrupt channels 1 to 8 (channels) are allocated directly to the inputs I1.0 to I1.7. The priority of the inputs is fixed: channel 1 (input 1.0) has the highest priority, channel 8 (input 1.7) has the lowest priority (see the example).

**Example for interrupt processing**

A "Basic" task contains a POU "PLC\_PRG". A further POU "Fastprog" should be processed if an L → H rising edge on the input I1.2 generates an interrupt.

- ▶ Create the POU's "PLC\_PRG" and "Fastprog" as shown in figure 28.

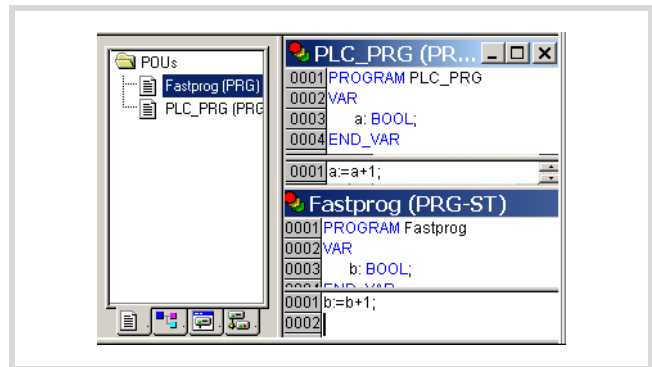


Figure 28: Writing a program

- ▶ Change over to the PLC Configuration and allocate the "rising edge" type to the input I1.2.

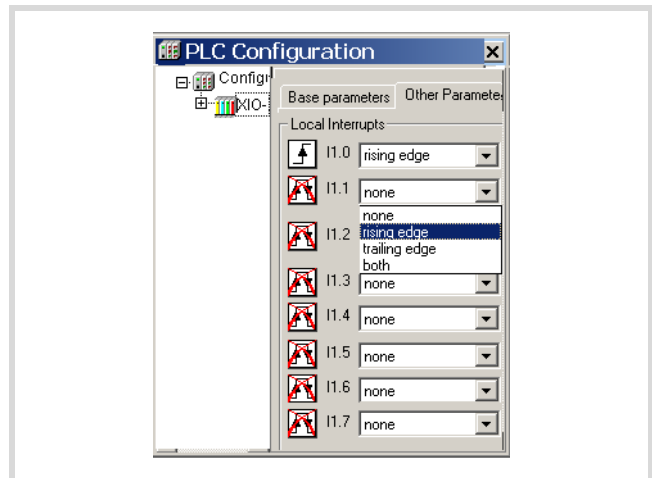


Figure 29: Interrupt edge selection

- ▶ Change over to the Task configuration and open the "System events" folder.

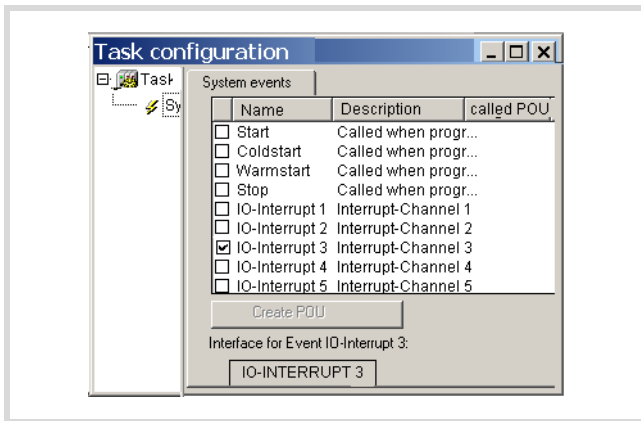


Figure 30: Enabling an interrupt

- ▶ Enable IO-Interrupt 3 by clicking in the checkbox on the left beside the name "IO-Interrupt 3". The box is checked to indicate that it has been activated.
- ▶ Mark the area of column "Called POU" and the area and the line "IO-Interrupt 3".
- ▶ Set the cursor on the marked area and press the function key F2.

The "Help Manager" window opens in which all predefined programs are listed:

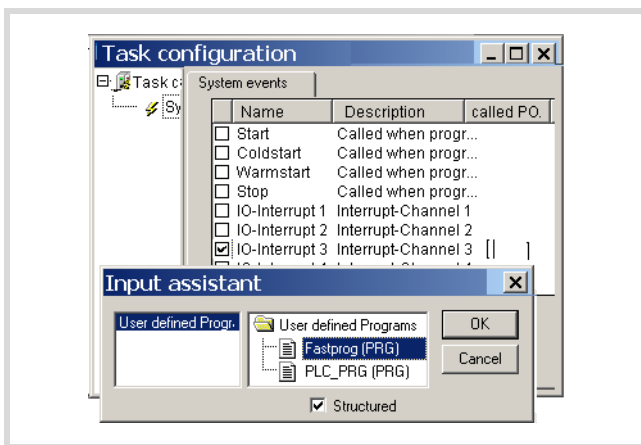


Figure 31: Allocation of Interrupt source → POU

- ▶ Select the "Fastprog" POU and confirm with OK.
- ▶ Save the project. You can now test it.

The variable "b" is incremented by one with every rising edge on input I1.2.

### Timer interrupt

With the timer interrupt a periodically active interrupt is triggered. The periodic duration can be set from 500 – 2500000 microseconds. The timer starts in dependence on a Boolean variable. It interrupts the user program and executes a user-defined application routine.

You have to include the XC121\_UTIL.lib library in your user program to program the "TimerInterruptEnable" function.

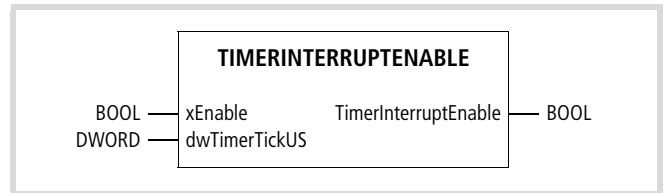


Figure 32: The TimerInterruptEnable function

At input dwTimerTickUS enter the delay time.

The value is accepted with the start of the timer and can not be modified for the run time. If the time falls below 500 or exceeds 2500000, the function returns FALSE and the timer is not started.

Creating application routine time\_Int:

- ▶ Open the "Task Configuration" sub-directory with a double click in the "Resources" directory.
- ▶ Click on the "System events" folder. The "System events" tab is active:

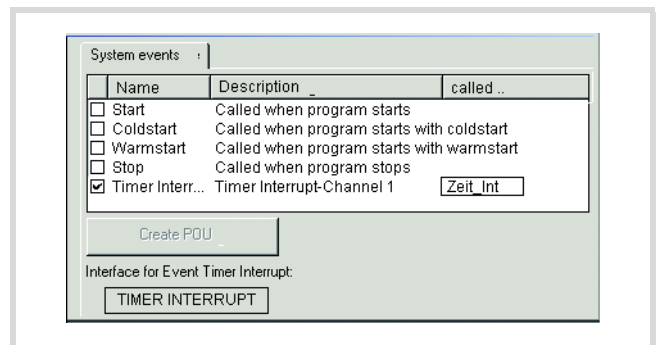


Figure 33: Select the "Timer Interrupt" system event

- ▶ Click the "Timer Interrupt" checkbox to activate the timer interrupt.
- ▶ In the field in the called POU column, enter the name "time\_Int" for the application routine.
- ▶ Click again on the name "Timer Interrupt". Now the "Create POU" button becomes active and indicates the name of the POU.
- ▶ Click on this button. In the "POUs" window a folder (POU) with the name is added.
- ▶ Open the POU and write your user program.

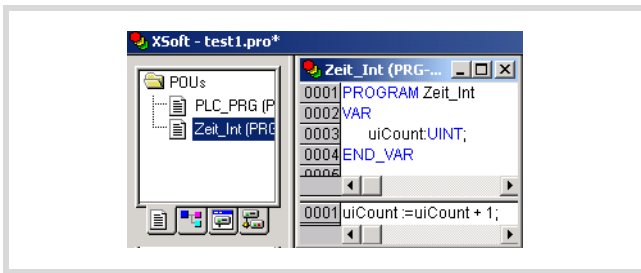


Figure 34: Writing an application routine

The interrupt can be interrupted by higher-priority system interrupts. Cycle time monitoring is active during execution of the timer interrupt.

The Timer interrupt can be inhibited and enabled from the user program. Functions "DisableInterrupt" and "EnableInterrupt" in library "XC121\_UTIL.lib" are available for this purpose.

### Direct I/O access

Via function "Direct IO access" the CPU directly accesses the local inputs and outputs of the XIO-EXT121-1 module. Access is not implemented via the input/output map.

→ Direct access to the data of the XI/OC modules is not supported.

Use functions such as e.g. "ReadBitDirect" of the library "XC121\_Util.lib" for "direct access" to the current IO states and data.

The function "ReadBitDirect" is described as an example for all other functions:

### ReadBitDirect

The bit of an input can be read directly with this function. The state of an input bit is stored in the variables, which point to the parameterized pointer "ptr\_xValue". The pointer variable will not be changed when a fault occurs during processing.

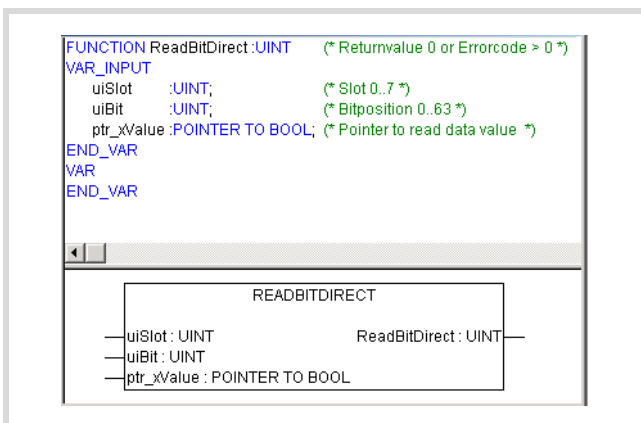


Figure 35: Function READBITDIRECT

### Bit access

Function: ReadBitDirect( uiSlot, uiBit, ptr\_xValue)

IX0.0 – IX0.7: uiSlot = 0, uiBit 0 – 7

IX1.0 – IX1.7: uiSlot = 0, uiBit 8 – 15

IX2.0 – IX2.7: uiSlot = 0, uiBit 16 – 23

Function: WriteBitDirect( uiSlot, uiBit, xValue)

QX0.0 – QX0.7: uiSlot = 0, uiBit 0 – 7

Further functions are:

### Byte access

Function: ReadByteDirect( uiSlot, uiByte, ptr\_byValue)

IB0: uiSlot = 0, uiByte 0

IB1: uiSlot = 0, uiByte 1

IB2: uiSlot = 0, uiByte 2

Function: WriteByteDirect( uiSlot, uiByte, byValue)

QB0: uiSlot = 0, uiByte 0

### WORD access

Function: ReadWordDirect( uiSlot, uiOffset, ptr\_wValue)

IW4: uiSlot = 0, uiOffset 2

IW6: uiSlot = 0, uiOffset 3

IW8: uiSlot = 0, uiOffset 4

IW10: uiSlot = 0, uiOffset 5

IW12: uiSlot = 0, uiOffset 6

IW14: uiSlot = 0, uiOffset 7

Function: WriteWordDirect( uiSlot, uiOffset, wValue)

QW2: uiSlot = 0, uiOffset 1

QW4: uiSlot = 0, uiOffset 2

### “Error code with direct peripheral access”

Verify all functions as far as possible for the validity of the call parameters. Verification is undertaken to determine if the access occurs in dependence on the parameterized signal module and the physical existence of the signal module. If a fault is determined, access is not undertaken and an error code is output. The data fields for the value transfer remain unchanged. The “DisableInterrupt” and “EnableInterrupt” functions do not generate an error code.

The following return values are possible:

Table 10: Error codes with direct peripheral access

|                               |   |
|-------------------------------|---|
| IO_ACCESS_NO_ERROR            | No error  |
| IO_ACCESS_INVALIDE_SLOTNUMBER | Slot = 0 or greater than 15   |
| IO_ACCESS_INVALID_OFFSET      | BitWord offset is too large   |
| IO_ACCESS_DENIED              | Invalid access, e.g. write access to input module, read access to output module or access to non-available address range (offset too large) |
| IO_ACCESS_NO_MODULE           | No module available at the parameterized slot   |
| IO_ACCESS_INVALID_Buffer      | No or incorrect pointer to the output variables   |
| IO_ACCESS_INVALID_Value       | Event is not “0” or “1” with “WriteBitDirect”   |

### Creating and transferring boot project

The CPU processes the user program stored in the main memory. As the main memory is not backed-up, the program is erased during a voltage loss. A boot project must be created in order to save the program retentively. The following steps are required:

- ▶ From the “Online” menu, select “Login”.
- ▶ Select the “Create boot project” command.

The following prompt appears:

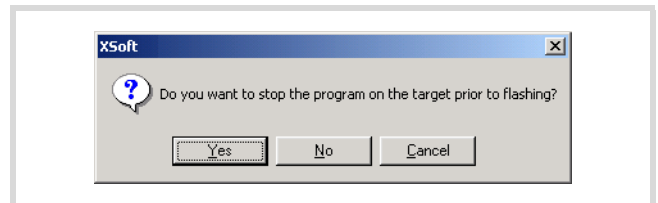


Figure 36: Create bootable project

- ▶ Click Yes.

The following dialog appears briefly:

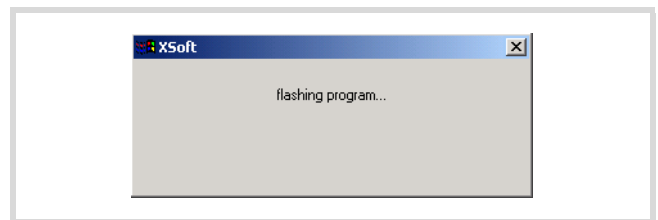


Figure 37: Creating a boot project

The boot project has been created when this dialog disappears again. You can now restart the PLC.

See Section “Switch on behaviour with boot project” on page 23.

### Saving boot project on MMC

The boot project in the system memory (Flash) can also be stored on the MMC. This occurs by using the “copyprojtommc” browser command.

### Erase boot project

The “Remove” browser command deletes the boot project stored in the system memory (Flash) as well as any project stored on the MMC. The boot project on the MMC is only erased with the “removeprojfrommmc” browser command.

### Operating system, download/update

You can replace the XC121 operating system (OS) with a current version, which is always available for download at the Moeller website (<http://www.moeller.net/support>). It is also included on each XSoft CD.

#### Important

When you download the OS, all files saved on the PLC are deleted (the existing boot project as well as the user program).

You have two options to transfer the OS:

- Directly from the PC to the PLC
- From the PC to the MMC.

### Transferring the operating system from the PC into the PLC

- ▶ Open an XSoft project and activate under Resources PLC Configuration and select the Other Parameters tab. → figure 5
- ▶ Click on the "Start" button.

The "Download operating system" dialog opens.

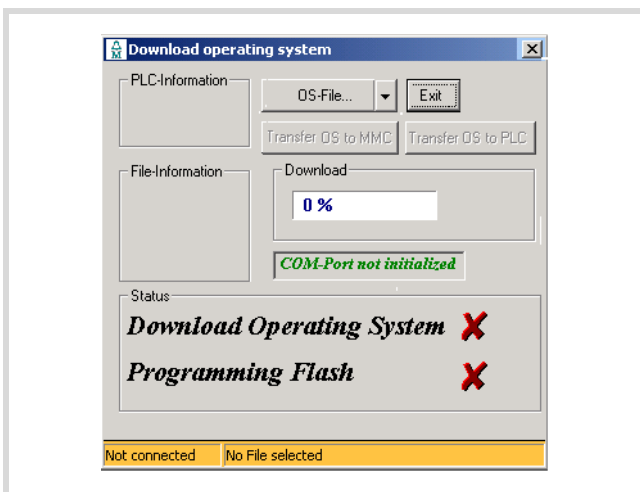


Figure 38: Download operating system

The system reports that the COM port is not initialized.

- ▶ Click the "OS-File" button and select the required operating system file (\*.hex).

→ The files last opened can be selected from the list field (dropdown menu).

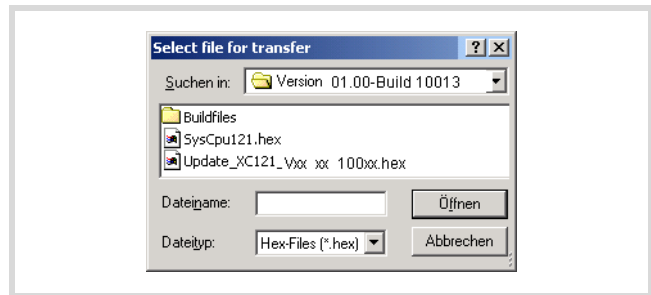


Figure 39: Operating system file selection

The target type and file version are displayed.

- ▶ Press the "Transfer OS to MMC" button.

The transfer begins. Programming of the flash EPROMs takes about 20 to 30 seconds.

→ Do not switch off the supply voltage during the transfer or while the warning symbol appears with "Programming Flash"!

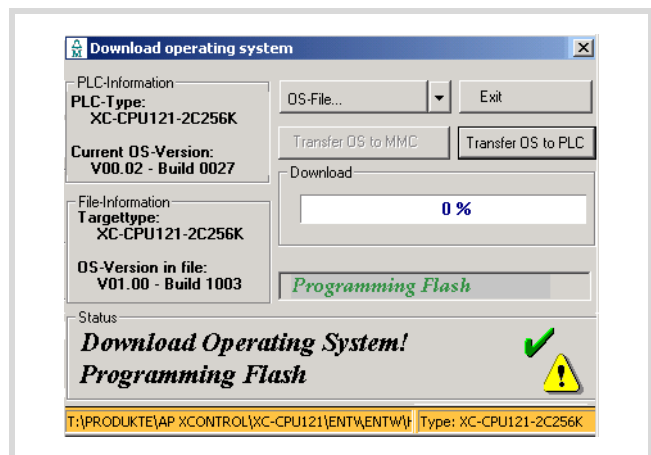


Figure 40: Warning during download

Wait for the following dialog.

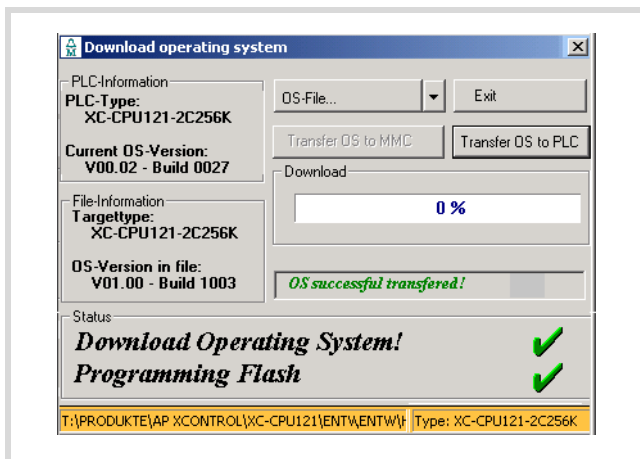


Figure 41: OS successfully transferred to the PLC

- ▶ Click in this window on the "Exit" button.

### Transferring the OS from the PC into the MMC

If an OS is loaded into the MMC, the OS and the boot project on the MMC and the user program in the PLC are deleted. The procedure is similar to the description in Section "Transferring the operating system from the PC into the PLC". Click in this case on the "Transfer OS to MMC" button, → figure 38 on page 32.

### Transferring the OS from the MMC into the PLC

- ▶ Insert the MMC into the PLC when it is switched off.
- ▶ Switch on the PLC.

The OS of the PLC is updated during the switch on process and a boot project is loaded into the PLC. The transfer can take more than 30 seconds as the CPU must be booted several times.

- Do not interrupt the process, e.g. by switching off the supply voltage!

### User program source code

It is possible, to save the source code of the user program on the MMC.



## 7 Browser commands

The PLC browser is a text based control monitor. Here the commands to query certain information from the control are entered in the input line and sent as a string to the control. The response string is indicated by a result window of the browser. This functionality can be used for diagnostics and debugging.

→ The browser commands can only be used online.

To run these commands:

- ▶ Under Resources in the object organiser, double-click PLC Browser.

A new window, PLC-Browser appears in the workspace.

- ▶ Click .

The selection field lists the available browser commands.

- ▶ Double-click the required command to select it.

The selected command now appears in the "PLC Browser" window.

- ▶ Press the enter button in order to view the response of the PLC to the browser command in the event window.

→ More detailed information concerning the selected browser command can be found if you place a "?" before the selected browser command followed by a space, and press the Enter (Return) key.

The description of the available commands can be found in the XSoft manual (h1437gb.pdf) in the section «Resources→ PLC Browser».

The XC-CPU121 supports the browser commands from Table 11.

Table 11: Browser commands

|                   |   |
|-------------------|---|
| ?                 | Get a list of implemented commands.                           |
| reflect*          | Mirror current command line for test purposes.                |
| mem               | memory dump, Syntax: mem <start-addr> <end-addr>              |
| memc              | As mem, addresses + start address of the code range           |
| memd              | As mem, addresses + start address of the data range           |
| pinf              | Output project information                                    |
| ppt               | Output module pointer table                                   |
| dpt               | Output data pointer table                                     |
| pid               | Output project ID   |
| cycle             | Output cycle time   |
| canload*          | Display traffic loading of the local CAN busses (CAN1 + CAN2) |
| copyprojtoMMC     | Copy the current boot project on the MMC                      |
| createstartupini  | Generation of the initialisation file on the MMC              |
| format            | Format the MMC memory card                                    |
| GetNodeId         | Display of the CANopen Node-IDs of both CAN interfaces        |
| GetRoutingId      | Display of the routing Node-ID and the routing interface      |
| metrics           | Output PLC information  |
| reload            | Load boot project from FLASH to the PLC                       |
| remove            | Erase boot project in the FLASH                               |
| removeprojfromMMC | Delete the boot project from the MMC                          |
| removestartupini  | Delete the initialisation file from the MMC                   |
| getswitchpos      | Output switch position  |
| getrtc            | Read real-time clock  |
| setrtc*           | Set real-time clock   |

Further information concerning the commands marked with \* can be found in the following pages.

**reflect**

Reflects the command line, for testing communications between browser and PLC.

This command is not transmitted to the PLC!

Example:

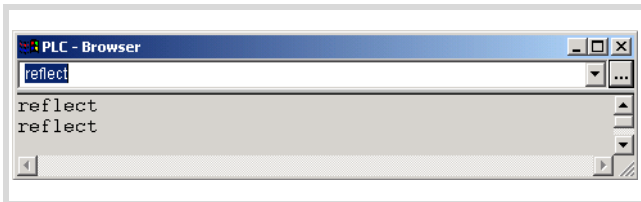


Figure 42: Browser command "reflect"

**canload**

Displays the utilization of the CANopen field bus.

Example:

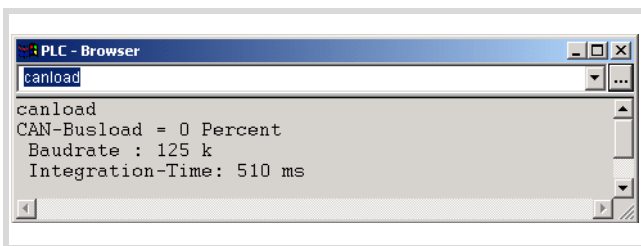


Figure 43: Browser command "canload"

This browser command returns, for example, the following information:

- CAN busload = 0 Percent
- Baud rate 125 Kbit/s
- Integration Time: 510 ms.

**Important**  
 With a bus utilization of 75 percent or higher, the warning ATTENTION: HIGH BUSLOAD also appears. Overload of the local CAN bus in conjunction with further short term load peaks can lead to CAN data loss.

→ In addition to the browser command, function CAN\_BUSLOAD can be used to determine the CAN bus utilization from the user program, see Section "Function CAN\_BUSLOAD" on page 38.

**setrtc**

Sets or changes the PLC date and/or time.

Syntax:

```
<setrtc_YY:MM:DD:DW_HH:MM:SS>
```

Legend:

- \_ Space
- YY The last two digits of the year (00 F YY F 99)
- MM Month (01 F YY F 12)
- DD Day (01 F DD F 31)
- DW Weekday (01 F DW F 07; 01 = Monday, 07 = Sunday)
- HH Hour (00 F HH F 23)
- MM Minute (00 F MM F 59)
- SS Second (00 F SS F 59)

## 8 Libraries, function blocks and functions

The libraries contain IEC function blocks and functions that you can use, for example, for the following tasks:

- Data exchange through the CANopen bus
- Controlling the real-time clock
- Determining bus load of the CANopen bus
- Triggering interrupts
- Sending/receiving data through the interfaces

The libraries are located in the folders:

- Lib\_Common for all PLCs
- Lib\_CPU121 for the PLC XC121.

### Using libraries

When you open a project, libraries Standard.lib and SYSLIBCALLBACK.lib are copied in to the Library Manager. If you need further libraries for your application, you have to install these manually.

The libraries in the Library Manager are assigned to the project after saving. When you open the project, the libraries are then automatically called up as well.

The following overview lists the documents in which the function blocks and functions are described.

| Document  | Library   |
|---|---|
| AWB 2700-1437   | Standard.lib<br>Util.lib<br>XC121_Util. Lib     |
| Online help or PDF files can be found in the directory "XsoftV2.3.4\Manuals\English\Software\XsoftProfessional\XsoftSyslibs\pdf". | SysLib... . lib                                 |
| AWB 2786-1456   | XS40_MoellerFB. Lib/ Visu. Lib/...              |
| AN2700K20   | 3S_CANopenDevice. Lib<br>3S_CANopenManager. Lib |
| AN2700K19   | 3S_CANopenNetVar. Lib                           |
| AN2700K27   | SysLibCan. Lib                                  |
| AWB 2786-1554   | CANUserLib. Lib<br>CANUser_Master. Lib          |

### Installing additional system libraries

You can install libraries manually as follows:

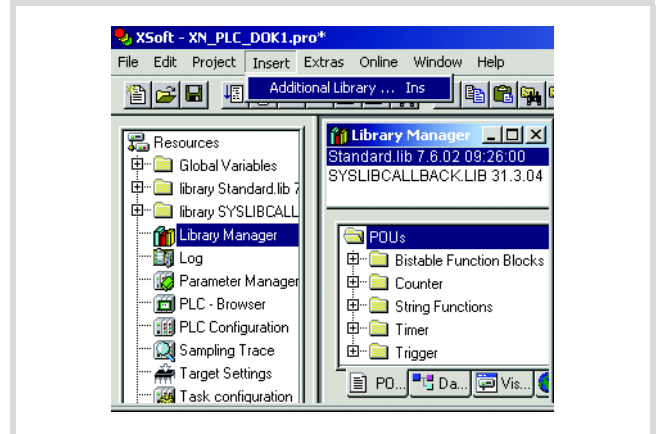


Figure 44: Libraries, installing manually

- ▶ In your project, click the "Resources" tab in the object organiser.
- ▶ Double-click the "Library Manager" element.
- ▶ From the Insert menu, select Additional Library....

The Open dialog appears.

- ▶ Select the library to install and click Open.

The library now appears in the Library Manager.

**XC121 specific functions**

**“Library XC121\_Util.lib”**

This library contains the functions shown in the illustration below:

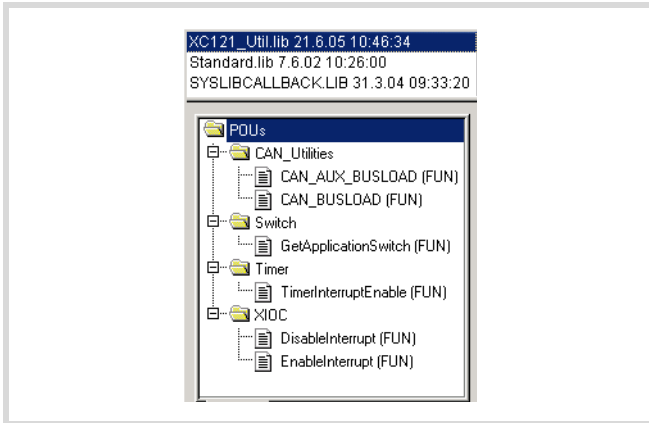


Figure 45:Functions of the library XC121\_Util.lib

→ The Timer functions are described in Section “Timer interrupt” on page 29.

**Function CAN\_BUSLOAD**

This function can be called cyclically in a user program. If a read cycle has been completed successfully, the function returns TRUE and writes the determined integration time and the bus utilization values to the passed addresses.

If the bus load calculation is not yet completed or the CAN controller has not yet been initialized, the function returns FALSE.

For information about evaluating the returned value, see „canload” on page 36.

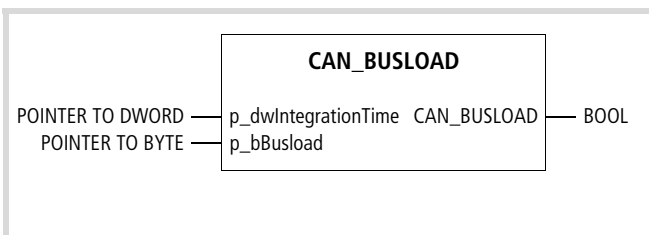


Figure 46:Function CAN\_BUSLOAD

**Function GETAPPLICATIONSWITCH**

With this function you can query the position of the application switch. After an H signal at input “xEnable”, the value to which the switch is set, is displayed.

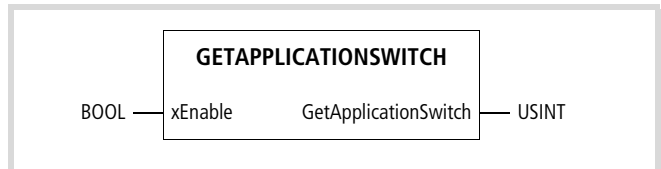


Figure 47:Function GETAPPLICATIONSWITCH

## 9 Connection set-up PC – XC121

To establish a connection between PC and XC121, the two devices' communication parameters must be the same.

To match them, first adjust the PC's communication settings to the CPU's settings. Use the CPU's default parameters, transferring them as shown in figure 48.

→ If you get an error message, the CPU's default settings have already been changed. In that case try a baud rate of 57600.

You can then change the CPU's parameters (→ figure 49), always making sure that you have the same settings on the PC.

### Communication settings of the PC

You can use either the COM1 or the COM2 port of the PC. Define the communication parameters of the interface in the XSoft.

- ▶ In the Online menu, select Communication Parameters
- ▶ Specify the port (COM1 or COM2), → section "Changing settings"
- ▶ Use the remaining settings as shown in figure 48.
- ▶ Confirm the settings with OK.
- ▶ Log on to the PLC.

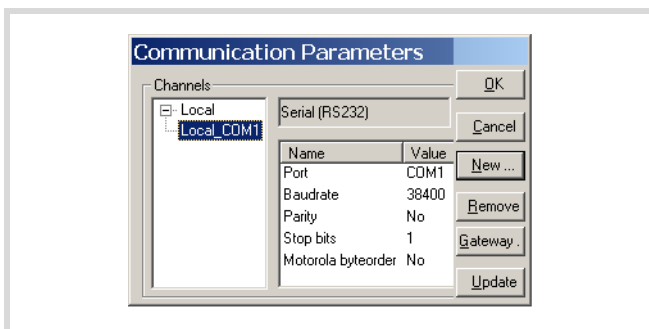


Figure 48: Defining the PC's communication settings

### Changing settings

To change settings such as the baud rate or the port, do the following:

- ▶ Double-click the displayed value, for example 38400. The field becomes grey.
- ▶ Enter the desired value.

You can double-click the field again to select the required baud rate, e.g. 57600 bit/s.

### Communication settings (baud rate) of the CPU

- ▶ In the Resources tab, select PLC Configuration.
- ▶ In the "PLC Configuration" dialog, click the "Other Parameters" tab.
- ▶ In the "Baudrate" list field, select the baud rate (for example 57600 bit/s as shown in figure 49).

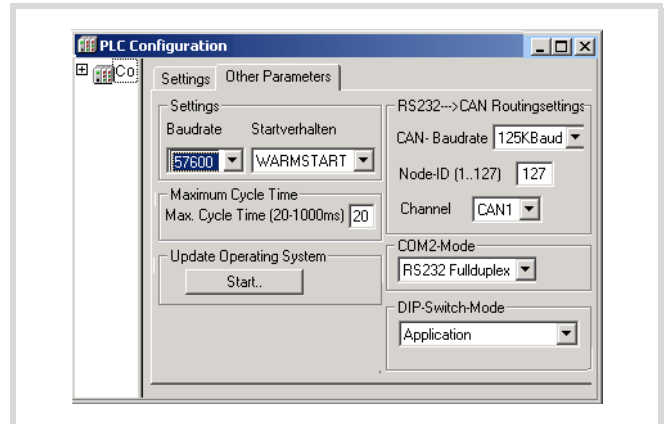


Figure 49: Specifying the CPU's communication settings

- ▶ Log on to the PLC.

The following prompt appears:

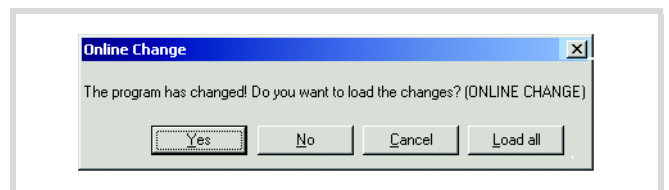


Figure 50: Query concerning program change

- ▶ Click Yes.

The program is loaded. After a short while, a communication error message appears, since the baud rate settings of PC and CPU are no longer the same:

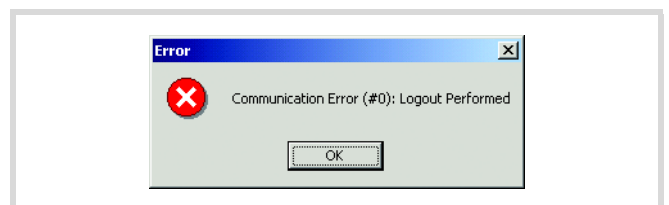


Figure 51: Communications fault

- ▶ Acknowledge the message with OK.

To reestablish communications, change the PC's baud rate again.



## 10 Set the system parameters via the STARTUP.INI file

### Parameter overview

You can define the system parameters with the INI file and save them on the MMC. The PLC accepts the parameters during start up. The INI file is always generated with all control-specific entries (→ table 12). Parameters which you have specified in the PLC configuration do not receive an entry.

Table 12: Parameters in the INI file

```
Entries
COM_BAUDRATE
CAN1_BAUDRATE
CAN1_NODEID
CAN2_BAUDRATE
CAN2_NODEID
CAN_ROUTINGID
CAN_ROUTING_CHANNEL
```

### Structure of the INI file

An INI file is a text file with a defined data format. From a section defined with a name (in square brackets), e.g. such as [STARTUP] the system parameters are listed followed by an equals sign and their value. Close the line by pressing RETURN.

```
COM_BAUDRATE=38400 Return
```

Lines commencing with a semicolon are interpreted by the PLC as comments and are ignored:

```
; CAN_ROUTING_CHANNEL=CAN1
```

Example: Default STARTUP.INI for XC121

```
[STARTUP]
TARGET=XC-CPU121-2C256K
COM_Baudrate=38400
CAN1_Baudrate=125
CAN1_NODEID=2
CAN2_Baudrate=125
CAN2_NODEID=2
CAN_ROUTINGID=127
CAN_ROUTING_CHANNEL=CAN1
```

### INI file generation

The INI file is generated with the browser command "createstartupini" on the MMC card. The current parameter values are applied to the system parameters. The programming interface is loaded with the following parameter:

```
COM_BAUDRATE=38400
```

The parameters from the program PLC configuration have no default values, e.g.:

```
CAN_ROUTINGID=
CAN_ROUTING_CHANNEL=
CAN1_BAUDRATE=
CAN1_NODEID=
```

A file which already exists cannot be changed or overwritten by the browser command "createstartupini". If you still enter the command, a warning appears. In order to create a new file the existing file must be deleted first.

→ section "Delete INI file" on page 42.

The parameters can be changed with a text editor if you insert the MMC into the MMC slot of a PC. The STARTUP.INI file is located in the "MOELLER/XC-CPU121-2C256K/BOOTPRJ/" directory. During online operation you can execute the "Load file from PLC" and "Write file in PLC" commands in the "Online" menu.

---

### Entries of the INI file

- Entry for specification of the target system.

```
TARGET=XC-CPU121-2C256K
```

- Parameters for programming via the serial communication with XSoft:

```
COM_BAUDRATE=38400
```

The parameters from the INI file have priority before the parameters of the PLC configuration. After a program download or after loading of the boot project the parameters from the PLC configuration are not used:

- Parameters of the PLC configuration for CAN Level 2Route:

```
CAN_ROUTINGID=127
CAN_ROUTING_CHANNEL=CAN1
CAN1_BAUDRATE=125
CAN1_NODEID=
CAN2_BAUDRATE=
CAN2_NODEID=
```

---

### Start behaviour of the XC121 with inserted MMC containing INI file

During switch-on of the XC121 the data of the INI-file are transferred to the XC121. They remain active after a new program is loaded until you run the browser command "removestartupini".

---

### Changing settings

If you insert an MMC with an INI file into the XC121 and switch on the XC121, the XC121 accepts the parameters of the INI file on the MMC. No parameters from the loaded project are accepted.

The parameters are retained until the browser command "removestartupini" has been entered and the XC121 is then switched on or off. The XC121 now operates with the parameters of the project.

---

### Delete INI file

There are two browser commands available to access the MMC.

- removestartupini: Deletes the INI file on the MMC and the data on the CPU. The data from the project is accepted next time the device is switched on.
- removeprojfrommmc: deletes the project and the INI file on the MMC. The data on the CPU is retained.

---

→ If a full reset command is executed in online mode in the PLC Configuration, the OS and the project on the MMC are deleted. The INI file is retained, → section "Memory card MCC" on page 11

## 11 Programming via CANopen network (Routing)

Routing means to establish an online connection from a programming device (PC) to any (routing-capable) PLC in a CAN network without having to directly connect the programming device to the target PLC. The target can instead be connected to any other PLC in the network. All actions that are available through a direct PC–PLC connection can also be implemented through the routing connection:

- Program download
- Online modifications
- Program test (Debugging)
- Generation of boot projects
- Writing files in the PLC
- Reading files from the PLC

Routing offers an advantage which makes it possible to access all routing capable PLCs on the CAN bus from any PLC which is connected with the programming device. You select the control with which you want to communicate by the project selection. This provides an easy way of controlling remote PLCs.

However, the data transfer from routing connections is significantly slower than with direct (serial or TCP/IP) connections. This results, for example, in slower display refresh rates of variables and longer download times.

### Prerequisites

The following prerequisites must be fulfilled to use routing:

- The routing PLC and the target PLC must both support routing.
- Both PLCs must be connected via the CAN bus.
- The PLCs must both have the same active CAN baud rate.
- The valid routing Node ID must be set on both PLCs.

### Routing properties of the XC121

The XC121 supports the routing via the CAN bus.

Routing can be implemented without prior download of a user program (default: CAN1, 125 kBaud, Node-Id 127). The target PLC must not be configured as a CAN Master or CAN Device for this purpose.

You can for example load a program from the PC via a PLC of the XC device series into the XC121. Assign a Routing Node-Id to the XC121 (target PLC) in this case.

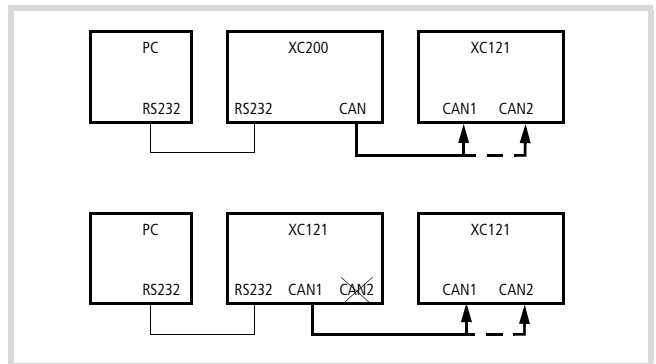


Figure 52: Program download per Routing

You can optionally connect the target PLC via the interfaces CAN1 or CAN2.

→ If you use the XC121 as the routing PLC you may only use the CAN1 channel for the connection leading further.

You can choose between two different methods to set the routing Node-Id:

- Setting via the PLC Configurator
- Setting via the APPLICATION switch

### Setting via the PLC Configurator

- ▶ Click on the "Other Parameters" tab in the "Configurator" folder (→ figure 5).

"Node-Id-Routing" may not be selected in the DIP-Switch-Mode field!

- ▶ Enter the CAN baud rate in the "RS232->CAN Routing settings" field, as well as the Node-Id and the Channel CAN1 or CAN2.

### Setting via the APPLICATION switch

- ▶ Click on the "Other Parameters" tab in the "PLC Configurator" tab (→ figure 5 on page 10).
- ▶ Set the "Node-Id Routing" mode in the "DIP-Switch-Mode" field.
- ▶ Set the baud rate in the "RS232->CAN Routing" field. The following baud rates are possible: 50, 100, 125, 250 and 500 kBaud (default: 125 kBaud)
- ▶ Set the Routing-Node-Id and the CAN interface on the APPLICATION switch:

Switch 1 – 7: (Routing-) Node ID 1 – 127; with invalid address 0 the default node Id 127 is used.

Switch 8: OFF = CAN1, ON = CAN2

→ The settings on the APPLICATION switch have priority over the configurator setting.

For further information chapter "APPLICATION switch (S2)" on page 10.

### Routing through XC200

To perform a program transfer or routing using TCP/IP through a connection between XC200 and PC, you must first set the block size for the transferred data. The packet size (4 KByte or 128 KByte) depends on the transfer type (program transfer or routing) and the operating system, → table 13.

Table 13: Block size for data transfer

|                               | Program/file transfer |                | Routing              |                |
|-------------------------------|-----------------------|----------------|----------------------|----------------|
|                               | BTS < V1.03.03        | BTS f V1.03.03 | BTS < V1.03.03       | BTS f V1.03.03 |
| Block size Default: 128 Kbyte | 128 Kbyte             | 4/128 kByte    | Routing not possible | 4 Kbyte        |

**Important**  
 The program download with a block size of 4 Kbyte to a PLC with an operating system version earlier than V1.03.03 will cause faulty behaviour!  
 If a program download is performed, the progress bar on the programming device monitor will only change erratically (about every 10 seconds).

The block size can be changed only directly in the Windows Registry.

→ You can change this setting only if you have administrator rights on your PC.

### Changing the block size

- ▶ Close all XSoft applications.
- ▶ Close the CoDeSys gateway server.

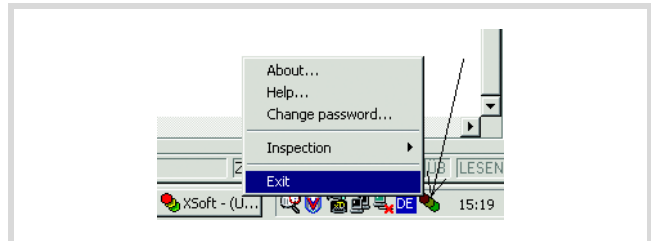


Figure 53: Closing the CoDeSys gateway server

- ▶ Change the block size to the required value.

The XSoft installation folder contains the following \*.reg files for entering the block size in the Windows Registry:

|                      |  |
|----------------------|--|
| BlockSizeDefault.reg | Enters a block size of 20000 <sub>hex</sub> = 128 KByte (default value) in the Registry. |
| BlockSizeRout.reg    | Enters a block size of 1000 <sub>hex</sub> = 4 Kbyte in the Registry.                    |

Alternatively, you can use the BlockSizeEditor application to change the block size.

The download block size is defined in the following Registry key:

```
[HKEY_LOCAL_MACHINE\SOFTWARE\3S-Smart Software Solutions GmbH\Gateway Server\Drivers\Standard\Settings\Tcp/Ip (Level 2 Route)]
"Blocksize"=dword:00020000
```

The default block size is 20000<sub>hex</sub> (= 128 KByte), the block size for routing is 1000<sub>hex</sub> (= 4 KByte).

### Notes

- If large files are written to the target PLC or read from the PLC, it is possible that the online connection will be interrupted after the transfer process has been completed. Renewed connection is possible.
- If a program with a modified routing node ID is loaded into the target PLC, the target PLC accepts the modified routing node ID; however, the communication connection will be interrupted. Reconnection with a corrected routing Node ID is possible.
- If a PLC receives a program without valid routing parameters (Baud rate / Node ID), this PLC cannot be connected via a routing connection.
- The routing is independent of the configuration (master/slave): a target PLC that has not been configured as a master or as a slave can be accessed. It must only receive the basic parameters such as Node ID and baud rate, as well as a simple program.

### Addressing

PLCs on the CANopen bus can be configured as a master or as a device. The PLCs are assigned with a node ID/node number (address) in order to uniquely identify them. To use the routing function to access a target PLC, you must assign a further node ID to the PLC.

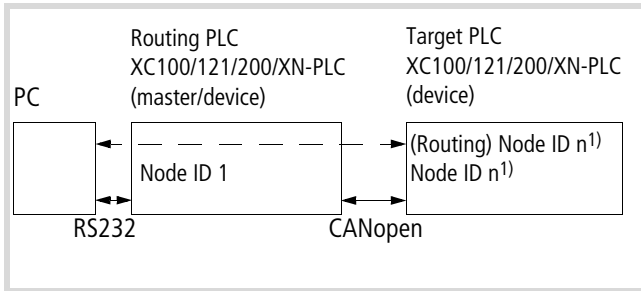


Figure 54:XC100/121/200, XN-PLC on the CANopen bus, routing principle

- 1) The following applies for the Node ID of the device function and the Node ID of the routing function:
  - The (Routing-) Node-ID must **not be equal to** the Node-ID (Device)!

### Procedure

- ▶ Connect the PC to the routing PLC.
- ▶ Select the target PLC with which you want to communicate for the project.
- ▶ First of all determine the communication parameters for the connection between the PC and the PLC which is connected to the PC.
- ▶ Enter the target PLC's target ID (target ID = node ID!) as shown in the example and log on.

You can run the following functions:

- Program download
- Online modification
- Program test (Debugging)
- Create bootable project
- Filing source code.

Note for project creation:

The Node ID/Node number and the baud rate of the target PLC to the routing function can be defined in the → figure 55 "Additional parameters" window in the PLC Configuration:

- ▶ Enter the baud rate on the CANopen bus and the Node-ID/node number in the "RS 232 → CAN routing settings" field.

Node ID and baud rate are transferred with the project download.

→ To guarantee a fast data transfer, the routing should be performed only with a CANopen baud rate of at least 125 Kbit/s.

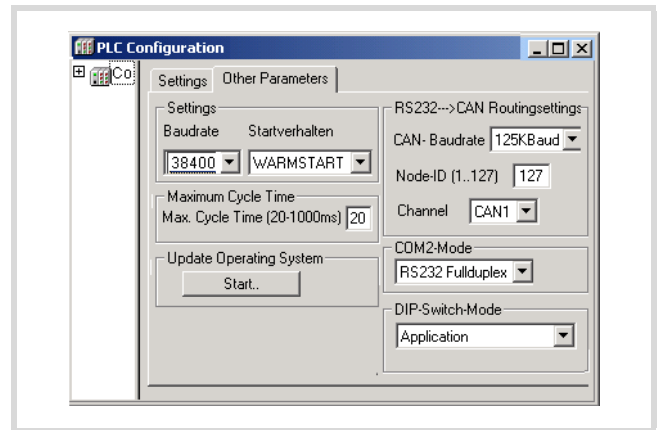


Figure 55: CANopen routing settings

The following illustrations indicate – independently of the routing settings – where the baud rate and the Node ID of the PLCs which have been configured as masters or devices are to be entered. The settings are to be made in the master PLC in the "CAN Parameters" tab or with the device PLC in the "CAN Settings" tab.

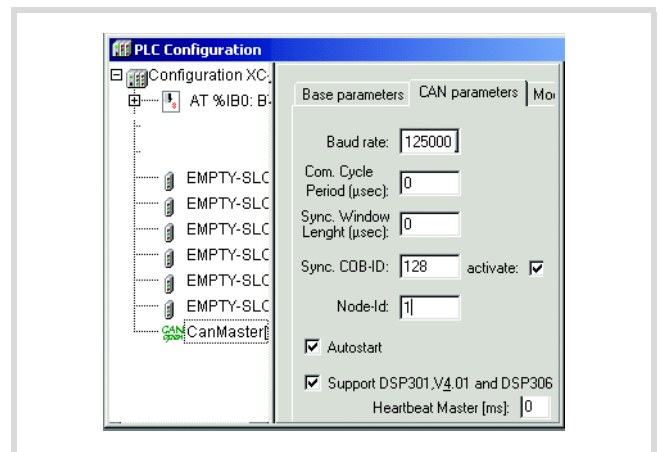


Figure 56: CAN master parameters

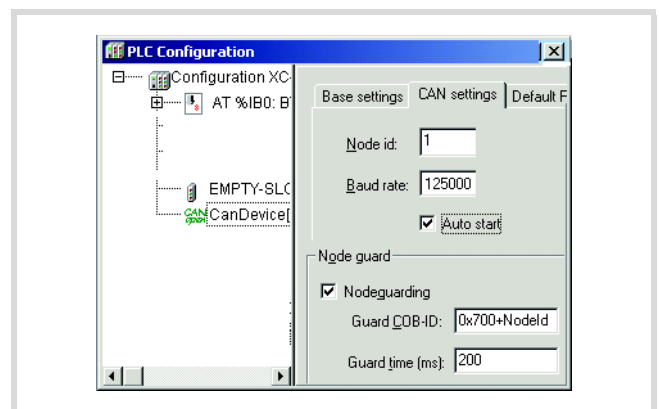


Figure 57: CAN device parameters

**Example**

The example below illustrates the procedure for accessing a PLC program.

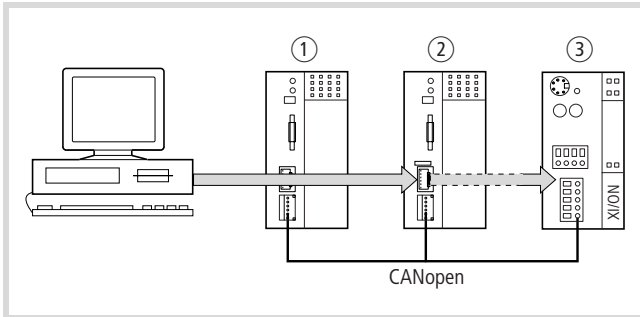


Figure 58: Diagnostics possibilities

- ① XC100 with node ID 1
- ② XC200 with node ID 2
- ③ XN-PLC with Node-ID 3 (instead of the XN-PLC you can also use an XC121)

You have connected the PC to the PLC with node ID "2" and want to access the target PLC with node ID "3".

- ▶ Open the project of the target PLC (Node ID 3) whose program you wish to edit or test.
- ▶ First configure the parameters for the hardware connection PC ↔ PLC (node ID 2).
- ▶ From the Online menu select Communication Parameters....
- ▶ Click the New button under "local" channels.

The "New Channel" window appears.

- ▶ Select the channel in the Device field.  
XC200: Serial [RS232] [Level 2 Route] or TCP/IP [Level 2 Route].
- ▶ In the Name field you can assign a new name, e.g. "Rout\_232".
- ▶ Confirm with OK and return to the original window.

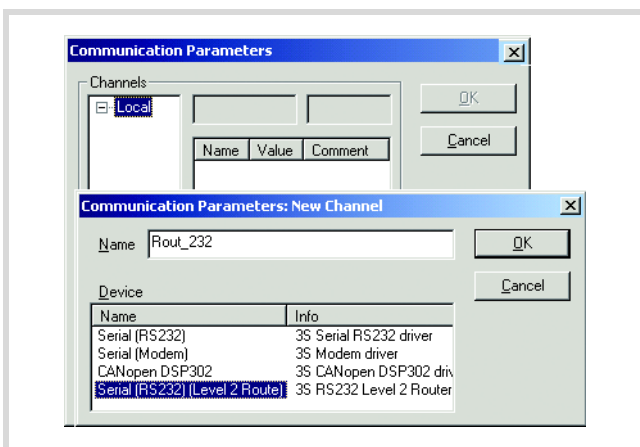


Figure 59: Channel parameter setting

You have now determined the parameters for the hardware connection between the PC and the PLC (node ID 2).

- ▶ Call up the communications parameters in the "Online" menu once again and select the control which you want to program/test.
- ▶ Enter the number 3 as the target ID in the example. The target ID is identical to the Node ID!  
Click in the field on the "Value" column on the right beside the target ID term in order to enter the target ID. Enter the figure 3 and confirm with OK.
- ▶ Log on and carry out the action.

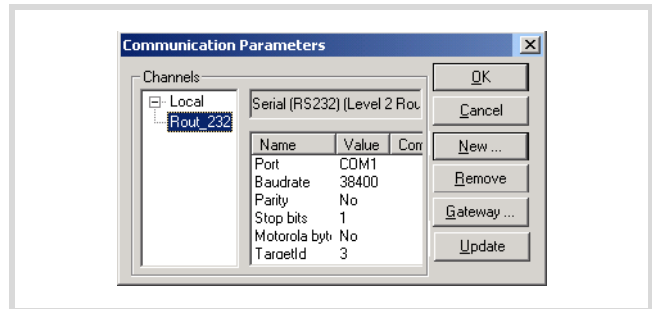


Figure 60: Setting the target ID of the target PLC

**PLC combinations for routing**

The following PLC support routing:

| From P         | XC100 | XC121 | XC200 | XN-PLC-CANopen |
|----------------|-------|-------|-------|----------------|
| To O           |       |       |       |                |
| XC100          | ×     | ×     | ×     | ×              |
| XC121          | ×     | ×     | ×     | ×              |
| XC200          | ×     | ×     | ×     | ×              |
| XN-PLC-CANopen | ×     | ×     | ×     | ×              |

## 12 RS232 interface in transparent mode

In transparent mode the data transfer occurs between the XC121 and data terminals (e.g. terminals, printers, PCs, measurement devices) without interpretation of the data. Switch the RS 232 serial interface of the XC121 (COM1) into transparent mode with the user program.

For running the transparent mode there are functions available for opening and closing the interface, for sending and receiving data and for setting the interface parameters.

→ Because the interface's control lines are not active, you can not use the SysComReadControl and SysComWriteControl functions.

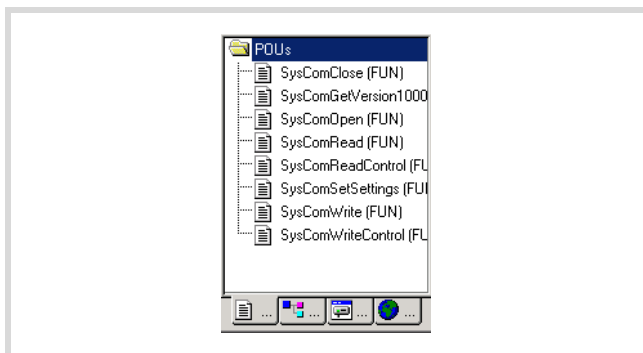


Figure 61: Function summary

The transparent mode functions are contained in the "XC121\_SysLibCom.lib" library. The library must therefore be included in the Library Manager. You can find the descriptions of the function blocks in the manual "Function Blocks for XSoft" (AWB2786-1452GB).

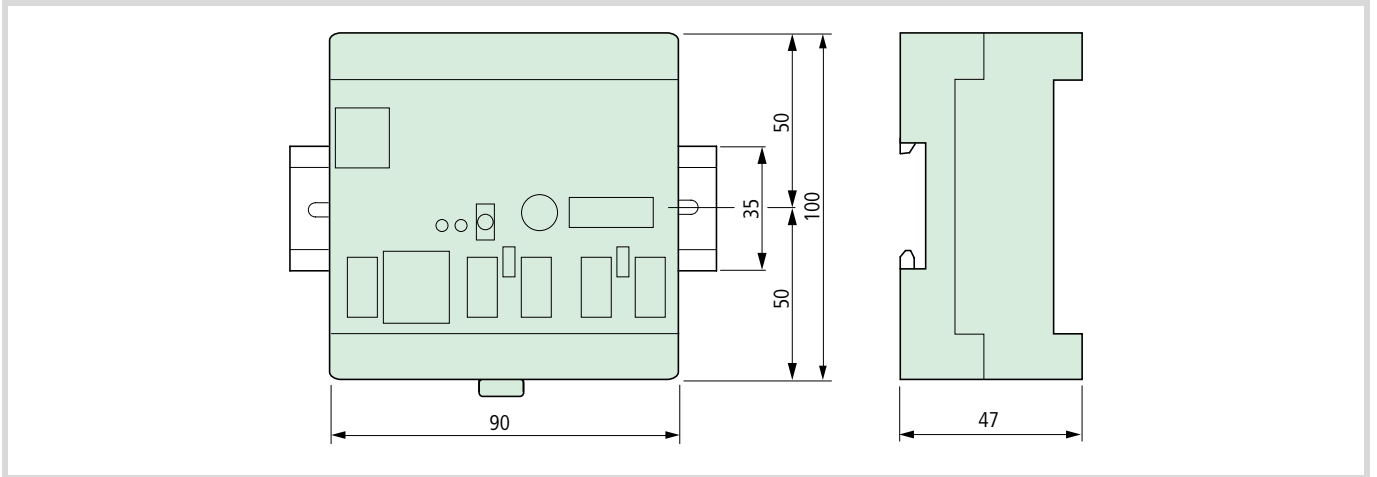
→ If the RS232 interface (COM1) of the XC121 is in transparent mode, programming via this interface is not possible. Transparent mode must first be disabled. When transparent mode is closed, the original communication parameters are reinitialised. The transparent mode is forcibly deactivated when the PLC state changes to the STOP mode or when the "SysComClose" function is accessed.



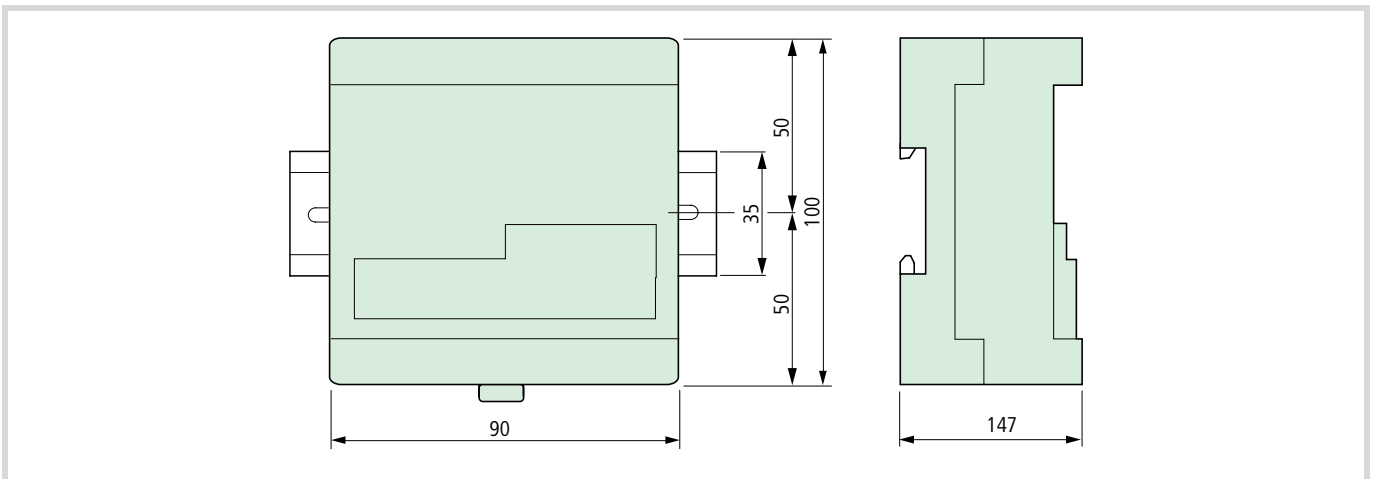
# Appendix

## Dimensions

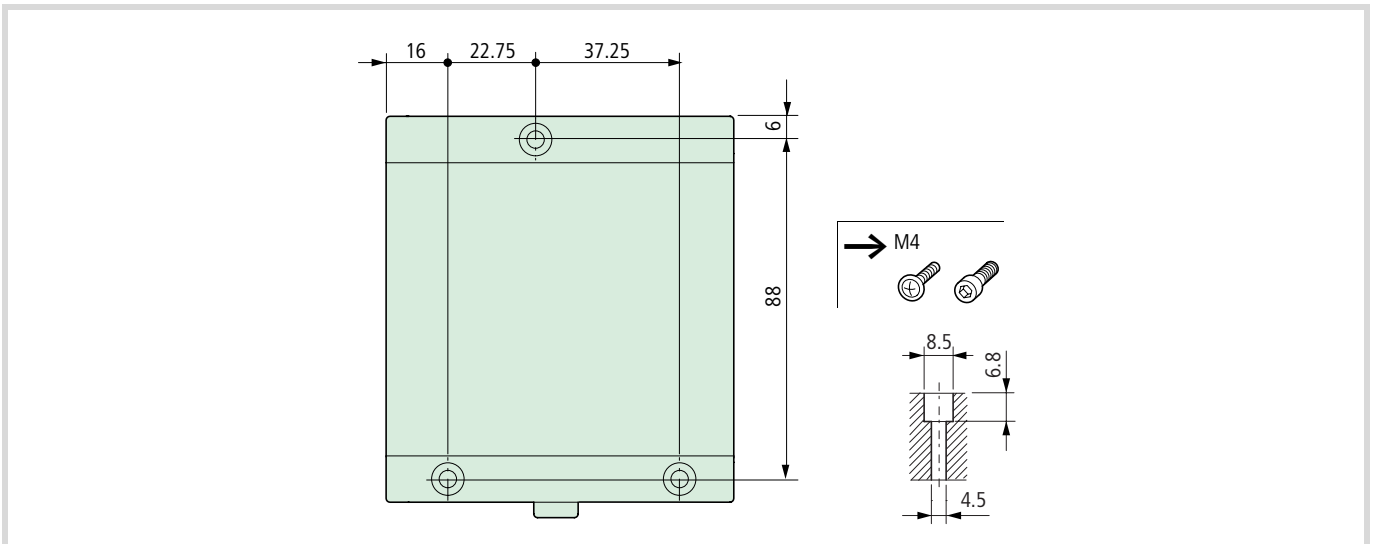
### XC-CPU121



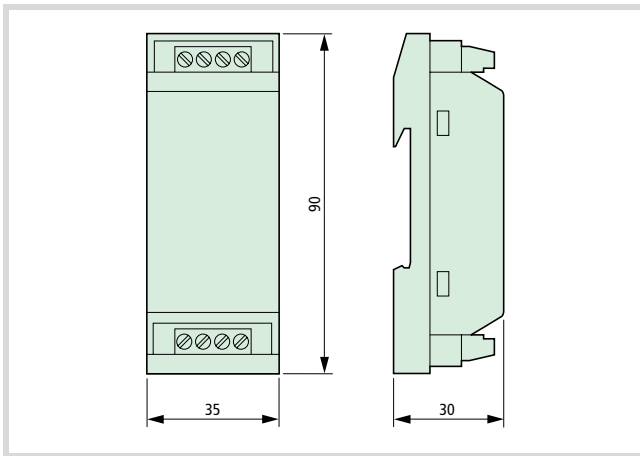
### XIO-EXT121-1



### XC-CPU121, XIO-EXT121-1



## 24 V DC line filter XT-FIL-1



## Technical data

## XC-CPU121/XIO-EXT121-1

| General  |     |   |
|--|-----|---|
| Standards and regulations                              |     | IEC/EN 61131-2<br>EN 50178  |
| Ambient temperature                                    | °C  | 0 to +55  |
| Storage  | °C  | -25 to +70  |
| Mounting position                                      |     | Horizontal  |
| Relative humidity, no condensation (IEC/EN 60068-2-30) | %   | 10 to 95  |
| Air pressure (in operation)                            | hPa | 795 to 1080   |
| Vibration resistance                                   |     | Frequency 5 to 9 Hz; amplitude 3.5 mm<br>9 to 150 Hz; 1.0 g constant acceleration |
| Mechanical shock resistance                            |     | 15 g/11 ms  |
| Overvoltage category                                   |     | II  |
| Pollution degree                                       |     | 2   |
| Enclosure protection                                   |     | IP 20   |
| Rated insulation voltage                               | V   | 500   |
| Interference emission (Industry)                       |     | EN 61000-6-4  |
| Interference immunity (Industry)                       |     | EN 61000-6-2  |
| Back-up of the clock                                   |     | at least 72 hours   |
| Weight   | kg  | 0,15  |
| Dimensions (W x H x D)                                 | mm  | 90 × 100 x 47   |

| <b>Electromagnetic Compatibility (EMC)</b>             |     |     |
|--|-----|-----|
| Electrostatic discharge IEC/EN 61000-4-2, Level 3, ESD |     |     |
| Contact discharge                                      | kV  | 4   |
| Air discharge  | kV  | 8   |
| Radiated (IEC/EN 61 000-4-3, RFI)                      | V/m | 10  |
| Burst Impulse (IEC/EN 61000-4-4, Level 3)              |     |     |
| Supply cables  | kV  | 2   |
| Signal cables  | kV  | 1   |
| Surge (IEC/EN 61 000-4-5)                              | kV  | 0,5 |
| Conducted (IEC/EN 61 000-4-6)                          | V   | 10  |

**XC-CPU121**

| <b>Connection by</b>  |      |   |
|---|------|---|
| Supply voltage  |      |   |
| Plug  |      | Springloaded terminal block, 4-pole                       |
| Terminal capacity   |      | 0.14 mm <sup>2</sup> to 1 mm <sup>2</sup> (AWG28-18)      |
| COM1 interface  |      |   |
| Plug  |      | RJ45  |
| COM2, CAN1, CAN2 interfaces                                 |      |   |
| Plug  |      | Springloaded terminal block, 6-pole                       |
| Terminal capacity   |      | 0.14 mm <sup>2</sup> to 0,5 mm <sup>2</sup> (AWG28-20)    |
| <b>Supply voltage (24 V/0 V)</b>                            |      |   |
| Input voltage   | V DC | 24  |
| Permissible range   | V DC | 20.4 to 28.8  |
| Power consumption   | W    | Up to 1.44  |
| Input current   | mA   | 60  |
| Residual hum and ripple                                     | %    | ≦ 5   |
| Maximum power dissipated P <sub>v</sub> (without local I/O) | W    | 6   |
| Oversvoltage protection                                     |      | Yes   |
| Polarity protection   |      | Yes   |
| Switch-on current surge                                     |      | Not limited, (limiting only by a supply-side 24 V DC PSU) |
| Hold-up time on supply drop-out                             |      |   |
| Dropout duration  | ms   | 10  |
| Repeat rate   | s    | 1   |
| External supply filter                                      |      | Type: XT-FIL-1, see the technical data on page 55         |
| Internal supply filter                                      |      | Yes   |

| <b>CPU</b>  |       |   |
|---|-------|---|
| Microprocessor  |       | Infineon XC161  |
| <b>Memory</b>   |       |   |
| Program code  | kByte | 256   |
| Program data  | kByte | 14 segments of 16 KB each   |
| Marker/Input/Output/Retain data   | kByte | 16/4/4/8  |
| Cycle time for 1 k instructions (Bit, Byte)   | ms    | < 0,3   |
| <b>Interfaces</b>   |       |   |
| <b>COM1 (RS 232) interface without handshake line</b>                                     |       |   |
| Data transfer rate for programming<br>Character format: 8 bit data, no parity, 1 stop bit | bit/s | 19 200, 38 400 (Default), 57 600 Bit/s                                      |
| Connection by   |       | RJ45 socket   |
| Electrical isolation  |       | none  |
| <b>in the transparent mode</b>  |       |   |
| Data transfer rate  | bit/s | 300, 600, 1 200, 2 400, 4 800, 9 600, 19 200, 38 400, 57 600, 115 200 Bit/s |
| Character formats   |       | 8E1, 8O1, 8N1, 8N2, 7E2, 7O2, 7N2, 7E1                                      |
| Number of transmission bytes in a block   |       | 190 bytes   |
| Number of received bytes in a block   |       | 190 bytes   |
| <b>COM2 (RS 232/RS 485) without handshake lines</b>                                       |       |   |
| Data transfer rate (transparent mode)<br>Setting via function blocks                      | bit/s | 300, 600, 1 200, 2 400, 4 800, 9 600, 19 200, 38 400, 57 600 Bit/s          |
| Character formats   |       | 8E1, 8O1, 8N1, 8N2, 7E2, 7O2, 7N2, 7E1<br>Setting via FB                    |
| Electrical isolation  |       | None  |
| Bus termination resistors for RS 485  |       | External  |
| <b>CAN1/CAN2 interface</b>  |       |   |
| Data transmission rate  |       | 10 Kbit/s to 500 kbit/s   |
| Electrical isolation  |       | No  |
| Participant   |       | 126   |
| Bus terminator  |       | Can be switched in for each interface (CAN1/CAN2)                           |
| PDO type  |       | Asynchronous, cyclic, acyclic   |

1) Higher transfer rates that are selectable in the PLC configuration are not possible!

**XIO-EXT121-1**

| <b>Connection by</b>             |      |  |
|----------------------------------|------|--|
| X1 connector                     |      |  |
| Connector type                   |      | Springloaded terminal block, 20-pole, B2L 3.5 (Weidmüller)                                 |
| Terminal capacity (solid)        |      | 0.5 mm <sup>2</sup> to 1 mm <sup>2</sup>   |
| X2/X3 connector                  |      |  |
| Connector type                   |      | Springloaded terminal block, 10-pole, BLZF 3.5/180 or BLI/O 3.5/10F with LEDs (Weidmüller) |
| Terminal capacity (solid)        |      | 0.5 mm <sup>2</sup> to 1 mm <sup>2</sup>   |
| <b>Supply voltage (24 V/0 V)</b> |      |  |
| Hold-up time on supply drop-out  |      |  |
| Dropout duration                 | ms   | 10   |
| Repeat rate                      | s    | 1  |
| Input voltage                    | V DC | 24   |
| Permissible range                | V DC | 20.4 to 28.8   |
| Power consumption                | W    | max. 1,68  |
| Input current                    | mA   | 70   |
| Residual hum and ripple          | %    | ≦ 5  |
| Overvoltage protection           |      | Yes  |
| Polarity protection              |      | Yes  |
| Switch-on current surge          | A    | max. 1   |
| max. field current IL            |      | 3.5 A  |
| <b>Digital inputs</b>            |      |  |
| Number with X2                   |      | 9 with connector BLI/O 3.5/10F<br>10 with connector BLZF 3.5/180                           |
| Number with X3                   |      | 8 (can also be used as outputs)  |
| rated voltage                    | VDC  | 24   |
| for 0 signal                     | V    | < 5  |
| for 1 signal                     | V    | > 15   |
| Rated current with 1 signal      | mA   | 3,3  |
| Delay time                       |      |  |
| 0 → 1                            | ms   | 20   |
| 1 → 0                            | ms   | 20   |
| Electrical isolation             |      | No   |

| <b>Digital outputs</b>                     |            |                                   |
|--|------------|-----------------------------------|
| Number with X3                             |            | 8 (can also be used as inputs)    |
| rated voltage $U_e$                        | VDC        | 24                                |
| Permissible range                          | VDC        | 20.4 to 28.8                      |
| Residual hum and ripple                    | V          | $\leq 5 \%$                       |
| rated current $I_e$ with 1 signal          | A          | 0.5 at 24 V DC                    |
| Simultaneity factor                        |            | 1                                 |
| Relative ON time                           | ms         | 100 %                             |
| Lamp load without $R_v$                    | W          | 5                                 |
| Electrical isolation                       |            | No                                |
| Residual current per channel with 0 signal | mA         | < 0,1                             |
| Max. output voltage                        |            |                                   |
| At 0 ext. load < 10 M $\Omega$             | V          | 2,5                               |
| at 1 with $I_e = 0.5$ A                    |            | $U = U_e - 1$ V                   |
| Short-circuit protection                   |            | Yes                               |
| Short-circuit detection threshold          |            |                                   |
| for $R_a < 10$ M $\Omega$                  |            | $0.7 \leq I_e \leq 2$ per output  |
| Total short-circuit current                | A          | 16                                |
| Peak short-circuit current                 | A          | 32                                |
| max. operating frequency                   | ops./h     | 40000                             |
| Can be switched in parallel                |            | Yes                               |
| <b>Data for the analog I/O</b>             |            |                                   |
| Analog inputs 0 ... 10 V                   |            |                                   |
| Number of channels                         |            | 2                                 |
| Input voltage range                        | V          | 0 ... 10                          |
| Resolution                                 | Bit        | 10                                |
| Conversion time                            | ms         | $\leq 5$                          |
| Overall accuracy                           |            | $\leq \pm 1 \%$ (of end of scale) |
| Input resistance                           | k $\Omega$ | 200                               |
| Analog inputs 0 ... 20 mA                  |            |                                   |
| Number of channels                         |            | 2                                 |
| Input voltage range                        | mA         | 0 ... 20                          |
| Resolution                                 | Bit        | 10                                |
| Conversion time                            | ms         | $\leq 5$                          |
| Overall accuracy                           |            | $\leq \pm 1 \%$ (of end of scale) |
| Input resistance                           | $\Omega$   | 50                                |
| Analog outputs                             |            |                                   |
| Number of channels                         |            | 2                                 |
| Output voltage range                       | V          | 0 ... 10                          |
| Resolution                                 | Bit        | 12                                |
| Conversion time                            | ms         | $\leq 5$                          |
| Overall accuracy                           |            | $\leq \pm 1 \%$ (of end of scale) |
| External load resistance                   | k $\Omega$ | 10                                |

**24 V DC line filter XT-FIL-1**

| <b>General</b>   |                 |   |
|--|-----------------|---|
| Standards and regulations                              |                 | IEC/EN 61131-2<br>EN 50178  |
| Ambient temperature                                    | °C              | 0 to +55  |
| Storage  | °C              | -25 to +70  |
| Mounting position                                      |                 | Horizontal/vertical   |
| Relative humidity, no condensation (IEC/EN 60068-2-30) | %               | 10 to 95  |
| Air pressure (in operation)                            | hPa             | 795 to 1080   |
| Vibration resistance                                   |                 | Frequency 5 to 9 Hz; amplitude 3.5 mm<br>9 to 150 Hz; 1.0 g constant acceleration |
| Mechanical shock resistance                            |                 | 15 g/11 ms  |
| Impact resistance                                      |                 | 500 g/∅ 50 mm ±25 g   |
| Overvoltage category                                   |                 | II  |
| Pollution degree                                       |                 | 2   |
| Enclosure protection                                   |                 | IP 20   |
| Rated impulse voltage                                  | V               | 850   |
| Interference emission (Industry)                       |                 | EN 61000-6-4  |
| Interference immunity (Industry)                       |                 | EN 61000-6-2  |
| Weight   | g               | 95  |
| Dimensions (W × H × D)                                 | mm              | 35 × 90 × 30  |
| Connecting terminals                                   |                 | Screw terminal  |
| Conductor cross-section                                |                 |   |
| Screw terminals  |                 |   |
| Stranded, with bootlace ferrule                        | mm <sup>2</sup> | 0.2 to 2.5 (AWG22-12)   |
| solid core   | mm <sup>2</sup> | 0.2 to 2.5 (AWG22-12)   |
| <b>Power supply</b>                                    |                 |   |
| Input voltage  | V DC            | 24  |
| Permissible range                                      | V DC            | 20.4 to 28.8  |
| Residual hum and ripple                                | %               | ≦ 5   |
| Overvoltage protection                                 |                 | Yes   |
| Electrical isolation                                   |                 |   |
| Input voltage to PE                                    |                 | Yes   |
| Input voltage to output voltage                        |                 | No  |
| Output voltage to PE                                   |                 | Yes   |
| Output voltage   | V DC            | 24  |
| Output current   | A               | 2,2   |



## Index

|          |   |            |          |   |        |
|----------|---|------------|----------|---|--------|
| <b>A</b> | Addressing, PLC on CANopen fieldbus       | 45         | <b>D</b> | Data access, to MMC                           | 11     |
|          | Analog inputs/outputs                     | 19         |          | Delay time entry                              | 29     |
|          | Application routine                       | 25, 29, 30 |          | Delete, memory card content                   | 11     |
|          | APPLICATION switch                        | 10         |          | Diagnostics possibilities                     | 46     |
|          | Setting                                   | 16         |          | DIP switch                                    | 10     |
|          |   |            |          | DIP switch mode                               | 10     |
| <b>B</b> | Backplane                                 | 7          |          | Direct IO access                              | 30     |
|          | Backup time, battery                      | 11         |          | Direct peripheral access                      | 31     |
|          | Basic expansion, maximum                  | 7          |          | Download, operating system                    | 32     |
|          | Battery buffer                            | 24         | <b>E</b> | Electromagnetic contamination                 | 17     |
|          | Baud rate, CAN bus                        | 13         |          | Engineering                                   | 17     |
|          | Baud rate, specifying/changing            | 39         |          | Event task                                    | 25     |
|          | Block size for data transfer              | 44         | <b>F</b> | Forcing                                       | 26     |
|          | Boot project                              | 23         |          | Forcing, variables and I/Os                   | 26     |
|          | Breakpoint                                | 26         |          | Format (browser command with delete function) | 11     |
|          | Browser commands                          | 35         |          | Function                                      |        |
|          | Bus length, CANopen                       | 14         |          | CAN_BUSLOAD                                   | 38     |
|          | Bus termination resistor                  | 13, 14     |          | Cold reset                                    | 9      |
|          | Bus utilization, CANopen fieldbus         | 36, 38     |          | EnableInterrupt                               | 27     |
| <b>C</b> | Cabinet layout                            | 17         |          | FileOpen                                      | 11     |
|          | Cable routing                             | 17         |          | FileRead                                      | 11     |
|          | Cable, CANopen (properties)               | 14         |          | GETAPPLICATIONSWITCH                          | 38     |
|          | CAN device parameters                     | 45         |          | GetApplicationSwitch                          | 10     |
|          | CAN Direct (Direct access to CAN objects) | 14         |          | Read...Direct                                 | 30     |
|          | CAN Master (configure XC121)              | 14         |          | removeprojfrommmc                             | 11     |
|          | CAN master parameters                     | 45         |          | TimerInterruptEnable                          | 29     |
|          | CAN settings (tab in the XSoft)           | 13         |          | Function blocks                               | 37     |
|          | CAN-Device (configure XC121)              | 13         |          | Functions                                     | 37     |
|          | canload, browser command                  | 36         |          | for transparent mode                          | 47     |
|          | CANopen                                   |            |          | on the real-time clock                        | 11     |
|          | Cable, properties                         | 14         |          | Operating mode switch                         | 9      |
|          | Interface                                 | 13         | <b>H</b> | Hardware timer                                | 25     |
|          | Network, demands                          | 14         | <b>I</b> | I/O module                                    | 7      |
|          | routing settings                          | 45         |          | Inductors                                     | 17     |
|          | Channel parameter setting                 | 46         |          | INI file                                      | 41     |
|          | Code, source                              | 33         |          | Initial value activation                      | 25     |
|          | CoDeSys gateway server                    | 44         |          | Interface                                     |        |
|          | Cold reset                                | 9          |          | CANopen                                       | 13     |
|          | Commissioning                             | 26         |          | Communication parameter definition            | 39     |
|          | Communication parameters                  | 39         |          | Connection with routing                       | 43     |
|          | Configuration                             |            |          | easy-NET                                      | 13     |
|          | XC121 as a CAN Master                     | 14         |          | Overview                                      | 9      |
|          | XC121 as a CAN-Device                     | 13         |          | Serial (COM1/COM2)                            | 12     |
|          | XIO-EXT121-1                              | 21         |          | Interference factors                          | 17     |
|          | Connecting actuators                      | 19         |          | Interrupt                                     | 27, 29 |
|          | Connecting sensors                        | 19         |          | Interrupt source                              | 29     |
|          | Connecting the power supply               | 18         |          | Interruption of power supply                  | 24     |
|          | Connection, PC – XN-PLC                   | 39         |          | IO access, direct                             | 30     |
|          | Core cross-section, CANopen cable         | 14         |          |   |        |
|          | Cycle time, monitoring                    | 25         |          |   |        |

|          |   |                |          |   |        |
|----------|---|----------------|----------|---|--------|
| <b>L</b> | Layout of units                             | 17             | <b>S</b> | Segments  | 12     |
|          | LED status indicator                        | 11             |          | Serial interface COM1/COM2                        | 12     |
|          | Libraries                                   |                |          | SET button  | 9      |
|          | CANUser.lib, CANUser_Master.lib             | 5              |          | Shielding   | 17     |
|          | SysLibRTC                                   | 11             |          | Single-cycle mode                                 | 26     |
|          | XC121_File.lib                              | 11             |          | Single-step mode                                  | 26     |
|          | XC121_SysLibCom.lib                         | 47             |          | Source code                                       | 33     |
|          | XC121_Util.lib                              | 10, 27, 29, 38 |          | Start behaviour                                   | 23, 42 |
|          | Libraries, installing                       | 37             |          | Starting the CPU                                  | 9      |
|          | Lighting protection                         | 18             |          | Start-up behaviour, setting in XSoft              | 24     |
|          | Local expansion module                      | 7              |          | STARTUP.INI                                       | 41     |
|          | Loop resistance, CANopen cable              | 14             |          | Status indicator                                  |        |
|          |   |                |          | in the XSoft                                      | 26     |
|          |   |                |          | on the LED  | 11     |
| <b>M</b> | Memory                                      | 12             |          | STOP  | 9, 25  |
|          | Memory card                                 | 11             |          | Structure, XC121                                  | 7      |
|          | Memory usage, limit values                  | 12             |          | Suppressor circuitry for interference sources     | 17     |
|          | MMC   | 11             |          | Switch off of the supply voltage                  | 24     |
|          | Mounting                                    |                |          | Switch-on behaviour                               | 23, 42 |
|          | XC121 on top-hat rail                       | 15             |          | System clock, back-up                             | 11     |
|          | XIO-EXT121-1 on the XC121                   | 15             |          | System events                                     | 25, 27 |
|          | Mounting position, PLC in the control panel | 17             |          | System parameter setting                          | 41     |
|          |   |                |          | System self test                                  | 23     |
|          |   |                |          | System time                                       | 25     |
| <b>N</b> | Node ID                                     | 45             | <b>T</b> | Target ID   | 45     |
|          | CAN1/CAN2                                   | 10             |          | TCP/IP connection (for routing)                   | 44     |
|          | Routing                                     | 10             |          | Test communication (from browser to PLC and back) | 36     |
|          | Setting                                     | 13             |          | Test functions                                    | 26     |
|          | Node number                                 | 45             |          | Timer interrupt                                   | 29     |
| <b>O</b> | Operating mode                              | 10             |          | Total expansion, maximum                          | 7      |
|          | Operating mode switch                       | 9              |          | Transfer rate, CANopen                            | 14     |
|          | Operating system update                     | 23             |          | Transparent mode                                  | 12, 47 |
|          | Operating system, download/update           | 32             | <b>U</b> | Uninterruptible power supply                      | 24     |
| <b>P</b> | PLC browser                                 | 35             | <b>V</b> | Variables   |        |
|          | Power off/interruption of the power supply  | 24             |          | Behaviour after reset                             | 25     |
|          | Program processing                          | 25             |          | Behaviour at start                                | 24     |
|          | Program stop                                | 25             |          | Ventilation                                       | 17     |
|          | Programming interface                       | 12             | <b>W</b> | Wiring  | 17     |
|          | Programming software                        | 7              |          | Wiring example, connecting the power supply       | 18     |
|          |   |                |          | Wiring, inputs/outputs (mounting example)         | 16     |
| <b>R</b> | ReadBitDirect                               | 30             |          |   |        |
|          | Real-time clock                             | 11             |          |   |        |
|          | reflect                                     | 36             |          |   |        |
|          | Removing, XC121 from top-hat rail           | 15             |          |   |        |
|          | Reset                                       | 25, 27         |          |   |        |
|          | Restoring factory default state             | 9              |          |   |        |
|          | Retentive variables                         | 24             |          |   |        |
|          | Routing-Id                                  | 10             |          |   |        |
|          | RUN   | 9              |          |   |        |
|          | Runtime system                              | 23             |          |   |        |