schleicher
control systems

Operating Manual

XCx 1100 / XCx 1200

Operating Manual XCx 1100 / XCx 1200 ● Version 06/16
Order No. R4.322.2390.0 (322 385 79)

Target Group

This Operating Manual has been written for trained personnel with specialized knowledge. There are special requirements for the selection and training of the personnel who work on the automation system. Suitable personnel include electricians and electrical engineers who have had the relevant training (see also Safety-related Information "Selection and Qualification of Personnel").

Applicability of this Operating Manual

hardware rev. 02 / software V09.05/2 version or higher

Previous Versions of this Operating Manual

02/09

Where to Get Operating Manuals

You can download all our operating manuals free of charge from our website at http://www.schleicher.berlin
or order them (please quote order no.) at:

Schleicher Electronic Berlin GmbH
Wilhelm-Kabus-Straße 21-35
10829 Berlin

Germany

Additional Documentation

See page 10

Copyright by

Schleicher Electronic Berlin GmbH
Wilhelm-Kabus-Straße 21-35
10829 Berlin

Germany

Phone +49 30 33005-0
Fax +49 30 33005-378
E-Mail: info@schleicher.berlin
Internet: http://www.schleicher.berlin

Errors and omissions reserved.

Contents

Document conventions

This programming manual uses the following symbols to indicate safety-related and handling warnings:

Warning!

Indicates possible injury to persons or damage to the automation system or the equipment if relevant warnings are not observed.

Italics: Information on preventing a hazard.

Important! or Note

Important information on the handling of the automation system or the respective part in the operating manual.

Other objects are represented as follows:

Object	Example
File names	MANUAL.DOC
Menus / Menu items	*Insert / Graphic / From file*
Paths / Directories	*C:\Windows\System*
Hyperlinks	*http://www.schleicher.berlin*

Program listings	`MaxTsdr_9.6   = 60`
	`MaxTsdr_93.75 = 60`

Keys	<Esc> <Enter> (press one after the other) <Ctrl+Alt+Del> (press all keys at the same time)
Configuration data identifiers	Q23
Name of variables	*mcMem.axSect[n].bContRel*

1 Safety-related information

The term automation system as used in this manual includes controllers, their components (modules), other parts (such as racks, cables), operator panels, and the software used for programming, commissioning and operating the controllers. This Operating Manual can only describe a part of the automation system (e.g. modules).

The technical design of Schleicher automation systems is based on the EN 61131-2 (IEC 61131-2) product norm for PLCs. The systems and devices have CE marking according to the EMC directive 2004/108/EC and, if applicable, the low-voltage directive 2006/95/EC.

The machinery directive 98/37/EC or 2006/42/EC is not applicable, because the safety objectives of the directive are covered by the low-voltage and EMC directives.

When Schleicher automation systems are part of the electrical equipment of a machine, the manufacturer must include them in the conformity evaluation process. In this case the DIN EN 60204-1 norm must be observed (safety of machines, general requirements for electrical equipment of machines).

When an automation system is properly maintained and used for its intended purpose, it will not normally cause damage to property or present health hazards. However, improper configuration, installation, maintenance or operation of the system or machine, ignoring the instructions in this manual, or intervention by insufficiently qualified personnel may result in connected actuators (such as motors, hydraulic units, etc.) becoming a source of danger.

1.1 Proper Use

SCHLEICHER automation systems are state-of-the-art products and manufactured to recognised safety requirements. All the same, their use can cause danger to the health and safety of operators and others, or damage machines, systems or other property.

The automation system must only be used in perfect technical condition for its intended purpose, with attention given to safety and danger, and observing the Operating Manual. Correct transport, storage, installation, operation and maintenance of the system are all prerequisites for smooth and safe operation of the control system. Malfunctions, in particular those which may affect safety, must be immediately resolved.

Automation systems are designed exclusively to control machines and systems. Automation systems are not intended for any other use than the above. The manufacturer will therefore accept no liability for any damages resulting from the incorrect use of the systems.

When using automation systems, all instructions given in this manual regarding mechanical and electrical setup, commissioning and operation must be observed.

1.2 Selection and Qualification of Personnel

Important!

All configuring, programming, installation, commissioning, operation and maintenance work on the automation system must be carried out by trained personnel such as electricians or electrical engineers. Personnel responsible for configuring and programming the system must be familiar with all safety-related issues in automation technology.

System operators must be instructed on the operation of the control system and be familiar with the relevant operating instructions.

All personnel responsible for installing, commissioning and maintaining the system must have had appropriate training qualifying them to work on automation systems.

1.3 Configuring, Programming, Installation, Startup and Operation

The automation system will in most cases be a part of a larger system in which machines are controlled. When configuring, installing and commissioning automation systems to control machines the machine manufacturer and the user must observe the safety regulations as defined in the machinery directive 98/37/EC or 2006/42/EC . For specific applications national accident prevention regulations such as VBG 4.0 will apply.

Safety-related components on the controlled machine must be designed such that they operate independently from the control system. Emergency stop components must remain operative in all operating modes of the controller. In an emergency stop the power supply to all switching elements controlled by the control system must be brought to a safe state.

Measures must be taken for restarting an interrupted control program following voltage dips or power failures. Operating conditions should never cause danger, not even for a short time. In the event of danger the emergency stop must be immediately triggered.

In order to prevent an open-circuit in the signal circuit causing non-controllable conditions in the control system, the relevant hardware and software safety precautions must be taken for I/O interfacing. Control elements and their assigned control panel elements must be installed in a place where they are sufficiently protected against inadvertent use.

1.4 Hazards due to Electrical Energy

⚠ **Warning!**

When the cabinet is opened or casing is removed from system components certain parts of the automation system are exposed. These parts may be subject to dangerous high voltages.

Switch off the voltage before working on the devices. Prevent short circuits when measuring live components.

The user must prevent any unauthorised and incorrect access to the system (for example, by ensuring that the cabinet is locked).

Personnel must be familiar with all sources of danger and measures for commissioning and maintaining the system in line with the instructions given in this manual.

1.5 Maintenance

Measuring and testing on active devices must be carried out in accordance with the regulations and instructions of national accident prevention regulations such as VBG 4.0. The appropriate power tools must be used.

Repairs on control components must be carried out at authorised repair shops only. Opening the components and repairs by unauthorised personnel may lead to personal injury or damage to property.

Always disconnect the device from the mains before opening it (either disconnect the mains plug or use the cut-out switch).

Control modules may only be replaced when the power is switched off. Disassembly and assembly must be carried out according to the directives for mechanical assembly.

Fuses may only be replaced with those types specified in Technical Data.

Batteries may only be replaced with those types specified in Technical Data. Batteries must always be disposed as hazardous waste.

1.6 Dealing with Used Batteries

When the batteries in the automation system are dead they must be disposed of in a battery return system or through public waste disposal facilities.

Batteries should be fully discharged before disposal. A battery is discharged when the function of the device is impaired due to insufficient battery capacity.

When batteries for disposal are not fully discharged precautions must be taken to prevent short circuits. For example by sticking tape over the poles of the battery.

2 Additional Operating Manuals

Important!

The XCx 1100 is a member of the XCx controller family that is based on a common software and hardware concept. For this reason, the following operating manuals must be used in addition to this Operating Manual.

Table 1: Additional Operating Manuals

Designation	Order no. or reference
For commissioning the field buses	
EMC Guidelines	R4.322.1070.0
Commissioning field bus systems	R4.322.1610.0
For programming the PLC and the CNC	
MULTIPROG programming system to IEC 61131-3	MULTIPROG manual (Quickstart_MWT.pdf) in the installation path from MULTIPROG
CNC Programming for XCx and ProNumeric	R4.322.2090.0
Shared RAM allocation of the XCx	Online help for the software packet for XCx
Operating Manual (German) sercos III-I/O	R6.322.0770.0
For the racks, power supplies and expansion modules	
Expansion modules for Promodul-U / XCx	R4.322.2410.0

All operating manuals are available as PDF files on the service CD for the XCx and can be downloaded for free at the website; http://www.schleicher.berlin

3 System Overview

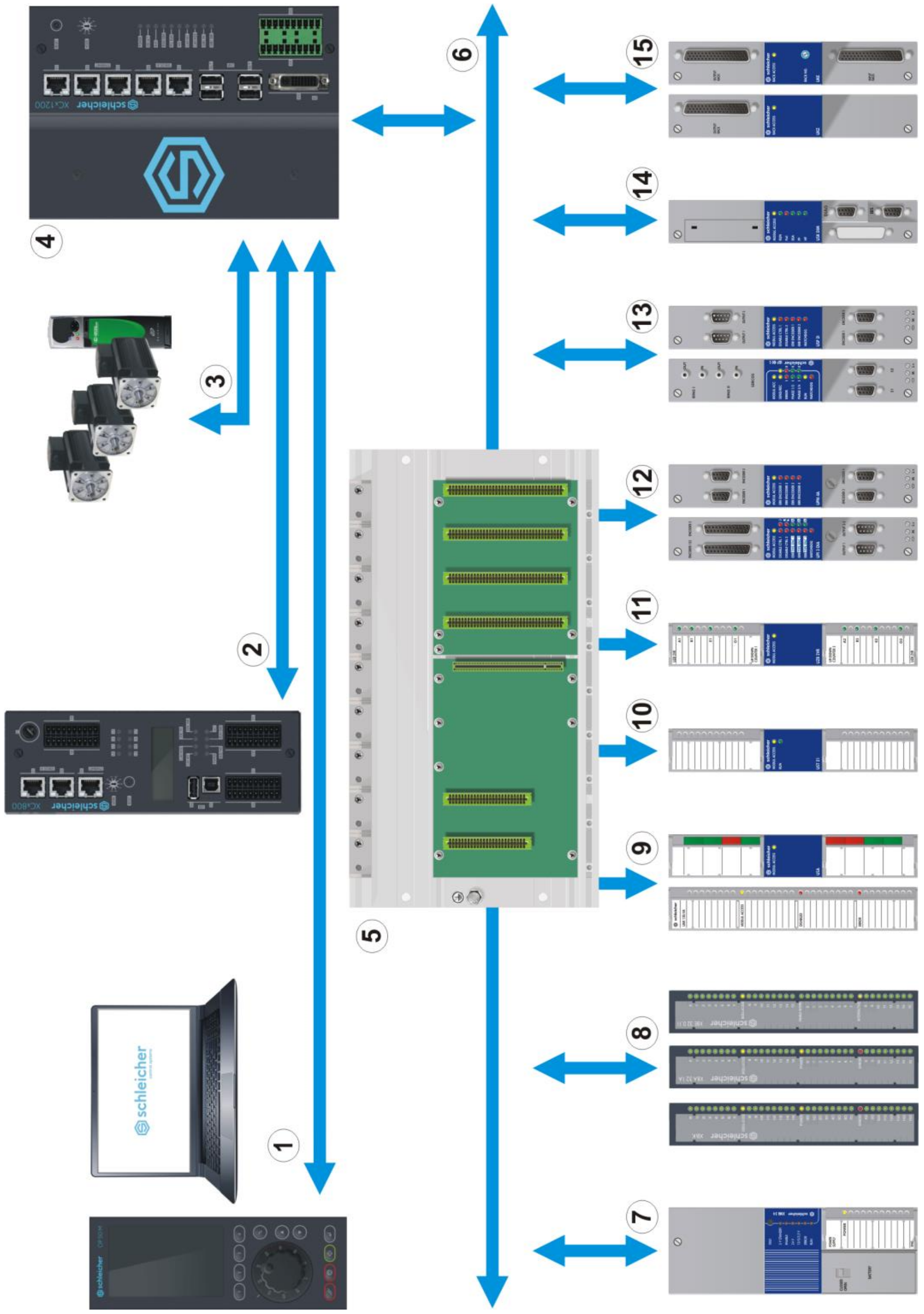
The XCA 1100 and XCA 1200 are not a PLC or an IPC in the classic sense but corresponds to the advanced concept of a ***P***rogrammable ***A***utomation ***C***ontroller (PAC) in its basic system characteristics.

It is capable of providing a number of complex automation tasks and scenarios of use for the highest level of performance and an open, modular architecture. Conventional requirements such as control, regulation, operation, diagnosis and reporting are operated by the XCx 1100 on a standard scalable platform.

The XCA 1100/1200 operates with VxWin, the established combination of the VwWorks real-time operating system and Windows embedded. VxWorks takes on the real-time component, i.e. control via PLC, CNC and Motion Control functions, while Windows provides the familiar environment for non-time-critical functions like visualisation and operator dialogs:

- NC operator dialogs
- Visualisation
- NC program memory
- Diagnosis
- Configuration
- PLC programming
- Manual
- Operational data logging

The operating systems operate separately from one another, because the XCx 1100 memory management unit (MMU) keeps the program memory areas separate. This ensures that instability on the Windows level has no effect on Schleicher CNC runtime or PLC runtime firmware running on VxWorks.



1	Programming, Visualization, Operation, Diagnostic
2	Fieldbus, Control Networking
3	Digital Servo Drives
4	Control Unitfor CNC/PLC
5	Module Rack
6	Systembus
7	Power Supply Units
8	Digital I/O Modules
9	Analog I/O Modules
10	Temperature Processor
11	Counter Module
12	Positioning Interface
13	Positioning Processors
14	Communication Module
15	Interface Modules

Figure 1: System overview XCx 1100, expansion modules and periphery

3.1 Controller Structure

The XCx 1100 is a modular automation system where up to 256 modules can be arranged on a maximum of 16 racks.

Important!

The automation system must be installed in earthed metal housings (e.g. enclosures). Observe the regulations described in the "EMC Guidelines for the Structure of Automation Devices" documentation (→page 10).

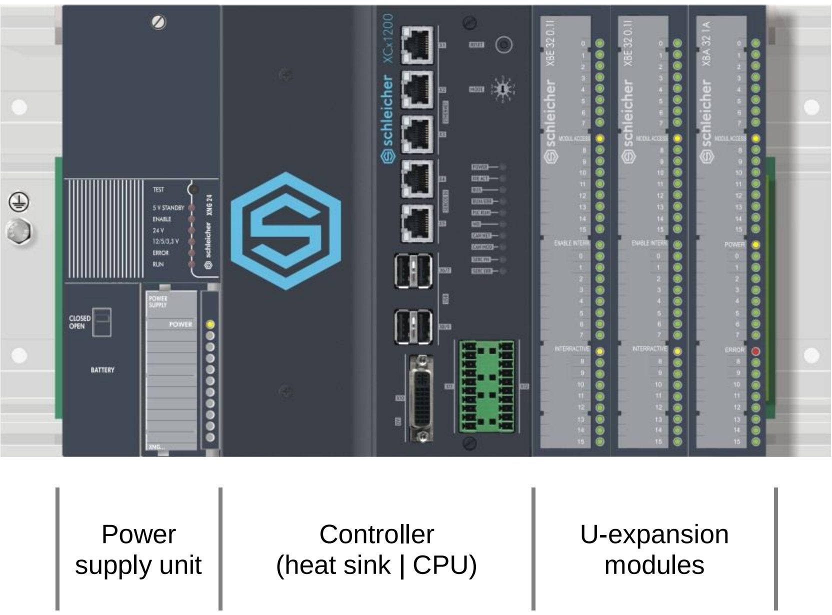
The CPU is a module with a Promodul-U system design. The installation height and depth are designed accordingly. The CPU module (CPU and heat sink) occupy the width of four standard U-modules in total.

Components that are susceptible to wear, such as fans or hard disks, are dispensed with to achieve the highest possible level of operating safety and the lowest level of maintenance.

Compact Flash or Solid State Discs are used for storing programs and data.

The backplane has a mechanically separated design. The U-periphery is located to the right of the CPU and the power supply unit is located on the left. This module supplies the CPU and the U bus with the required operating voltages.

Figure 2: Structure of the complete system



3.2 Assembly

The controller slot on a rack of the XBT series (→ Figure3) is located between the power supply unit (left) and the expansion modules (right). This slot order must be observed!

Figure3: Slot of the controller on the rack

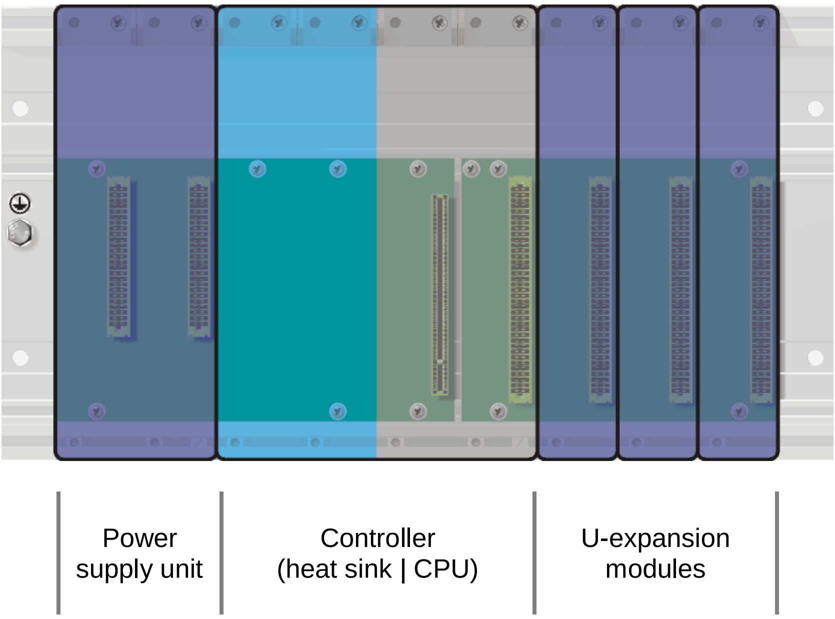
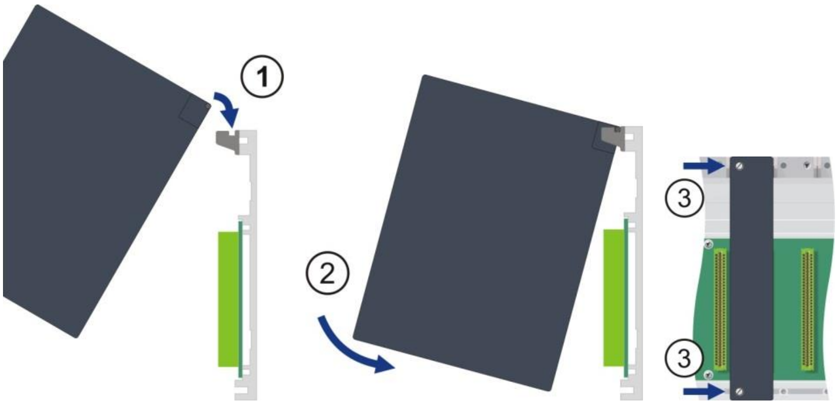


Figure4: Install the modules on the rack



1.
Hook the module with the lateral pin into the rack from above

2.
Press module firmly onto contact strip

3.
Tighten attachment screws

Note

Further information on the complete system, rack fitting, measurement of the required power supply units and selection of the expansion modules can be found in the "Expansion Modules for XCx and Promodul-U" operating manual (→ page 10)

4 XCA 1100 / XCA 1200 Control Unit

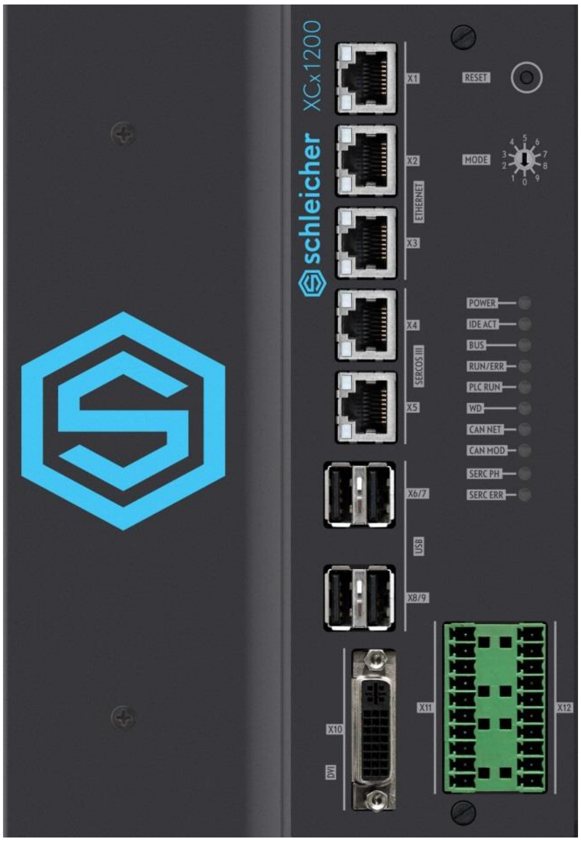


Figure 5: Control unit XCA, CPU and heat sink

The XCx 1100 control units are fitted with a PLC operating system and a powerful CNC operating system.

A Windows operating system is also available for visualisation, operation and programming.

PLC

- Operating system: ProConOS
- Programming: MULTIPROG acc. to IEC 61131-3

CNC

- Programming: to DIN 66025
- Machine-specific special functions and transformations
- Communication with the PLC via shared RAM

Windows

- Windows XP embedded

All control units have:

- An internal Compact Flash memory card
- 3 Ethernet interfaces with integrated Ethernet switch
- 4 USB 2.0 interfaces
- DVI interface
- Serial interfaces
- Integrated web server

Options:

- Various processor and memory specifications
- Sercos III
- CANopen

(for more details about the versions, see "Variants XCA 1100", page 26)

4.1 Interfaces, Control Panel Elements, Displays

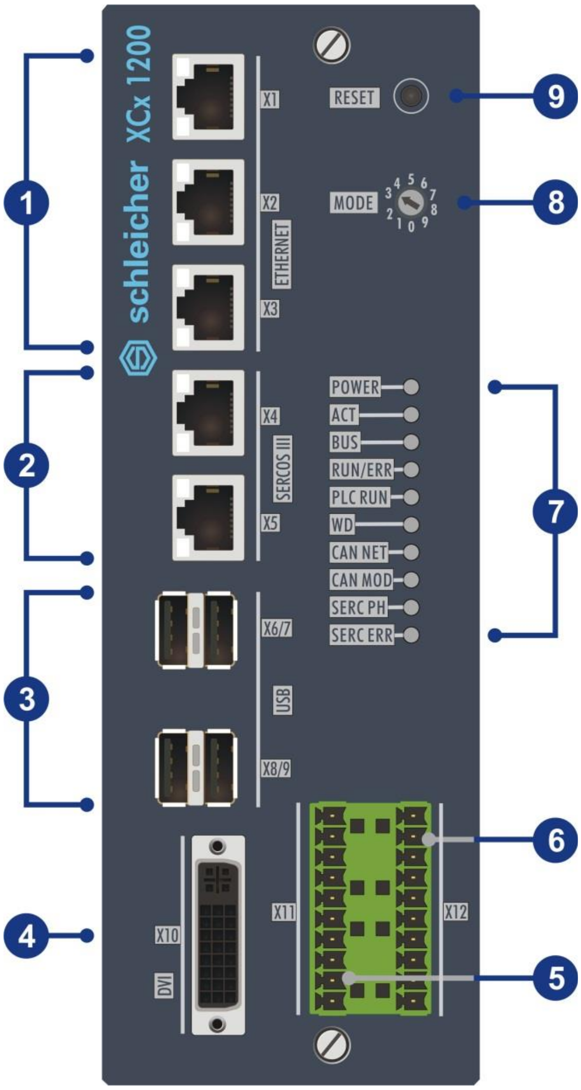


Figure 6: XCx 1100 control unit, interfaces, operator panel elements and LED displays

1 X1, X2, X3
Ethernet interfaces, RJ 45

2 X4, X5
Sercos III interfaces, RJ 45

3 X6/7, X8/9
USB interfaces

4 X10
DVI Interface

5 X11
CAN Interfaces

6 X12
RS 232 / RS 422 / RS 485
for connection of operator panels and displays

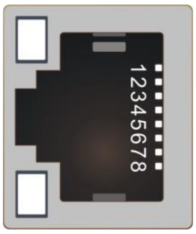
7 LED displays

8 Operating mode switch

9 Reset button

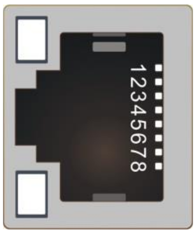
4.1.1 X1, X2, X3 – Ethernet-Interfaces

Table 2:
Pin assignment of Ethernet interfaces X1, X2, X3 (RJ 45)

	Pin	Designation	Explanation
	**1**	TX+	Transmitted data plus
	2	TX+	Transmitted data minus
	3	RX+	Received data plus
	4	nc	Not connected
	5	nc	Not connected
RJ 45 socket connector	**6**	RX-	Received data minus
	7	nc	Not connected
	8	nc	Not connected

4.1.2 X4, X5 – Sercos III Interfaces

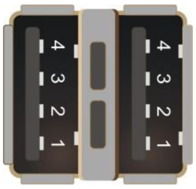
Table 3:
Pin assignment of the Sercos III interfaces X4, X5

	Pin	Designation	Explanation
	**1**	TX+	Transmitted data plus
	2	TX+	Transmitted data minus
	3	RX+	Received data plus
	4	nc	Not connected
	5	nc	Not connected
RJ 45 socket connector	**6**	RX-	Received data minus
	7	nc	Not connected
	8	nc	Not connected

The Sercos III interfaces are only active for control units that are fitted accordingly (→ "Variants XCA 1100", p. 26).

4.1.3 X6/7, X8/9 – USB 2.0 Interfaces

Table 4:
Pin assignment of USB interfaces X6/7, X8/9

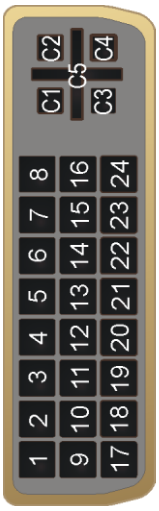
	Pin	Designation	Explanation
	**1**	VCC	+5 V
	2	D-	Data minus
	3	D+	Data plus
	4	GND	Ground
USB socket connector Standard A			

4.1.4 X10 – DVI Interface

The monitor interface is designed as a DVI-I (single link with 18+5 contacts). Both digital and, via a DVI AGA adapter, analogue monitors can be operated.

Table 5: Pin assignment of DVI interface X10

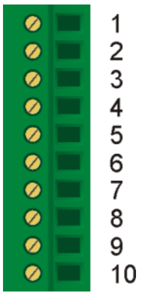
Pin	Designation	Explanation
1	TDMS data 2-	Digital red minus (link 1)
2	TDMS data 2+	Digital red plus (link 1)
3	TDMS data 2/4 shield	Shield data 2,4
4	nc	Not connected
5	nc	Not connected
6	DDC clock	DDC clock pulse
7	DDC data	DDC data
8	Analogue vertical sync	V sync
9	TDMS data 1-	Digital green minus (link 1)
10	TDMS data 1+	Digital green plus (link 1)
11	TDMS data 1/3 shield	Shield data 1,3
12	nc	Not connected
13	nc	Not connected
14	+5V	
15	Ground	Ground for 5 V
16	Hotplug detect	
17	TDMS data 0-	Digital blue minus (link 1)
18	TDMS data 0+	Digital blue plus (link 1)
19	TDMS data 0/5 shield	Shield data 0.5
20	nc	Not connected
21	nc	Not connected
22	TDMS clock shield	Shield clock pulse
23	TDMS clock+	Clock pulse plus
24	TDMS clock-	Clock pulse minus
C1	Analogue red	Analogue red
C2	Analogue green	Analogue green
C3	Analogue blue	Analogue blue
C4	Analogue horizontal sync	H sync
C5	Analogue ground	Ground



DVI-I socket connector
Single link

4.1.5 X11 – CAN Interfaces

Table 6:
Pin assignment of CAN interface X11

	Pin	Designation	Explanation
 Screw block terminal 10-pin	**1**	**V+**	Power supply +24DCV
	2	**CAN_H**	CAN high
	3	**DRAIN**	Shield connection (optional)
	4	**CAN_L**	CAN low
	5	**V-**	Ground 0V
	6	**V+**	Power supply +24DCV
	7	**CAN_H**	CAN high
	8	**DRAIN**	Shield connection (optional)
	9	**CAN_L**	CAN low
	10	**V-**	Ground 0V

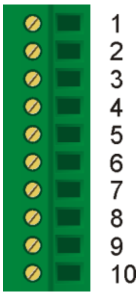
Pin groups 1..5 and 6..10 are connected in parallel.

The CAN interfaces are only active for control units that are fitted accordingly (→ "Variants XCA 1100", page 26).

The screw terminal block is coded to prevent reversal of the interfaces X11/X12.

4.1.6 X12 – RS 232 / RS 422 / RS 485 Interfaces

Table 7:
Pin assignment of RS 232 / 422 / 485 interfaces X12232



Screw block terminal 10-pin

Pin	Designation	Explanation
1	**SHLD**	Shield RS 232
2	**TxD**	RS 232 transmitted data
3	**RxD**	RS 232 received data
4	**M_{ext}**	Ground for RS 232
5	**M_{ext}**	Ground for RS 422 / RS 485
6	**TD-**	Transmitted data / transmitted and received data
7	**TD+**	Transmitted data / transmitted and received data
8	**RD-**	Received data / bus terminating resistors
9	**RD+**	Received data / bus terminating resistors
10	**SHLD**	Shield RS 422 / RS 485

The RS interfaces are used to connect the operator panels and displays. Connect RD+ with TD+ and RD- with TD- for activating the bus terminating resistors when using the RS 485 interface. Both ground pins for RS 232 and RS 485 have the same potential.

The screw terminal block is coded to prevent reversal of the interfaces X11/X12.

Caution!

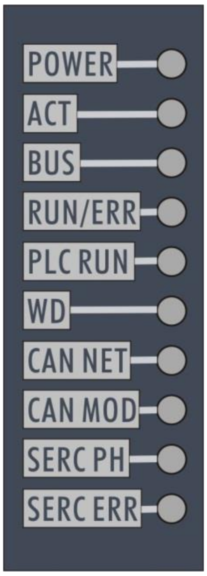
Do not connect open cables to serial ports (RS 232 and RS 485), which are temporary connected to service computers for example.

Open cables can cause linkages between transmitter and receiver. This can lead to errors within the control.

If not needed, remove the plug from the control.

4.1.7 Controller LED Displays

Table 8: LED displays on the controller



Designation	Colour	Status	Meaning
POWER			**Power**
		off	Device switched off
	green	on	Unit switched on
ACT			**IDE (CF) / SATA (SSD) activity**
		off	No access
	green	flashing	Access successful
BUS			**Bus access**
		off	No access for PLC stop or (real-time) operating system inactive
	green	on	Bus access OK
	red	flashing	Bus access error / configuration error
RUN/ERR			**CPU status**
		off	CPU faulty
	green	on	CPU boots
	green	on	CPU running, operating voltage OK, no errors
	red	flashing	Fatal error: CPU can not boot
PLC RUN			**PLC status**
		off	PLC stop
	green	on	PLC running
	yellow	flashing	PLC running, but outputs shut down (ready-for-operation relay released)
WD			**Watchdog**
		off	Watchdog did not respond
	red	on	Serious error or (real-time) operating system inactive
CAN NET			**CAN network status**
		off	CAN state prepared
	green	on	CAN state operational
		flashing	CAN state pre-operational
	red	on	Bus off
		flashing	CAN error
CAN MOD			**CAN module status**
	green	on	CAN stack initialized
		flashing	Invalid CAN configuration
	red	on	Control unit not ready, or serious error
		flashing	Error in controller

(continuation)

Table 8:
LED displays on the controller (continuation)

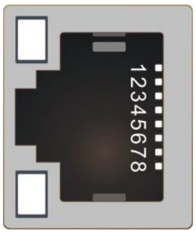
Designation	Colour	Status	Meaning
SERC PH			**SERCOS phases**
	red	on	SERCOS phase 0
	red	flashing	SERCOS phase 1
	yellow	flashing	SERCOS phase 2
	green	flashing	SERCOS phase 3
		on	SERCOS phase 4
SERC ERR			**SERCOS error**
	red	off	No error
		on	Communication error
		flashing	Drive error

Error messages are saved in the active error buffer and the error logbook and have error numbers and additional information.

The active error buffer and logbook can be called on any operating level using the <Ctrl+?> key combination.

4.2 Ethernet and Sercos III LED Displays

Table 9:
LED displays at the Ethernet and Sercos-III socket connectors (5x RJ45)



Designation	Colour	Status	Meaning
1			**Link / Activity / Speed**
		off	No network connection, no activity
	yellow	on	10 Mbit/s
		flashing	Activity
	green	on	100 Mbit/s
		flashing	Activity
2			**Duplex / Collision**
		off	Half duplex operation
	yellow	on	Full duplex operation
		flashing	Collision

4.2.1 Operating mode switch

The operating mode switch has ten positions to set the startup behaviour of the controller.

Table 10: Operating mode switch

	Position / designation	**Meaning**
	0	Default initialisation / diagnosis (Start of the real-time operation system in safe mode and reset of retentive data memory, → p. 81)
	1 / prog	Programming mode (PLC stop)
	2 / Warm (also 4..9)	PLC warm start to IEC 61131-3 (default setting)
	3 / Cold	PLC cold start to IEC 61131-3 (Reinitialisation of the retain variables)

The current setting of the operating mode switch can be requested in the PLC program. The variable is already created in the templates for the XCx control units supplied with MULTIPROG (worksheet *Global_Variables, PLC_COMMON:* `cmpSwrdPlcRd_lXModeSwitch`).

4.2.2 Reset button

The reset button allows the CPU module to be switched off or reset:

- Short push of the button = Reset
- Long push of the button = Shutdown

Table 11: Reset button

Important!

If RESET is activated while accessing the CF card or SSD (LED *ACT* flashes green), data may be lost.

4.3 Technical Data for Control Unit XCx 1100

Electrical data		
Internal power supply		DC 12 V, DC 5 V, DC 3.3 V
Internal power consumption		< 40 W
Isolation (from internal electronics)	X1, X2, X3 Ethernet	Yes
	X4, X5 Sercos III	Yes
	X6/7, X8/9 USB	No
	X10 DVI	No
	X11 CAN	Yes
	X12 (RS 422)	Yes
	X12 (RS 232)	Yes

Interfaces		
Ethernet	RJ 45	Programming, diagnosis and operating panel interface
Sercos III	RJ 45	Sercos III drive interface (Ethernet)
USB	Standard A	USB interface (e.g. mouse, keyboard, USB memory stick)
DVI	DVI-I single link	Monitor interface (DVI-I)
CAN	10-pin plug-in terminal	CANopen field bus interface
RS 232	10-pin plug-in terminal	for stationary connection of serial units
RS 422	10-pin plug-in terminal	Serial operating panel interface

Hardware and memory			
Processor		Performance versions	
		CPU Intel Celeron M 370, 1.50 GHz, 1 MB L2 cache	CPU Intel Core 2 Duo SU 9300 Ultra Low Voltage, 1.2 GHz, 3 MB L2 cache
Memory	SDRAM	512 MB to 4 GB	
	SRAM (buffered)	1 MB	
	Solid state drive	32 GB (optional, alternative to CF)	
Real-time clock		Battery-buffered with calendar and leap year, resolution: 1s	
Buffering		Supercap min. 3 hours, rechargeable battery after at least of 4 hours charge time, min. 3 months.	

CNC/PLC properties		
PLC processing times every 1000 instructions	Bit	0.064 ms
	Byte / Word / DWord	0,033 ms
	Integer (Add / Mul)	0,038 ms
	Real (Add)	0.064 ms
PLC signal propagation time (input to output)		< 2 ms (for task periods = 1 ms)
Function blocks		Any number of firmware functions and function blocks
Number of NC axes / sub-systems		64 / 32
CNC interpolation cycle from		1 ms
Block cycle time from		1 ms
Operating system	Controller	VxWorks, multitasking operating system (time-driven / priority-driven)
	PLC runtime	ProConOS
	PC	Windows Embedded
Configuring		MULTIPROG acc. to IEC 61131-3
Number of user tasks		18
Task cycle times		Programmable ≥ 1 ms (whole number)
Real-time memory (can be set)	Operating system (data / program)	32768 kB
PLC memory	Programs	4096 kB
	Flag retentive	256 kB
	Flag not retentive	2048 kB
Memory management		Dynamic
Times and counters		Any number programmable from 1 ms ... 290 h (number limited only by memory capacity)

Dimensions / weight	
Dimensions (W x H x D)	142 mm x 200 mm x 150 mm
Modular spacing	4
Weight	2500 g

The information in chapter "Technical Data of all Modules", page 128 also applies.

4.4 Variants XCA 1100 and XCA 1200

Variants XCA 1100 and XCA 1200	
XCA 1100	Celeron M, 1,5 GHz, 512 MB, 4 GB Compact Flash
XCA 1110	Celeron M, 1,5 GHz, 1 GB DDR2-SODIMM, 32 GB SSD
XCA 1120	Core 2 Duo, 1,2 GHz, 1 GB DDR2-SODIMM, 32 GB SSD
XCA 1125	Core 2 Duo, 1,8 GHz, 1 GB DDR2-SODIMM, 32 GB SSD
XCA 1200	Core 2 Duo, 1,8 GHz, 2 GB, DDR2-SODIMM, 64 GB SSD, Hypervisor
Basic Funktions	
	IPC Control Unit 64 interpolating Axes
	PC-based Control Unit
	CNC-Operating System with synchr. PLC
	sercos III I/O
	More than 4 Axes the Software-Option SCR xx is required
	Windows Embedded
	Ethernet Switch
	DVI
	USB
Optionale Funktions	
C	CANopen
S	sercos III für Drives
E	Export Version, IPC Control Unit for max. 4 interpolating Axes
Software-Option	
SRC8	8- Axes sercos III
SRC16	16- Axes sercos III
SRC24	24 Axes sercos III
SRC32	32- Axes sercos III

5 Expansion Modules for XCx and Promodul-U

There are a number of racks , power supply units and expansion modules available for the XCx 1100 control units. These modules are described in a separate operating manual (→ page 10).

Module	Order No.	Comment
Racks		
XBT 0 / 1100	R4.507.0040.0	Basic rack, NT, CPU
XBT 3 / 1100	R4.507.0010.0	Basic rack, NT, CPU and 3 slots
XBT 4 / 1100	R4.507.0030.0	Basic rack, NT, CPU and 4 slots
XBT 7 / 1100	R4.507.0020.0	Basic rack, NT, CPU and 7 slots
XBT 11 / 1100	R4.507.0050.0	Basic rack, NT, CPU and 11 slots
UBT 4 x*	R4.311.0010.0	Basic rack/expansion rack, 4 slots
UBT 8 x*	R4.311.0020.0	Basic rack/expansion rack, 8 slots
UBT 12 x*	R4.311.0030.0	Basic rack/expansion rack, 12 slots
UBT 16 x*	R4.311.0040.0	Basic rack/expansion rack, 16 slots
Interface modules		
UKZ	R4.318.0030.0	Interface module for basic rack
UKE	R4.318.0040.B	Interface module for expansion rack
Power supply units		
UNG 230A x**	R4.312.0030.F	Power supply unit 230 V, 2 unit width
UNG 115A x**	R4.312.0040.F	Power supply unit 115 V, 2 unit width
UNG 24 x**	R4.312.0020.B	Power supply unit 24 V, 1 unit width
XNG 24 x	R4.507.0100.0	Power supply unit 24 V, 2 unit width
Digital I/O modules		
UBE 32 0,1I	R4.314.0100.E	32 inputs, 4 interrupts, 0.1 ms input delay.
UBE 32 1D	R4.314.0120.E	32 inputs, 1 ms input delay
UBE 32 10D	R4.314.0090.E	32 inputs, 10 ms input delay
UBA 32/2A	R4.314.0080.D	32 24 V DC / 2 A semiconductor outputs
UBK 16E 1D/16A	R4.314.0130.E	16 inputs, 1 ms input delay / 16 outputs
UBK 16E 10D/16A	R4.314.0110.E	16 inputs, 10 ms input delay / 16 outputs
XBE 32 1D	R4.314.0140.0	32 inputs, 1 ms input delay
XBE 32 10D	R4.314.0180.0	32 inputs, 10 ms input delay
XBE 32 0,1I	R4.314.0170.0	32 inputs, 4 Interrupts, 0,1 ms input delay.
XBA 32/1A	R4.314.0150.0	32 semiconductor outputs DC 24V / 1A
XBK 16E 1D/16A	R4.314.0160.0	16 inputs, 1 ms input delay / 16 outputs
XBK 16E 10D/16A	R4.314.0190.0	16 inputs, 10 ms input delay / 16 outputs
Counter modules		
UZB 2VR	R4.315.0010.B	2 counters, 24 V input voltage
UZB 2VR/5V	R4.315.0040.B	2 counters, 5 V input voltage

Module	Order No.	Comment
Analogue and temperature modules		
UAK12E/4A	R4.315.0230.0	12 inputs 0..10 V, 4 outputs ±10 V
USA 8/1	R4.315.0090.F	Analogue processor, 8 Slots for USA modules
USA E1/1	R4.315.0100.0	1 voltage input
USA E1/2.1	R4.315.0120.0	1 current input
USA E1/6	R4.315.0140.0	1 resistance temperature measurement Pt100
USA E1/7	R4.315.0150.0	1 thermo-element input Fe-CuNi
USA A1/1	R4.315.0110.B	1 voltage output
USA A1/2	R4.315.0130.0	1 current output
UST 2	R4.315.0170.0	Temperature module, 8 inputs
UST 21	R4.315.0180.0	Temperature module, 8 inputs, adaptive control
Positioning modules		
USP 200S	R4.315.0300.0	Sercos master, 1 ring, 8 axes, kinematics funct.
USP 400S	R4.315.0330.0	Sercos master, 2 rings, 16 axes.
USP 2I	R4.315.0020.0	Positioning processor, 2 axes, incremental encoder
USP 2A	R4.315.0030.0	Positioning processor, 2 axes, absolute encoder SSI
UPI 2 DIA	R4.318.0180.B	Positioning interface, 2 axes
UPI 3 DIA	R4.318.0160.B	Positioning interface, 3 axes
UPM 3I	R4.315.0080.B	Position detection, 3 channels, incremental encoder
UPM 4A	R4.315.0060.C	Position detection, 4 channels, absolute encoder
UPM 4U	R4.315.0310.C	Position detection, 4 channels, ultrasound encoder
Multifunktion modules		
XSF 05	R4.315.0340.0	8 I/O 24 V DC, 14 I/O 5 V DC
XSF 24	R4.315.0350.0	22 I/O 24 V DC
XSL 05	R4.315.0360.0	Laser Control, 8 I/O 24 V DC, 14 I/O 5 V DC
XSL 24	R4.315.0370.0	Laser Control, 22 I/O 24 V DC
Communication modules		
USK DIM	R4.318.0170.0	Interbus-S master
USK DPM	R4.318.0370.0	PROFIBUS-DP master
USK DPS	R4.318.0360.0	Profibus-DP slave
Accessories		
UBT LA	R4.318.0120.0	Empty slot covers for UBT
UKK 24	R4.318.0020.0	Cable UKZ ↔ UKE, without power supply
UKK 24V	R4.318.0060.0	Cable UKZ ↔ UKE, with power supply
UNB 115/230	R4.318.0050.0	Buffer battery for UNG 230A/115A
UNB 24	R4.318.0130.0	Buffer battery for UNG 24
UST	R4.315.0160.F	Temperature control (spare part for UST 2 / UST 21)

x* = Use only as expansion rack with basic rack XBT
x** = Use only as expansion rack

Table 12: List of available racks, power supply units and expansion modules for XCx and Promodul-U

6 Commissioning

The XCx startup described in this section can be carried out without in-depth knowledge. The startup steps must be followed precisely and the specified conditions must be observed (e.g. I/O configuration).

Note

The screenshots shown for the software installation and startup examples in the following chapters are only examples. Version numbers of software or unit designations may differ from the current version.

6.1 Installation of MULTIPROG, OPC Server and Add-Ons

Important!

The entire programming software consists of the MULTIPROG software components, OPC server, add-ons for MULTIPROG and the Schleicher dialog.

All software components must be installed one after the other in this order before startup.

The programming software can be installed on an external PC or even on the controller. The installation on an external PC will be described.

If it is necessary, two CDs are supplied with the controller:

Table 13: Contents of the CD

Name	Contents
MULTIPROG	• Programming software MULTIPROG • OPC server
Service Pack	• Controller software for all Schleicher control units • Add-ons • Schleicher Dialog • Other tools such as documentation and service information

6.2 System requirements

Observe the following system requirements for installation and operation of the software:

Table 14: System requirements

Windows PC	Pentium 4, 2 GHz [1]
RAM	512 MB [1]
Hard disk	250 MB free Memory [1]
Monitor	1024 x 768 (True Color) [1]
Interfaces	TCP/IP or RS232
Mouse	
PC operating system	Microsoft Windows XP SP3 Microsoft Windows Vista SP2 Microsoft Windows 7 (32 or 64 Bit) Microsoft Windows 8 Microsoft NET Framework 3.5 [2] Microsoft Visual C++ 2005 [3]Redistributables and Microsoft Visual C++ 2008 [3]Redistributables

[1] Minimum requirement

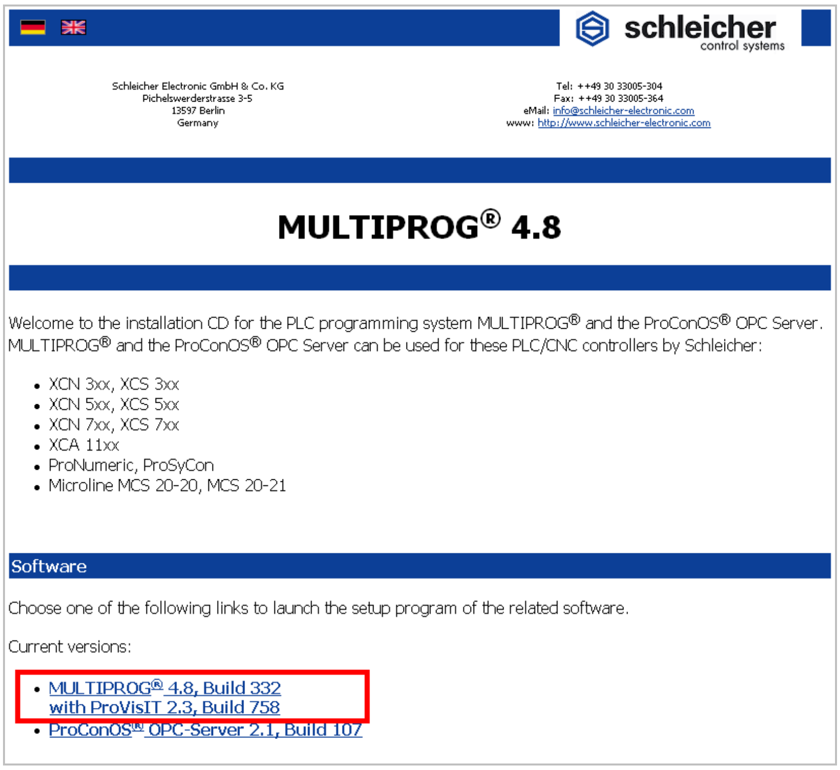
[2] Microsoft .NET Framework 3.5 is not included.

[3] Microsoft Visual C++ 2005 Redistributables und Microsoft Visual C++ 2008 Redistributables are included.

6.2.1 MULTIPROG Installation

Insert the MULTIPROG CD in the drive of the PC. The AutoRun function on the CD starts Internet Explorer. Now select *MULTIPROG* (version 4.0 here) and start installation (→ Figure 7).

Figure 7:
Installation of MULTIPROG



schleicher
control systems

Schleicher Electronic GmbH & Co. KG
Pichelswerderstrasse 3-5
13597 Berlin
Germany

Tel: ++49 30 33005-304
Fax: ++49 30 33005-364
eMail: info@schleicher-electronic.com
www: http://www.schleicher-electronic.com

MULTIPROG® 4.8

Welcome to the installation CD for the PLC programming system MULTIPROG® and the ProConOS® OPC Server. MULTIPROG® and the ProConOS® OPC Server can be used for these PLC/CNC controllers by Schleicher:

- XCN 3xx, XCS 3xx
- XCN 5xx, XCS 5xx
- XCN 7xx, XCS 7xx
- XCA 11xx
- ProNumeric, ProSyCon
- Microline MCS 20-20, MCS 20-21

Software

Choose one of the following links to launch the setup program of the related software.

Current versions:

- MULTIPROG® 4.8, Build 332 with ProVisIT 2.3, Build 758
- ProConOS® OPC-Server 2.1, Build 107

Note

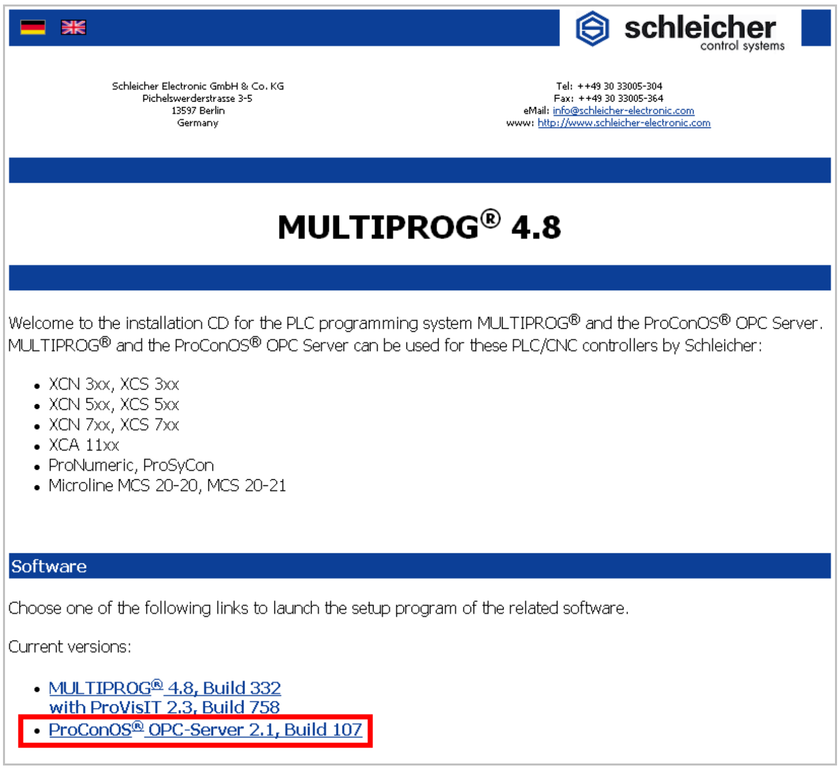
If a version of MULTIPROG below version 4 is already installed, the installed version must not be overwritten if the old projects are needed for further work.

MULTIPROG must then be installed on a new path. All other installation settings can remain unchanged.

You are prompted to restart your computer when installation is complete. A computer restart is not yet required if you want to now install the ProConOS OPC server.

6.2.2 OPC Server Installation

To install the OPC server in Internet Explorer, select the *ProConOS OPC server* and start installation (→ Figure 8).



Schleicher Electronic GmbH & Co. KG
Pichelswerderstrasse 3-5
13597 Berlin
Germany

Tel: ++49 30 33005-304
Fax: ++49 30 33005-364
eMail: info@schleicher-electronic.com
www: http://www.schleicher-electronic.com

MULTIPROG® 4.8

Welcome to the installation CD for the PLC programming system MULTIPROG® and the ProConOS® OPC Server. MULTIPROG® and the ProConOS® OPC Server can be used for these PLC/CNC controllers by Schleicher:

- XCN 3xx, XCS 3xx
- XCN 5xx, XCS 5xx
- XCN 7xx, XCS 7xx
- XCA 11xx
- ProNumeric, ProSyCon
- Microline MCS 20-20, MCS 20-21

Software

Choose one of the following links to launch the setup program of the related software.

Current versions:

- MULTIPROG® 4.8, Build 332
  with ProVisIT 2.3, Build 758
- ProConOS® OPC-Server 2.1, Build 107

Figure 8:
Installation of OPC server

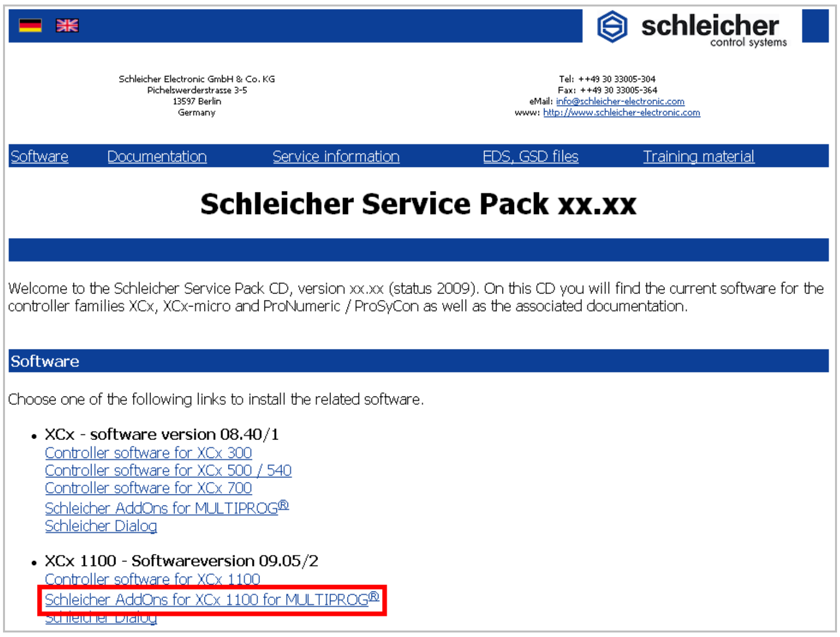
The OPC server should be installed in the MULTIPROG directory. All other installation settings can remain unchanged. A PC restart is required after installation.

6.2.3 Add-ons Installation

The add-ons for MULTIPROG must be installed in the next step.

Insert the *Service Pack* CD. The AutoRun function on the CD starts Internet Explorer. Now select *Add-ons for MULTIPROG* under the category for the existing controller and start installation (→ Figure 9).

Figure 9: Installation of AddOns



schleicher control systems

Schleicher Electronic GmbH & Co. KG
Pichelswerderstrasse 3-5
13597 Berlin
Germany

Tel: ++49 30 33005-304
Fax: ++49 30 33005-364
eMail: info@schleicher-electronic.com
www: http://www.schleicher-electronic.com

Software Documentation Service information EDS, GSD files Training material

Schleicher Service Pack xx.xx

Welcome to the Schleicher Service Pack CD, version xx.xx (status 2009). On this CD you will find the current software for the controller families XCx, XCx-micro and ProNumeric / ProSyCon as well as the associated documentation.

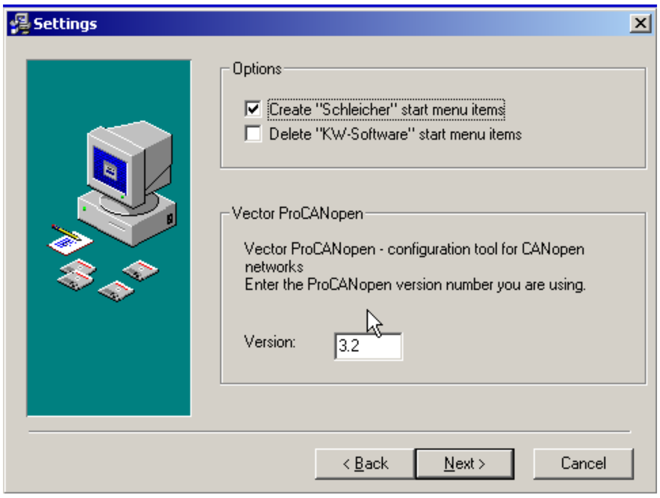
Software

Choose one of the following links to install the related software.

- XCx - software version 08.40/1
  Controller software for XCx 300
  Controller software for XCx 500 / 540
  Controller software for XCx 700
  Schleicher AddOns for MULTIPROG®
  Schleicher Dialog
- XCx 1100 - Softwareversion 09.05/2
  Controller software for XCx 1100
  Schleicher AddOns for XCx 1100 for MULTIPROG®
  Schleicher Dialog

During installation, the ProCANopen version (3.2 here) currently used by the user must be entered (→ Figure 10). You will need ProCANopen later for starting up the CANopen network. If ProCANopen is not used, you can adopt the default settings.

Figure 10: AddOn installation, entry of ProCANopen version



Settings

Options
☑ Create "Schleicher" start menu items
☐ Delete "KW-Software" start menu items

Vector ProCANopen
Vector ProCANopen - configuration tool for CANopen networks
Enter the ProCANopen version number you are using.

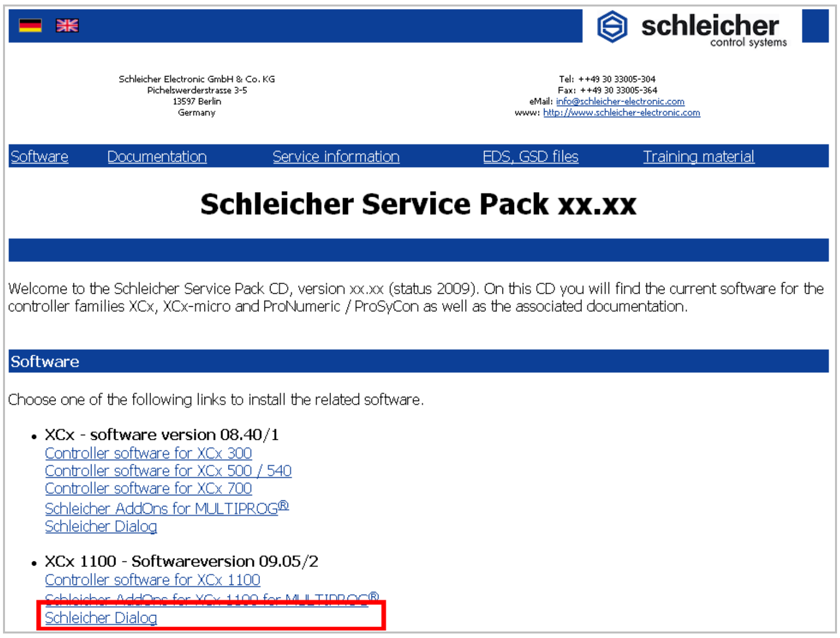
Version: 3.2

The PC does not have to be restarted after installation.

6.2.4 Schleicher Dialog Installation

The Schleicher dialog user interface is now installed. Select the category for the existing controller in *Internet Explorer* and start (→ Figure 11).

Figure 11: Installation of Schleicher Dialog



Schleicher Electronic GmbH & Co. KG
Pichelswerderstrasse 3-5
13597 Berlin
Germany

Tel: ++49 30 33005-304
Fax: ++49 30 33005-364
eMail: info@schleicher-electronic.com
www: http://www.schleicher-electronic.com

Software Documentation Service information EDS, GSD files Training material

Schleicher Service Pack xx.xx

Welcome to the Schleicher Service Pack CD, version xx.xx (status 2009). On this CD you will find the current software for the controller families XCx, XCx-micro and ProNumeric / ProSyCon as well as the associated documentation.

Software

Choose one of the following links to install the related software.

- XCx - software version 08.40/1
  Controller software for XCx 300
  Controller software for XCx 500 / 540
  Controller software for XCx 700
  Schleicher AddOns for MULTIPROG®
  Schleicher Dialog
- XCx 1100 - Softwareversion 09.05/2
  Controller software for XCx 1100
  Schleicher AddOns for XCx 1100 for MULTIPROG®
  Schleicher Dialog

6.3 Starting up the Network Interface

Important!

The network-specific characteristics and the procedure are to be clarified with the network administrator for your in-house network.

All the identifiers and addresses stated or specified in the following installation information are examples and must be adapted to your local circumstances.

All examples of these instructions refer to Windows XP. The procedure for other operating systems may differ from those described here.

The information given here is non-binding!

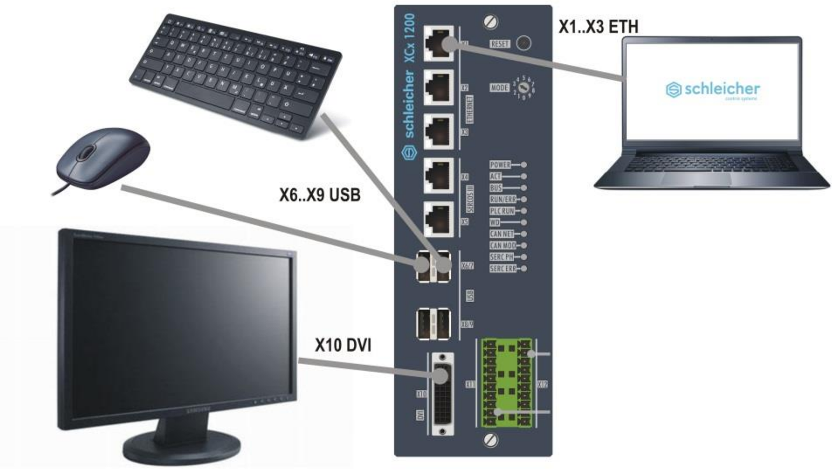
6.3.1 Preparation

Connect the following devices to the controller to prepare for the network interface startup (→ Figure12, left):

- A digital monitor directly (or an analogue monitor via a VGA-DVI adapter) to the DVI interface X10;
- A keyboard to one of the USB interfaces X6..X9;
- A mouse to one of the USB interfaces X6..X9.

The operating mode switch can be at any position. Start the controller by applying the operating voltage to the power supply unit.

Figure12: Connecting the input/output devices to the XCx 1100



An alternative to the direct input is to start up using the remote control software VNC (→ page 127) via an Ethernet connection (→ Figure12, right). Use the following IP addresses in the PC for the initial connection with VNC.

6.3.2 Communication connections

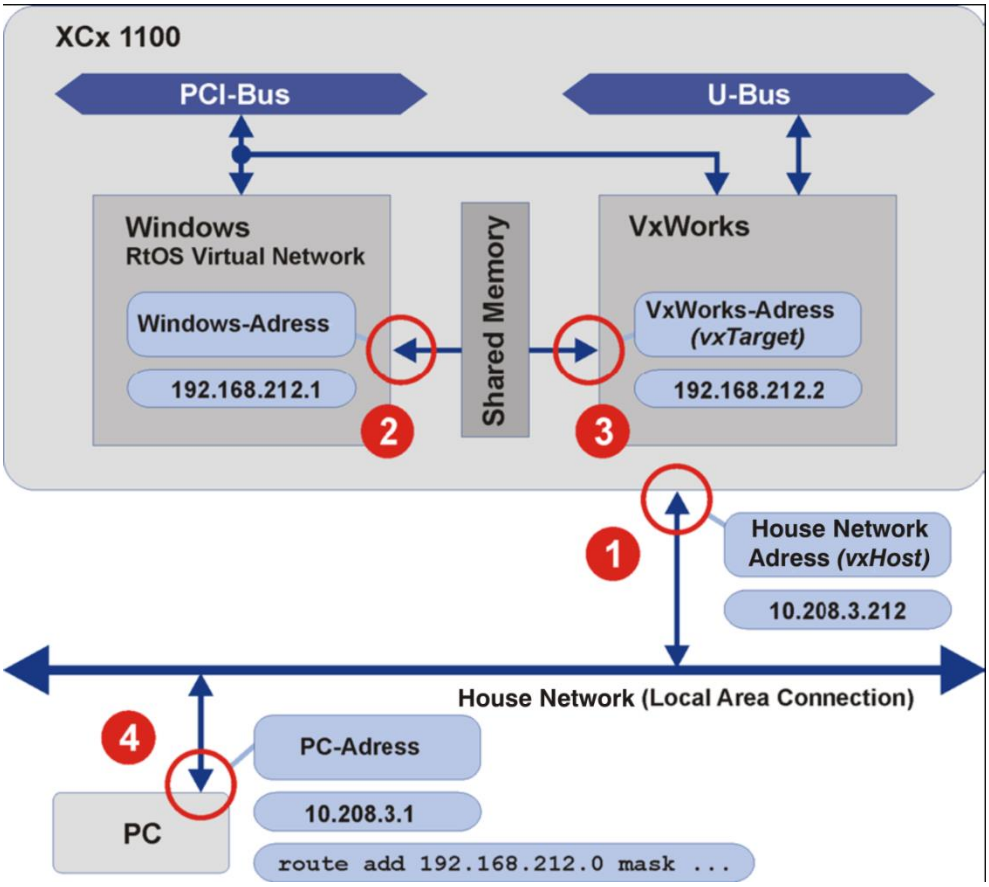


Figure 13: Start up the network interfaces in four steps

The XCx 1100 contains the VxWorks real-time operating system and a Windows component (XP embedded). Both communicate with each other via a common memory (shared memory). The expansion modules and the required supply voltages (power supply unit) are connected via the U-bus. The PCI bus is used for quick extensions in the future.

Startup of the network interfaces is described in four steps in the following chapter (→ Figure 13):

(1) Assignment of an **in-house network address** for connecting the controller to an existing in-house network; the specified IP address is an example, the alias name (vxHost) is specified by the manufacturer.

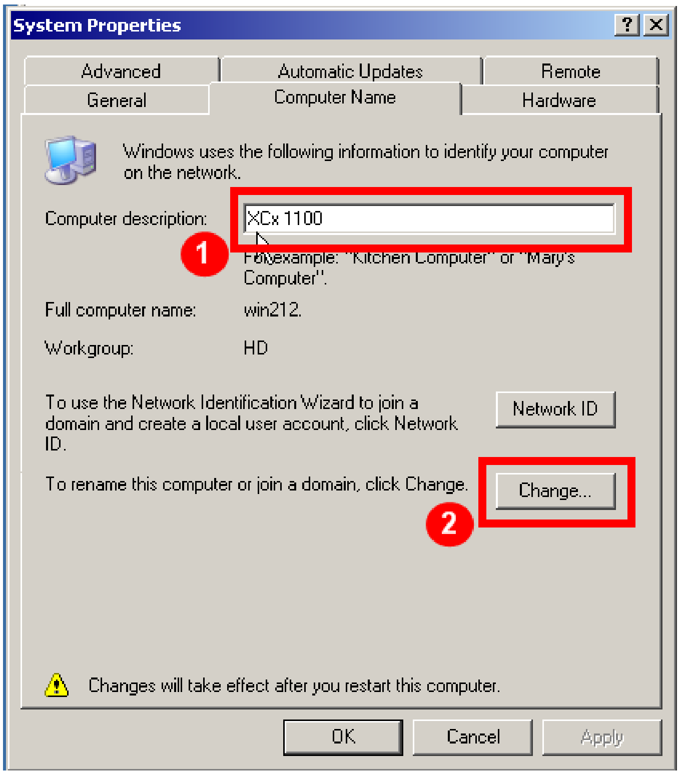
(2) Change the **Windows address**; the specified address (192.168.212.1) should be retained if possible.

(3) Change the **VxWorks address**; the specified address (192.168.212.2) should be retained if possible; the alias name (vxTarget) is specified by the manufacturer.

(4) Set up a **PC address** for communication with the controller via the in-house network (programming with MULTIPROG).

6.3.3 Specification of the Computer Name for XCx

In the *System Properties* dialog window (under *Start / Control Panel / System / Computer Name)*, enter "XCx 1100" as the *Computer Description* (1) (→ Figure 14).



Windows uses the following information to identify your computer on the network.

Computer description: XCx 1100

For example: "Kitchen Computer" or "Mary's Computer".

Full computer name: win212.

Workgroup: HD

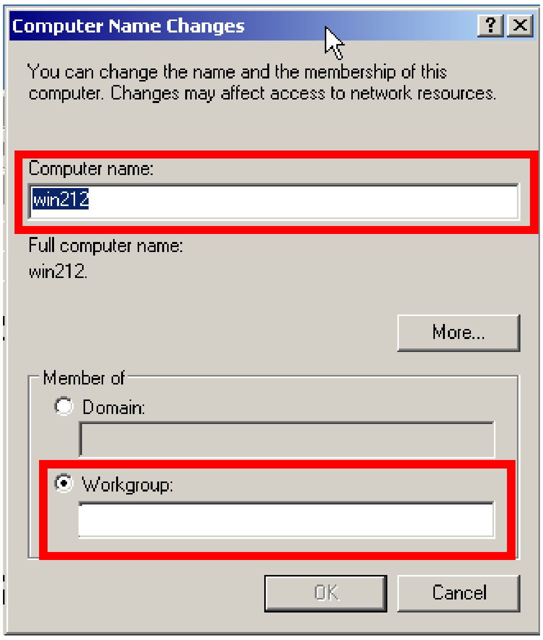
To use the Network Identification Wizard to join a domain and create a local user account, click Network ID.

To rename this computer or join a domain, click Change.

Changes will take effect after you restart this computer.

Figure 14: "System properties" dialog window, enter the computer description

Click the *Change* button (2) to access the *Change computer name* dialog window where you can enter a computer name and work group of you choice (→ Figure 15). The "win212" name is used in the following examples; retain this for the sake of clarity. Ask your network administrator if required.



You can change the name and the membership of this computer. Changes may affect access to network resources.

Computer name: win212

Full computer name: win212.

Member of

○ Domain:

● Workgroup:

Figure 15: "Change computer name " dialog window

6.3.4 Adaptation of the XCx-TCP/IP Settings

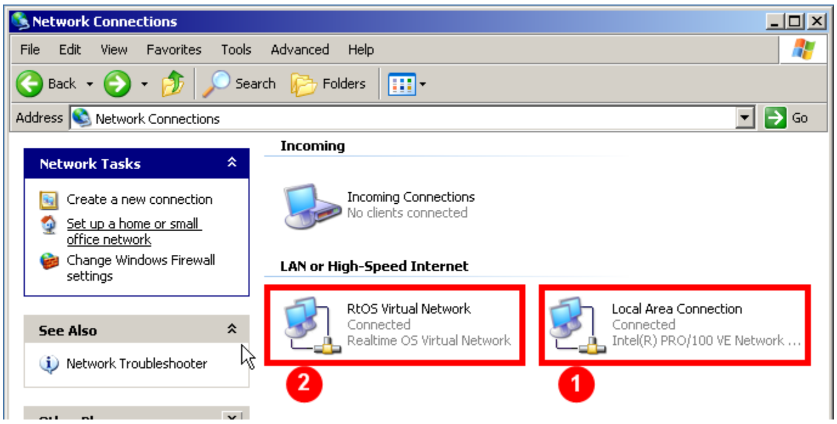
Important!

First clarify the adaptation of the IP addresses with your network administrator for your in-house network.

The information given here is non-binding!

Open *Network Connections* via *Start / Control Panel* to make the necessary TCP/IP settings for the network card (→ Figure 16).

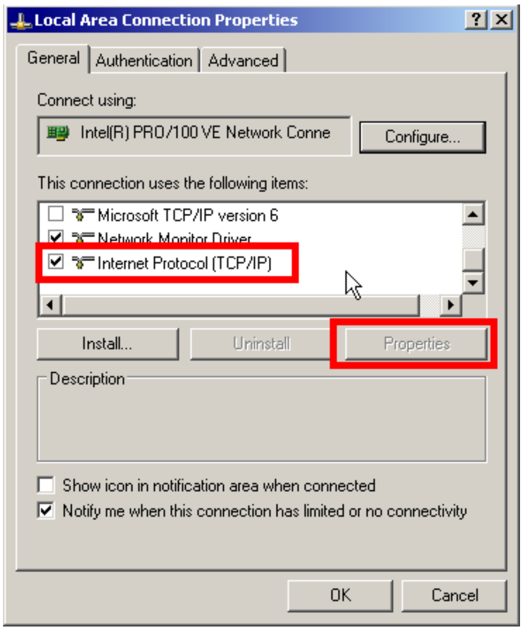
Figure 16: Select the network connections



IP addresses for Local Area Connection (step 1)

Double-click *Local Area Connection* (1) to open the associated Properties window. Now select *Internet Protocol (TCP/IP)* and click the *Properties* button (→ Figure 17).

Figure 17: Properties of Local Area Connection



Local Area Connection Properties

Connect using:

Intel(R) PRO/100 VE Network Conne

This connection uses the following items:

- □ Microsoft TCP/IP version 6
- ☑ Network Monitor Driver
- ☑ Internet Protocol (TCP/IP)

Description

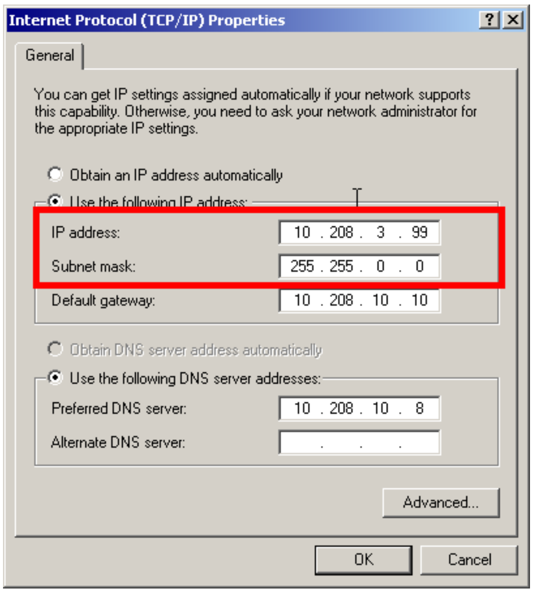
- □ Show icon in notification area when connected
- ☑ Notify me when this connection has limited or no connectivity

Enter the following values in the next *Properties of Internet Protocol (TCP/IP)* window (→ Figure 18):

- IP address: 10.208.3.212 (default)
- Subnetwork mask: 255.255.0.0 (default)

The rest of the fields can remain unchanged.

Figure 18: Enter the IP address and subnetwork mask for the Local Area Connection



Internet Protocol (TCP/IP) Properties

You can get IP settings assigned automatically if your network supports this capability. Otherwise, you need to ask your network administrator for the appropriate IP settings.

○ Obtain an IP address automatically

● Use the following IP address:

IP address: 10 . 208 . 3 . 99

Subnet mask: 255 . 255 . 0 . 0

Default gateway: 10 . 208 . 10 . 10

○ Obtain DNS server address automatically

● Use the following DNS server addresses:

Preferred DNS server: 10 . 208 . 10 . 8

Alternate DNS server: . . .

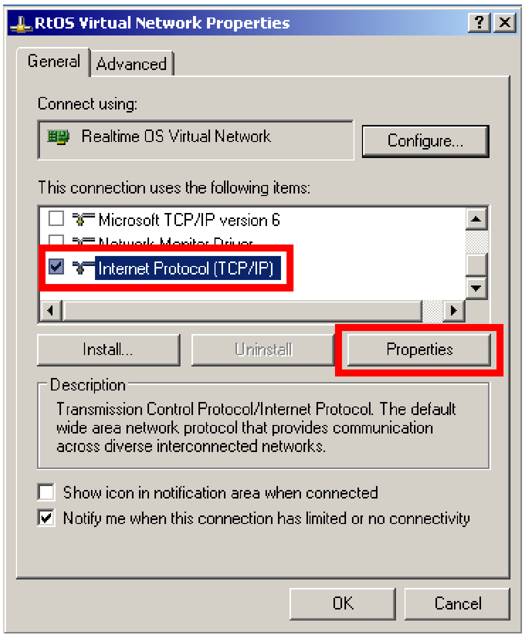
Important!

First clarify the adaptation of IP addresses with your network administrator for installation with a connection to an existing in-house network.

IP addresses for RtOS Virtual Network (step 2)

Double-click the entry of the *same name (2)* in Network Connections to make the IP settings2 for the RtOS Virtual Network (→ Figure 16) and open the Properties window. Now select *Internet Protocol (TCP/IP)* and click the *Properties* button (→ Figure 19).

Figure 19: Properties of RtOS Virtual Network



RtOS Virtual Network Properties

General | Advanced

Connect using:

Realtime OS Virtual Network [Configure...]

This connection uses the following items:

- □ Microsoft TCP/IP version 6
- □ Network Monitor Driver
- ☑ Internet Protocol (TCP/IP)

[Install...] [Uninstall] [Properties]

Description

Transmission Control Protocol/Internet Protocol. The default wide area network protocol that provides communication across diverse interconnected networks.

□ Show icon in notification area when connected

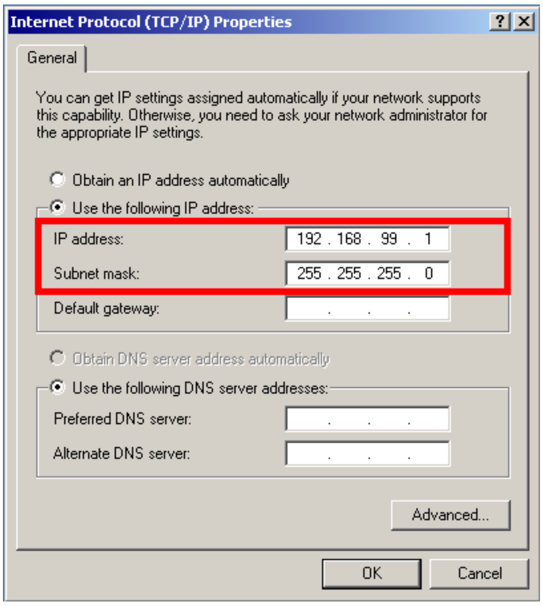
☑ Notify me when this connection has limited or no connectivity

[OK] [Cancel]

Enter the following values in the next *Properties of Internet Protocol (TCP/IP)* window (→ Figure: 20):

• IP address:	192.168.212.1
• Subnet mask:	255.255.255.0
• Standard gateway:	empty
• DNS server addresses:	empty

Figure: 20: Enter the IP address and subnetwork mask for the RtOS Virtual Network



Internet Protocol (TCP/IP) Properties

General

You can get IP settings assigned automatically if your network supports this capability. Otherwise, you need to ask your network administrator for the appropriate IP settings.

○ Obtain an IP address automatically

● Use the following IP address:

IP address: 192 . 168 . 99 . 1

Subnet mask: 255 . 255 . 255 . 0

Default gateway: . . .

○ Obtain DNS server address automatically

● Use the following DNS server addresses:

Preferred DNS server: . . .

Alternate DNS server: . . .

[Advanced...]

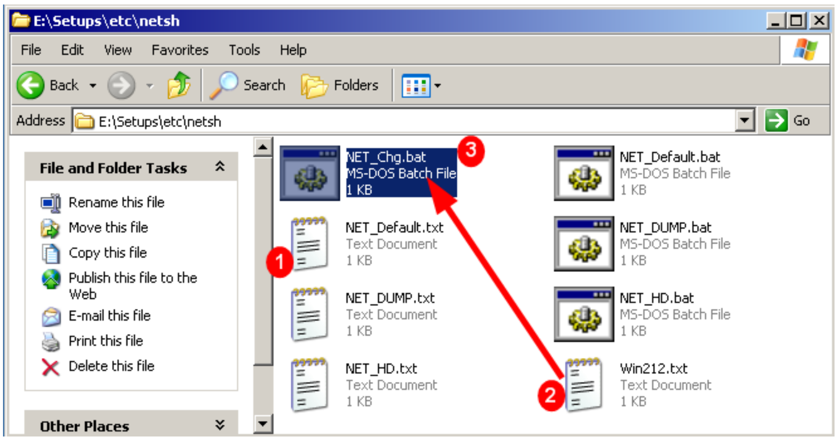
[OK] [Cancel]

Change IP addresses using text form

The IP addresses can be changed in a text form as an alternative to the previous description.

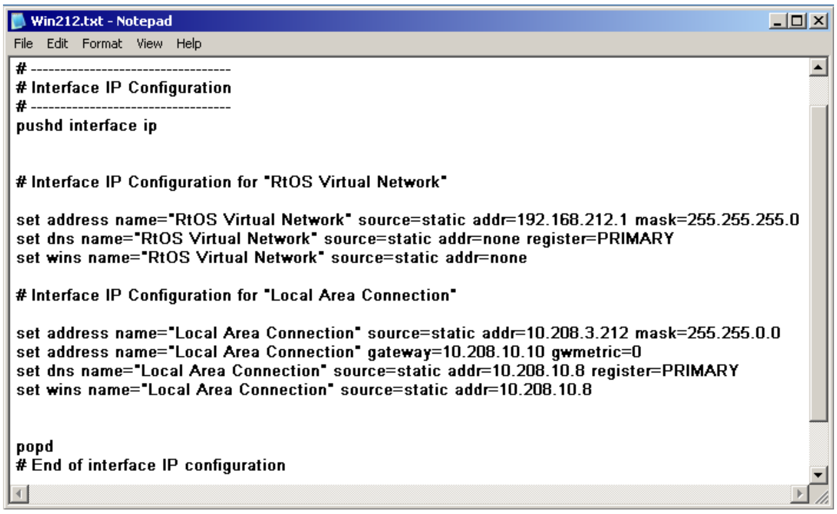
The *NET_Default.txt* IN THE *E:\Setups\etc\netsh* (1) directory can be used as a template (→ Figure 21).

Figure 21: Change and set the IP addresses using text form


E:\Setups\etc\netsh
File Edit View Favorites Tools Help
Back Search Folders
Address E:\Setups\etc\netsh Go
File and Folder Tasks
Rename this file
Move this file
Copy this file
Publish this file to the Web
E-mail this file
Print this file
Delete this file
Other Places
NET_Chg.bat MS-DOS Batch File 1 KB
NET_Default.bat MS-DOS Batch File 1 KB
NET_Default.txt Text Document 1 KB
NET_DUMP.bat MS-DOS Batch File 1 KB
NET_DUMP.txt Text Document 1 KB
NET_HD.bat MS-DOS Batch File 1 KB
NET_HD.txt Text Document 1 KB
Win212.txt Text Document 1 KB
1
2
3

The existing IP addresses in this text file are replaced by the required IP addresses and saved as a new text file (e.g. *Win212.txt*) using *Save as* (→ Figure 22).

Figure 22: Enter the IP addresses in the text form



```
# ----------------------------------
# Interface IP Configuration
# ----------------------------------
pushd interface ip


# Interface IP Configuration for "RtOS Virtual Network"

set address name="RtOS Virtual Network" source=static addr=192.168.212.1 mask=255.255.255.0
set dns name="RtOS Virtual Network" source=static addr=none register=PRIMARY
set wins name="RtOS Virtual Network" source=static addr=none

# Interface IP Configuration for "Local Area Connection"

set address name="Local Area Connection" source=static addr=10.208.3.212 mask=255.255.0.0
set address name="Local Area Connection" gateway=10.208.10.10 gwmetric=0
set dns name="Local Area Connection" source=static addr=10.208.10.8 register=PRIMARY
set wins name="Local Area Connection" source=static addr=10.208.10.8


popd
# End of interface IP configuration
```

To make the change, use the mouse to drag the new text file (*Win212.txt* (2) in the example) onto the *NET_Chg.bat* (3) file (→ Figure 21). The batch file adds the IP addresses entered into the corresponding files.

6.3.5 Communication for programming with MULTIPROG (step 3)

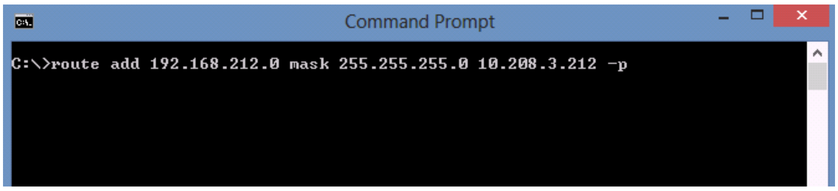
A suitable network route must first be set up for communication with an external PC for programming with MULTIPROG via the in-house network.

The parameters are entered under *Start / Programs /Accessories /Prompt* (→ Figure 23).

Example:

```
route add 192.168.212.0 mask 255.255.255.0
10.208.3.212 -p
```

(-p for permanent), apply with <Enter>.



```
C:\>route add 192.168.212.0 mask 255.255.255.0 10.208.3.212 -p
```

Figure 23:
Set up network route for programming with MULTIPROG

Use the PING command to perform a test to see whether the communication connection is working (the XCx 1100 must be connected and started):

`ping 192.168.212.2` *<Enter>*

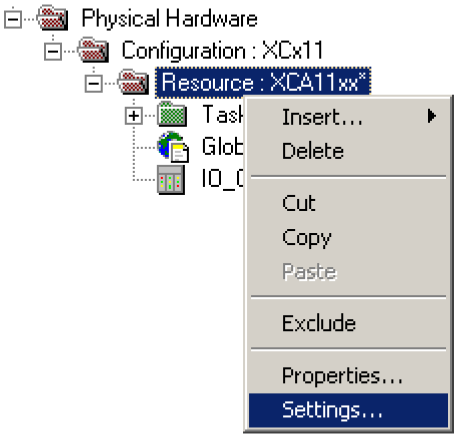
Display for correct connection:

```
Response from 192.168.212.2: Bytes=32 Time<1ms TTL=63
Response from 192.168.212.2: Bytes=32 Time<1ms TTL=63
Response from 192.168.212.2: Bytes=32 Time<1ms TTL=63
Response from 192.168.212.2: Bytes=32 Time=1ms TTL=63

Ping statistics for 192.168.99.2:
    Packages: Sent = 4, Received = 4, Lost = 0
    (0% Loss),

Approx. time in milliseconds:
    Minimum = 0ms, Maximum = 1ms, Mean = 0ms
```

MULTIPROG can now be started on the external PC. Select *New Project / XCA11xx* (or open an existing project with the current controller). A more detailed description of the procedure is given in the Section "First Steps with MULTIPROG" on page 45.

In the PLC project, right-click on the entry *Resource : XCx11* and select *Settings* in the context menu
(→ Figure 24).

Figure 24:
Call the resource settings in MULTIPROG

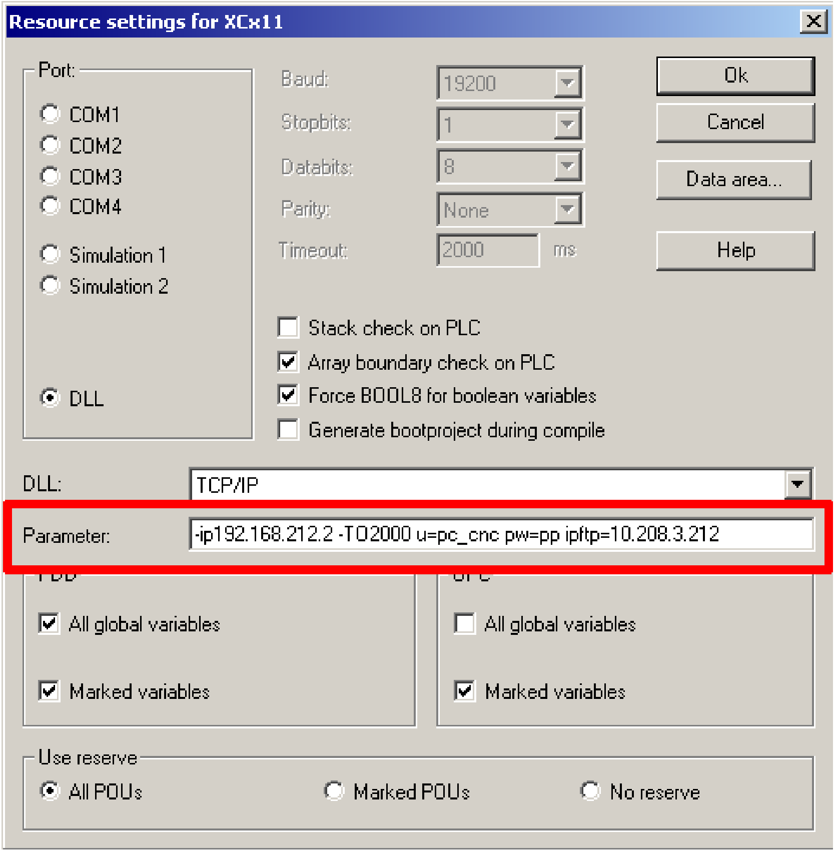


The IP address is specified in the *Resource settings for XCx11* dialog window (→ Figure 25).

Example (if all default settings are adopted):

```
-ip192.168.212.2 -TO2000 u=pc_cnc pw=pp
ipftp=10.208.3.212
```

Figure 25:
Enter the IP address in MULTIPROG



Resource settings for XCx11

Port:
○ COM1
○ COM2
○ COM3
○ COM4
○ Simulation 1
○ Simulation 2
● DLL

Baud: 19200
Stopbits: 1
Databits: 8
Parity: None
Timeout: 2000 ms

□ Stack check on PLC
☑ Array boundary check on PLC
☑ Force BOOL8 for boolean variables
□ Generate bootproject during compile

DLL: TCP/IP
Parameter: -ip192.168.212.2 -TO2000 u=pc_cnc pw=pp ipftp=10.208.3.212

PDD
☑ All global variables
☑ Marked variables

OPC
□ All global variables
☑ Marked variables

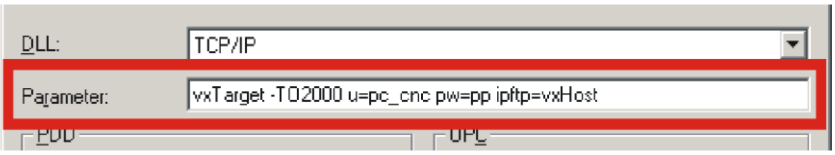
Use reserve
● All POUs ○ Marked POUs ○ No reserve

The IP address can also be specified with the alias addresses (→ Figure 26).

Example:

```
vxTarget -TO2000 u=pc_cnc pw=pp ipftp=vxHost
```

Figure 26: Enter alias IP address in MULTIPROG

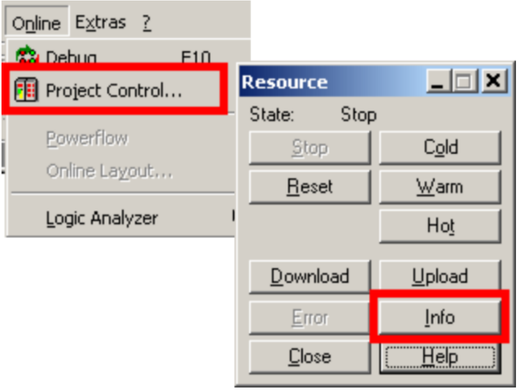


DLL: TCP/IP

Parameter: vxTarget -TO2000 u=pc_cnc pw=pp ipftp=vxHost

After confirming with *OK*, the connection to the XCx 1100 is called via *Online / Project Control* and then the *Info* button in the *Resource* window (→ Figure 27).

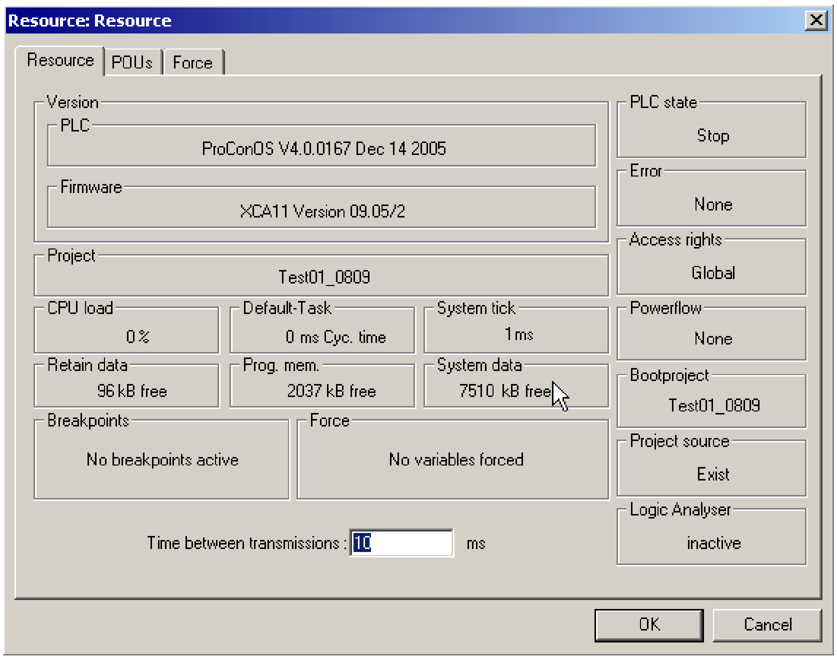
Figure 27: Call the connection for XCx 1100 in MULTIPROG



The ´Resource:Resource´ window is displayed when a successful connection is established (→ Figure 28). *Timeout* is reported for a faulty connection. In this case, check all settings again or consult your network administrator.

Other settings may have to be considered for the subnetwork determination for gateways.

Figure 28: Resource window for online connection for XCx 1100



Resource: Resource

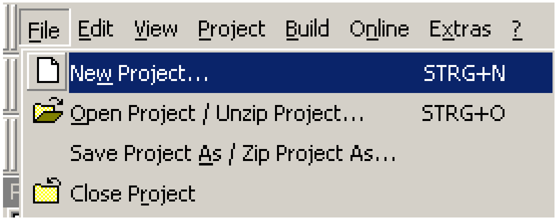
Version
PLC: ProConOS V4.0.0167 Dec 14 2005
Firmware: XCA11 Version 09.05/2
Project: Test01_0809
CPU load: 0 %
Default-Task: 0 ms Cyc. time
System tick: 1 ms
Retain data: 96 kB free
Prog. mem.: 2037 kB free
System data: 7510 kB free
Breakpoints: No breakpoints active
Force: No variables forced
Time between transmissions : 10 ms
PLC state: Stop
Error: None
Access rights: Global
Powerflow: None
Bootproject: Test01_0809
Project source: Exist
Logic Analyser: inactive

6.4 First Steps with MULTIPROG

6.4.1 Start MULTIPROG, Open and Save a New Project

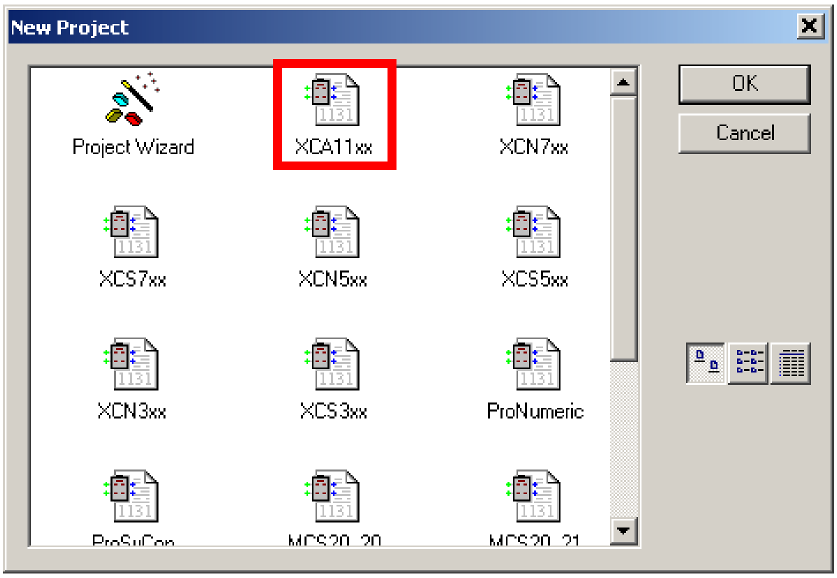
Start MULTIPROG, select *File / New Project* (→ Figure 29).

Figure 29: Open new project in MULTIPROG


File
Edit
View
Project
Build
Online
Extras
?
New Project... STRG+N
Open Project / Unzip Project... STRG+O
Save Project As / Zip Project As...
Close Project

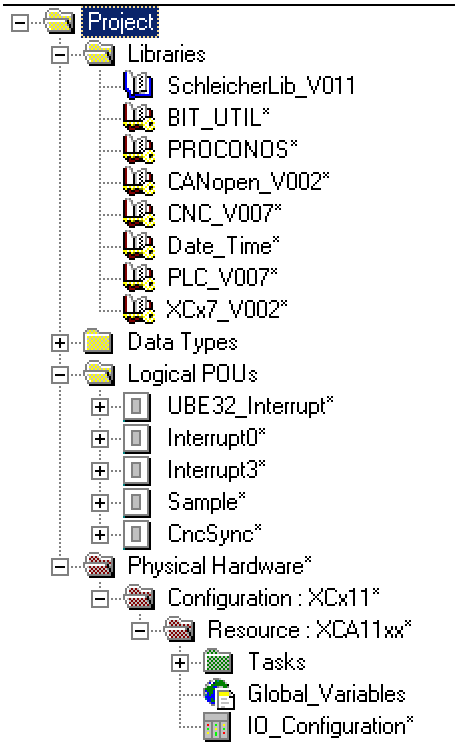
Select a project for the available controller type (XCA 11xx here) and click *OK* (→ Figure: 30).

Figure: 30: Select controller type for new project



If the project opened successfully, the project tree is displayed in the project window (→ Figure 31). There are already logical POUs contained that are fully functional and sufficient for a simple quick startup.

Figure 31:
Example for project tree in MULTIPROG



```
Project
  Libraries
    SchleicherLib_V011
    BIT_UTIL*
    PROCONOS*
    CANopen_V002*
    CNC_V007*
    Date_Time*
    PLC_V007*
    XCx7_V002*
  Data Types
  Logical POUs
    UBE32_Interrupt*
    Interrupt0*
    Interrupt3*
    Sample*
    CncSync*
  Physical Hardware*
    Configuration : XCx11*
      Resource : XCA11xx*
        Tasks
        Global_Variables
        IO_Configuration*
```

Save the project with a new name under *File / Save project as* (START here) (→ Figure 32 and Figure 33).

Figure 32:
Call memory dialog

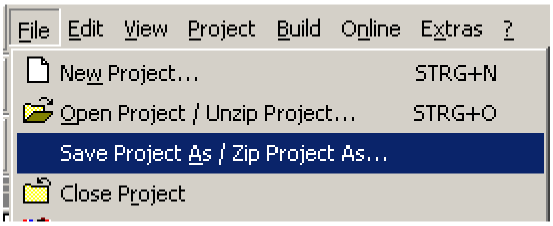
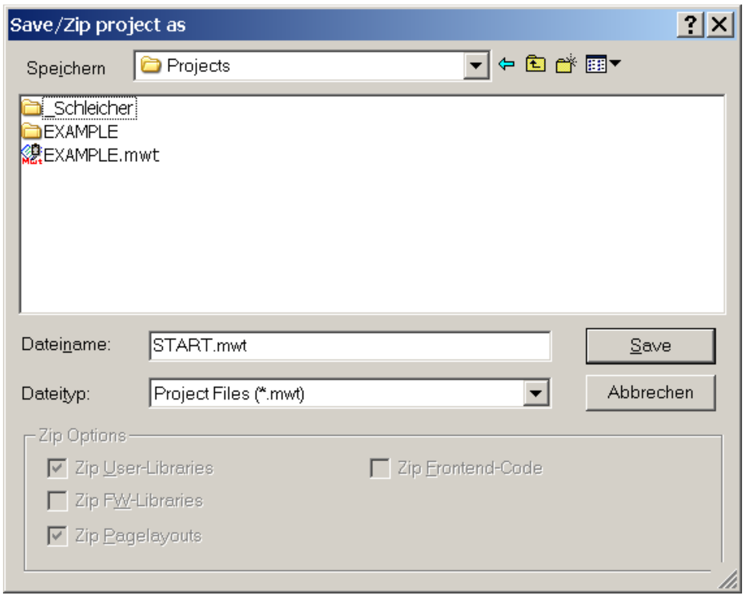

File
Edit
View
Project
Build
Online
Extras
?
New Project... STRG+N
Open Project / Unzip Project... STRG+O
Save Project As / Zip Project As...
Close Project

Figure 33:
"Save project" dialog window



Dateiname: START.mwt

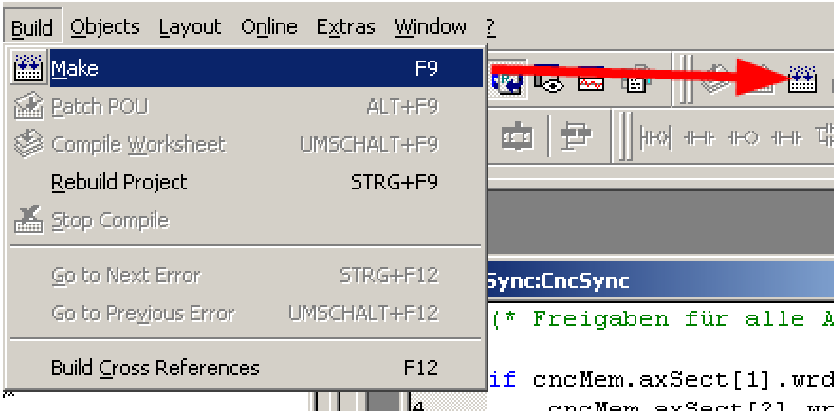
Dateityp: Project Files (*.mwt)

☑ Zip User-Libraries
☐ Zip Frontend-Code
☐ Zip FW-Libraries
☑ Zip Pagelayouts

6.4.2 Compiling a Project and Sending to the XCx

To compile a project, select *Code / Make* (→ Figure 34) (alternatively click <F9> or *Make* button, see arrow).

Figure 34: Compile project


Build
Objects
Layout
Online
Extras
Window
?
Make F9
Patch POU ALT+F9
Compile Worksheet UMSCHALT+F9
Rebuild Project STRG+F9
Stop Compile
Go to Next Error STRG+F12
Go to Previous Error UMSCHALT+F12
Build Cross References F12
Sync:CncSync

```
(* Freigaben für alle A
if cncMem.axSect[1].wrd
```

The progress of compiling is displayed in the message window below. Error messages, warnings and other information are displayed here. Click "Error", "Warning", etc., to display the message in more detail.

If errors are displayed, you can jump directly to the line of the PLC program that caused the error by double-clicking in the error line.

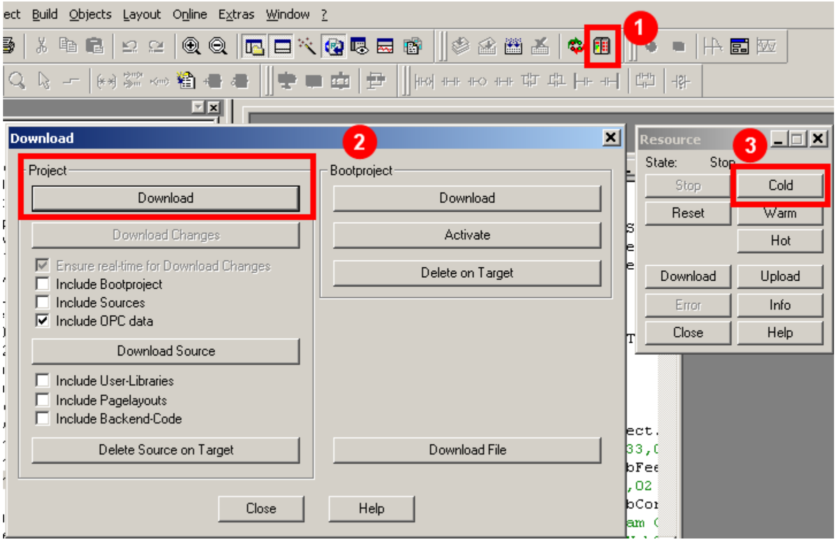
Send the project to the controller via the Ethernet connection (→ Figure 35):

(1) Click the *Project control dialog* button. Click the *Send* button in the *XCx* control dialog that opens.

(2) Under *Project*, select *Send* again to overwrite the existing project in the XCx.

(3) Use the *Cold* (cold start) button in the *Resource* control dialog to start the program on the XCx. The *PLC RUN* LED on the controller permanently lights up in green.

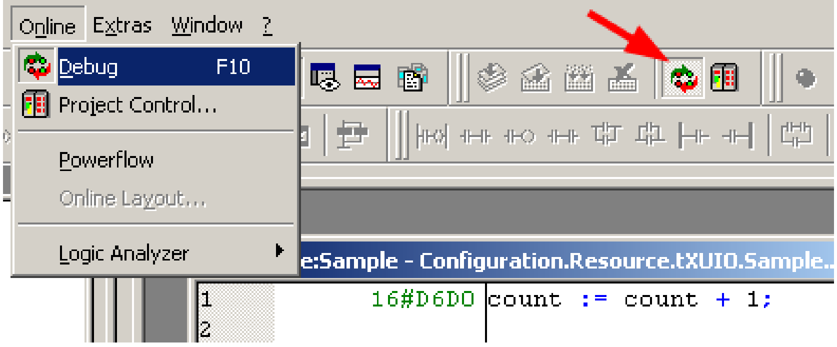
Figure 35: Transfer project to controller



☑ Ensure real-time for Download Changes
□ Include Bootproject
□ Include Sources
☑ Include OPC data
□ Include User-Libraries
□ Include Pagelayouts
□ Include Backend-Code

With the *Debug on/off* button (see arrow), you can display the content of the variables online on the worksheet (→ Figure 36).

Figure 36:
Online display of variables



```
1      16#D6D0 count := count + 1;
2
```

6.5 Insert the Shared RAM

To get full access to the the predefined variables, it is appropriate for startup to insert the shared RAM structure in the project.

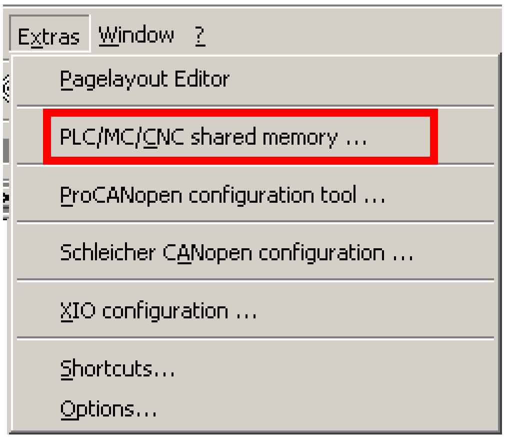
The shared RAM has a comprehensive data structure that is used for communication between the PLC, CNC and visualisation systems. Data areas such as the version number, error page and logbook can also be accessed via the shared RAM.

An introduction on the subject of shared RAM is given on page 119. A more detailed description of all variables of the shared RAM is available as online help in the "Schleicher dialog" software. This can also be called using the help menu of MULTIPROG.

The corresponding data types and variables must be inserted in the PLC project to enable access to the variables of the shared RAM. These are not yet included in the project templates of MULTIPROG. They must be inserted in the PLC project by the user with the aid of the shared RAM add-ons for MULTIPROG. It must be ensured that the user works with the shared RAM version that is suitable for the controller software.

Insert the shared RAM data types and variable in a PLC project via the *Extras / PLC/MC/CNC shared RAM menu* (→ Figure 37).

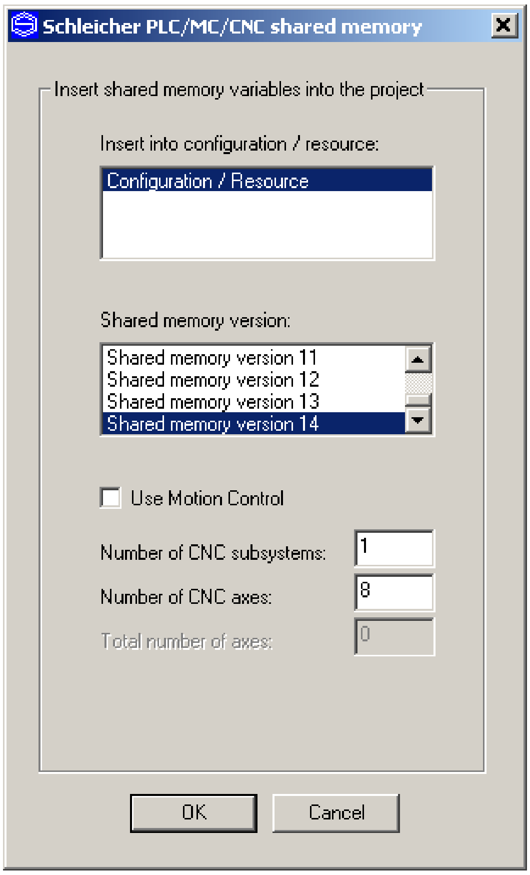
Figure 37: Call the "PLC/MC/CNC shared RAM" dialog window


Extras
Window
?
Pagelayout Editor
PLC/MC/CNC shared memory ...
ProCANopen configuration tool ...
Schleicher CANopen configuration ...
XIO configuration ...
Shortcuts...
Options...

A dialog window opens with the following selection and entry options (→ Figure 38).

- *Insert in configuration / Resource*:
  Selection of the resource of the PLC project where the shared RAM variables should be inserted.
- *Version of shared RAM*:
  Selection of the shared RAM version.
- *Number of CNC sub-systems / CNC axes*:
  Entry of the number of sub-systems and axes for CNC controllers (both these entry fields are deactivated for pure PLC controllers).
- *OK button*:
  Press OK to exit this dialog window and insert the shared RAM data types and variables in the PLC project.
- *Cancel button*:
  Click Cancel to exit the dialog window and the PLC project is not changed.

Figure 38: "PLC/MC/CNC shared RAM" dialog window



Schleicher PLC/MC/CNC shared memory

Insert shared memory variables into the project

Insert into configuration / resource:

Configuration / Resource

Shared memory version:

Shared memory version 11
Shared memory version 12
Shared memory version 13
Shared memory version 14

□ Use Motion Control

Number of CNC subsystems: 1

Number of CNC axes: 8

Total number of axes: 0

Important!

The selected shard RAM version must correspond to the controller operating system version (see "Information on the Selection of the Shared RAM Version", page 51.)

The PLC project must be recompiled and transferred to the controller after inserting the shared RAM data types and variables.

6.5.1 Access the Shared RAM

The PLC program has access to the entire shared RAM via the global variable `plcMem` (for PLC controllers) or `cncMem` (for CNC controllers). The individual components of the shared RAM can be accessed with the use of full stops. For example, the PLC program can read the version number of the controller operating software as follows: `cncMem.plcSect.lOSVersion`.

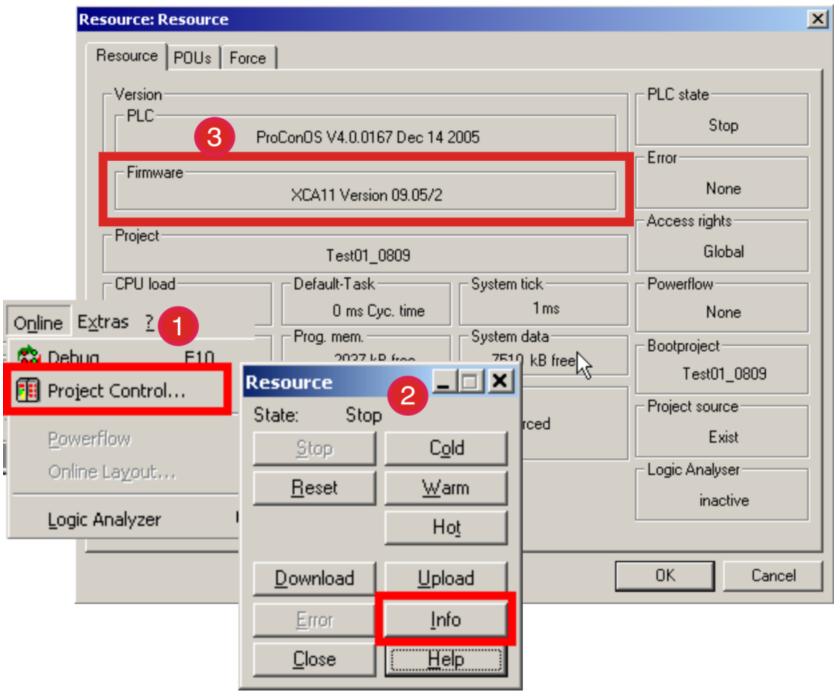
Visualisation systems have access to the shared RAM via the OPC interface. For example, the version number of the operating system can be read from the OPC variables `cmpS_lOSVersion`.

6.5.2 Information on the Selection of the Shared RAM Version

The shared RAM structure is updated and enhanced by Schleicher from time to time. A version number is used to distinguish between the individual versions. The version number is increased for large changes to the shared RAM structure where a change of variable addresses is required. For this reason, shared RAM versions with different version numbers are incompatible. Compatibility is only ensured for matching version numbers.

The latest shared RAM version must always be used. However, if the controller has old operating software, the older shared RAM version that is suitable must be used. The "XCx11xx Revision History" document has information on which shared RAM version can be used with which operating software version. The document is available on the Schleicher website at http://www.schleicher-electronic.comunder "XCx operating manuals" or can be supplied on request.

The version of the controller operating software can be displayed in the Info dialog window (3) for the corresponding resource in the PLC project (via (1) *Online / Project control*, then (2) *Resource / Info*) (→ Figure 39).



Resource: Resource

Version
PLC: ProConOS V4.0.0167 Dec 14 2005
Firmware: XCA11 Version 09.05/2
Project: Test01_0809
CPU load
Default-Task: 0 ms Cyc. time
System tick: 1 ms
Prog. mem.: 2027 kB free
System data: 7510 kB free

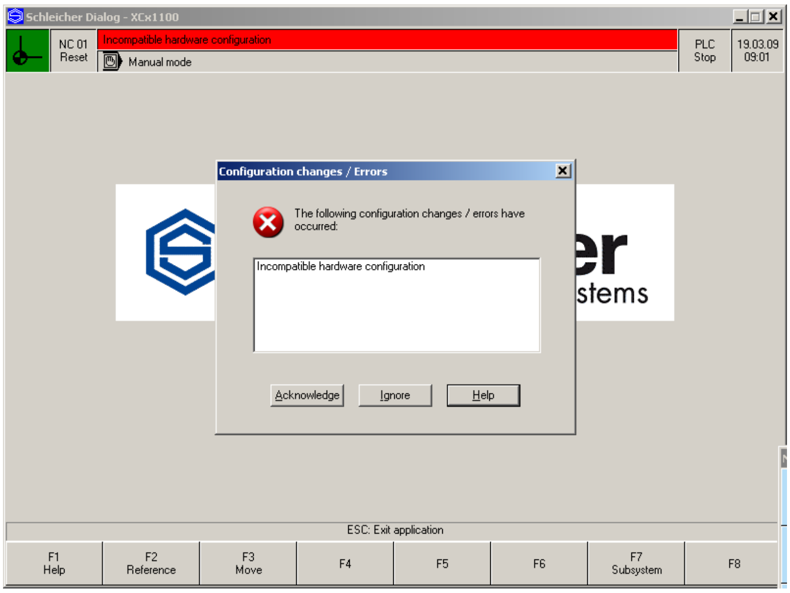
PLC state: Stop
Error: None
Access rights: Global
Powerflow: None
Bootproject: Test01_0809
Project source: Exist
Logic Analyser: inactive

Resource
State: Stop

Figure 39: Read the operating software version

If the controller operating software and the shared RAM version used do not correspond, an error message is entered in the error memory when the PLC starts. This error message is displayed in the Schleicher Dialog as follows (→ Figure: 40).

Figure: 40:
Display of the shared RAM error message in the Schleicher Dialog



Configuration changes / Errors

The following configuration changes / errors have occurred:

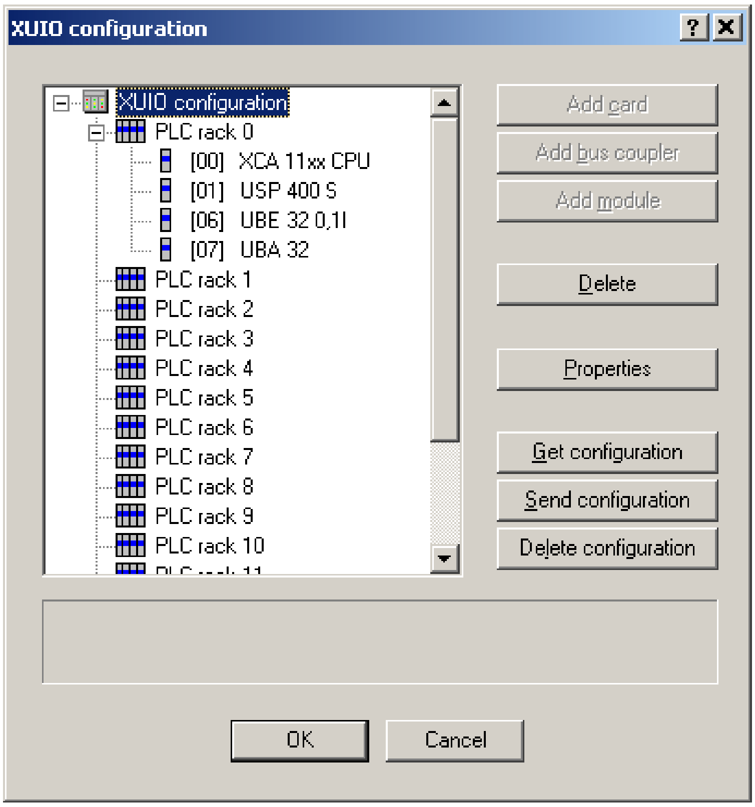
Incompatible hardware configuration

If such an error message appears, the PLC project must be corrected by inserting the data types and variables for the appropriate shared RAM version. The PLC project must now be recompiled and transferred to the controller.

6.6 Access to I/O Level

6.6.1 The Buttons in The XUIO Configuration Dialog Window

Figure 41: "XUIO configuration" dialog window



Add card; add bus coupler

Not relevant for XCx 1100.

Add module

Add a new module. A selection list appears with all modules for the XCx 1100. The properties of the module (slot number, option, module addresses) can now be specified in another dialog window. Click *OK* to confirm and add the module. It is sorted in the hardware configuration according to the slot number selected. A new module can then only be added when a PLC rack or a different module was selected in the tree view.

Delete

Delete one or more modules. If a module was selected in the tree view, this module is removed from the hardware configuration. If a PLC rack was selected in the tree view, all the modules belonging to this rack are removed. All modules are removed from the hardware configuration if the *XUIO configuration* node was selected in the tree view.

Properties

Display the properties of the complete configuration or a module.

The properties of the complete configuration are displayed if the *XUIO configuration* node was selected in the tree view. Here you can specify which variables are to be generated for the digital inputs and outputs and which entries should be added in the I/O configuration of the PLC project.

No properties are available for PLC racks.

If a module was selected in the tree view, the properties of this module are displayed. The slot, option and module addresses can be changed here.

Fetch configuration

Fetch the current hardware configuration from the controller and display in the tree view. This function can only be performed if a connection to the controller can be established.

Send configuration

Transfer the hardware configuration displayed in the tree view to the controller. This function can only be performed if a connection to the controller can be established. The hardware configuration created is stored on the Compact Flash memory of the controller. The previous hardware configuration is overwritten by this. The controller must be restarted for the transferred hardware configuration to become effective.

Delete configuration

Delete the hardware configuration on the controller. This function can only be performed if a connection to the controller can be established. The hardware configuration saved on the Compact Flash memory of the controller is deleted.

OK

Save the hardware configuration created, insert variables and I/O configuration entries to the PLC project according to the hardware configuration created and close the window.

Cancel

Close the window without changes to the PLC project and without saving the hardware configuration created.

6.6.2 Loading the Hardware Configuration

A configurator tool is provided with the installation of the add-ons for MULTIPROG that performs the following tasks:

- Display and edit the hardware configuration (slot list of the input and output modules)
- Insert the required variables in the PLC project which aids the PLC programmer to access the input and output modules.
- Insert the required entries in the I/O configuration of the PLC project

The configurator can be started via the *Extras / XIO configuration* menu item (→ Figure 42).

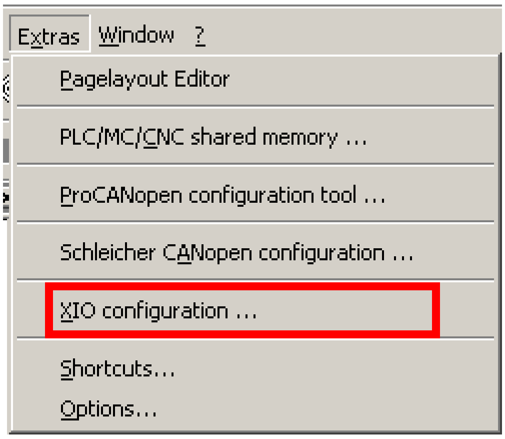
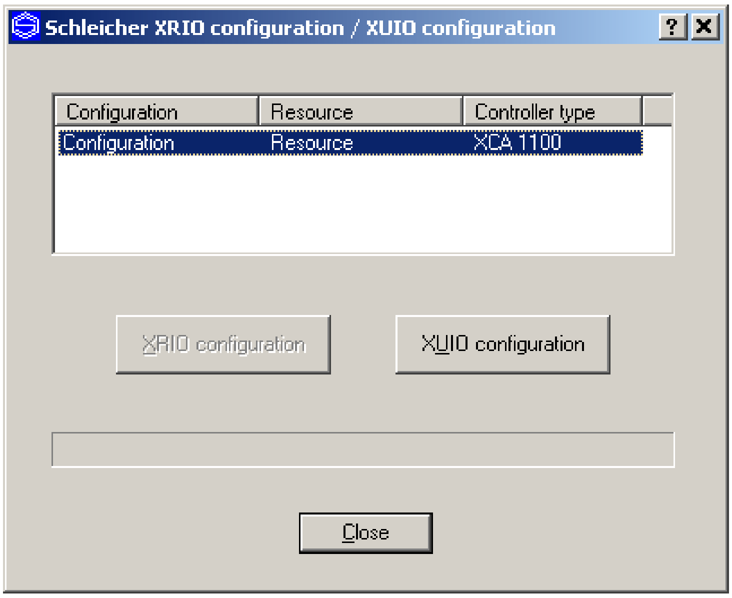

Extras
Window
?
Pagelayout Editor
PLC/MC/CNC shared memory ...
ProCANopen configuration tool ...
Schleicher CANopen configuration ...
XIO configuration ...
Shortcuts...
Options...

Figure 42: "Extras / XIO configuration" menu item

The *XRIO configuration / XUIO configuration* dialog window first appears displaying the resources available in the PLC project and the associated controller types for selection. Click the *XUIO configuration* button to continue the configuration for the selected resource (→ Figure 43).



Configuration	Resource	Controller type
Configuration	Resource	XCA 1100

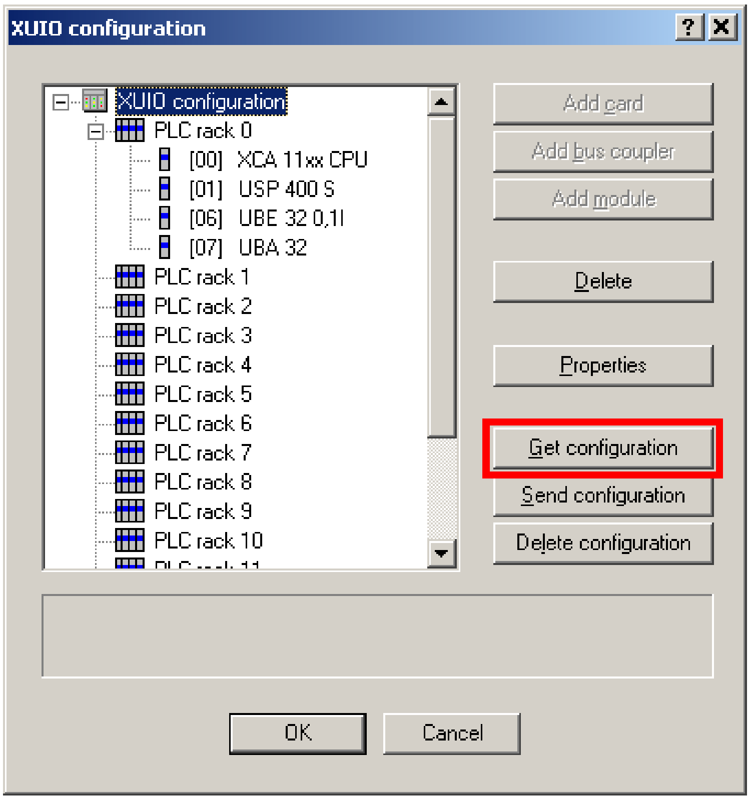
Figure 43: "XUIO configuration / XUIO configuration" dialog window

Important!

If errors occur during communication via the Ethernet connection to the XCx, the correct parameterisation of the interface in MULTIPROG must be checked.

The hardware configuration saved in the PLC project is displayed in the dialog window. The hardware configuration is initially empty when you first open this window for a new PLC project. In this case, the hardware configuration must only be read once from the controller (*Fetch configuration* button) and added to the PLC project with *OK* (→ Figure 44).

Figure 44: "XUIO configuration" dialog window



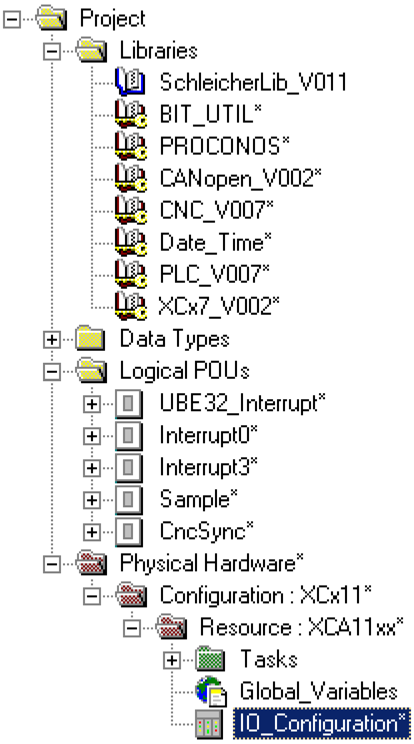
The saved hardware configuration is compared with the actual hardware configuration available each time the controller is started. If there any differences, the controller startup is interrupted with an error message. The *BUS* LED on the control unit flashes red and an error message is saved in the active error buffer:

0x01100001	Incompatible hardware configuration

The active error buffer and error log book can be viewed on each operating level in the Schleicher dialog via the key combination <Ctrl+?>.

The required entries in the I/O configuration for the created hardware configuration are inserted in the PLC project (the I/O configuration can be opened by double-clicking on the *IO_Configuration* node in the project tree → Figure 45).

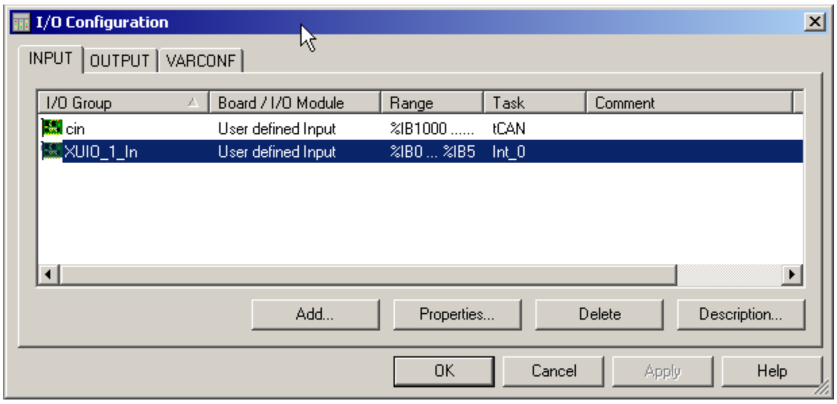
Figure 45:
"IO_Configuration" node



- Project
  - Libraries
    - SchleicherLib_V011
    - BIT_UTIL*
    - PROCONOS*
    - CANopen_V002*
    - CNC_V007*
    - Date_Time*
    - PLC_V007*
    - XCx7_V002*
  - Data Types
  - Logical POUs
    - UBE32_Interrupt*
    - Interrupt0*
    - Interrupt3*
    - Sample*
    - CncSync*
  - Physical Hardware*
    - Configuration : XCx11*
      - Resource : XCA11xx*
        - Tasks
        - Global_Variables
        - IO_Configuration*

For the example above, the entries *XUIO_1_In* (under INPUT) and *XUIO_1_Out* (under OUTPUT) are inserted in the I/O configuration (→ Figure 46).

Figure 46:
"I/O configuration"
dialog window

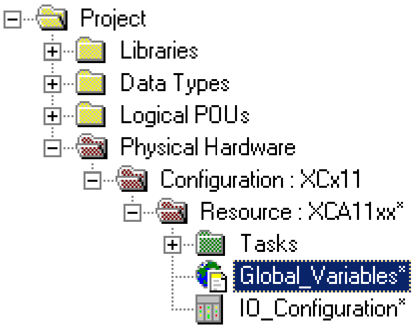


I/O Configuration

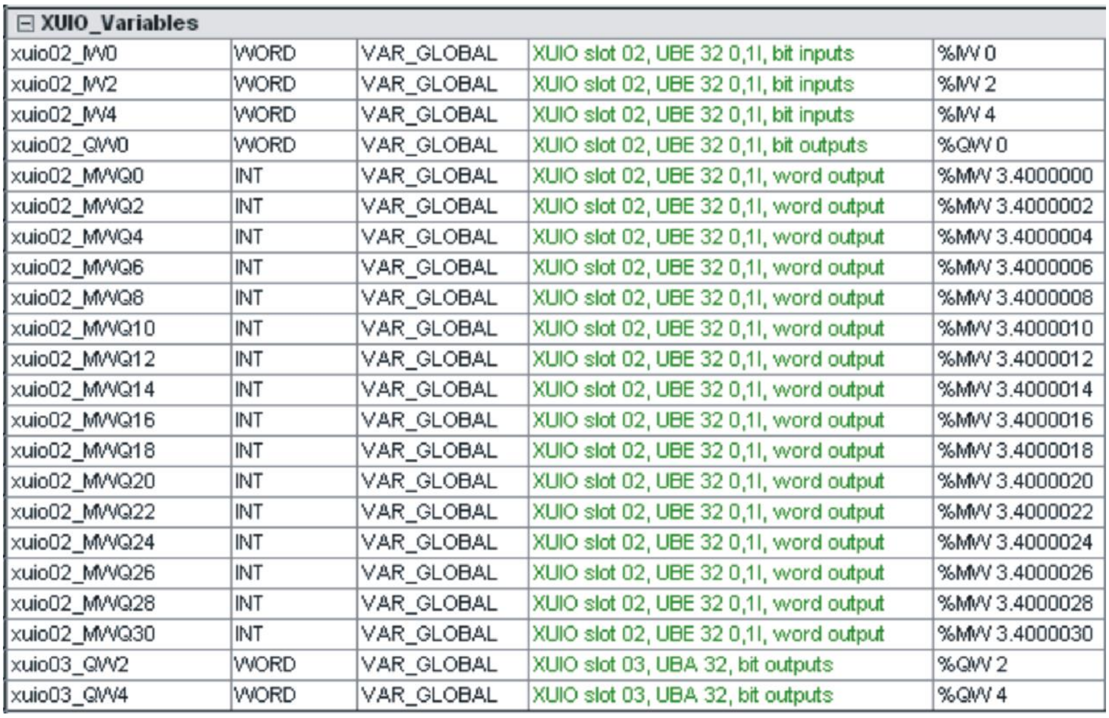
I/O Group	Board / I/O Module	Range	Task	Comment
cin	User defined Input	%IB1000	tCAN	
XUIO_1_In	User defined Input	%IB0 ... %IB5	Int_0	

The entries required in the variables table for the created I/O configuration are inserted in the PLC project (→ Figure 48). The variables table can be opened by double-clicking on the *Global_Variables* node in the project tree (→ Figure 47).

Figure 47: "Global_Variables" node



Project
Libraries
Data Types
Logical POUs
Physical Hardware
Configuration : XCx11
Resource : XCA11xx*
Tasks
Global_Variables*
IO_Configuration*



⊟ **XUIO_Variables**

xuio02_IW0	WORD	VAR_GLOBAL	XUIO slot 02, UBE 32 0,1I, bit inputs	%IW 0
xuio02_IW2	WORD	VAR_GLOBAL	XUIO slot 02, UBE 32 0,1I, bit inputs	%IW 2
xuio02_IW4	WORD	VAR_GLOBAL	XUIO slot 02, UBE 32 0,1I, bit inputs	%IW 4
xuio02_QW0	WORD	VAR_GLOBAL	XUIO slot 02, UBE 32 0,1I, bit outputs	%QW 0
xuio02_MWQ0	INT	VAR_GLOBAL	XUIO slot 02, UBE 32 0,1I, word output	%MW 3.4000000
xuio02_MWQ2	INT	VAR_GLOBAL	XUIO slot 02, UBE 32 0,1I, word output	%MW 3.4000002
xuio02_MWQ4	INT	VAR_GLOBAL	XUIO slot 02, UBE 32 0,1I, word output	%MW 3.4000004
xuio02_MWQ6	INT	VAR_GLOBAL	XUIO slot 02, UBE 32 0,1I, word output	%MW 3.4000006
xuio02_MWQ8	INT	VAR_GLOBAL	XUIO slot 02, UBE 32 0,1I, word output	%MW 3.4000008
xuio02_MWQ10	INT	VAR_GLOBAL	XUIO slot 02, UBE 32 0,1I, word output	%MW 3.4000010
xuio02_MWQ12	INT	VAR_GLOBAL	XUIO slot 02, UBE 32 0,1I, word output	%MW 3.4000012
xuio02_MWQ14	INT	VAR_GLOBAL	XUIO slot 02, UBE 32 0,1I, word output	%MW 3.4000014
xuio02_MWQ16	INT	VAR_GLOBAL	XUIO slot 02, UBE 32 0,1I, word output	%MW 3.4000016
xuio02_MWQ18	INT	VAR_GLOBAL	XUIO slot 02, UBE 32 0,1I, word output	%MW 3.4000018
xuio02_MWQ20	INT	VAR_GLOBAL	XUIO slot 02, UBE 32 0,1I, word output	%MW 3.4000020
xuio02_MWQ22	INT	VAR_GLOBAL	XUIO slot 02, UBE 32 0,1I, word output	%MW 3.4000022
xuio02_MWQ24	INT	VAR_GLOBAL	XUIO slot 02, UBE 32 0,1I, word output	%MW 3.4000024
xuio02_MWQ26	INT	VAR_GLOBAL	XUIO slot 02, UBE 32 0,1I, word output	%MW 3.4000026
xuio02_MWQ28	INT	VAR_GLOBAL	XUIO slot 02, UBE 32 0,1I, word output	%MW 3.4000028
xuio02_MWQ30	INT	VAR_GLOBAL	XUIO slot 02, UBE 32 0,1I, word output	%MW 3.4000030
xuio03_QW2	WORD	VAR_GLOBAL	XUIO slot 03, UBA 32, bit outputs	%QW 2
xuio03_QW4	WORD	VAR_GLOBAL	XUIO slot 03, UBA 32, bit outputs	%QW 4

Figure 48: Insert the global variables to the PLC project

For the above example, variables of the type WORD were created for the digital inputs and outputs. Here, each of the 16 bits of this type of variable corresponds to a digital input or output. However, a separate variable of type BOOL can be generated for each digital input or output if required. To do this, the *Generation of BOOL variables for digital inputs/outputs* option must be activated in the properties of the complete configuration.

6.6.3 Set Hardware Configurations Options

Note

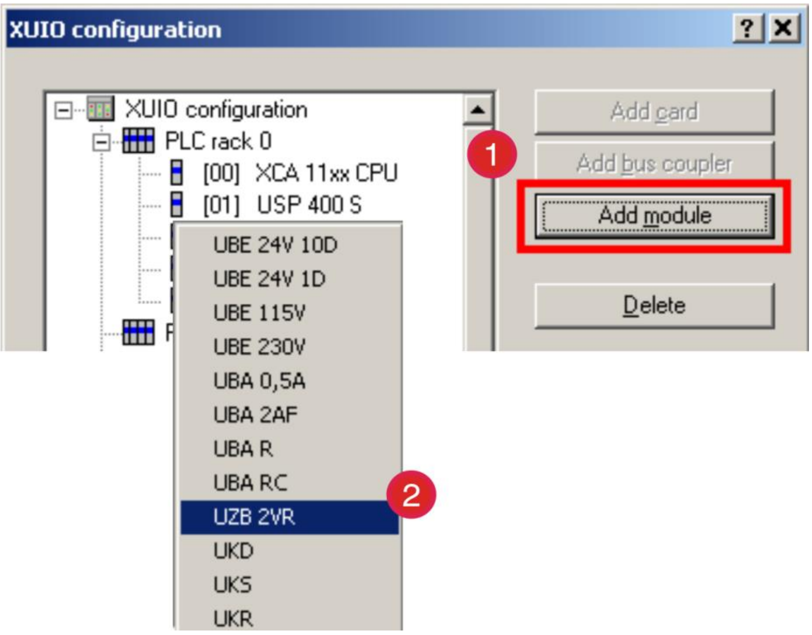
The following instructions are only important for advanced users.

Hardware configurations options (called options in the following) allow for effective creation of PLC programs. A PLC program can be adapted to different hardware configurations without program changes by requesting options.

Example

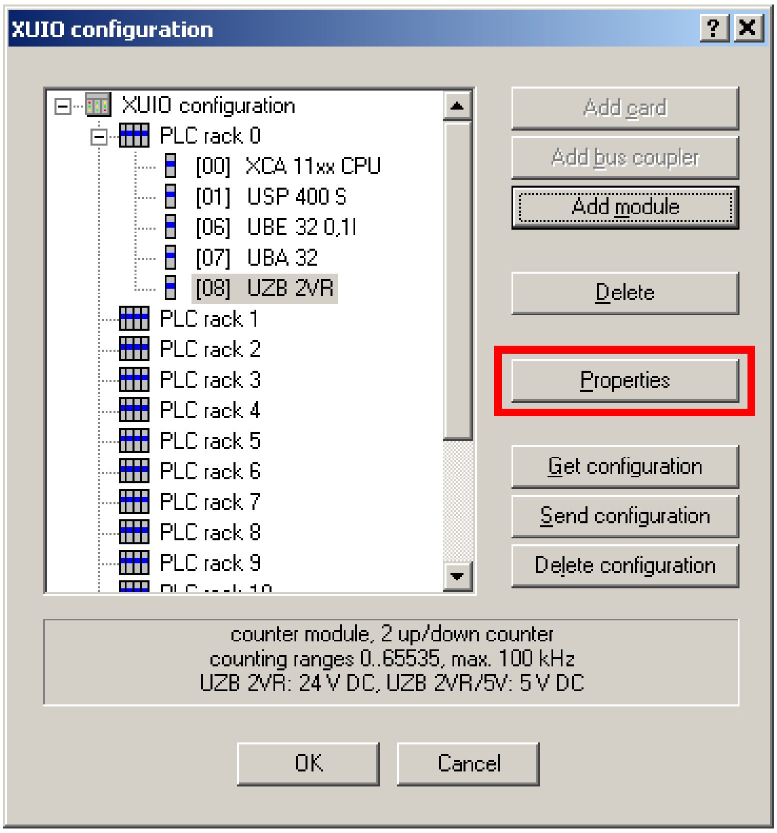
The I/O configurator is restarted via the *Extras / XIO configuration* menu item. The hardware configuration loaded above is then supplemented with two counter modules UZB 2VR (2) using the *Add module* button (1) (→ Figure 49).

Figure 49: "XUIO configuration" dialog window, add module



The *Module properties* dialog window is then called via the *Properties* button for each of the two new modules (→ Figure 50).

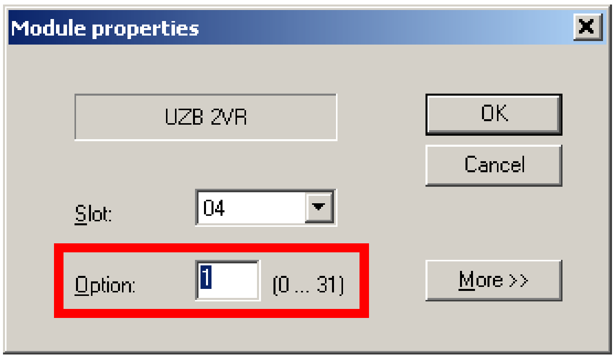
Figure 50: "XUIO configuration" dialog window, properties



counter module, 2 up/down counter
counting ranges 0..65535, max. 100 kHz
UZB 2VR: 24 V DC, UZB 2VR/5V: 5 V DC

The *Option* setting in the *Module properties* dialog is set to 1 for both modules (→ Figure 51). A total of 32 options 0...31 can be set. All other modules retain their setting. The preset is option 0.

Figure 51: "Module properties" dialog window, set options



UZB 2VR

Slot: 04

Option: 1 (0 ... 31)

Both UZB 2VR units belong to option 1, all other modules to option 0.

The saved hardware configuration is compared to the actual hardware available when the controller starts up. Either all modules of an option must be available (the option is then active), or no modules must be available for this option (the option is then inactive). The address ranges of inactive options are remain reserved.

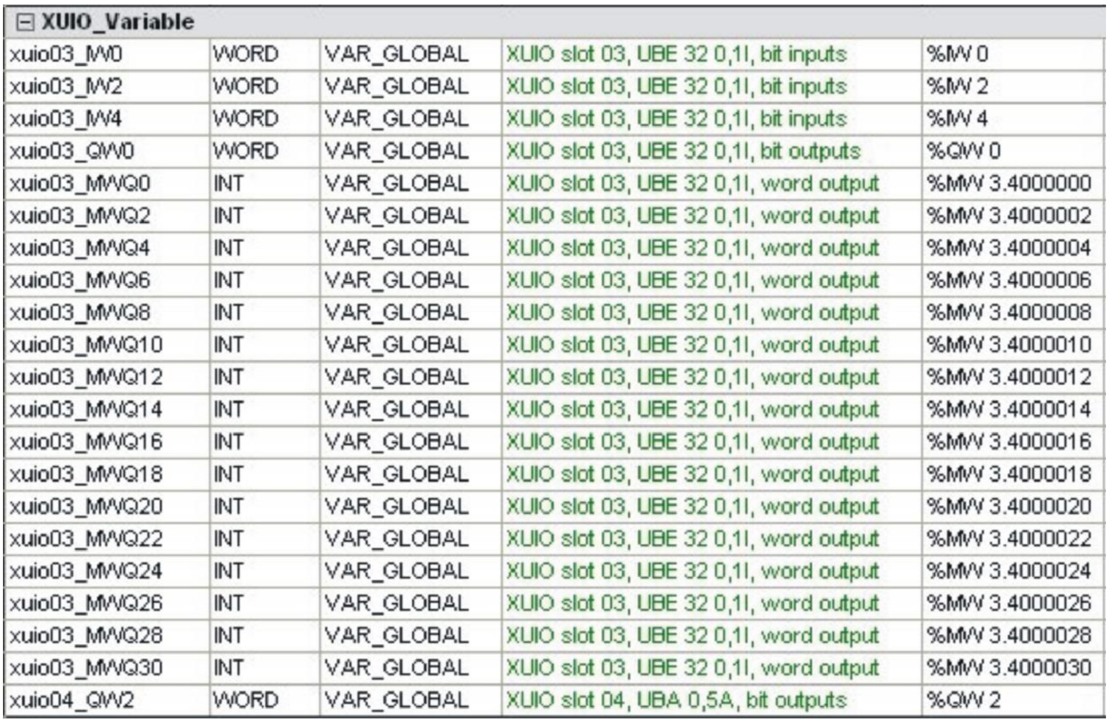
If the modules of an option are only partly available, a configuration error is detected. (BUS LED on the control unit flashes red and an error message is saved in the active error buffer).

6.7 Access to Interrupt Inputs

The XCx 1100 can process four interrupts that are triggered by the digital inputs. The first four inputs of an input module UBE 32 0,1I are used for this.

6.7.1 Variables for the Interrupt Processing

The variables of the physical inputs and the variables required for the interrupt processing are created during the hardware configuration of the controller structure with a UBE 32 0,1I (see above) (→ Figure 52).



⊟ XUIO_Variable				
xuio03_IW0	WORD	VAR_GLOBAL	XUIO slot 03, UBE 32 0,1I, bit inputs	%IW 0
xuio03_IW2	WORD	VAR_GLOBAL	XUIO slot 03, UBE 32 0,1I, bit inputs	%IW 2
xuio03_IW4	WORD	VAR_GLOBAL	XUIO slot 03, UBE 32 0,1I, bit inputs	%IW 4
xuio03_QW0	WORD	VAR_GLOBAL	XUIO slot 03, UBE 32 0,1I, bit outputs	%QW 0
xuio03_MWQ0	INT	VAR_GLOBAL	XUIO slot 03, UBE 32 0,1I, word output	%MW 3.4000000
xuio03_MWQ2	INT	VAR_GLOBAL	XUIO slot 03, UBE 32 0,1I, word output	%MW 3.4000002
xuio03_MWQ4	INT	VAR_GLOBAL	XUIO slot 03, UBE 32 0,1I, word output	%MW 3.4000004
xuio03_MWQ6	INT	VAR_GLOBAL	XUIO slot 03, UBE 32 0,1I, word output	%MW 3.4000006
xuio03_MWQ8	INT	VAR_GLOBAL	XUIO slot 03, UBE 32 0,1I, word output	%MW 3.4000008
xuio03_MWQ10	INT	VAR_GLOBAL	XUIO slot 03, UBE 32 0,1I, word output	%MW 3.4000010
xuio03_MWQ12	INT	VAR_GLOBAL	XUIO slot 03, UBE 32 0,1I, word output	%MW 3.4000012
xuio03_MWQ14	INT	VAR_GLOBAL	XUIO slot 03, UBE 32 0,1I, word output	%MW 3.4000014
xuio03_MWQ16	INT	VAR_GLOBAL	XUIO slot 03, UBE 32 0,1I, word output	%MW 3.4000016
xuio03_MWQ18	INT	VAR_GLOBAL	XUIO slot 03, UBE 32 0,1I, word output	%MW 3.4000018
xuio03_MWQ20	INT	VAR_GLOBAL	XUIO slot 03, UBE 32 0,1I, word output	%MW 3.4000020
xuio03_MWQ22	INT	VAR_GLOBAL	XUIO slot 03, UBE 32 0,1I, word output	%MW 3.4000022
xuio03_MWQ24	INT	VAR_GLOBAL	XUIO slot 03, UBE 32 0,1I, word output	%MW 3.4000024
xuio03_MWQ26	INT	VAR_GLOBAL	XUIO slot 03, UBE 32 0,1I, word output	%MW 3.4000026
xuio03_MWQ28	INT	VAR_GLOBAL	XUIO slot 03, UBE 32 0,1I, word output	%MW 3.4000028
xuio03_MWQ30	INT	VAR_GLOBAL	XUIO slot 03, UBE 32 0,1I, word output	%MW 3.4000030
xuio04_QW2	WORD	VAR_GLOBAL	XUIO slot 04, UBA 0,5A, bit outputs	%QW 2

Figure 52: Create variables for the interrupt processing

Important Variables of the UBE 32 0,1I for Interrupt Processing

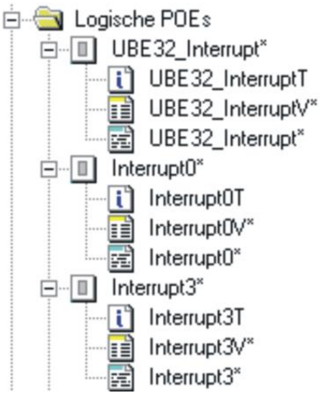
Inputs		
xuio.._IW0	Input bits 0...15 (input bits 0...3 with interrupt functionality)	
xuio.._IW2	Input bits 16...31	
xuio.._IW4	Map of input bits 0...3 for enabled interrupt functionality (These bits must be reset via xuio.._MWQ0-14 for the interrupt enable, see below)	
	Bit 0	Set for rising edge at input 0
	Bit 1	Set for rising edge at input 1
	Bit 2	Set for rising edge at input 2
	Bit 3	Set for rising edge at input 3
	Bit 4	Set for falling edge at input 0
	Bit 5	Set for falling edge at input 1
	Bit 6	Set for falling edge at input 2
	Bit 7	Set for falling edge at input 3
Outputs		
xuio.._QW0	Mask to enable the interrupt	
	Bit 0	Input bit 0 with rising edge
	Bit 1	Input bit 1 with rising edge
	Bit 2	Input bit 2 with rising edge
	Bit 3	Input bit 3 with rising edge
	Bit 4	Input bit 0 with falling edge
	Bit 5	Input bit 1 with falling edge
	Bit 6	Input bit 2 with falling edge
	Bit 7	Input bit 3 with falling edge
xuio.._MWQ0 to xuio.._MWQ14	Acknowledge the interrupt	
	xuio.._MWQ0	Input bit 0 with rising edge
	xuio.._MWQ2	Input bit 1 with rising edge
	xuio.._MWQ4	Input bit 2 with rising edge
	xuio.._MWQ6	Input bit 3 with rising edge
	xuio.._MWQ8	Input bit 0 with falling edge
	xuio.._MWQ10	Input bit 1 with falling edge
	xuio.._MWQ12	Input bit 2 with falling edge
	xuio.._MWQ14	Input bit 3 with falling edge

Table 15: Important variables of the UBE 32 0,1I for the interrupt processing

The slot number of the UBE 32 0,1I is shown in the table with two points (xuio..).

6.7.2 Example POUs for the interrupt process

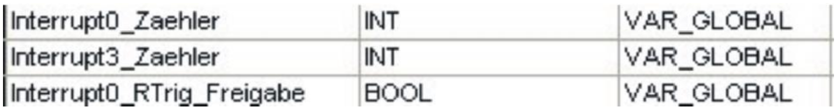
Three POUs are created and integrated in the task structure. The POUs are used for the interrupt enable *UBE32_Interrupt* and the interrupt processing of a tracer *Interrupt 0* and a counter pulse *Interrupt3* (→ Figure53).



- Logische POEs
  - UBE32_Interrupt*
    - UBE32_InterruptT
    - UBE32_InterruptV*
    - UBE32_Interrupt*
  - Interrupt0*
    - Interrupt0T
    - Interrupt0V*
    - Interrupt0*
  - Interrupt3*
    - Interrupt3T
    - Interrupt3V*
    - Interrupt3*

Figure53: Node for interrupt enable and processing

Global variables are created for counting interrupts that occurred and for communication of the POUs with each other (→ Figure 54).



Interrupt0_Zaehler	INT	VAR_GLOBAL
Interrupt3_Zaehler	INT	VAR_GLOBAL
Interrupt0_RTrig_Freigabe	BOOL	VAR_GLOBAL

Figure 54: Global variables for interrupt count and communication

POE "UBE32_Interrupt"

```
(* Example 1: POE Interrupt0/ Task I_E0 Event 0
 Interrupt 0 (UBE32 0,1I Input 0)
 Evaluation of first rising edge at the input (e.g. a tracer),
 Here the interrupt 0_RTrig_Freigabe must be set manually by
 forcing. It is then reset in the Interrupt0 program
 ----------------------------------------------------------------- *)
IF    Interrupt0_RTrig_Freigabe
      THEN
             xuio03_QW0  := S_BIT_IN_WORD(TRUE,xuio03_QW0,SINT#0);
      ELSE
             xuio03_QW0  := R_BIT_IN_WORD(TRUE,xuio03_QW0,SINT#0);
             xuio03_MWQ0 := 0;
END_IF;

(* Example 2: POE Interrupt3/ Task I_E3 Event 3
 Interrupt 3 (UBE32 0,1I Input 3)
 Evaluation of all falling edges at the input (e.g. Counting with a
 light barrier). Interrupt 3_RTrig_Freigabe must be manually set and reset.
 ----------------------------------------------------------------- *)
IF    Interrupt3_FTrig_Freigabe
      THEN
             xuio03_QW0   := S_BIT_IN_WORD(TRUE,xuio03_QW0,SINT#7);
      ELSE
             xuio03_QW0   := R_BIT_IN_WORD(TRUE,xuio03_QW0,SINT#7);
             xuio03_MWQ14 := 0;
END_IF;
```

POE "Interrupt0" (tracer)

```
(* Example 1: POE Interrupt0/ Task I_E0 Event 0
 Interrupt 0 (UBE32 0,1I Input 0)
 Evaluation of first rising edge at the input as a tracer
 ----------------------------------------------------------------- *)
Interrupt0_Zaehler        := Interrupt0_Zaehler + 1;

Interrupt0_RTrig_Freigabe := FALSE;

RETURN;
```

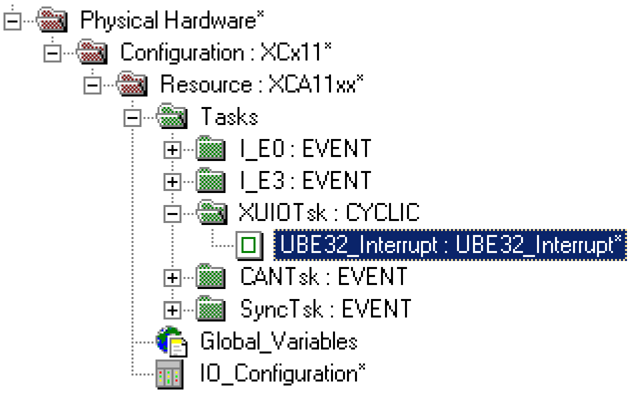
POE "Interrupt3" (counter)

```
(* Example 2: POE Interrupt3/ Task I_E3 Event 3
 Interrupt 3 (UBE32 0,1I Input 3)
 Evaluation of all falling edges at the input as a counter
 ----------------------------------------------------------------- *)
Interrupt3_Zaehler  := Interrupt3_Zaehler + 1;

xuio03_MWQ14   := 0;

RETURN;
```

6.7.3 Task Structure for Interrupt Processing

For the Interrupt enable, a cyclic task with the name *XUIOTsk* is created that is assigned to the POU *UBE32_Interrupt* (→ Figure 55).

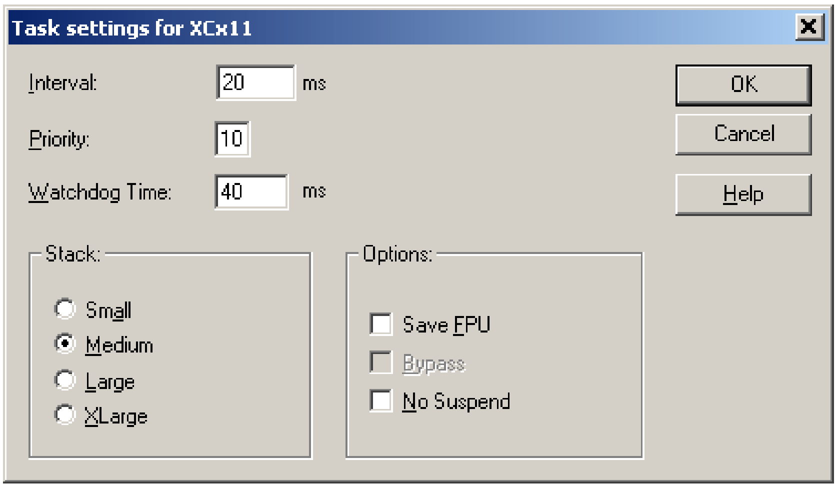
Figure 55: Cyclic task "XUIO" with POU "UBE32_Interrupt"



```
Physical Hardware*
  Configuration : XCx11*
    Resource : XCA11xx*
      Tasks
        I_E0 : EVENT
        I_E3 : EVENT
        XUIOTsk : CYCLIC
          UBE32_Interrupt : UBE32_Interrupt*
        CANTsk : EVENT
        SyncTsk : EVENT
      Global_Variables
      IO_Configuration*
```

The following task settings are selected as an example (→ Figure 56):

Figure 56: Settings for cyclic task



Task settings for XCx11

Interval: 20 ms

Priority: 10

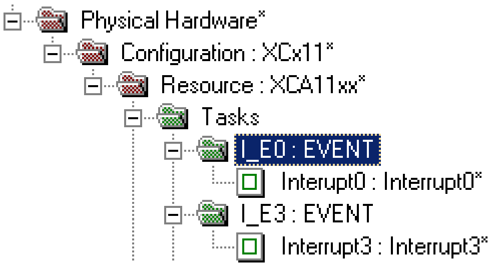
Watchdog Time: 40 ms

Stack:
- ○ Small
- ● Medium
- ○ Large
- ○ XLarge

Options:
- □ Save FPU
- □ Bypass
- □ No Suspend

Finally, event tasks are created that are started independent of the interrupt and are assigned by the interrupt processing POUs (→ Figure 57).

Figure 57: Create the event tasks for the interrupts



```
Physical Hardware*
  Configuration : XCx11*
    Resource : XCA11xx*
      Tasks
        I_E0 : EVENT
          Interupt0 : Interrupt0*
        I_E3 : EVENT
          Interrupt3 : Interrupt3*
```

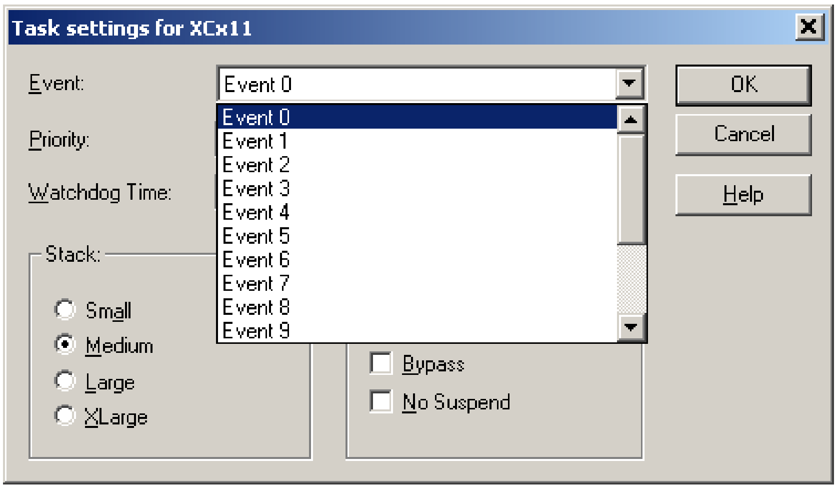
Two event tasks were created here:

- *I_E0:EVENT* for interrupt via input bit 0 with POU *Interrupt0*
- *I_E3:EVENT* for interrupt via input bit 3 with POU *Interrupt3*

The assignment of the interrupt (input bits) to the event tasks must be specified in the task setting using the event number specified (→ Figure 58). The following specification applies here:

Input bit 0	Event 0
Input bit 1	Event 1
Input bit 2	Event 2
Input bit 3	Event 3

Figure 58:
Settings for event task

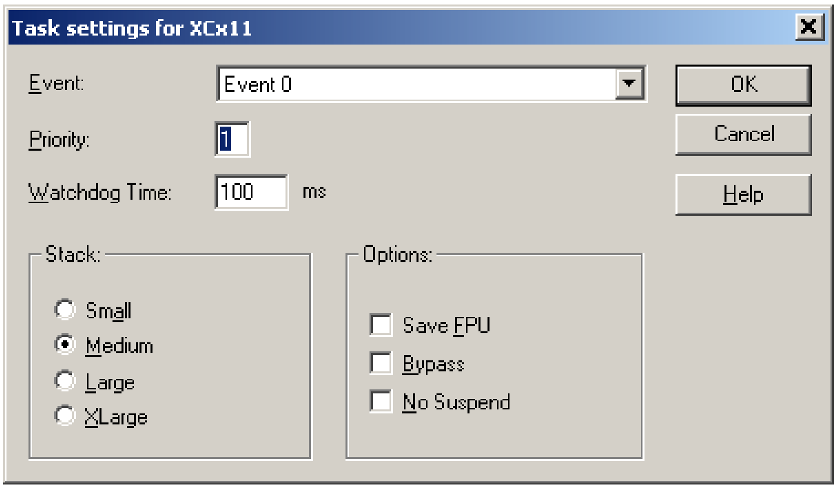


Task settings for XCx11

Event: Event 0

Event 0
Event 1
Event 2
Event 3
Event 4
Event 5
Event 6
Event 7
Event 8
Event 9

Figure 59:
Settings for event task



Task settings for XCx11

Event: Event 0

Priority: 1

Watchdog Time: 100 ms

Stack:
○ Small
● Medium
○ Large
○ XLarge

Options:
□ Save FPU
□ Bypass
□ No Suspend

6.8 CANopen for Remote I/O

The XCx 1100 has an integrated CANopen interface for the field bus connection or the control of digital drives. This section describes the startup and configuration of the CANopen network using a minimal configuration.

6.8.1 Specifications

- CANopen operates with two types of telegrams:
    - **SDO** (Service Data Objects) are telegrams that must be **confirmed** by the recipient
    - **PDO** (Process Data Objects) are telegrams that must **not** be **confirmed** by the recipient.
- The PDOs for data exchange are defined during network configuration and given a so-called COB ID. The recipient of a message always knows which telegram is addressed for this node.
- Certain components only use default mapping and work with fixed COD IDs, which are defined in the CANopen definition.
- The standard communication for PDO is **COS** (**C**hange **O**f **S**tate): A PDO is only sent if the information in the PDO changes.

6.8.2 Connection and Wiring

The described setup is a minimum configuration, which serves as an example for further commissioning.

Figure 60: Connection principle for CANopen network

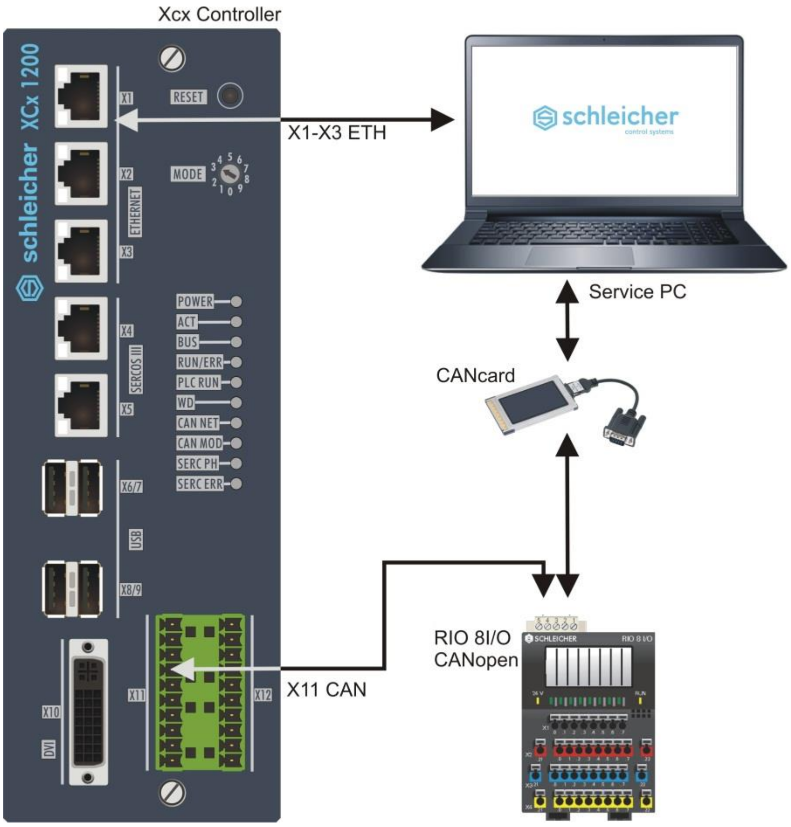
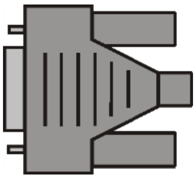
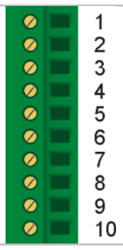
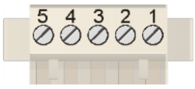


Table 16: Pin assignment of the connector used

			
	1	**2**	**3**
	Subminiature, 9-pin, socket connector For CANcard in service PC	Plug-in terminal 10-pin X11, to XCx 1100	Plug-in terminal 5-pin on RIO 8 I/O CANopen
CAN	**Pin**	**Pin**	**Pin**
0 V		1 / 6	1
CAN_L	2	2 / 7	2**
Drain		3 / 8	3
CAN_H	7	4 / 9	4**
DC +24 V		5 / 10	5

* Pin groups 1..5 and 6..10 are connected in parallel.

** A 120 ohm terminator must be connected between pin 2 and 4 on the RIO 8 I/O CANopen.

6.8.3 Settings on the RIO 8 I/O CANopen module

On the RIO 8 I/O CANopen compact module, set the node number to 2 and the data transmission rate to 125 kBaud (→ Figure 61). Set the DIP switches on top of the module as follows:

Table 17:
Set node number and data transfer rate

	Node number MAC ID							Data transfer rate BAUD		
Switch	1	2	3	4	5	6	7	8	9	10
Position	OFF	ON	OFF	OFF	OFF	OFF	OFF	ON	ON	OFF

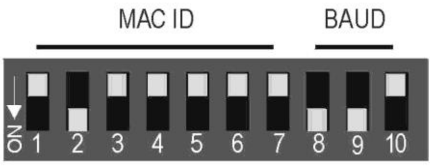
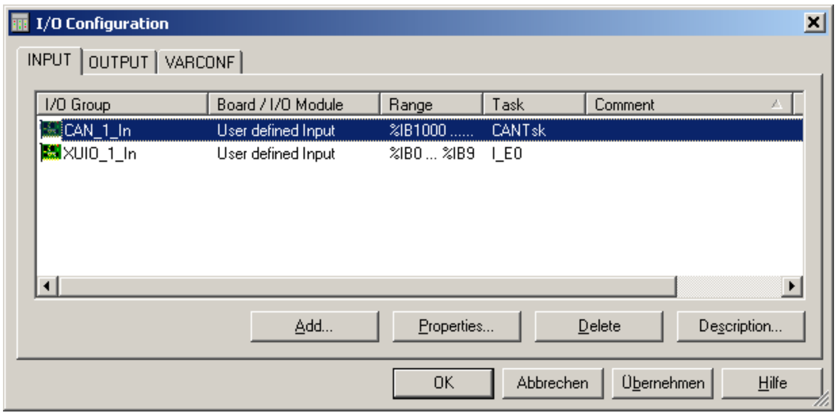


Figure 61:
DIP switch at compact module RIO 8 I/O CANopen

6.8.4 Declaring the I/O driver for CANopen

The I/O driver for the CANopen network is declared at the same place and in the same way as the XUIO driver. The *IO_Configuration* container is at the end of the project tree. The I/O configurations *CAN_1_In* (under INPUT) and *CAN_1_Out* (under OUTPUT) are already created there (→ Figure 62).

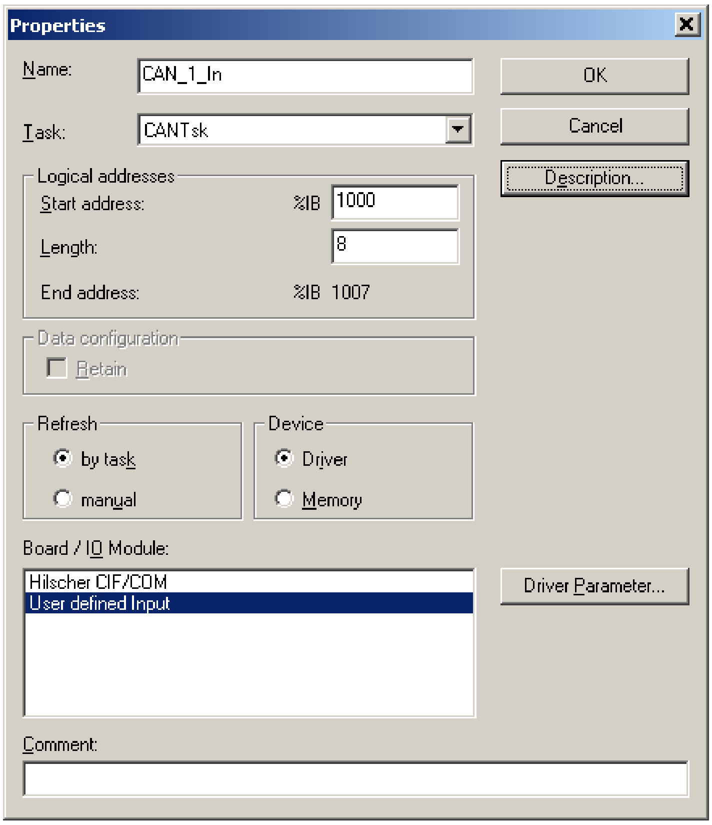


I/O Group	Board / I/O Module	Range	Task	Comment
CAN_1_In	User defined Input	%IB1000	CANTsk	
XUIO_1_In	User defined Input	%IB0 ... %IB9	I_E0	

Figure 62:
CANopen configuration "I/O configuration" dialog window

For this example you have to enter the following parameters: (→ Figure 63):

- The **Task** with which the I/O address space is synchronised must be *CanTsk*.
- As **Start address**, the logical addresses *IB1000* are entered for *CAN_1_In* and *QB1000* for *CAN_1_Out*.
- In the **Length** parameter, the number of I/O bytes to be exchanged in the CANopen network are declared (here 4, because minimal double word spacing is being used).



Properties

Name: CAN_1_In

Task: CANTsk

Logical addresses

Start address: %IB 1000

Length: 8

End address: %IB 1007

Data configuration

□ Retain

Refresh

● by task

○ manual

Device

● Driver

○ Memory

Board / IO Module:

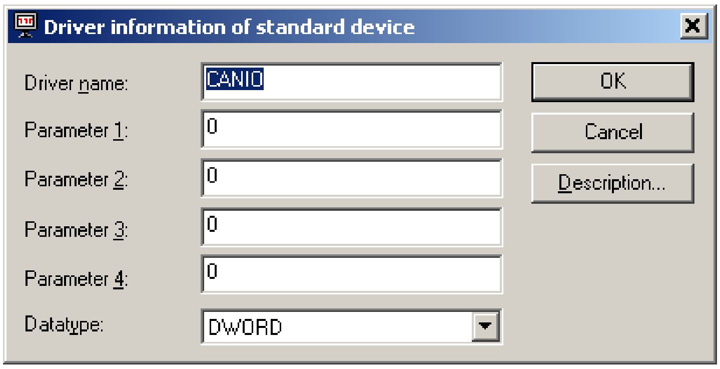
Hilscher CIF/COM

User defined Input

Comment:

Figure 63: CAN open configuration, "Properties" dialog window

The driver name **CANIO** must be set in *Driver parameters*, the data type is DWORD (→ Figure 64).



Driver information of standard device

Driver name: CANIO

Parameter 1: 0

Parameter 2: 0

Parameter 3: 0

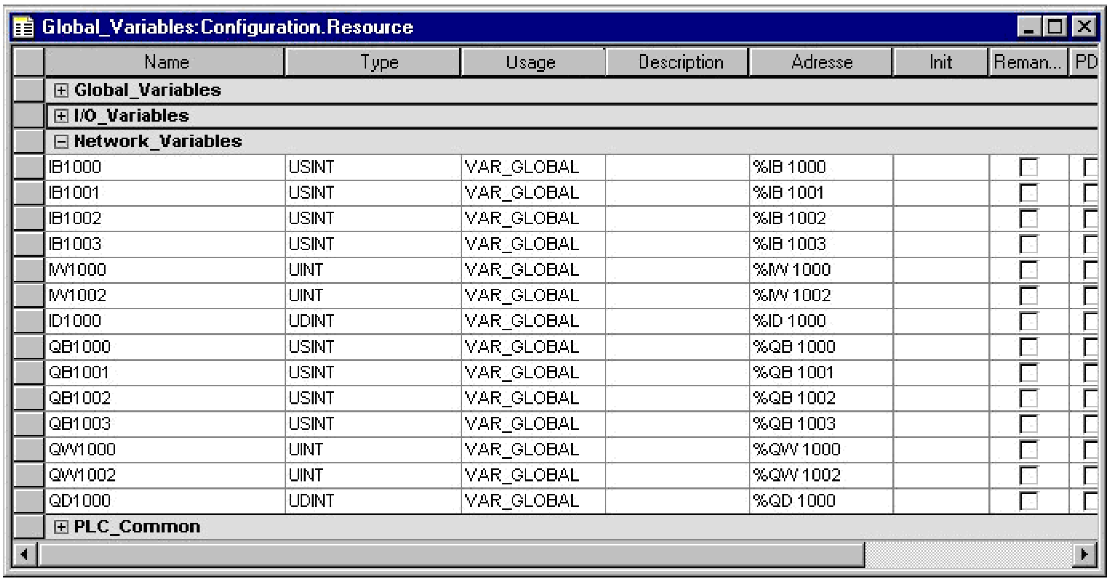
Parameter 4: 0

Datatype: DWORD

Figure 64: CAN open configuration, set "Driver properties"

6.8.5 Declaring Network Variables in MULTIPROG

The required variables are predefined in the *Network_Variables* folder in *Global_Variables* in the project tree window (→ Figure 65).

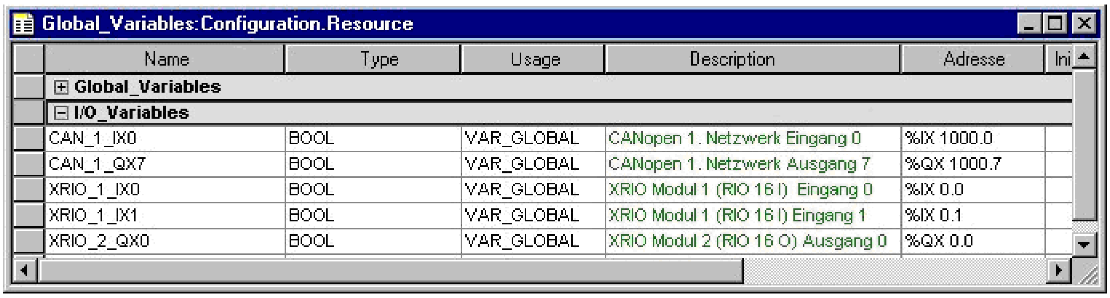


Global_Variables:Configuration.Resource

Name	Type	Usage	Description	Adresse	Init	Reman...	PD
⊞ Global_Variables							
⊞ I/O_Variables							
⊟ Network_Variables							
IB1000	USINT	VAR_GLOBAL		%IB 1000		□	□
IB1001	USINT	VAR_GLOBAL		%IB 1001		□	□
IB1002	USINT	VAR_GLOBAL		%IB 1002		□	□
IB1003	USINT	VAR_GLOBAL		%IB 1003		□	□
IW1000	UINT	VAR_GLOBAL		%IW 1000		□	□
IW1002	UINT	VAR_GLOBAL		%IW 1002		□	□
ID1000	UDINT	VAR_GLOBAL		%ID 1000		□	□
QB1000	USINT	VAR_GLOBAL		%QB 1000		□	□
QB1001	USINT	VAR_GLOBAL		%QB 1001		□	□
QB1002	USINT	VAR_GLOBAL		%QB 1002		□	□
QB1003	USINT	VAR_GLOBAL		%QB 1003		□	□
QW1000	UINT	VAR_GLOBAL		%QW 1000		□	□
QW1002	UINT	VAR_GLOBAL		%QW 1002		□	□
QD1000	UDINT	VAR_GLOBAL		%QD 1000		□	□
⊞ PLC_Common							

Figure 65: CANopen configuration, worksheet "Network_Variables"

I/O bits are declared in the *I/O_Variables* worksheet with addresses *IX1000.0* and *QX1000.0* (→ Figure 66). (*QX1000.7* is used in the example, to make the result visible on the RIO 8 I/O),



Global_Variables:Configuration.Resource

Name	Type	Usage	Description	Adresse	Ini
⊞ Global_Variables					
⊟ I/O_Variables					
CAN_1_IX0	BOOL	VAR_GLOBAL	CANopen 1. Netzwerk Eingang 0	%IX 1000.0	
CAN_1_QX7	BOOL	VAR_GLOBAL	CANopen 1. Netzwerk Ausgang 7	%QX 1000.7	
XRIO_1_IX0	BOOL	VAR_GLOBAL	XRIO Modul 1 (RIO 16 I) Eingang 0	%IX 0.0	
XRIO_1_IX1	BOOL	VAR_GLOBAL	XRIO Modul 1 (RIO 16 I) Eingang 1	%IX 0.1	
XRIO_2_QX0	BOOL	VAR_GLOBAL	XRIO Modul 2 (RIO 16 O) Ausgang 0	%QX 0.0	

Figure 66: CANopen configuration, worksheet "I/O_Variables"

6.8.6 CANopen Configuration with "Schleicher CANopen Configuration"

The **Schleicher CANopen configuration** that is installed together with the MULTIPROG add-ons is provided for setup of simple CANopen networks. The configurator is a pure offline tool, i.e. there is no communication to the CAN card of the controller. The tool generates program code that is listed when the PLC is started. A series of SDOs are transferred to the controller that configure the network.

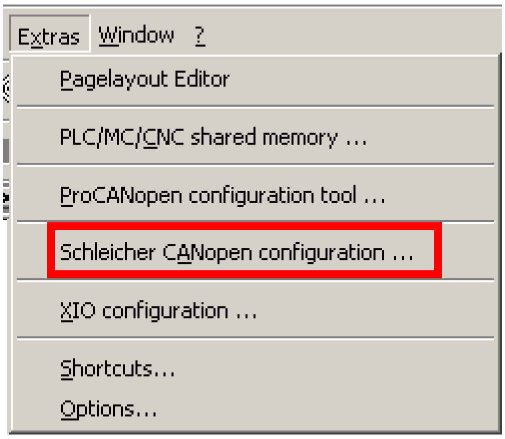
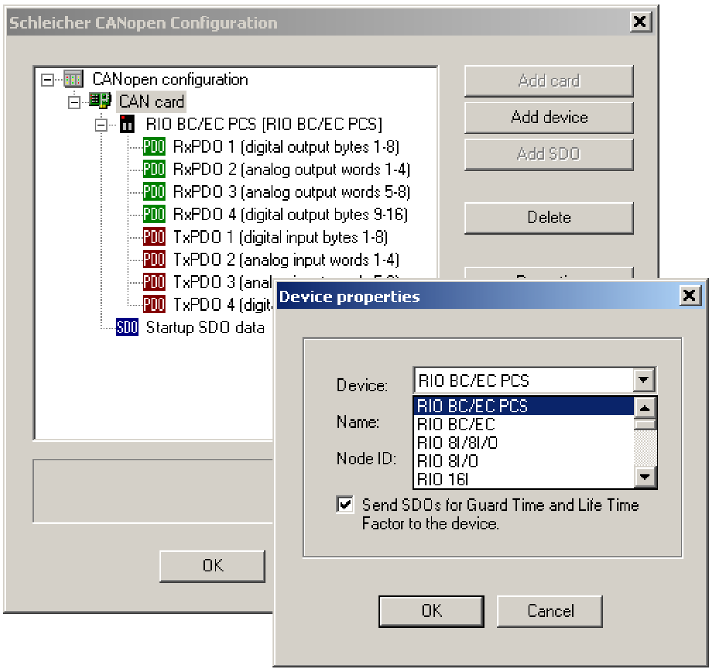

Extras
Window
?
Pagelayout Editor
PLC/MC/CNC shared memory ...
ProCANopen configuration tool ...
Schleicher CANopen configuration ...
XIO configuration ...
Shortcuts...
Options...

Figure 67: Call the "Schleicher CANopen configuration" in MULTIPROG


Schleicher CANopen Configuration

- CANopen configuration
  - CAN card
    - RIO BC/EC PCS [RIO BC/EC PCS]
      - RxPDO 1 (digital output bytes 1-8)
      - RxPDO 2 (analog output words 1-4)
      - RxPDO 3 (analog output words 5-8)
      - RxPDO 4 (digital output bytes 9-16)
      - TxPDO 1 (digital input bytes 1-8)
      - TxPDO 2 (analog input words 1-4)
      - TxPDO 3 (anal
      - TxPDO 4 (digit
      - Startup SDO data

Device properties

Device: RIO BC/EC PCS

RIO BC/EC PCS
RIO BC/EC
RIO 8I/8I/O
RIO 8I/O
RIO 16I

Name:

Node ID:

☑ Send SDOs for Guard Time and Life Time Factor to the device.

Figure 68: Schleicher CANopen configuration

One or more CAN cards can be configured in the configuration window (according to the controller). CAN devices such as bus couplers or drives can easily be added using a selection list. The receive and send PDOs supported by the device are automatically added. The created configuration is saved and all required changes to the PLC project are made (incl. generation of the PLC program code).

As the configurator operates offline, a direct error diagnosis with the tool itself is not possible. However, PLC program code and variables are generated that allow the configuration of the CANopen network to be checked after the start of the PLC. The values of these variables can be checked in the online mode of MULTIPROG.

6.8.7 CANopen Configuration with "ProCANopen"

ProCANopen is a configuration tool from Vector Informatik GmbH.

Complex CANopen networks can be configured with the **ProCANopen configuration tool**. The "ProCANopen" software is required for this. It is not within the scope of delivery of MULTIPROG. A CAN field bus card is also required in the service PC, e.g. "CANcardY" (→ page 128).

The properties and capabilities of the components are declared in an EDS file (**E**lectronic **D**ata **S**heet). The EDS file must be copied to the subdirectory of ProCANopen with the name EDS.

ProCANopen maps the (mapable) objects of the nodes. For example, output bytes of the RIO modules (that represent the input bits of RIO) are linked with input bytes of the XCx. Additional information for the field bus is also configured:

- Which node is the "NMT manager"?
- Which node is the "configuration manager"?
- "Guarding" and "guarding time".
- "Sync time" and "sync window length".

After the network has been configured they can be saved in the network. Saving in the network means: the node selected as "configuration manager" (normally XCx) receives the information about how the network is to be configured via CANopen. The "configuration manager" saves the information (e.g. XCx on flash disk) and the XCx configures the network when it is switched on.

Once the network has been configured, the "NMT manager" can start the CAN network. Starting the network means: Status "operational" and data exchanged from PDOs (**P**rocess **D**ata **O**bjects).

6.8.8 Installation of ProCANopen

Note

For users who have already installed ProCANopen version 2.1 and MULTIPROG 1.2 for Schleicher MicroLine and ProNumeric/ProSycon controllers:

- **You only need the update version of ProCANopen.**
- **Do not overwrite the installed version!**
- **Install ProCANopen V3.2 in a new path on the hard disk, e.g. \ProCANopen3.**
- **If you are using a CANCardX you may need to update the firmware and the options on the card. Please note the serial number of the card and contact your local supplier.**
- **Depending on your PC operating system, you may need to install a number of drivers. Some new drivers are incompatible with the older ProCANopen version 2.1. This means that ProCANopen version 2.1 with the new V3.x drivers will not function online with the field bus.**
- **Configuration files configured with ProCANopen V2.1 can still be used with ProCANopen V3.2.**

To install ProCANopen, please follow the documentation supplied with the software and the CAN card. You have to install the drivers and the ProCANopen software in two steps.

- Copy the latest EDS files for the controller type of the service CD to the *\...\ProCANopen\EDS* directory.
- If you want English as the dialog language, open the *\....\ProCANopen\EXE\VECTOR.INI* file and change the line "language=0049" to "language=001".

6.8.9 Integrationn of ProCANopen in MULTIPROG

The installation of the add-ons is prepared by MULTIPROG so that ProCANopen can be started directly from MULTIPROG. To do this, select the configuration tool in the *Extras / ProCANopen* menu (→ Figure 69).

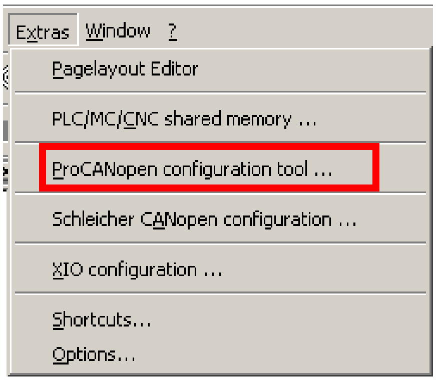
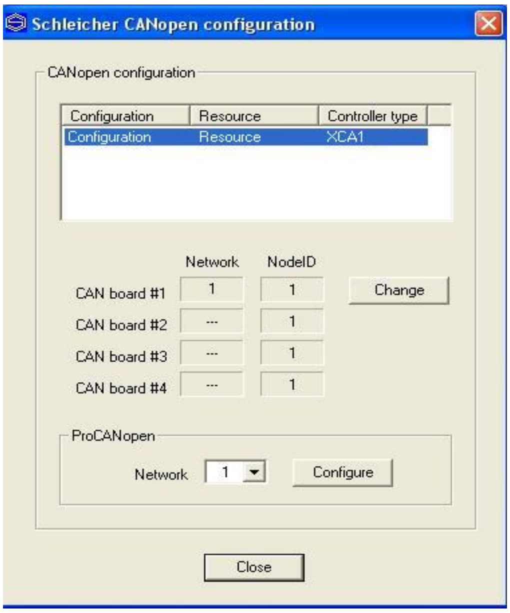

Extras
Window
?
Pagelayout Editor
PLC/MC/CNC shared memory ...
ProCANopen configuration tool ...
Schleicher CANopen configuration ...
XIO configuration ...
Shortcuts...
Options...

Figure 69: Call the ""ProCANopen configuration tool" in MULTIPROG

Select the CAN card of the XCx. In the example, only the single standard card is used. Node number (NodeID) 1 can be retained. Click the *Configuration* button to start ProCANopen (→ Figure 70).


Schleicher CANopen configuration

CANopen configuration

Configuration	Resource	Controller type
Configuration	Resource	XCA1

	Network	NodeID
CAN board #1	1	1
CAN board #2	---	1
CAN board #3	---	1
CAN board #4	---	1

ProCANopen

Network: 1

Figure 70: Select the CAN card and start ProCANopen

ProCANopen starts directly with the correct CANopen project (→ Figure 71).

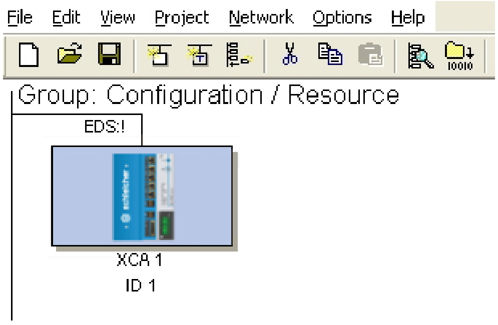
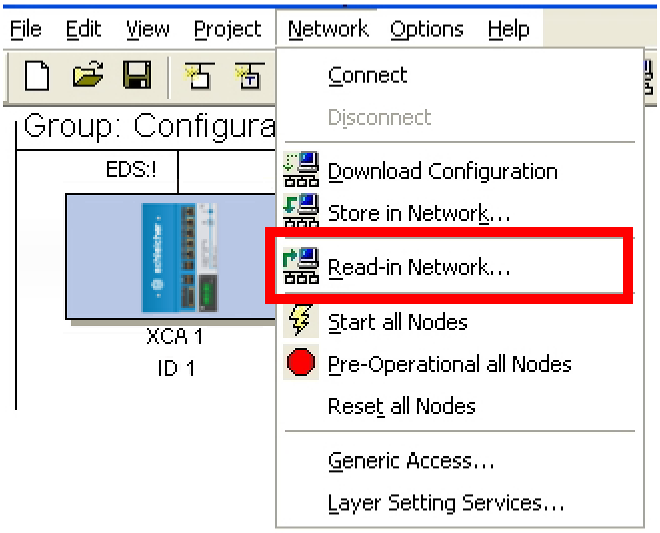


Figure 71: ProCANopen with current CANopen project

6.8.10 First Connections with ProCANopen

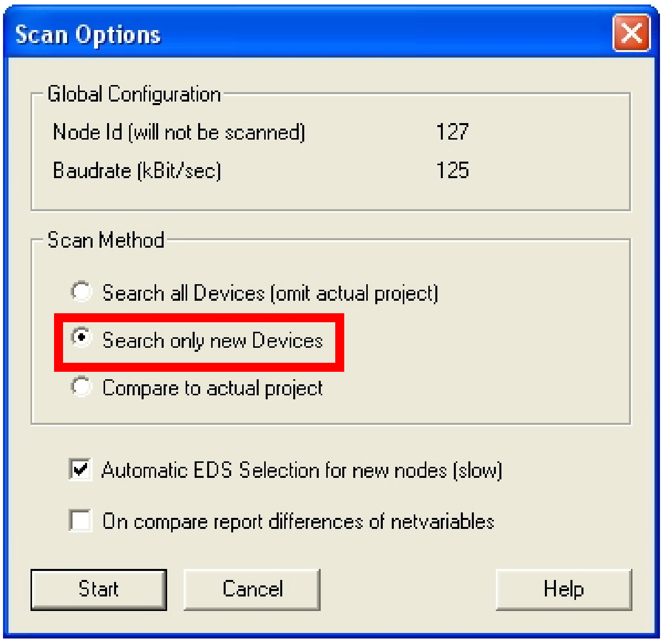
First you have to load the network (→ Figure 72).

Figure 72: Load the network with ProCANopen



Because the network is already configured with "node 1 XCA" you have to load with the *Search only new devices* scan option (→ Figure 73).

Figure 73: Setting the scan options


Scan Options

Global Configuration

Node Id (will not be scanned) 127

Baudrate (kBit/sec) 125

Scan Method

○ Search all Devices (omit actual project)

● Search only new Devices

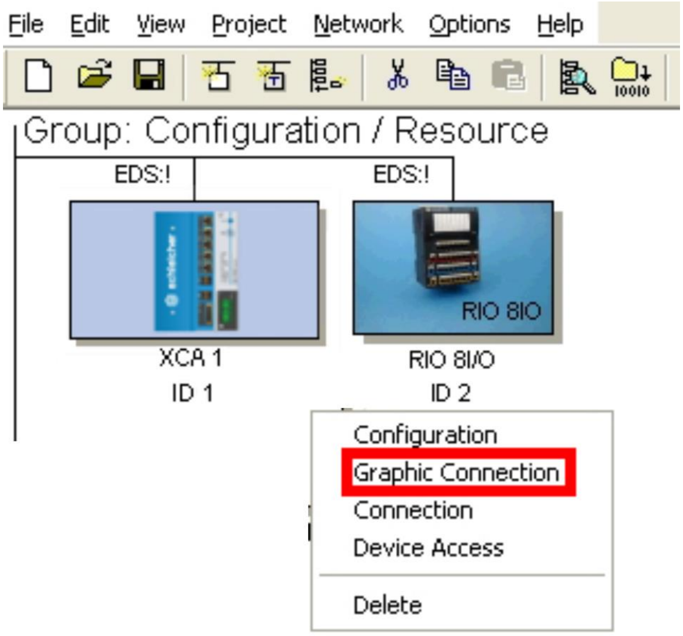
○ Compare to actual project

☑ Automatic EDS Selection for new nodes (slow)

☐ On compare report differences of netvariables

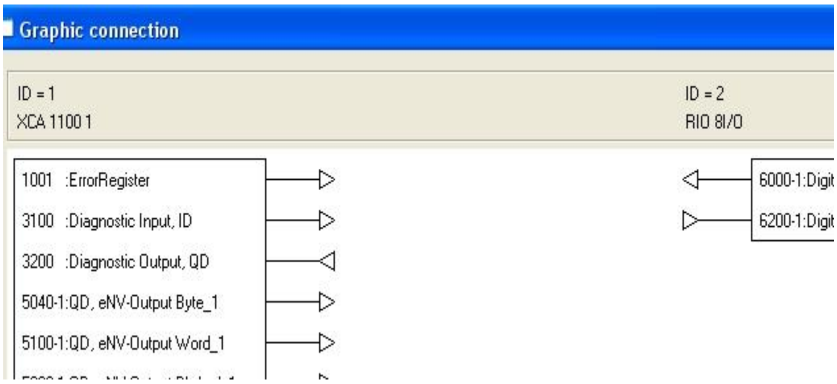
Then you can configure the network node connections. Right-click on the XCx, select *Graphic connection* in the context menu of the node, then click on the node to which you wish to connect (in the example 2 RIO 8 I/O) (→ Figure 74).

Figure 74: Graphic connection of the network nodes



The nest figure shows an example of the communication relationships between the controller and the I/O module (→ Figure 75).

Figure 75: Communication - relationship between controller and I/O module



Graphic connection

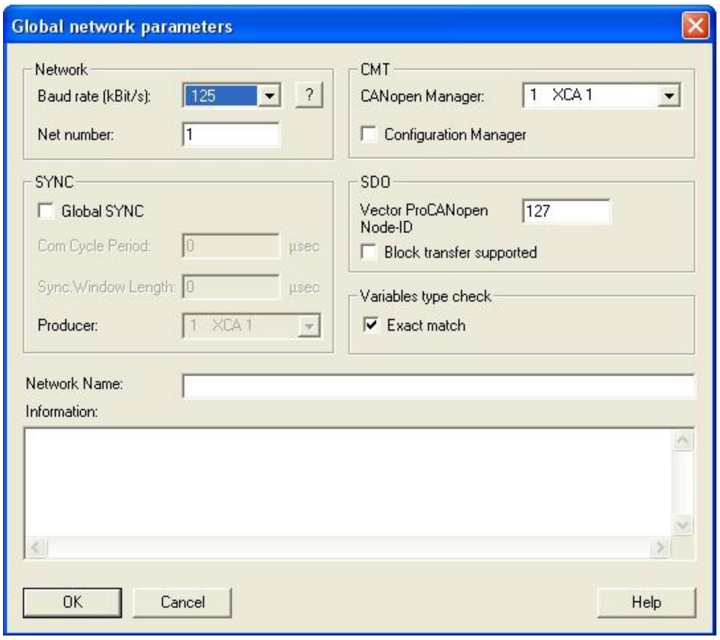
ID = 1
XCA 1100 1

ID = 2
RIO 8I/O

1001 :ErrorRegister
3100 :Diagnostic Input, ID
3200 :Diagnostic Output, QD
5040-1:QD, eNV-Output Byte_1
5100-1:QD, eNV-Output Word_1

6000-1:Digit
6200-1:Digit

Select the CANopen manager XCA 1 as the configuration manager in *Project / Global configuration* (→ Figure 76).



Global network parameters

Network

Baud rate (kBit/s): 125 ?

Net number: 1

CMT

CANopen Manager: 1 XCA 1

□ Configuration Manager

SYNC

□ Global SYNC

Com Cycle Period: 0 µsec

Sync.Window Length: 0 µsec

Producer: 1 XCA 1

SDO

Vector ProCANopen Node-ID 127

□ Block transfer supported

Variables type check

☑ Exact match

Network Name:

Information:

OK Cancel Help

Figure 76:
Select the configuration manager

Save in network saves the CAN configuration in the configuration - manager (→ Figure 77). The XCx saves the data in the Compact Flash. When it is switched on the network is booted.

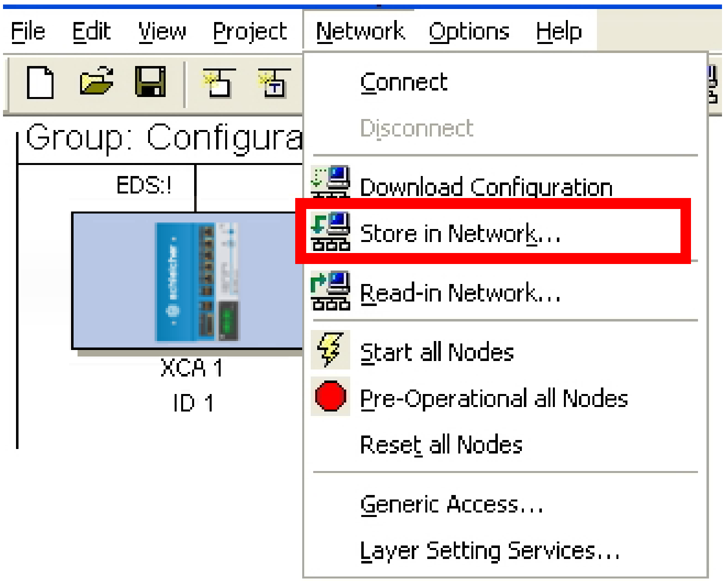


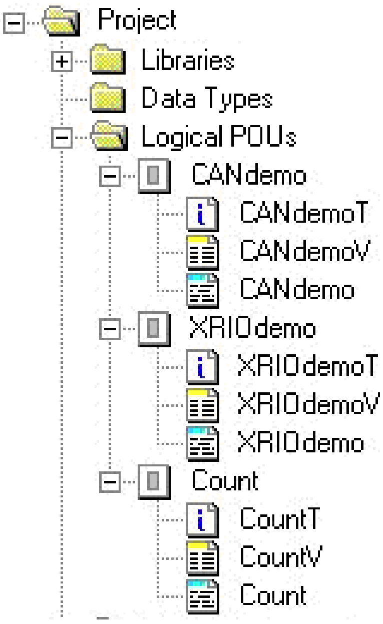
Figure 77:
Save the CAN configuration in the configuration manager

Note

Further information on the CAN configuration can be found in the "Commissioning Field Bus Systems" operating manual (→ 10).

To test the network connection, you must create a new POU (here *CANdemo*) and instance it in *CanTsk* (→ Figure 78).

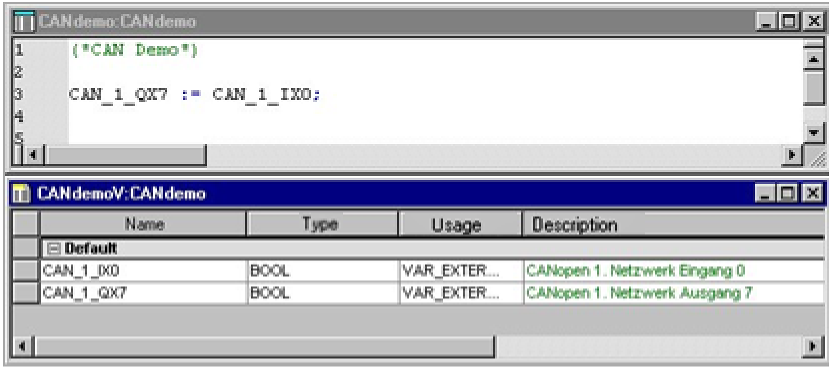
Figure 78:
Test the network connection with POU "CANdemo"



- Project
  - Libraries
  - Data Types
  - Logical POUs
    - CANdemo
      - CANdemoT
      - CANdemoV
      - CANdemo
    - XRIOdemo
      - XRIOdemoT
      - XRIOdemoV
      - XRIOdemo
    - Count
      - CountT
      - CountV
      - Count

The POU *CANdemo* with the associated variables worksheet (→ Figure 79):

Figure 79:
POE CANdemo with associated variables worksheet



CANdemo:CANdemo

```
(*CAN Demo*)

CAN_1_QX7 := CAN_1_IX0;
```

CANdemoV:CANdemo

Name	Type	Usage	Description
⊟ Default			
CAN_1_IX0	BOOL	VAR_EXTER...	CANopen 1. Netzwerk Eingang 0
CAN_1_QX7	BOOL	VAR_EXTER...	CANopen 1. Netzwerk Ausgang 7

If 24 V is connected to input 0 of the RIO 8 I/O CANopen 24 V, output 7 is set to 1.

6.9 The Web Server

6.9.1 General Functions and Concept

- The main advantage of web server technology is the storage of the complete visualisation application (HTML, JavaScript) on the controller. Additional configuration in an operator panel is not required.
- The web server is integrated in the operating system of the XCx.
- The browser is the "thin client" for data visualization.
- Other visualisation systems must be installed separately on each operating panel. This concept is known as a "fat client".

6.9.2 Schleicher-specific Applet

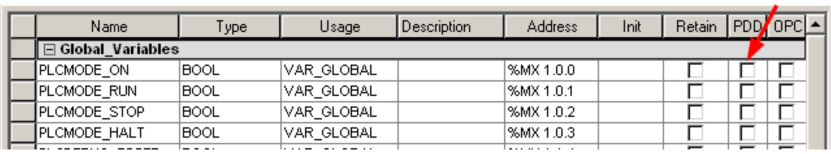
Normally web technology means single-direction downloading to the browser, and the web page itself is dynamic (animation gifs or flash files). Cyclic parameter refresh is not possible.

Schleicher supplies a special Java Applet to allow bidirectional data exchange between the browser and the controller. This applet supports functions that can be called by the HTML/ JavaScript language. These functions allow the application to write one or more PLC variable values.

6.9.3 Declaration of Variables for Visualisation

The variables that are to be visualised must be marked in MULTIPROG by the *PDD* checkbox (PDD = **P**rocess **D**ata **D**irectory) (→).

Figure 80: Declaration of PDD-variables in MULTIPROG



Name	Type	Usage	Description	Address	Init	Retain	PDD	OPC
⊟ **Global_Variables**								
PLCMODE_ON	BOOL	VAR_GLOBAL		%MX 1.0.0		□	□	□
PLCMODE_RUN	BOOL	VAR_GLOBAL		%MX 1.0.1		□	□	□
PLCMODE_STOP	BOOL	VAR_GLOBAL		%MX 1.0.2		□	□	□
PLCMODE_HALT	BOOL	VAR_GLOBAL		%MX 1.0.3		□	□	□

These variables are updated in an internal list in the XCx. The web server can read and write the variables in this list.

6.9.4 Application example

The XCx is supplied with a standard browser application. This application allows you to read and write PDD-marked variables. A status overview is also provided.

6.9.5 Browser / Components

- You can use any standard PC with Ethernet.
- MS Internet Explorer v5 or higher is required.
- Certain terminals with Windows CE can be used, if the browser meets the requirements for Java Script 1.5, Java 2, HTTP1.1.

6.10 General information on Commissioning

Default Initialisation

To ensure operation of the real-time operating system for faulty PLC projects, Q parameters or invalid data in the retentive data memory (e.g. when the buffer battery fails), the controller can start in safe mode with the aid of the so-called default initialisation. The default initialisation causes a SRAM reset.

Here there is an option for a diagnosis of the controller data.

Perform the Default Initialisation

Please note: A timeout of 4 s applies for all transitions.

- Move operating mode switch to position "0"
- Switch on the controller (or "START XCA 11xx" in MFA tool).
- The simultaneous yellow flashing of all LEDs indicates the start of "Default init." detection.
- Operating mode switch now in position "9". Now only the *RUN/ERR* and *PLC RUN* LEDs continue to flash red.
- Switch back operating mode switch to position "0". All LEDs now flash red and indicate the performance of the default initialisation.

After default initialisation, the controller is started in "Safe mode". In "safe mode, neither the PLC nor the CNC run but access to the controller is possible via FTP.

Note

If the rotary switch is left in position "0" after switching on the controller, the PLC boot project is not loaded and the PLC does not start.

7 Operation

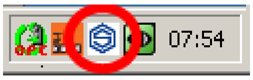
7.1 Multi Function Application MFA

MFA (Multi Function Application) is the basis for operating the controller. With MFA you can set the start behaviour and start and stop the controller and the PLC.

7.1.1 Start the MFA

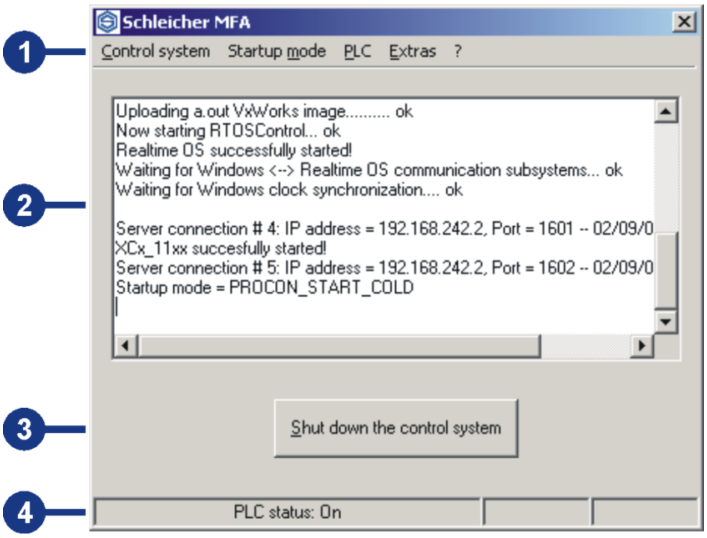
MFA starts automatically when the controller starts and is entered in the system tray of the taskbar (→ Figure 81). You can activate it from there by double-clicking on the blue Schleicher logo.

Figure 81: Taskbar with Schleicher logo



7.1.2 The MFA window

Figure 82: MFA window



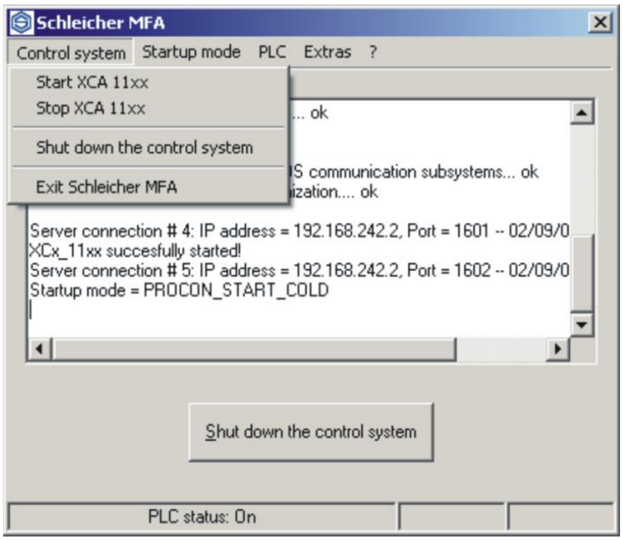
```
Uploading a.out VxWorks image.......... ok
Now starting RTOSControl... ok
Realtime OS successfully started!
Waiting for Windows <--> Realtime OS communication subsystems... ok
Waiting for Windows clock synchronization.... ok

Server connection # 4: IP address = 192.168.242.2, Port = 1601 -- 02/09/0
XCx_11xx succesfully started!
Server connection # 5: IP address = 192.168.242.2, Port = 1602 -- 02/09/0
Startup mode = PROCON_START_COLD
```

1	**Menu** For an explanation, see "MFA functions".
2	**Messages** With information on memory, real-time operating system and controller software.
3	**Buttons**
4	**Status** Information on PLC status.

7.1.3 MFA functions

"Control system" menu

Figure 83:
MFA, "Control system" menu



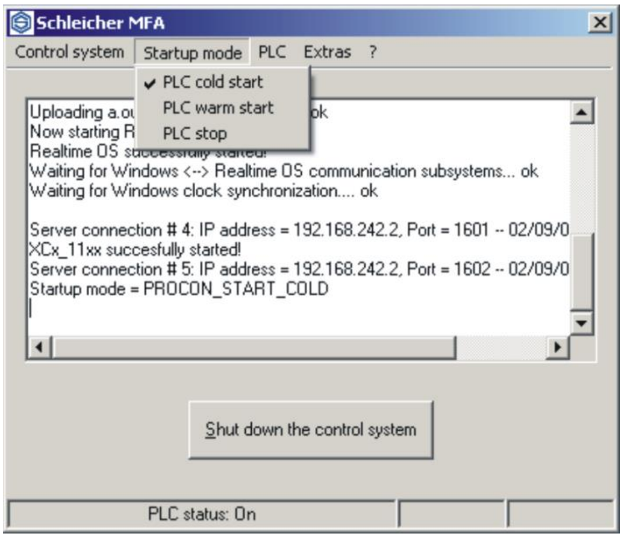
```
... ok
S communication subsystems... ok
ization.... ok
Server connection # 4: IP address = 192.168.242.2, Port = 1601 -- 02/09/0
XCx_11xx succesfully started!
Server connection # 5: IP address = 192.168.242.2, Port = 1602 -- 02/09/0
Startup mode = PROCON_START_COLD
```

The *Start ... / Stop ...* menu items are used to start and stop the real-time operating system and the controller software.

The *Shut down the control system* menu item has the same function as the <Shut down the control system> button. The controller software including the PC operating system is shutdown and the controller is switched off.

"Startup mode" menu

Figure 84:
MFA, "Startup mode" menu

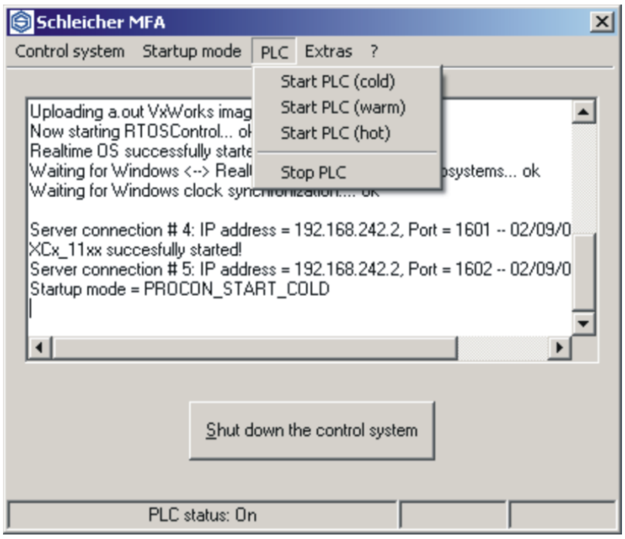


```
Uploading a.ou
Now starting R
Realtime OS successfully started!
Waiting for Windows <--> Realtime OS communication subsystems... ok
Waiting for Windows clock synchronization.... ok

Server connection # 4: IP address = 192.168.242.2, Port = 1601 -- 02/09/0
XCx_11xx succesfully started!
Server connection # 5: IP address = 192.168.242.2, Port = 1602 -- 02/09/0
Startup mode = PROCON_START_COLD
```

In the *Startup mode* menu, you can set how the PLC behaves after controller startup. The start behaviour is described in detail in section "The PLC" (→ page 92).

"PLC" menu

Figure 85: MFA, "PLC" menu

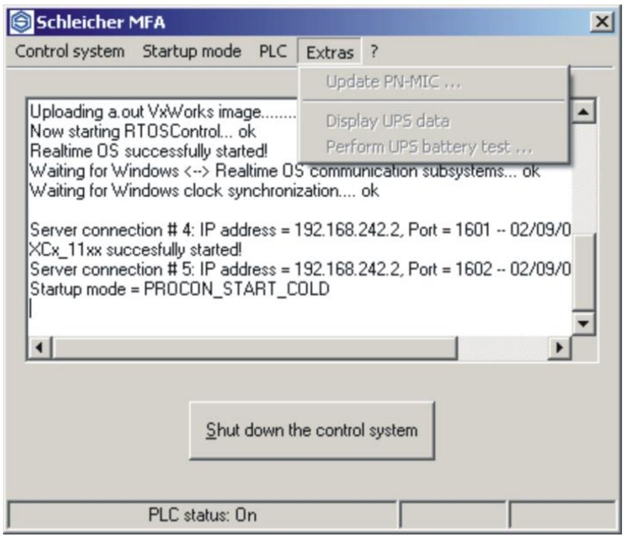


```
Uploading a.out VxWorks imag
Now starting RTOSControl... ok
Realtime OS successfully starte
Waiting for Windows <--> Reall                    bsystems... ok
Waiting for Windows clock synchronization.... ok

Server connection # 4: IP address = 192.168.242.2, Port = 1601 -- 02/09/0
XCx_11xx succesfully started!
Server connection # 5: IP address = 192.168.242.2, Port = 1602 -- 02/09/0
Startup mode = PROCON_START_COLD
```

Only the PLC is started and stopped in the *PLC* menu. The start behaviour is described in detail in section "The PLC" (→ page 92).

"Extras" menu

Figure 86: MFA, "Extras" menu



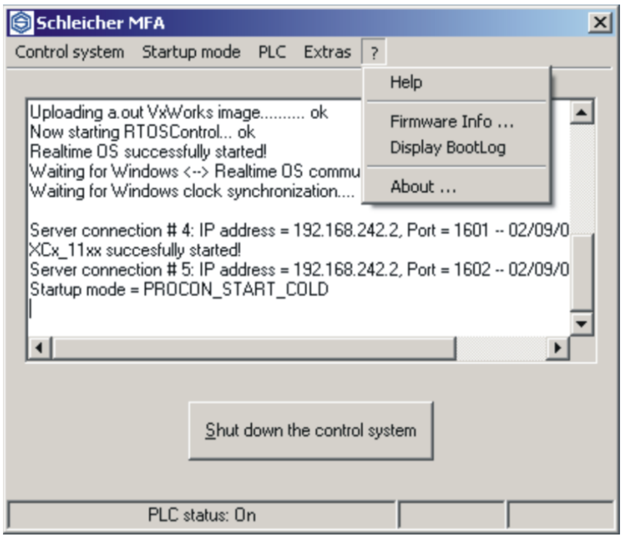
```
Uploading a.out VxWorks image........
Now starting RTOSControl... ok
Realtime OS successfully started!
Waiting for Windows <--> Realtime OS communication subsystems... ok
Waiting for Windows clock synchronization.... ok

Server connection # 4: IP address = 192.168.242.2, Port = 1601 -- 02/09/0
XCx_11xx succesfully started!
Server connection # 5: IP address = 192.168.242.2, Port = 1602 -- 02/09/0
Startup mode = PROCON_START_COLD
```

The state of a UPS (USV / uninterruptable power supply) that may be connected via USB is displayed and tested in the *Extras* menu. UPS from the Eaton Powerware Company (e.g. Powerware 5115 500 VA) is currently supported. Support for the 24-V-UPS is in preparation.

A PN-MIC PCI card located in the system can also be addressed (currently only ProNumeric/ ProSycon).

"?" Menu

Figure 87: MFA, "?" menu



```
Uploading a.out VxWorks image.......... ok
Now starting RTOSControl... ok
Realtime OS successfully started!
Waiting for Windows <--> Realtime OS commu
Waiting for Windows clock synchronization....

Server connection # 4: IP address = 192.168.242.2, Port = 1601 -- 02/09/0
XCx_11xx succesfully started!
Server connection # 5: IP address = 192.168.242.2, Port = 1602 -- 02/09/0
Startup mode = PROCON_START_COLD
```

The MFA documentation can be called via the *?* menu. The help file contains information on the program settings and command line parameters as well as programming examples.

The *Display BootLog* menu item displays the Bootlog file of the real-time operating system VxWorks that is generated when the controller starts up.

Memory location and name of file: *SCHLEICHER/Os/Log/bootlog.txt*

7.1.4 The Log File of the MFA

MFA generates a log file with error messages and boot protocol from the real-time operating system VxWorks. These files are generated with each restart of the controller. If the controller is in continuous operation, the LogBook_xx.txt file is saved every 24 hours (default setting). A max of 10 files of type BootLog_xx.txt and LogBook_xx.txt are stored in the log folder. The current log file can be determined in the FileList.ini file. The file name _xx corresponds to the content of the count (e.g. count =5)

Memory location and name of file:

SCHLEICHER/Os/Log/ BootLog_05.txt

SCHLEICHER/Os/Log/ Logbook_05.txt

Note

The log files are intended primarily for diagnosis investigations by the controller manufacturer.

7.1.5 Communication with other Applications

MFA has an interface for communication with other Windows applications (e.g. a visualisation program). The MFA communicates with another application via a registered Windows message. By calling the Windows API function *RegisterWindowMessage()* with the string "Schleicher MFA" as the parameter, an application receives the numeric value of the Windows message that is required for communication with MFA.

Further information and a programming example are given in the MFA help (→ "?" Menu).

7.2 Schleicher Dialog

The Schleicher dialog provides all the dialogs for operating the CNC and PLC. The Schleicher Dialog is permanently installed on the controller and starts automatically after controller startup.

7.2.1 Structure of the User Interface

Figure 88: Schleicher Dialog, start window

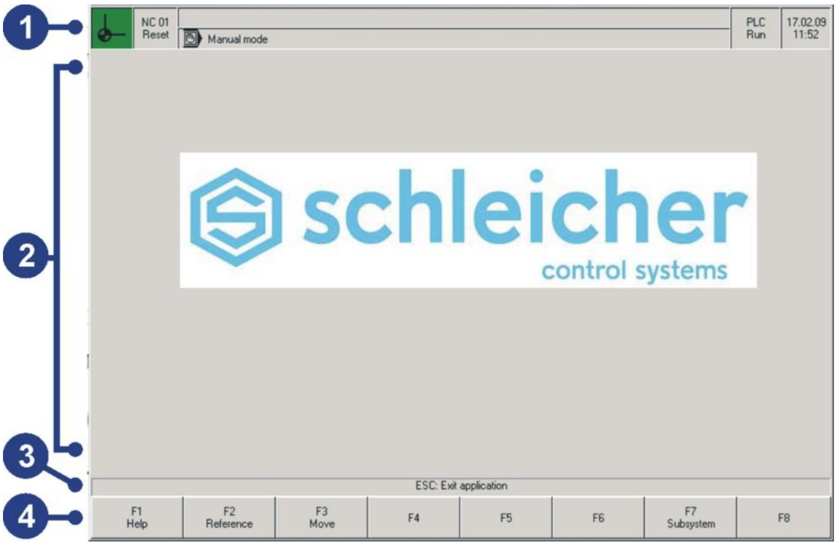


Table 18: Schleicher Dialog, division of the user interface

Range	Meaning
1	Status and messages
2	Workspace for settings and information
3	Hints
4	Softkeys with functional information

Figure 89: Schleicher Dialog, status and messages

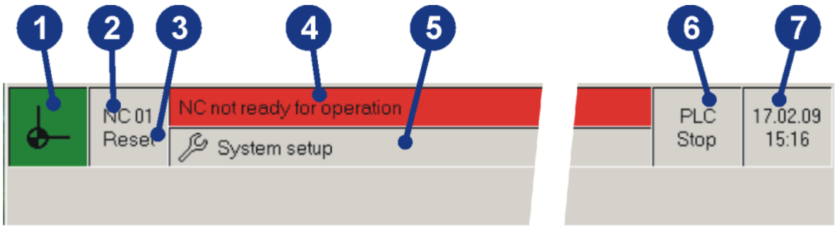


Table 19: Schleicher Dialog, status and messages

Range	Meaning
1	Display current operating mode
2	Selected NC sub-system
3	Current NC state
4	Message window
5	Current position in controller menu
6	PLC status
7	Date and time

7.2.2 Schleicher Dialog PLC/CNC

Controller Menu and Operating Elements of the XCx

The highest level of the controller menu consists of the operating elements that are oriented towards important activities for the machine (manual mode, automatic, programming, etc.). They are called with the key combination <Ctrl + Function key>.

Associated options are called using the subordinated softkey level (function keys F1..F8).

Softkey *F1* is always used for calling help pages. The help page contains further information on the content of the subsequent operating levels.

Manual mode <Ctrl+F1>

F1 Help	F2 Reference	F3 Procedure	F4	F5	F6	F7 Sub-system	F8
	Axes	Jog					
		Incremental					
		Target position					
		Handwheel					
		Axis					
		Rapid feed					
		Zero set					
		Override					

Automatic <Ctrl+F2>

F1 Help	F2 Program	F3	F4	F5	F6	F7 Sub-system	F8
	Activate						
	MDI						
	Single record						
	Block record						
	Record sequence						
	Rapid feed						
	Override						

Programming <Ctrl+F3>

F1	F2	F3	F4	F5	F6	F7	F8
Help	CNC programs	R parameters	Tool data	Zero point offsets	Coordinate system		
	Edit program	Change value	Change value	Change value	Change value		
	New program	Edit					
	Activate program	New					
	Copy program	Delete					
	Delete program						
	Protect program						
	New Project						
	Activate project						
	Copy project						
	Delete project						
	Protect project						
	View						
	Edit						
	New						
	Transfer						
	Update						
	Delete						
	Directory						

Access authorisation <Ctrl+F4> (also applies for XCS)

F1	F2	F3	F4	F5	F6	F7	F8
Help							

Start external software <Ctrl+F5> (also applies for XCS)

F1	F2	F3	F4	F5	F6	F7	F8
Help							
	Start application 1	Start application 2	Start application 3	Start application 4	Start application 5	Start application 6	Start application 7

Commissioning <Ctrl+F6> (Softkey level 1)

				(also applies for XCS)			
F1	F2	F3	F4	F5	F6	F7	F8
Help	Default setting	CNC system	Drive configuration	Boot settings	OPC variables	Info	More >>
	Edit	Edit	Edit	Edit	Change value		
		Display mode	Drive parameter	CAN settings	Edit		
		Axis setting	DriveTop		New		
					Delete		

Commissioning <Ctrl+F6> (Softkey level 2)

(also applies for XCS)				(also applies for XCS)			
F1	F2	F3	F4	F5	F6	F7	F8
Help	Program settings	PLC/CNC options	Logging	Set clock	Connect	Info	<< Back
	Properties	Enable options	Transfer				
	Add resource		Add system parameter				
	Delete resource		Delete system parameter				
	Resource up		Add drive parameter				
	Resource down		Delete drive parameter				

7.2.3 Calling Active Error Buffer and Logbook

The error messages in the active error buffer and logbook can be called on each operating level via the key combination <Ctrl+?>.

Error <Ctrl+?>

F1 Help	F2 Logbook	F3 Bootlog	F4 Problem - report	F5	F6	F7	F8
	Opening						
	Saving						
	Printing						
	Update						
	Delete						

8 The PLC

- Operating system: ProConOS
- Programming: MULTIPROG acc. to IEC 61131-3
- Communication with the CNC via shared RAM

8.1 Programming

The XCx is programmed on a PC using the MULTIPROG programming software to IEC 61131-3.

Important!

The programming software consists of the MULTIPROG software and the add-ons for MULTIPROG from Schleicher.

The programming system and programming instructions can be ordered as accessories (→ page 128).

The PLC is supplied with a ready-configured project, which you can use as the basis for programming the PLC (→"First Steps with MULTIPROG", page 45).

8.2 PLC Operating States and Start Behaviour

8.2.1 Operating States

Operating state	Description
ON	• No program loaded
STOP	• Program loaded • User tasks inactive • Process map memory inputs are not updated • Output signals are not sent to inputs and outputs
RUN	• Program execution active • User tasks active • Process map memory inputs updated according to I/O configuration • Process map memory outputs updated according to I/O configuration and program execution
HALT	• Program execution stopped at a breakpoint • User tasks inactive • Process map memory inputs are not updated • Process map memory outputs are not updated

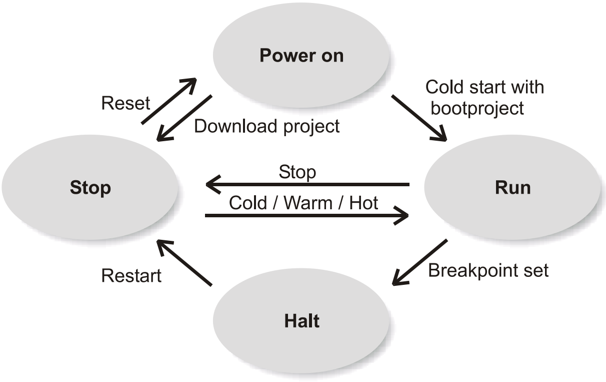
Table 20: PLC operating states

The current state of the PLC is displayed in the project control dialog in the *Status* line in MULTIPROG. If “debug” is displayed behind the current state in the control dialog it means that breakpoints have been set or variables forced.

8.2.2 Change the Operating States with MULTIPROG

You can use the graphic user interface of MULTIPROG to control when program execution on the PLC starts and stops. The buttons for changes which are not possible in the current operating state are shaded in the project control dialog (→ Figure 90).

Figure 90: MULTIPROG, change operating states



Start program execution

State change from → to	Button in control dialog	Description of what happens
Stop → Run	Cold	• Cold start • All data is initialised • SPG 1 is called • All user tasks are activated • Program execution activated
Stop → Run	Warm	• Warm start • Only non-buffered data is initialised • SPG 0 is called • All user tasks are activated • Program execution activated
Stop → Run	Hot	• Hot start • No data is initialised • All user tasks are activated • Program execution activated • Not available when you start program execution for the first time after downloading

Table 21: MULTIPROG, start program execution

Stop program execution

State change from → to	Button in control dialog	Description of what happens
Run → Stop		- All user tasks are deactivated when their operating cycle is complete - SPG 2 is called - Process map memory outputs are written - Program execution stops - Physical outputs are set to zero or preferred shut-off state

Table 22: MULTIPROG, stop program execution

General reset

State change from → to	Button in control dialog	Description of what happens
Stop → On		- The project is deleted - General reset

Table 23: MULTIPROG, general reset

8.2.3 PLC starting behaviour after power supply is switched on

The PLC starting behaviour is set with the operating mode switch. The following options are available:

Table 24: Operating mode switch

	Position / designation	Meaning
	0	Default initialisation / diagnosis (Start of the real-time operation system in safe mode and reset of retentive data memory, → p. 81)
	1 / prog	Programming mode (PLC stop)
	2 / Warm (also 4..9)	PLC warm start to IEC 61131-3 (default setting)
	3 / Cold	PLC cold start to IEC 61131-3 (Reinitialisation of the retain variables)

8.3 System Variables

System variables provide information about the status of the system, for example about forced variables, CPU performance, etc. These variables have fixed memory addresses and can be used by the PLC program to obtain the corresponding information.

All the system variables in the following table are already declared in the *Global_Variables* area of the *Global_Variables* worksheet.

Name	Data type	Log. addr. (byte)	Log. addr. (bit)	Description
PLCMODE_ON	BOOL	0	0	TRUE := current PLC state is ON
PLCMODE_RUN	BOOL	0	1	TRUE := current PLC state is RUN
PLCMODE_STOP	BOOL	0	2	TRUE := current PLC state is STOP
PLCMODE_HALT	BOOL	0	3	TRUE := current PLC state is HALT
PLCDEBUG_BPSET	BOOL	1	4	TRUE := one or more breakpoints have been set
PLCDEBUG_FORCE	BOOL	2	0	TRUE := one or more variables have been forced
PLCDEBUG_POWERFLOW	BOOL	2	3	TRUE := powerflow active
PLC_TICKS_PER_SEC	INT	44	-	Number of system ticks per second, used by the PLC as the basis for the system time. This value determines the time resolution of the PLC for time delay function blocks like TON, TOF and TP, and the shortest cycle time for the DEFAULT task and cyclical tasks.
PLC_SYS_TICK_CNT	DINT	52	-	Number of counted PLC system ticks

Table 25: System variables

As well as these system variables, other variables are also defined, containing information on the system. The type definitions of the variables can be found in the *PLC_Types* section of the "SchleicherLib" library.

8.4 Libraries and Function Blocks in MULTIPROG

Function blocks are combined in libraries. They are automatically integrated according to the controller type when a new MULTIPROG project is created or can be manually integrated if required.

Important!

The latest version of the libraries that match the controller operating system must always be used.

Libraries	XCA 11xx	XCN 7xx	XCS 7xx	XCN 5xx	XCS 5xx	XCN 3xx	XCS 3xx	MCS 2x	ProNumeric	ProSyCon	MCS 20-20	MCS 20-21	Simulation
PROCONOS	+	+	+	+	+	+	+	+	+	+	+	+	+
BIT_UTIL	+	+	+	+	+	+	+	+	+	+	+	+	+
CANopen_Vxxx	+	+	+	+	+	o	o	o	+	+	–	+	–
CFB_Vxxx	o	o	o	o	o	o	o	–	o	o	–	–	–
CNC_Vxxx	+	+	–	+	–	+	–	–	+	–	–	–	–
Date_Time	+	+	+	+	+	+	+	o	+	+	+	+	–
MC_Vxxx	o	+	–	+	–	+	–	–	–	–	–	–	–
Microline	-	–	–	–	–	–	–	+	–	–	+	+	–
MMI	+	o	o	o	o	o	o	-	o	o	o	o	–
PLC_Vxxx	+	+	+	+	+	+	+	o	+	+	–	–	–
PNS_Vxxx	-	–	–	–	+	+	–	–	–	–	–	–	–
Profibus_Vxxx	o	o	o	o	o	o	o	-	o	o	–	–	–
SchleicherLib_Vxxx	+	+	+	+	+	+	+	-	+	+	–	–	–
Serial	o	o	o	o	o	o	o	o	o	o	o	o	–
XCx7_Vxxx	+	+	+	–	–	–	–	–	–	–	–	–	–

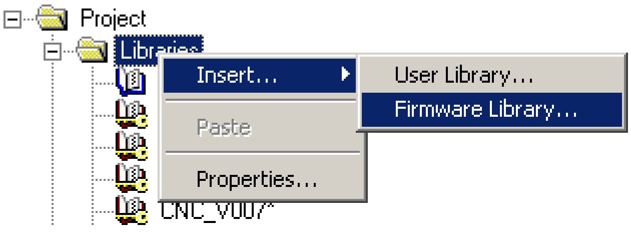
+ Automatically integrated when creating a new project.
o Can be manually integrated if required.
– Not possible or unnecessary.

Table 26: Libraries and function blocks in MULTIPROG

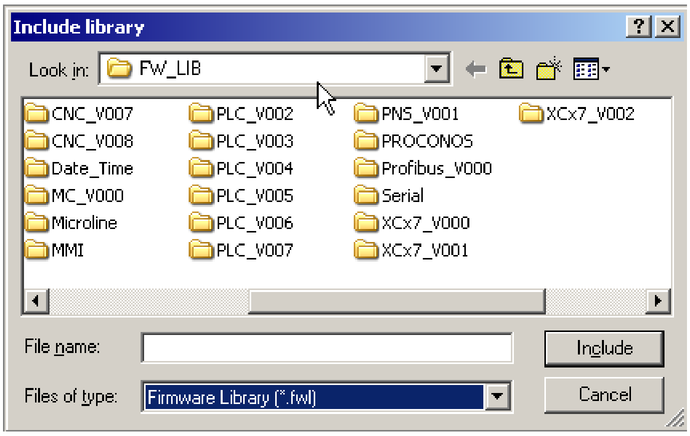
Function blocks can be integrated as follows:

- In the project tree of MULTIPROG, right click and open the *Libraries/ Insert/ Library* context menu (→ Figure 91).

Figure 91: MULTIPROG, "Libraries" context menu



- Select the path *.\KWSoft\MWT\PLC\FW_LIB* and the file type *Firmware library(*.fwl)* (→ Figure 92).

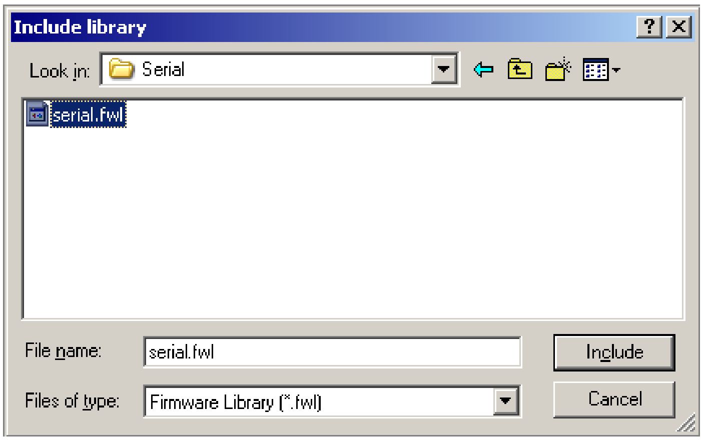


File name:

Files of type: Firmware Library (*.fwl)

Figure 92: MULTIPROG, "Integrate library" dialog window

Each library is stored in a separate path. For example, if the *Serial* library is to be integrated, it must be selected in the library path of the same name (→ Figure 93).



File name: serial.fwl

Files of type: Firmware Library (*.fwl)

Figure 93: MULTIPROG, integrate "Serial" library

Online help is available for the libraries (except Schleicher dialog). Online help can be accessed via the context menu of the respective library. The context menu is active by right clicking on the icon of the library.

8.4.1 Information on the Variable Declarations of the Example Programs of FBs

The example programs for the function blocks contain variable declarations to IEC 61131-3 with keywords VAR and END_VAR. If you wish to use the example programs with MULTIPROGRAM, you have to enter the variable declarations manually, in tabular form on the variables worksheet of the respective POU.

8.4.2 CANopen_Vxxx library

The "CANopen_Vxxx" library contains function blocks for parameterizing and diagnosis on the CANopen network.

Function block	No.	Brief description	Controller types
CO_NET_SDO_WRITE	150	Sends a Service Data Object (SDO)	XCx ProNumeric ProSyCon MCS 20-21
CO_NET_SDO_READ	151	Receives a Service Data Object (SDO)	
CO_NET_GET_LOCAL_NODE_ID	152	Returns own node ID	
CO_NET_GET_STATE	153	Supplies current CANopen status	
CO_NET_GET_KERNEL_STATUS	154	Supplies current extended CANopen kernel status	
CO_NET_NMT	155	Sets status of one or all devices in the CANopen network	
CO_NET_RECV_EMY_DEV	156	Reads any emergency messages from a particular network node	
CO_NET_RECV_EMY	157	Reads any emergency messages from any network node	
CO_NET_RECV_ERR_DEV	160	Reads any error messages from a particular network node	
CO_NET_RECV_ERR	161	Reads any error messages from any network node	
CO_NET_SENDL2	162	Sends any CAN Layer 2 messages	
CO_NET_PING	163	Executes a ping on a particular network node	
CO_NET_RESTART_CAN	164	Restarts CANopen communication (e.g. after "bus- off")	
CO_NET_RESTART_ALL	165	Restarts the complete CANopen stack	
CO_NET_SHUTDOWN	166	Stops the CANopen stack	
CO_NET_CAN_SYNC	170	Allows synchronisation between PLC task and CANopen stack	

Table 27: CANopen_Vxxx library

8.4.3 Library CFB_Vxxx

The "CFB_Vxxx" library (based on IEC 61131-5) contains function blocks for peer-to-peer communication via TCP/IP.

Function block	No.	Brief description	Controller types
CONNECT_V	60	Provides a peer-to-peer connection between two stations	XCx ProNumeric ProSyCon
USEND_V	61	Sends any data	
URCV_V	62	Receives any data	

Table 28: Library CFB_Vxxx

8.4.4 CNC_Vxxx library

The "CNC_Vxxx" library contains function blocks for reading and writing system data, SERCOS, XRIO and CAN drive parameters and PROFIBU-DP drive parameters.

Function block	No.	Brief description	Controller types
READ_Q_PARAM_*	200 to 207	Reads a CNC system data parameter	XCN ProNumeric
WRITE_Q_PARAM_*	208 to 215	Writes a CNC system data parameter	
SAVE_Q_PARAM_*	221	Saves the CNC system data parameter to the hard disk	
SAVE_R_PARAM_*	220	Saves the CNC arithmetic parameter to the hard disk	
READ_SERC_PARAM	302	Reads a SERCOS parameter	
WRITE_SERC_PARAM	303	Writes a SERCOS parameter	
SET_SERC_PHASE	304	Sets the SERCOS communication phase	
SET_SERC_COMMAND	308	Runs a SERCOS command	
MC_ANALOG	300	XRIO motion control block (with position controller)	XCN
MC_ANALOG_1_AXIS	307	XRIO motion control block (with position controller)	
READ_AXIS_PAGE	305	Reads a parameter from an axis assigned to a so-called remote page.	
WRITE_AXIS_PAGE	306	Writes a parameter to an axis assigned to a so-called remote page.	
MC_CAN	301	CAN motion control block	Not XCN700 XCN1100
MC_DP	309	PROFIBUS-DP Motion control block	
MC_DP_1_AXIS	310	PROFIBUS-DP Motion control block for an axis	

Table 29: CNC_Vxxx library

The READ_AXIS_PAGE and WRITE_AXIS_PAGE function block were transferred from the XCx7_Vxxx library to the CNC_Vxxx library (from CNC_V006 / XCx7_V002).

8.4.5 Date_Time library

The XCx has a buffered real-time clock with a calendar (which takes leap years into account) and a resolution of 1 second.

You can read and set the time and date using function blocks from the "Date_Time" library.

Function block	No.	Brief description	Controller types
GET_TIME	130	Read time	XCx ProNumeric ProSyCon MCS xx-xx
GET_DATE	128	Read date	
SET_TIME	131	Set time	
SET_DATE	129	Set date	

Table 30: Date_Time library

8.4.6 MC_Vxxx Library

The "MC_Vxxx" library (Motion Control) contains function blocks for programming the motion sequences in the PLC.

Function block	No.	Brief description	Controller types
MC_MoveAbsolute	320	The axis is instructed to drive to an absolute position.	CXN 300 XCN 5xx XCN 700
MC_MoveRelative	321	The axis is instructed to drive along a path.	
MC_MoveAdditive	322	The axis is instructed to drive to an absolute position.	
MC_MoveVelocity	324	Axis is instructed to move with the specified speed	
MC_Home	325	The axis is instructed to reference	
MC_Stop	326	The axis is instructed to terminate axis movement	
MC_Power	327	The axis is instructed to switch on torque (controller enable)	
MC_ReadStatus	328	The status informations of the axes are read	
MC_ReadAxisError	329	Current error number is read	
MC_Reset	330	Reset (error acknowledgement) is performed	
MC_ReadParameter	331	A parameter of the axis is read	
MC_ReadBoolParameter	332	A Boolean parameter of the axis is read	
MC_WriteParameter	333	A parameter of the axis is written	
MC_WriteBoolParameter	334	A Boolean parameter of the axis is written	
MC_ReadActualPosition	335	Current axis position is read	
MC_GetCncAxis	345	The axis is borrowed from the CNC so that it can be moved more easily in the PLC	
MC_ReleaseCncAxis	346	Borrowed axis is returned to CNC	

Table 31: MC_Vxxx Library

8.4.7 MMI library

The "MMI" library realises the communication with an operator panel of the COP family via the serial interface of the controller.

Function block	No.	Brief description	Controller types
PPF_COP_COMM	140	Communicates with a COP operator panel (PNet protocol)	XCx ProNumeric ProSyCon MCS xx-xx

Table 32: MMI library

8.4.8 PLC_Vxxx library

This library provides controller-specific firmware function blocks over and above the standard IEC/ProConOS function blocks.

Function block	No.	Brief description	Controller types
PUT_ERROR	400	Generates a user-defined error message (please do not use any more)	XCx ProNumeric ProSyCon
PUT_ERROR2	401	Generates a user-defined error message	
CLEAR_ERROR	402	Deletes an error message sent with a lock flag	
READ_FILE	405	File access read	
WRITE_FILE	406	File access write	
SEND_MAIL	410	Sends an E-MAIL (SMTP client)	
XFIO_CONFIG	420	XFIO interrupt configuration	XCx
XRIO_STATE	422	XRIO status information	
GET_MTS	430	Supplies current time value in µs ticks	XCx ProNumeric ProSyCon
OPEN_PROFILE	431	Opens a file in INI format	
NEW_PROFILE	432	Creates a new file in INI format	
FLUSH_PROFILE	433	Writes updated file in INI format	
CLOSE_PROFILE	434	Closes a file in INI format	
GET_PROFILE_STRING	435	Reads a string from a file in INI format	
GET_PROFILE_INT	436	Reads an integer value from a file in INI format	
GET_PROFILE_REAL	437	Reads a real value from a file in INI format	
WRITE_PROFILE_STRING	438	Writes a string to a file in INI format	
WRITE_PROFILE_INT	439	Writes an integer value to a file in INI format	
WRITE_PROFILE_REAL	440	Writes a real value to a file in INI format	

Table 33: PLC_Vxxx library

8.4.9 PNS_Vxxx library

The "PNS_Vxxx" library contains function blocks for parameterizing and diagnosis on the PROFINET network.

Function block	No.	Brief description	Controller types
PNSReadIOData	530	Reads IO data	XCx 5xx
PNSWriteIOData	531	Writes IO data	
PNSCommunicating	532	Supplies status of PROFINET connection	

Table 34: PNS_Vxxx library

8.4.10 Profibus_Vxxx library

The "Profibus_Vxxx" library contains function blocks for the communication via the PROFIBUS card.

Function block	No.	Brief description	Controller types
DP_NET_GET_STATE	190	Supplies the status of the PROFIBUS card	XCx ProNumeric ProSyCon
DP_NET_PUT_MSG	191	Sends a message to the message interface of the Hilscher card	
DP_NET_GET_MSG	192	Fetches a message from the message interface of the Hilscher card	

Table 35: Profibus_Vxxx library

8.4.11 SchleicherLib_Vxxx library

The "SchleicherLib_Vxxx" library contains data type definitions of the firmware that is prepared for MULTIPROG. Function blocks are not contained in this library.

8.4.12 Serial library

The "Serial" library contains function blocks for the serial communication of he controllers.

Function block	No.	Brief description	Controller types
PORT_OPEN	135	opens a serial interface	XCx ProNumeric ProSyCon MCS xx-xx
PORT_CLOSE	136	closes a serial interface	
PORT_READ	137	outputs characters to a serial interface	
PORT_WRITE	138	reads characters from a serial interface	
PORT_STATE	139	supplies status information of a serial interface	

Table 36: Serial library

8.4.13 XCx7_Vxxx library

The "XCx7_Vxxx" library contains function blocks that are exclusively used for the XCx 700 and XCx 1100 controller types.

Function block	No.	Brief description	Controller types
UZB_VR	250	Function block for the operation of the UZB 2VR modules	XCx 7xx XCx 11xx
UBA_ERR_CTRL	251	Error handling of the UBA expansion modules	
READ_RP	252	read access to system U-remote pages	
WRITE_RP	253	write access to system U-remote pages	
IBSM	254	InterBus-S Master (USK-DIM)	

Table 37: XCx7_Vxxx library

8.5 PLC Operating System ProConOS

8.5.1 ProConOS.INI initialisation file

You can use the ProConOS.INI file to make application-specific changes to advanced settings (e.g. communication drivers, system tasks, CANopen stacks). If ProConOS.INI does not exist or has been deleted, a file called "initial" containing default values will be generated when the controller software starts up.

Path of the file on the Compact Flash: */ata0/OS/PLC/ProConOS.INI*

8.5.2 Description of ProConOS.INI section and key entries

Section PLC

```
[PLC]
; max. size of the PLC program memory: 512…12288 kByte
PC_PROGRAM_SIZE =4096

; use the ProConOS socket communication driver
PC_SOCKET_DRV  = 1      ; yes = 1 (default), no = 0
; max. number of ProConOS clients
; for simultaneous access to controller
PC_SOCKET_BLOG = 4      ; default

; Use serial ProConOS- or HBG-communication-driver
; no = 0, ProConOS protocol = 1, HBG protocol = 2, DriveTop = 3
PC_SERIAL0_DRV = 0      ; RS 422 / RS 485
PC_SERIAL0_BR  = 19200  ; Baud rate = 19200 (default)
PC_SERIAL1_DRV = 0      ; RS 232
PC_SERIAL1_BR  = 19200  ; Baud rate = 19200 (default)
PC_SERIAL2_DRV = 0      ; RS 232, XCx7 only
PC_SERIAL2_BR  = 19200  ; Baud rate = 19200 (default, RS 232, XCx7 only)

PC_SERIAL_DELAY = 100   ; ProConOS driver receive delay (2 .. 200ms)
```

Section CNC

```
currently no entry (obsolete)
```

Section CAN

```
[CAN]
; 1 =enable (default), 0 = disable CAN driver
CAN_ENABLE  =0

; Priority of CANopen Process Task
CAN_PRIO_HIGH = 35      ; 35 = default (1..200)
; Priority of CANopen Interrupt Handling Task
CAN_PRIO_INT  = 10      ; 10 = default (0..20)

; Restart the CANopen process after PLC STOP (NMT master!)
RESTART_CAN    = 0      ; yes=1, no=0 (default)

; PLC STOP after CAN heartbeat error
HBE_STOP_PLC   = 1      ; yes = 1 (default), no = 0

; PLC STOP after CAN Bus Off
CBO_STOP_PLC   = 0      ; yes = 1 (default), no = 0

; obsolete, without function!! - start the CANopen Task with higher priority
CAN_HIGH_PRIO = 0       ; yes = 1, no=0 (default)
```

Section CANxx

```
; CAN parameter definitions:
; Attention: Overwrites some settings in the CANconf.dat!!
[CAN00] ; e.g. card# 0
; any description
DESCRIPTION     = CANconf #0 (XCx 500 series)
NODE_ID         = 1      ; bus address, 1..127
; baud rate
; (1=10,2=20,3=50 obsolete),4=125,5=250,6=500,7=800,8=1000 MBit/s
BAUDRATE        = 4
BOOTUP_DELAY    = 0      ; bootup delay, 0..60 s
CYCLE_TIME      = 4      ; CAN cycle time, 0..255 ms
```

Section SERCOS

```
[SERCOS]
; default IP Address for NRT communication
SERCOS_IP=192.168.0.1
```

Section DPxx

```
; DP parameter definitions:
[DP00]                  ; e.g. card# 0
USE_COM_DRV        = 1                 ; use the simple COM slave driver,
                        ; 1 = yes (default), 0 = no
; any description
DESCRIPTION        = Profibus DPS #0 (XCx5 series)
BUS_ADDR           = 2                 ; 0..126
MASTER_FCONF       = 1                 ; the master force slaves
configuration,
                        ; no further configuration necessary
                        ; 1 = yes (default)/ 0 = no
FirmwareFileFolder =”/ata0/OS/DP/”      ; location of the firmware file
terminated
                ; with ”/” (for automated updates)
```

Section IODriver (CIF Driver for DPM Cards)

(s. ProConOS Manual for Hilscher CIF 30/50 and ProConOS CIF Driver Manual for Hilscher field bus CIF interfaces)

```
[IODriver]      ; Name of section
; The 0 at the end of the parameter name characterises the first of n
possible
; further drivers IODriver[n]. The name refers to the current version of the
; driver. Thus in future versions only this place has to be changed but not
the
; user projects (see I/O configuration).
IODriver0       = "CIF_KW_V2.0"

[CIF_KW_V2.0@0]         ; One instance for one CIF board.
; If this key is set, the hardware communication  must be started and
stopped
; manually by the user. Specific function blocks are provided for this
purpose.
; Otherwise the driver will start the hardware communication at PLC_RUN and
; stop the hardware communication at PLC_STOP automatically.
StartUpManual   = 0

; If this key is set, the Hilscher configuration tool SyCon is able to
connect to
; and to configure installed CIF boards via TCP/IP. The TCP/IP address is
the
; same as of ProConOS. This key can only be set in the section of the first
CIF
; board.
ComServer       = 0
```

```
; This key determines the bus type of the installed boards. All installed
boards
; must be of the same bus type. This key can only be set in the section of
the first board.
BusTyp          = ”USR”         ; The proper Bus type for XCx controller.

; The default mode is "HostControlledBuffered". If there is no ComMode entry
; in the PROCONOS.INI file the default settings are applied. If ComMode =
; "NoChangeOfMode" the actual CIF card mode is applied. In this case the
; mode can be changed e.g. by application of the SyCon tool.
; "DirectDeviceControlled"      Direct Data Transfer, DEVICE Controlled
; "BufferedDeviceControlled"    Buffered Data Transfer, DEVICE Controlled
; "UncontrolledDirect"  Uncontrolled Direct Data Transfer
; "HostControlledBuffered"      HOST Controlled, Buffered Data Transfer
; "HostControlledDirect"        HOST Controlled, Direct Data Transfer
; "NoChangeOfMode"      Keep the mode set by Sycon,
;       needs to be set for every board
ComMode         = "HostControlledDirect"

; The Startup DPM configuration.
ConfigurationFile="/ata0/OS/DP/DPMconf0.dbm"
; The COM module firmware path - location of the firmware file terminated
with ”/”!
FirmwareFileFolder="/ata0/OS/DP/"
More then one board using:
If more than one board are placed in the respective hardware, or more than
one board are assigned in the
ProConOS-IO-Groups, an assignment for the hardware board and ProConOS-IO-
Group-Board is necessary.

[CIF_KW_V2.0@0]         ; possible are up to four instances of the same
driver: @0 ... @3.

;1 = CIF hardware driver assigns board number, only possible in section of
card 0
ManualBoardAssign = 1

; The device number of the card can be found on the card used. Also it is
possible to read out this with the SyCon.
DeviceNr    = 10504000
; The series number of the card can be found on the card used. Also it is
possible to read out this with the SyCon.
SerNr           = 00003930

[CIF_KW_V2.0@1]         ; possible are up to four instances of the same
driver: @0 ... @3.
; The device number of the card can be found on the card used. Also it is
possible to read out this with the SyCon.
DeviceNr    = 10304100
; The series number of the card can be found on the card used. Also it is
possible to read out this with the SyCon.
SerNr           = 00005648
```

Important!

Reading the entries from the sections [IODriver] and [CIF_KW_V2.0@0] uses other operating system routines. No space characters must be before "=" e.g.:

not	**ManualBoardAssign = 1**
but	**ManualBoardAssign=1**

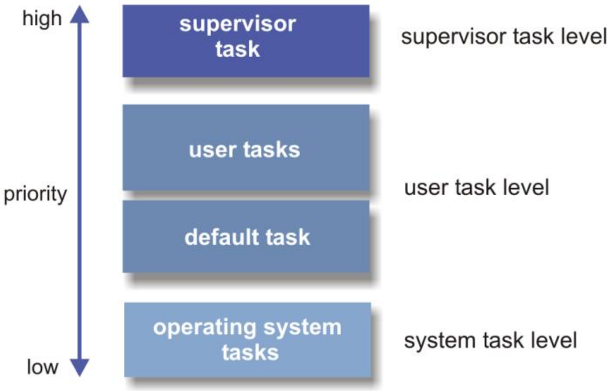
9 The Multi-Task System

9.1 Overview

This is based on a real-time operating system controlled by task priorities. A program is assigned to a task in the MULTIPROG programming system . The tasks in turn are assigned to different priority levels and times that ensure the order and duration of the processing according to their importance.

There are three priority levels for tasks (→ Figure 94):

Figure 94: Multi-Task-System, priority levels



- **Supervisor task**
  The supervisor task is a specially protected operating system task that operates on the highest priority level. It detects errors such as division by zero and task time overrun, and activates the appropriate operating system task.
- **User and default tasks**
  All tasks that are inserted by the user run on the user and default level. Certain important firmware tasks that have to be considered when parameterizing user tasks also run on this level. See section "Task Priorities", page 115. The user tasks are time-monitored (Watchdog).
  - **Cyclical tasks**
    execute the programs assigned to them within a defined interval under a user-defined priority. The task with the highest priority is called first.
  - **Event tasks**
    are started by the controller operating system when particular events occur, for example interrupt signal, CANopen task or interpolation task.
  - **The default task**
    is the user task with the lowest priority. It is not time-monitored and is activated as a background task when no higher priority user task is active at the time.
- **Operating system tasks**
  Tasks for communication, debugging, memory management and system control run unaffected by the user on the priority level for the operating system task.

9.2 User tasks

Attention!

Incorrectly or inappropriately selected user task settings for type, priority, interrupt mode, etc. – especially in conjunction with longer program runtimes – can lead to controller malfunction when essential operating system tasks are displaced.

Observe the description in section Task Priorities.

User tasks are all tasks that can be inserted by the application programmer.

The default task is also on the user task level. It is the user task with the lowest priority. The default task is executed when no other user task is active.

You can use various types of user task.

9.2.1 Cyclical tasks

Cyclical tasks execute the programs assigned to them within a defined interval under a user-defined priority.

In MULTIPROG you can give the individual tasks a priority between 0 and 31. Task 0 has the highest priority, task 31 the lowest. The task with the highest priority is called first. The user-task priorities are mapped to the priority levels of the real-time operating system (see section Task Priorities).

If the watchdog time of a cyclical task is higher than the set interval time and task execution has not been completed, one or more execution cycles will be omitted.

9.2.2 Event tasks

The operating system starts event tasks when particular events occur. The following events are currently defined:

Internal designation	Event no.	Comments
Interrupts		
PLC_EVENT_XFIO_I0	0x00	XFIO Interrupt (Input 0, XCx 3/5)
PLC_EVENT_XUIO_0	0x00	U-Bus Interrupt 0 (XCx7, UBE32 0,1I input 0)
PLC_EVENT_XFIO_I1	0x01	XFIO Interrupt (Input 1, XCx 3/5)
PLC_EVENT_XUIO_1	0x01	U-Bus Interrupt 1 (XCx7/11, UBE32 0,1I input 1)
PLC_EVENT_XUIO_2	0x02	U-Bus Interrupt 2 (XCx7/11, UBE32 0,1I input 2)
PLC_EVENT_XUIO_3	0x03	U-Bus Interrupt 3 (XCx7/11, UBE32 0,1I input 3)
Synchronisation		
PLC_EVENT_POS	0x04	Position controller task (XCN only)
PLC_EVENT_CAN	0x05	- CANopen task, - also applies for Profibus task (microLine, XCx micro)
PLC_EVENT_IPO	0x06	CNC IPO task (XCN only)
PLC_EVENT_DECO	0x07	CNC DECO task (XCN only)
PLC_EVENT_MCSIO	0x08	MCS / XCS20 I/O driver synchronisation (microLine, XCx micro)
Reserved	0x09	
PLC_EVENT_XFIO_I10	0x0A	Measurement interrupt active 0 (only XCx3/5)
PLC_EVENT_XFIO_I11	0x0B	Measurement interrupt active 1 (only XCx3/5)
Reserved	0x0C	
Reserved	0x0D	
Reserved	0x0E	
PLC_EVENT_AC_FAIL	0x0F	AC Fail (XCx 11, ProNumeric)

Table 38: The Multi-Tasking System, event tasks

The event number is used in the MULTIPROG task setting to specify the event that starts the event task.

The specified priority is used unless a bypass option is set by the system. (Bypass cancels the normal task change so that the assigned programs are executed immediately when the event occurs.)

Up to 16 events will be put in a queue. So these events are not lost, and will be executed later. This also applies if new events occur before the assigned event task is executed.

9.2.3 System tasks

System tasks and system programs (SPGs) are started automatically by the operating system when an event occurs in connection with the operating system. The SPGs which can be used are listed in the following table:

No.	Name	Event	Actions
SPG 0	WARM_START	Is executed during a warm start	• Retentive data is not initialised • Non-buffered data is initialized • The open function of the I/O driver is executed • User tasks are activated • PLC switches to run status
SPG 1	COLD_START	Is executed during a cold start	• All data is initialised • The open function of the I/O driver is executed • User tasks are activated • PLC switches to run status
SPG 2	TO_STOP	Is executed when program execution is stopped	• User tasks are deactivated • All outputs are updated • The close function of the I/O driver is executed • PLC switches to STOP
SPG 10	WATCHDOG	Is executed when a task has not been completed within its watchdog time	• User tasks are deactivated • All outputs are updated • The close function of the I/O driver is executed • PLC switches to STOP
SPG 11	ZERODIV	Is executed if division by zero occurs during program execution	• User tasks are deactivated • All outputs are updated • The close function of the I/O driver is executed • PLC switches to STOP
SPG 12	STACKOVER	Is executed if a stack overflow has occurred. Is only executed if the “Stack-Prüfung” [“Stack check”] checkbox in the “Ressource ... einrichten” [Resource ... Set up] dialog in MULTIPROG was activated.	• User tasks are deactivated • All outputs are updated • The close function of the I/O driver is executed • PLC switches to STOP
SPG 13	BADCAL	Is executed if a non-existent manufacturer-specific POU is called	• User tasks are deactivated • All outputs are updated • The close function of the I/O driver is executed • PLC switches to STOP
SPG 14	IOERROR	Is executed if an error occurs in the I/O driver while the process is running	• PLC continues execution
SPG 16	MATHERR	Is executed if a sliding point error occurs in an arithmetic function	• User tasks are deactivated • All outputs are updated • The close function of the I/O driver is executed • PLC switches to STOP

No.	Name	Event	Actions
SPG 17	**CPU_OVERLOAD**	Is executed if a CPU overload occurs	• User tasks are deactivated • All outputs are updated • The close function of the I/O driver is executed • PLC switches to STOP
SPG 18	**INITIODRV_ERR**	Is executed if an error occurs in I/O driver initialization during a cold or warm start	• PLC does not start
SPG 19	**BOUNDS_ERR**	Is executed if the limits of an array or a structure are exceeded. Is only executed if the “Index-Prüfung” [“Index check”] or “Feldbegrenzungs-Prüfung” [“Array limit check”] checkbox in the “Ressource ... einrichten” [Resource ... Set up] dialog in MULTIPROG was activated.	• User tasks are deactivated • All outputs are updated • The close function of the I/O driver is executed • PLC switches to STOP
SPG 20	**BUS_ERR**	Is executed if variables with a data type ≥ 2 bytes and uneven addresses were used or an internal error has occurred in MULTIPROG. Only on Motorola platforms.	• User tasks are deactivated • All outputs are updated • The close function of the I/O driver is executed • PLC switches to STOP
SPG 21	**STRING_ERR**	Is executed if an error has occurred in a character string operation, e.g. if one character string is to be replaced by another, but cannot be found.	• The behaviour of a character string exception has changed! In the standard setting SPG 21 is called after a character string exception has occurred. An entry with the module number and line number is also made in the error catalogue. The PLC remains in “RUN” status.

Table 39: The Multi-Tasking system, system tasks

Note

System tasks are not monitored by the watchdog.

9.2.4 Default task

The default task runs as a background task with the lowest possible user priority and is not time-monitored. It is activated when all higher-priority user tasks have been processed. The default task is configured so that it only uses some of the available residual time. Only one default task is permitted in each resource. It is recommended only to use cyclical tasks.

ℹ **Note**

All drivers in the I/O configuration that are not explicitly assigned to a user task automatically activate the default task and are executed in the context of the default task.

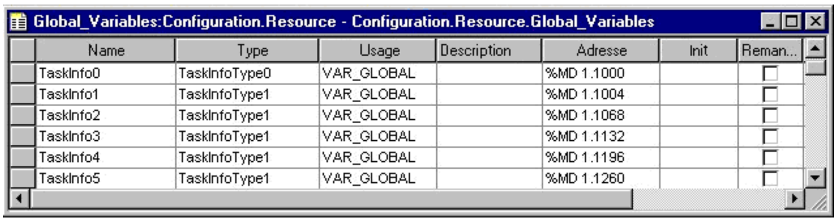
9.3 User task information

Information is mapped to system variables for each user task. The type definitions listed below for the system variables can be found in the *PLC_Types* section of the SchleicherLib library.

Type definition	Comment
TYPE	
TaskNameType : ARRAY [1..10] OF BYTE;	
END_TYPE	
TYPE	
TaskInfoType0 : STRUCT	
MaxTask : INT; (* 00: *)	Max. poss. number of tasks
CurTask : INT; (* 02: *)	Current number of tasks
END_STRUCT (* TaskInfoType0 *);	
END_TYPE	
TYPE	
TaskInfoType1 : STRUCT	
TaskName : TaskNameType; (* 04: *)	Task name
TaskPrio : INT; (* 14: *)	Task priority
TaskMode : INT; (* 16: *)	Task mode
TaskPeriod : INT; (* 18: [ms] *)	Task period in ms
TaskStack : INT; (* 20: *)	Size of used task stack
MainPoe : INT; (* 22: assigned PLC program *)	Assigned PLC program
TaskWatchDog : INT; (* 24: [ms] *)	Watchdog time in ms
reserve0 : DINT; (* 26: *)	
MaxStack : INT; (* 30: max. used stack *)	Size of poss. task stack
CurDuration : INT; (* 32: [ticks] *)	Current task duration including prioritised calls
MinDuration : INT; (* 34: [ticks] *)	Minimum task duration
MaxDuration : INT; (* 36: [ticks] *)	Maximum task duration
AveDuration : INT; (* 38: [ticks] *)	Average task duration
CurDelay : INT; (* 40: [ticks] *)	Current task delay
MinDelay : INT; (* 42: [ticks] *)	Minimum task delay
MaxDelay : INT; (* 44: [ticks] *)	Maximum task delay
AveDelay : INT; (* 46: [ticks] *)	Average task delay
END_STRUCT (* TaskInfoType1 *);	
END_TYPE	

The variables are declared with *TaskInfoType0* and *TaskInfoType1* (→ Figure 95).

Figure 95: Multi-Task-System, variable declaration



Global_Variables:Configuration.Resource - Configuration.Resource.Global_Variables

Name	Type	Usage	Description	Adresse	Init	Reman...
TaskInfo0	TaskInfoType0	VAR_GLOBAL		%MD 1.1000		☐
TaskInfo1	TaskInfoType1	VAR_GLOBAL		%MD 1.1004		☐
TaskInfo2	TaskInfoType1	VAR_GLOBAL		%MD 1.1068		☐
TaskInfo3	TaskInfoType1	VAR_GLOBAL		%MD 1.1132		☐
TaskInfo4	TaskInfoType1	VAR_GLOBAL		%MD 1.1196		☐
TaskInfo5	TaskInfoType1	VAR_GLOBAL		%MD 1.1260		☐

The following user task information is declared with an offset of 64 starting at 1004 (1004 + 64 = 1068 etc.).

The sequence of tasks is defined by the rank of the task in the *Physical Hardware/Configuration/Resource/Tasks* project tree.

9.4 Task Priorities

The table gives an overview of recommended task priorities and their relationship to important reserved firmware tasks (*tfwLAGE, tfwCANhigh, tfwIPO*).

Warning!

Incorrectly or inappropriately selected user task settings for type, priority, interrupt mode, etc. – especially in conjunction with longer program runtimes – can lead to controller malfunction when essential firmware tasks are displaced (*tfwLAGE, tfwCANhigh, tfwIPO*).

Check and adapt the task assignment and task time setting

MULTIPROG Priority	**RTOS* priority (default)**	**RTOS* task name**	**Application**
0	30	Any	E.g. user task (event 0)
1	31	Any	E.g. user task (event 1)
2	32	Any	E.g. user task (event 4)
3	33	*tfwLAGE*	Reserved for position controller task (XCN only)
4	34	Any	E.g. user task (event 4, 5)
5	35	*tfwCANhigh*	Reserved for CAN stack task (option CAN_HIGH_PRIO = 1)
6	36	Any	E.g. user task (event 5, 6)
7	37	*tfwIPO*	Reserved for IPO task (XCN only)
8	38	Any	E.g. user task (event 5)
9	39	*tfwCANhigh*	Reserved for CAN stack task (option CAN_HIGH_PRIO = 0)
10	40	Any	E.g. user task (event 5)
11..15	41..45	Any	E.g. cyclical user tasks
16..31	46	Any	E.g. other cyclical user tasks
Default	127	*default*	Background task
			***R**eal **T**ime **O**perating **S**ystem

Table 40: The Multi-Tasking System, task priorities

Note

The system supports 18 user tasks (priority levels 0..16 and the default task). Tasks with priority ≥ 16 are executed with priority 16.

9.5 Tasks and watchdogs

Each user-defined task has its own settable watchdog. The watchdog checks that task execution has been completed by the end of the watchdog interval. If task execution is not complete at the end of this time the system task *SPG 10 'WATCHDOG'* is executed and the PLC switches to *'STOP'* state if no other actions were programmed. An entry is also made in the error catalogue. The watchdog time starts when the task is ready to execute. The watchdog interval is defined in the *“Task ... Set up”* dialog in MULTIPROG.

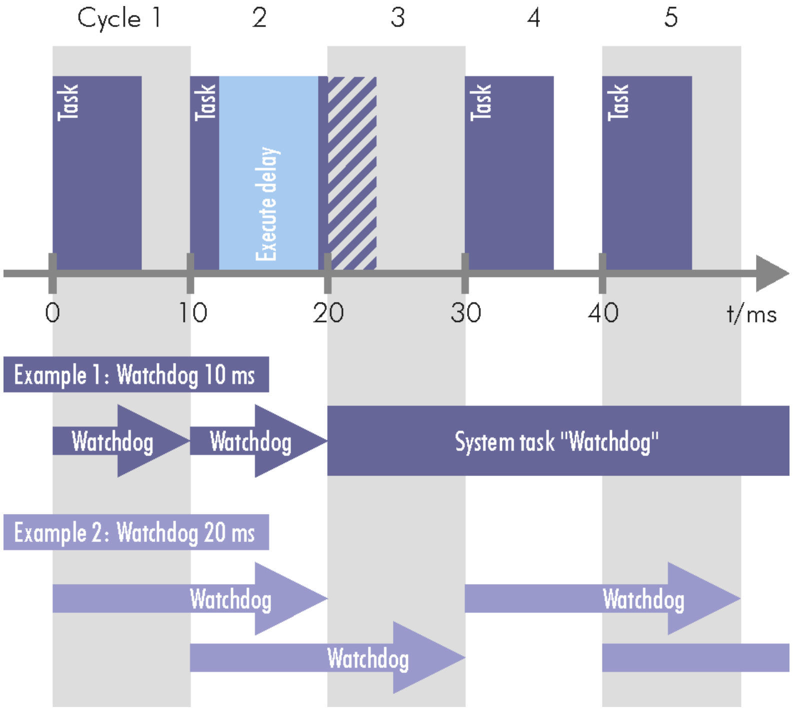
Note

If the execution time of the task and the watchdog time are roughly the same, and the CPU workload is high, the watchdog time may be exceeded during certain online operating steps.

The reason for this may be that you selected address status with powerflow when debugging in online mode.

Example

Figure 96: Multi-Task-System, example for tasks and watchdogs



In **example 1** the watchdog time of the displayed task is set to 10 ms. The watchdog time is exceeded in the second cycle by 20 ms. The execution of the task is interrupted and the "Watchdog" system task is called.

In **example 2** the watchdog time is set to 20 ms. For this reason, it does not address the time overrun of the task in the second cycle. The task is only interrupted for the next cycle and is executed again after 30 ms in the fourth cycle.

9.6 Insert Tasks and Assign Programs

Insert tasks

To insert a task, you have to carry out the following steps in MULTIPROG:

- In the project tree under the resource for the respective controller, right click on the *Tasks* folder to open the context menu (→ Figure 97).

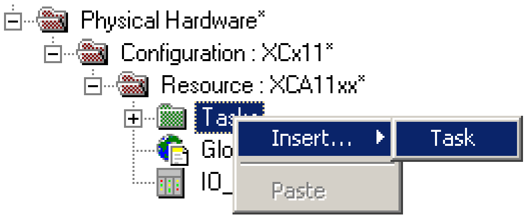


Figure 97: Multi-Task-System, insert a task in MULTIPROG

- Select the *Insert/Select task* menu item. The *Insert* dialog appears.
- Enter the name for the task.
- Set the required task type in the *Task type* list.
  Choose from a default task, cyclical task, event task or system task.
  Note: If task type 'DEFAULT' is not listed, the resource already has a default task.
- Confirm the dialog with *OK*.

The *Task settings for ...* dialog appears. The dialog contains text and list fields, depending on the previously selected task.

You have to enter the following parameters for the task:

Table 41: Multi-Task-System, task parameters

Task	Parameter
Cyclical task	Time interval
Event task	Event number (number of interrupt)
System task	Number of a system program

The instructions in section Task Priorities must be observed when assigning priorities.

Programs must be assigned to tasks before they can be executed. Assigning a program to a task means that an instance of the program will be executed when the task is activated. Different instances of a program can be assigned to different tasks.

Several programs can be assigned to one task. In this case the first program in the task directory will be executed first. Then the next program will be executed, and so on.

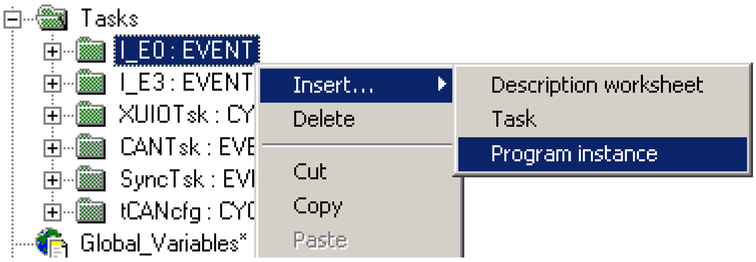
Assign programs

To insert programs you have to carry out the following steps in MULTIPROG.

- Right click on the project tree icon of the task in which the program is to be inserted (→ Figure 98).
- Select *Insert/Program instance* in the context menu.
- Enter an instance name for the program in the *Program instance* field.
- Set the required program in the *Program type* list box.
- Confirm the dialog with OK.

The program symbol is inserted in the project tree.

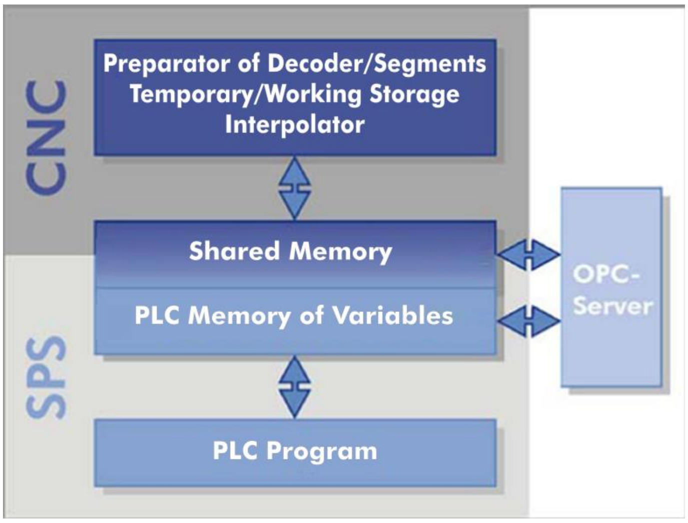
Figure 98: Multi-Task-System, assign programs in MULTIPROG



10 The Shared RAM

The shared RAM connects the sequence control of the PLC and the motion functions of the CNC. Both controller systems operate synchronously for data exchange on the memory and the PLC can take on a master function. Visualisation systems are also integrated via OPC in the communication.

Figure 99: Shared RAM as connection of PLC and CNC



The close link between the CNC and the PLC system now enables you to carry out complex processes which would not be possible with separate CNC and PLC controllers. The classic PLC interface enables PLC functions in NC programs, e.g. the setting and requesting of PLC flags. The synchronisation of the PLC task with the CNC position control provides further options:

- There are no waiting times or communication overhead.
- The PLC can monitor all actions of the CNC
- The CNC output setpoints via the PLC.
- The PLC can specify the CNC management sizes in the position control cycle

10.1 Variables and Tasks

Shared RAM data takes the form of variables as per IEC 61131-361131, which are declared as global variables during configuring in the MULTIPROG programming system. They are accessible to the OPC server as standard and are displayed in the Schleicher dialog operating tool.

In the multi-tasking operating system, PLC task 6 is synchronised with the interpolation task of the CNC controller. The cycle time of task 6 is then oriented on the interpolation cycle of the CNC.

10.2 Access the Shared RAM

The data structure of the shared RAM is created during the installation of the PLC or CNC operating system. For pure PLC, only the variable areas for PLC specification and errors are created (*plcSect* and *errSect*, see below). The PLC program has access to the entire shared RAM via the global variable *plcMem* (for PLC controllers) or *cncMem* (for CNC controllers). The individual components for read and write access (e.g. version numbers, error messages, bit signals, word ranges, NC data, CAN data, etc.) are combined in sections. The retentive variables (retain) occupy their own sections.

- PLC-specific section plcSect
- Error section errSect
- General section comSect
- General section (retain) comSect
- System section sysSect
- System section (Retain) sysSect
- Axis section axSect
- Axis section (Retain) axSect

You can access the individual sections and components of the shared RAM with the *Globale_Variable.Section.Components* notation. For example, the PLC program can read the version number of the controller operating software from the `cncMem.plcSect.lOSVersion` variables. For integration of the shared RAM in the MULTIPROG programming software→ page 49.

Visualisation systems have access to the shared RAM via the OPC interface. The OPC server cannot handle structured variables so the whole data structure of the shared RAM is represented as a one-dimensional list. The names are composed of two parts separated by a "_". The first part is the access path, while the second part is identical to the component name of the PLC variable. For example, the version number of the operating system could be read from the OPC variables `cmpS_lOSVersion`.

10.3 Help about Shared RAM

A more detailed description of the shared RAM structure and all versions can be accessed via the online help in both MULTIPROG and the Schleicher dialog.

10.4 Further Background Information on Shared RAM

The following elements are added or updated when inserting the shared RAM in a PLC project:

- The *SharedMemory_Types* data type worksheet; the data structure of the shared RAM is declared here.

  The data type worksheet is inserted for shared RAM version 8 or higher. For earlier versions, no data type worksheet is inserted as the declaration of the shared RAM structure is contained in the associated user library *SchleicherLib_Vxxx*(shared RAM version 7 in *SchleicherLib_V007*, shared RAM version 6 in *SchleicherLib_V006*, etc.).
- The global variables `plcMem` (for PLC controllers) or `cncMem` (for CNC controllers). See worksheet *Global_Variable*s, group *SharedMemory_Variables*.

  These variables represent the complete non-retentive (non retain) section of the shared RAM. The PLC program can access the individual components (variables) of the shared RAM using full stops as explained above.

  With shared RAM version 8 or higher, there is a retentive (retain) section of the shared RAM as well as the non-retentive section. Unlike the non-retentive section, the values of the variables of this section are retained after switching off the controller. For this, the global variable `cncRMem` is also inserted.
- The global variables `cmpS...,` `cmeS...,` `cmcS...,` `cmsS...,` `cmaS....` See worksheet *Global_Variable*s, groups *PLC_Common*, *CNC_Common*, *CNC_System_x* (x stands for the number of the CNC sub-system. Such a group exists with system-specific variables for each sub-system) and *CNC_Axis_y* (y stands for the number of CNC axes. Such a group exists with axis-specific variables for each axis).

  These variables are provided for the visualisation systems or similar programs to access the shared RAM. They provide the complete shared RAM via the OPC interface.

  These variables provide the shared RAM in an unstructured form. Only simple data types (BOOL, DINT, REAL, STRING) and fields of simple data types are used. This procedure is required as structured data (like the ones the variables `plcMem` und `cncMem` contain) can not be transferred via the OPC interface.

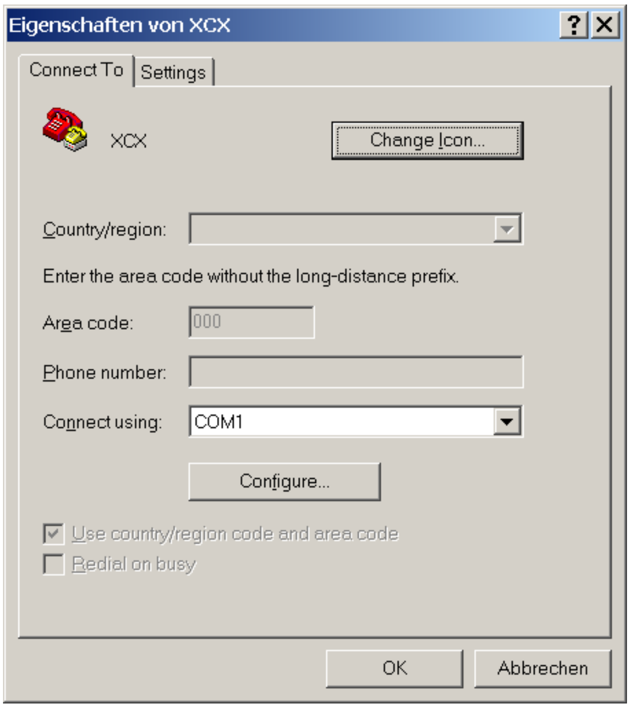
11 RS232 Serial Interface

11.1 Starting up the Serial Connection via the RS232 Interface

The virtual interface is assigned to the real-time operating system and is used to connect operating devices and for system diagnosis. It is used for the output of the bootlog when starting up the XCx, for example.

- Establish the cable connection between service PC COM1 or COM2 and the XCx connection X12 (pin assignment → page 20).
- Go to *Start/Programs/Accessories/Communication* on the PC and start the HyperTerminal *program* . Enter a name, for example XCx, and select a symbol.
- In the *Properties of <Name>*dialog window under*Establish connection via*, select the Direct connection via COM1 and click the *Configure* button.

Figure 100: "Properties of ..." dialog window,select connection



Country/region:

Enter the area code without the long-distance prefix.

Area code: 000

Phone number:

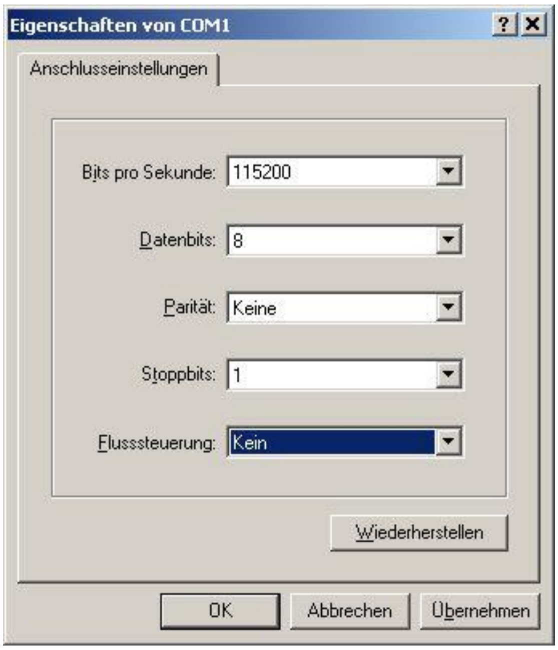
Connect using: COM1

☑ Use country/region code and area code

□ Redial on busy

- Set the these parameters in *Properties of COM1*.
  - Bits per second: 115200
  - Data bits: 8
  - Parity: none
  - Stop bits: 1
  - Flow control: none

Figure 101: "Properties of COM1", connection settings



Eigenschaften von COM1

Anschlusseinstellungen

Bits pro Sekunde: 115200

Datenbits: 8

Parität: Keine

Stoppbits: 1

Flusssteuerung: Kein

- Switch the XCx on or RESET.

The bootlog appears in the hyperterminal dialog window while the controller is starting up.

12 The CNC

The XCx 1100 is a CNC with up to 64 axes/spindles and an integrated, powerful PLC.

Overview of Functions

- Up to 32 sub-systems with a total of 64 axes/spindles
- Technologies for drilling, milling, grinding, handling
- Endless rotating round axes
- spindle packet with comprehensive functionality, e.g. thread cutting function, variable pulse evaluation, oriented spindle stop
- Synchronous spindle
- Programmable acceleration
- Electronic gears
- 2D+n-helical curve interpolation
- Feed rate and rapid feed:
  0.001 mm/min to 999 m/min
- Tool radius compensation with approach and departure strategy
- Tool length compensation
- Interpolar lead screw and measurement system error compensation
- Backlash compensation
- Field of work limit
- Software limit switch

The CNC programming of the XCx is described in detail in a separate operating manual (→ page 10).

13 Other Operating Software

13.1 Windows embedded

As well as the real-time operating system for PLC and CNC, the XCA is equipped with the Windows operating system that enables non-time critical tasks such as visualisation or diagnosis in the familiar environment (→ page 11).

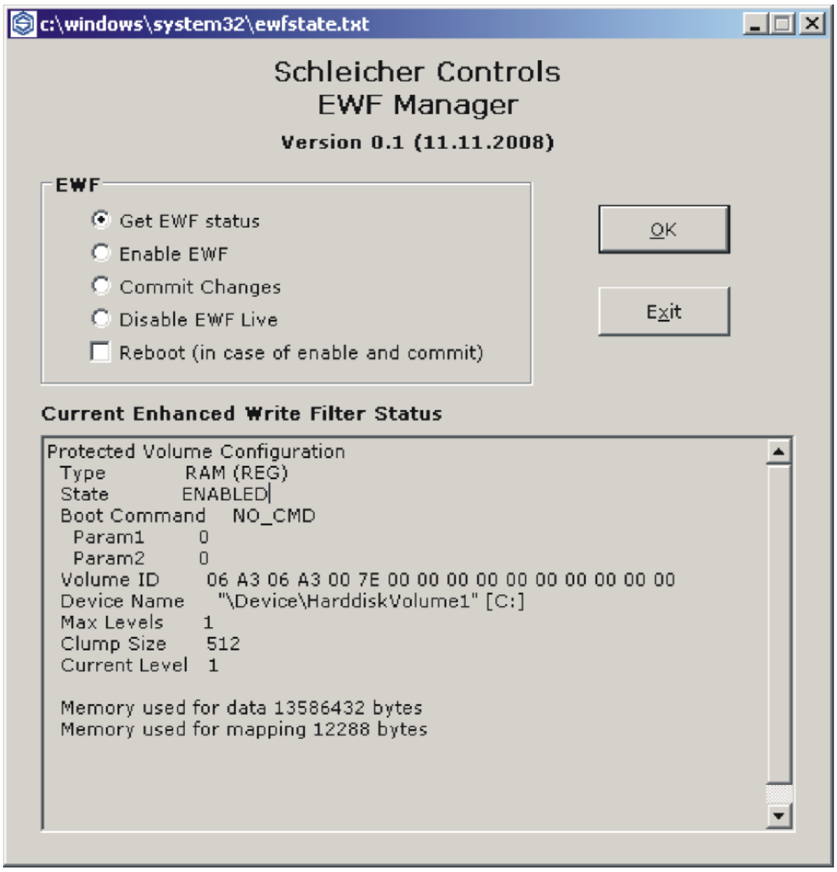
Windows embedded is a Microsoft operating system that is based on the same source code as Windows. The operating system has a compete modular structure and so provides the manufacturer with the option to adapt it to the requirements of the device.

13.2 Enhanced Write Filter EWF

The Enhanced Write Filter is a component in the Windows XP embedded PC operating system of the XCx 1100. It is used to protect one or more partitions (volumes, e.g. on a CF drive) against modifications. Write access is diverted to a so-called overlay (e.g. in system RAM) for the activated EWF. This is lost after switching off the system. However, to be able to make the required changes to the system, the contents of the overlay memory can be written back using the corresponding system command or the EWF can be temporarily deactivated.

The "EWF Manager" tool is available on the XCx 1100 for the control of the EWF function (→ Figure 102).

Figure 102: EWF manager



c:\windows\system32\ewfstate.txt

Schleicher Controls
EWF Manager
Version 0.1 (11.11.2008)

EWF

● Get EWF status
○ Enable EWF
○ Commit Changes
○ Disable EWF Live
□ Reboot (in case of enable and commit)

Current Enhanced Write Filter Status

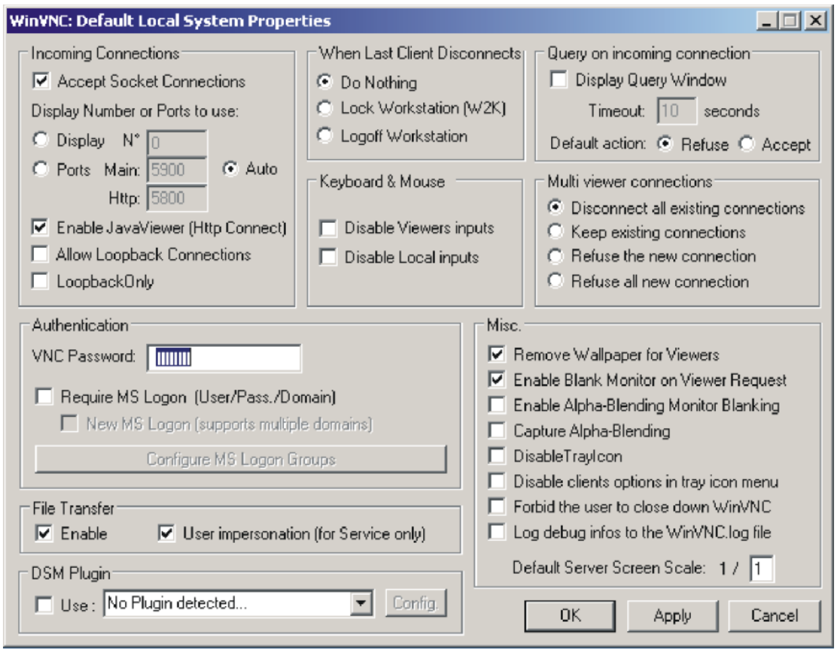
```
Protected Volume Configuration
  Type           RAM (REG)
  State          ENABLED
  Boot Command     NO_CMD
    Param1        0
    Param2        0
  Volume ID       06 A3 06 A3 00 7E 00 00 00 00 00 00 00 00 00 00
  Device Name     "\Device\HarddiskVolume1" [C:]
  Max Levels      1
  Clump Size      512
  Current Level   1

  Memory used for data 13586432 bytes
  Memory used for mapping 12288 bytes
```

- **Get EWF status**
  outputs the current configuration and the status of the EWF.
- **Enable EWF**
  activates the EWF.
- **Commit changes**
  writes back the contents of the overlay memory.
- **Disable EWF Live**
  deactivates the EWF.
- **Reboot**
  A status modification of the EWF (with the exception of deactivation) is only instructed and is actually executed after a system reboot. The restart can be triggered immediately with the set "reboot" option together with the corresponding EWF command.

13.3 Remote Desktop UltraVNC

UltraVNC is remote maintenance software under Windows that enables remote access from a PC via the network or Internet on the desktop of a remote Windows computer.

The software operates according to the client-server model where the server runs on the computer to be monitored (under Windows XP embedded for the XCx). The client receives the screen outputs of the server and sends these to the mouse and keyboard entries.



WinVNC: Default Local System Properties

Incoming Connections
☑ Accept Socket Connections
Display Number or Ports to use:
○ Display N° 0
○ Ports Main: 5900 ● Auto
Http: 5800
☑ Enable JavaViewer (Http Connect)
□ Allow Loopback Connections
□ LoopbackOnly

When Last Client Disconnects
● Do Nothing
○ Lock Workstation (W2K)
○ Logoff Workstation

Query on incoming connection
□ Display Query Window
Timeout: 10 seconds
Default action: ● Refuse ○ Accept

Keyboard & Mouse
□ Disable Viewers inputs
□ Disable Local inputs

Multi viewer connections
● Disconnect all existing connections
○ Keep existing connections
○ Refuse the new connection
○ Refuse all new connection

Authentication
VNC Password: ||||||||
□ Require MS Logon (User/Pass./Domain)
□ New MS Logon (supports multiple domains)
Configure MS Logon Groups

Misc.
☑ Remove Wallpaper for Viewers
☑ Enable Blank Monitor on Viewer Request
□ Enable Alpha-Blending Monitor Blanking
□ Capture Alpha-Blending
□ DisableTrayIcon
□ Disable clients options in tray icon menu
□ Forbid the user to close down WinVNC
□ Log debug infos to the WinVNC.log file
Default Server Screen Scale: 1 / 1

File Transfer
☑ Enable ☑ User impersonation (for Service only)

DSM Plugin
□ Use : No Plugin detected... Config

OK Apply Cancel

Figure 103: Remote maintenance software UltraVNC

UltraVNC is based on the VNC (Virtual Network Computing) network protocol and is available for free download as an open source version under the GNU General Public License. The program can be run under Windows 95/98/NT/2000/XP/Vista. It provides functions such as data encryption, password request, mirror video driver (for the read access on the remote desktop), file transfer, directory transfer and a text chat.

The server of UltraVNC is pre-installed on the XCx controller. Load the current version of UltraVNC from the website of the manufacturer to install the client on a maintenance computer. http://www.uvnc.com. Help for the setup and operation, (online) FAQs, forums and tutorials can be found on this page.

14 Annex

14.1 Technical Data of all Modules

Climatic conditions	
Ambient operating temperature	0 ... +55°C (category KV to DIN 40040), vertical installation, free air circulation
Storage temperature	-25 ... +70°C (category HS to DIN 40040)
Relative humidity	10 ... 95% (category F to DIN 40040), no condensation
Air pressure in operation	860 ... 1060 hPa

Mechanical strength	
Vibration	Acc. to DIN EN 60068-2-6 10 ... 57 Hz constant amplitude 0.075mm 57 ... 150 Hz constant acceleration 1 g
Shock	Acc. to DIN EN 60068-2-27, sinusoidal half-wave 15g / 11 ms
Free fall	Acc. to DIN EN 60068-2-32, fall height 1m (with original packaging)

Electrical safety	
Protection class	Class I, basic insulation and PE terminal (to IEC 60536)
Protection type	IP 00 to EN 60529
Clearance/creepage distance	DIN EN 61131-2 between electrical circuits and objects as well as between decoupled electrical circuits, corresponding to overload category II, contamination level 2
Test voltage	AC 350 V/50Hz for device rated voltage DC 24V AC 1350 V/50Hz for device rated voltage AC 230V

Electromagnetic compatibility	
Electrostatic discharge	EN 61000-4-2: 8 KV air discharge, 4kV contact discharge
Electromagnetic fields	EN 61000-4-3, field intensity 10 V/m, 80 ... 1000 MHz
Rapid transients (bursts)	EN 61000-4-4: 2kV on DC supply lines , 1kV on I/O signal lines
Interference emissions	EN 55011, limit category A, Group 1

14.2 Accessories and Software

Designation	Description	Article number
MULTIPROG 4.x	PLC programming system to IEC61131-3	R4.320.0640.0
Service Pack	Controller software for all Schleicher controllers, add-ons, Schleicher dialog, documentation and service information	R4.320.0590.0
ProCANopen	Network configuration software	R4.320.0500.0
CANcardY	Single CANopen interface, PCMCIA card	R4.321.0020.0

Table 42: Accessories and spare parts

14.3 Trademarks

- WINDOWS is a registered trademark of Microsoft Corporation.
- CANopen is a registered trademark of CAN in Automation e.V.,
- ProCANopen is a registered trademark of Vector Informatik GmbH.
- VxWorks is a registered trademark of Wind River Systems Inc.
- PROFIBUS is a registered trademark of the PROFIBUS users organization.
- MULTIPROG is a registered trademark of KW-Software GmbH.

All other trademarks or product names are registered trademarks of their respective owners.

14.4 List of Figures

14.5 List of tables

14.6 Index