

---

# User manual

## LON SMI Controller DR 4x16M LoVo

MTN887281

Table of contents

---

1	Introduction .....	4
2	System structure .....	5
2.1	Network structure .....	5
2.1.1	Direct coupling to an IP backbone.....	5
2.1.2	Coupling to a TP/FT 10 line.....	6
3	General device characteristics.....	7
3.1	Description of the LEDs .....	7
3.2	Description of the device buttons.....	9
4	Installation.....	10
4.1	Installing the SMI Controller DR 4x16M LoVo .....	10
4.2	Installing the SMI lines .....	10
4.3	Installing the LON line .....	11
5	Commissioning .....	12
6	Configuration.....	13
6.1	IP configuration .....	13
6.2	Installing the configuration tool .....	14
6.3	Installation und starting the IRC Project Manager and IRC Configurator .....	14
7	Creating a project.....	18
7.1	Configuration of the function objects.....	19
7.2	LON control panels .....	21
7.2.1	Switch Object.....	21
7.2.2	Scene Panel Object.....	22
7.2.3	Occupancy Panel Object.....	22
7.2.4	fb_0-Object.....	23
7.3	Simulation and test.....	24
7.4	Saving a project .....	25
7.5	Basic configuration.....	25
7.6	Configuration of the network address .....	26
7.7	Addressing the extension modules at TP/FT 10.....	27
7.8	Installation procedure.....	29
7.9	Commissioning the SMI lines.....	31
7.9.1	Addressing the SMI devices.....	32
7.10	Creating binding links.....	34
7.11	Application in a LON network.....	37
7.11.1	Creating a LON template (XIF).....	37
7.11.2	Program ID settings.....	38
7.11.3	Structure of an IP channel.....	38
7.11.4	CNIP settings.....	40
7.12	Tools.....	40
7.13	Configuration using the Web browser.....	41
7.13.1	IP SETTING.....	42
7.13.2	CNIP SETTING .....	43
7.13.3	LOG SETTING .....	44
7.13.4	TIME .....	45
7.13.5	LIST DEVICES .....	46
8	Appendix A: Description of the function objects .....	47
8.1	LonMark®-object Blind Controller .....	47
8.1.1	Introduction.....	48
8.1.2	Priority Control.....	48
8.1.3	Internal automatic Control .....	49
8.1.4	Occupancy control.....	50
8.1.5	Glare shield control.....	50
8.1.6	HVAC support.....	50

Table of contents

---

8.1.7	Slat tracing (not supported) .....	51
8.1.8	State monitoring .....	51
8.1.9	Behaviour after reset .....	52
8.1.10	Node Object (LonMark Object #0).....	53
8.1.11	SbController (LonMark Object #5).....	55
8.1.12	SharedIn (LonMark Object #3) .....	64
8.2	LonMark®-object SMI Actuator .....	66
8.2.1	fb_0 Object .....	66
8.2.2	Object SMIActuator .....	69
8.3	LonMark®-object Safety Position.....	73
8.3.1	Introduction .....	73
8.3.2	fb_0 Object (LonMark Object #0) .....	75
8.3.3	SafetyPosCntlr (LonMark Object #5).....	77
8.4	LonMark®-object Scene Controller .....	80
8.4.1	Introduction .....	80
8.4.2	fb_0Object (LonMark Object #0) .....	81
8.4.3	SceneController (LonMark Object #3251) [4].....	83
8.5	LonMark®-object Logic controller (#) switch.....	91

## 1 Introduction

The LON SMI Controller DR 4x16M (MTN887281) is for controlling SMI sunblind systems using LON. The device has four independent SMI LoVo interfaces. In addition it has one TP/FT-10 interface for connecting conventional LON devices such as LON control panels.

Incorporation into a LON network and configuration of the device is preferably performed using Ethernet (LON over IP) but can also be realised by the TP/FT-10 interface, which in the same way allows the connection of LON devices, e.g. LON-panels.

The SE configuration tool can be downloaded free of charge for configuration and creation of applications. The application of the device is created by the user from a library using device templates which correspond to the physical devices to be connected. The device templates are constructed of function objects which match the LonMark function profiles. A description of the function objects can be found in Appendix A.

The device templates are sorted into categories which correspond to their physical connections: SMI, TP/FT-10. In addition there is the further category "Internal", which contains controller functions such as "Sunblind Control", "Scene Control", "Logic Control" etc.

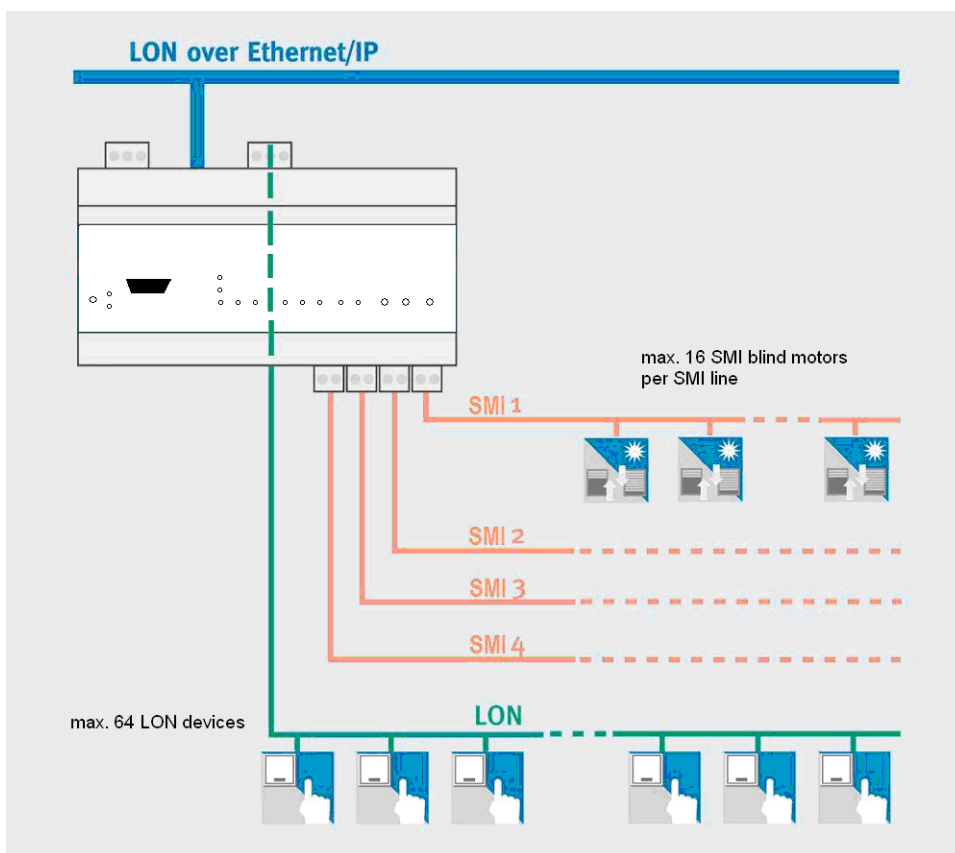
Please note that this document contains only explanation for functional objects that are useful in combination with LON SMI Controller. The Configuration Tool that is used to configure the LON SMI Controller contains further functional objects that are used in combination with lighting control. These functional objects are explained in the corresponding documentation for a device used for lighting control.

## 2 System structure

### 2.1 Network structure

The SMI Controller can be linked into a LON network in different ways.

#### 2.1.1 Direct coupling to an IP backbone



**Fig. 2.1.1: Infrastructure with IP linking**

The device is directly connected to the Ethernet using the 100 Base-T interface. Extension devices are connected to the TP/FT 10 interfaces.

The LON commissioning tool views the IP port as a logical interface. For communication with other devices, the device should be bound into an IP channel by means of a configuration server.

### 2.1.2 Coupling to a TP/FT 10 line

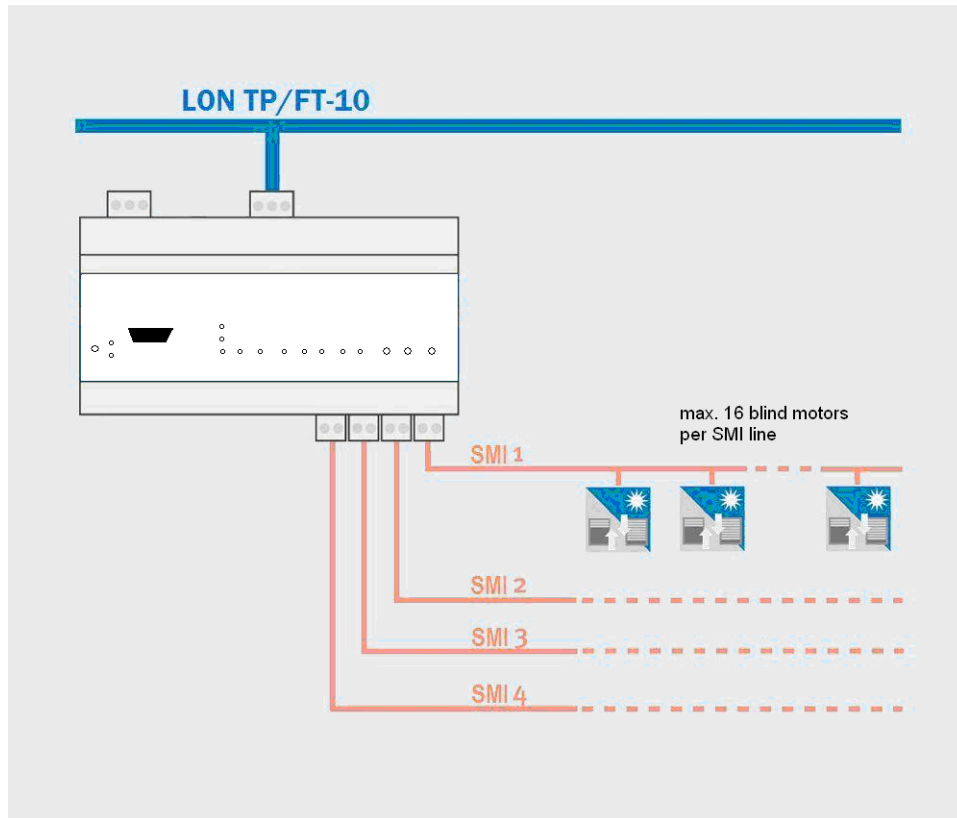


Fig. 2.1.2: Infrastructure with TP/FT 10 linking

The device is connected to the LON segment using the TP/FT-10 interface.

The LON commissioning tool views the TP/FT-10 port as a logical interface. Other LON devices that are connected to the same LON segment are managed using the LON management tool.

General device characteristics

---

**3 General device characteristics**

The device offers interfaces for connection of four SMI segments, one LON interface, preferably for connection of LON control panels and an Ethernet interface for a higher-level system or for networking with other CNIP - controllers.

In accordance with the SMI standard, up to 16 SMI LoVo motors can be connected to each SMI interface. The SMI control-voltage is supplied by the interface itself. LON devices can be installed on the LON interface. These must be held as templates in the configuration tool. Templates for devices can be added on request.

The functions of the LEDs and the device buttons are listed in the tables below.

**3.1 Description of the LEDs**

<b>Service</b>	
RED	Loading the firmware.
OFF	Application is started – the boot-up process is complete.
Flashing at 1 Hz	Boot-up process (data points created, operating system started)

<b>RUN</b>	
GREEN	Device is supplied with power.
OFF	No voltage is present.

<b>CFG</b>	
GREEN	IP stack has been configured.
OFF	IP stack has not been configured. 1) Boot-up process running 2) invalid netmask, 3) IP collision.

<b>MSG</b>	
OFF	No data traffic at 100 base T
Random flashing GREEN	Data traffic via 100 base T

<b>LINK</b>	
GREEN	100 base T link.
OFF	No 100 base T link.

General device characteristics

<b>CNIP</b> (the CNIP port is relevant only when using a LON over IP network)	
GREEN	CNIP port configuration is fully configured and updated.
YELLOW	CNIP port configuration is fully configured but not updated, e.g. because the configuration server cannot be accessed.
RED	CNIP port configuration is incomplete (i.e. not implemented or the initialisation has failed). In this case check the CNIP configuration using the IRC Configurator or check the settings on the configuration server.
OFF	No valid CNIP packet detected.
Flashing RED	CNIP port is unconfigured.
Flashing GREEN or YELLOW	Data traffic via the CNIP.

<b>TP/FT 10</b>	
GREEN	TP/FT 10 port is configured and online. Heavy data traffic at the port.
RED	TP/FT 10 port is defective or a LON management tool has unconfigured this device.
OFF	TP/FT 10 port is configured and online. No data packet was received.
Flashing RED	Data packets were received, but at least one device on this line is defective.
Flashing YELLOW	TP/FT 10 port is unconfigured.
Flashing GREEN	TP/FT 10 port is configured and online. Data traffic at the port and all devices operating normally.

<b>IRC*</b>	
GREEN	Port is configured and online. Heavy data traffic at the port.
RED	Port is defective or unused.
Off	Port is configured and online. No data packet was received.
Flashing RED	Data packets were received, but at least one device on this line is defective.
Flashing YELLOW	Port is unconfigured.
Flashing GREEN	TP/FT 10 port is configured and online. Data traffic at the port.

<b>SMI 1 – SMI 4</b>	
Flashing GREEN	Port configured. Data traffic at the port and all addressed SMI devices on the line are OK.
Flashing YELLOW	Manual operating mode active.
Flashing RED	At least one SMI device is defective, but data traffic at the port.
Off	1) BUS mode: No data traffic at the port. 2) Manual Mode: tbd
GREEN	Manual mode Motors on the SMI channel are running.
RED	SMI channel is no longer configured or is defective.
YELLOW	Manual mode / programming mode: Exchanging a SMI motor. Colour changes when the CHANNEL or ON/OFF/BUS button is pressed.
* This port is not existing at the SMI Controller	



General device characteristics

---

### 3.2 Description of the device buttons

Service	Send a "service message" for each LON channel: FTT10, CNIP). If this button is kept pressed during the boot-up process (until the SERVICE LED stops flashing), the standard configuration will be restored.
---------	---

Channel	SMI manual mode: Activate manual mode with a long button push (more than 3 seconds). A further short button push allows the consecutive SMI channel to be selected (cycle: channel 1- channel 2- channel 3- channel 4- all channels).
---------	---

UP/DOWN/Bus	This button is effective only in SMI manual mode. A short button push in toggle mode time toggles all devices on the selected SMI channel UP or DOWN. A long button push (> 3s) is necessary in order to switch to channel selection mode.
-------------	--

Program	This button allows manual exchange of a defective SMI device. The command is effective only in manual mode for the respective channel. Procedure for exchange: <ol style="list-style-type: none"><li>1) Mount the exchange device.</li><li>2) First select manual mode for the respective channel.</li><li>3) A long button push on the "Program" button exchanges the device in the database. Completion of the device exchange is signalled by the exchanged device "driving". The device will be exchanged only when a new and a defective device are found on the channel.</li></ol>
---------	--

## 4 Installation

### 4.1 Installing the SMI Controller DR 4x16M LoVo

A 24V supply (SELV) is necessary for operating the LON SMI Controller DR 4x16M LoVo.

Connect the devices as described below.

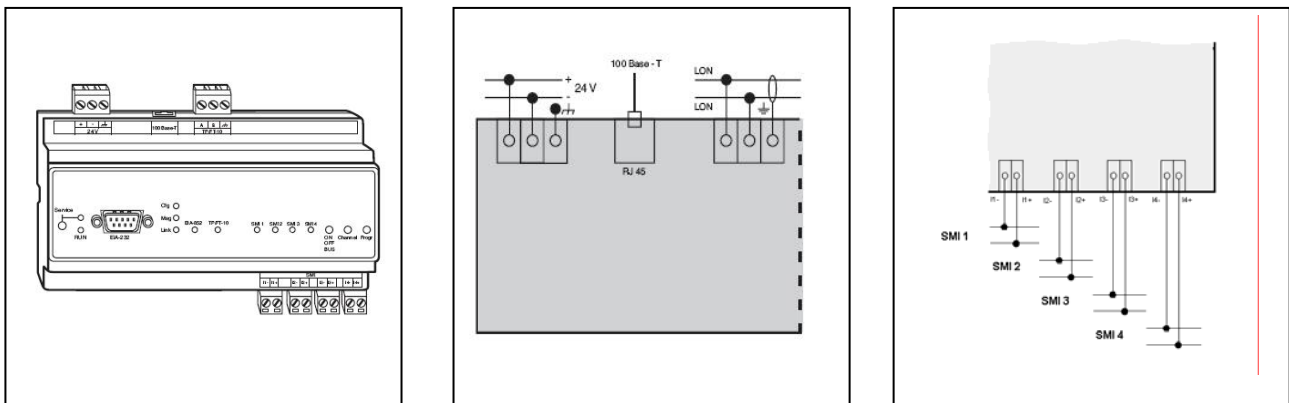


Fig. 4.1: Wiring diagram LON SMI Controller DR 4x16M LoVo

### 4.2 Installing the SMI lines

SMI stands for "Standard Motor Interface" and is the definition for the standardised digital operating device interface for an interface standard spanning different companies in the field of sunblind systems. The SMI standard is described at [www.smi-group.com](http://www.smi-group.com).

SMI supplies a simplified digital interface for sunblind control devices. The intelligent components communicate in a simple and interference-proof way within a local system with distributed intelligence. The data communications wiring requires no special features, neither must terminating resistors be fitted.

SMI is defined for a maximum of 16 individual devices (individual addresses), which can be divided into a maximum of 16 groups (group addresses).

The SMI Controller provides the bus power of approx. 18VDC on each of its SMI interface (I1-/I1+ ... I4-/I4+). The SMI line should be restricted to 350m.

**Attention: For operation, the SMI Controller requires a DC 24V supply voltage. It is necessary to ensure that the SMI Controller and all connected SMI LoVo motors are powered from one common power supply!**

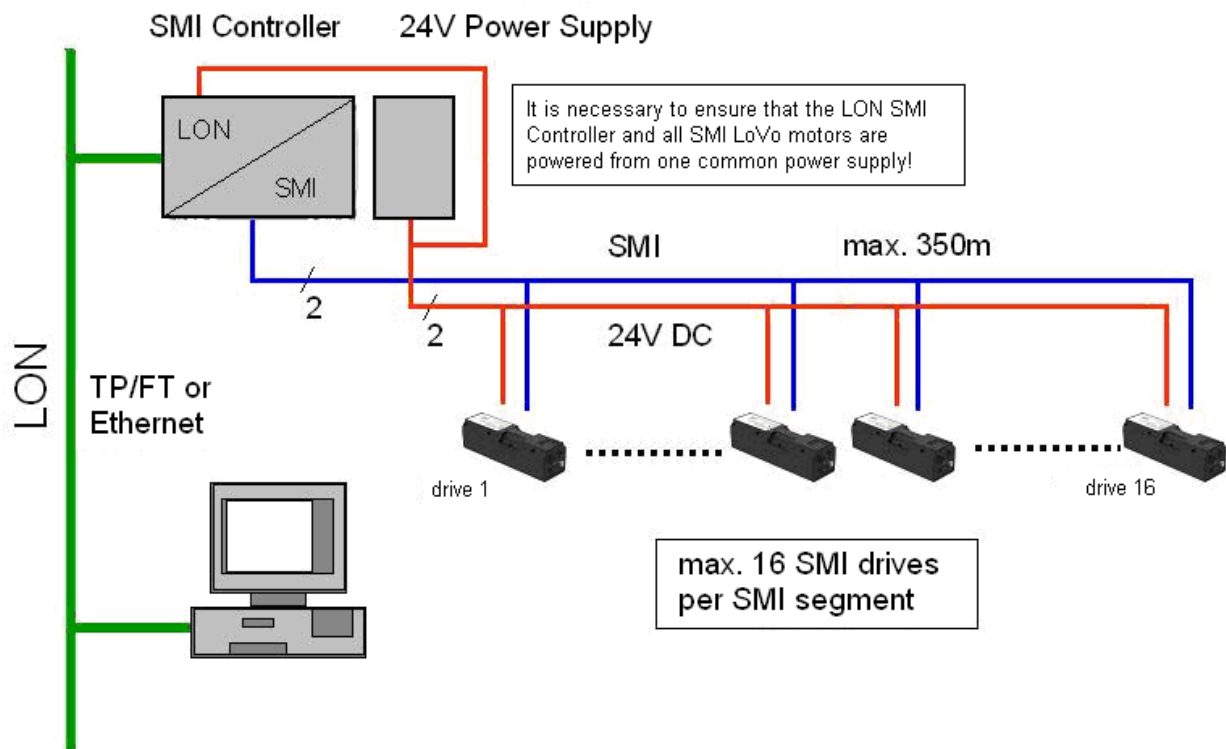


Fig. 4.2: Installation scheme of the SMI lines

### 4.3 Installing the LON line

The TP/FT 10 connection can be used as an alternative in two application cases:

LON control panels or standard LON devices can be connected to this interface as I/O extensions. This is specially designed for connection of the LON control panels from Schneider Electric. The management and configuration of these LON devices is performed using the configuration tool. Links (bindings) between the devices can be created using the configuration tool, when using the SMI Controller in stand-alone use. Retrospective changes must always be performed using the configuration tool. In this case the 100 base T interface acts as the interface for binding into an LNS database. The LON installation guidelines are applicable. We recommend a maximum extension of 10 LON control panels. When integrating the SMI Controller into an LNS database, the bindings created by the configuration toll will be overwritten respectively deleted.

Alternatively the LON SMI Controller can be bound into the LON network using the TP/FT 10 connection. In this case the device functions are depicted at this interface. The LON XIF should be generated accordingly (setting in the IRC configuration menu, program ID settings). The configuration of the LON SMI Controller can be performed only via the 100 base T or the RS-232 serial interface.

## 5 Commissioning

After you have switched on the power supply, the device boot-up process starts. This takes a little time. The process is divided into 2 phases:

- 1) Self-test: During the self-test the green RUN LED is switched on and the red SERVICE LED is switched off.
- 2) Initialising the interfaces: During this phase the green RUN LED remains switched on and the red SERVICE LED flashes cyclically. All interface LEDs light up red continuously. As soon as each interface has been tested successfully the respective LED goes out. If an LED remains red this indicates a fault at the respective interface. In this case please check the connections.

In normal operation the data traffic over the interfaces is indicated by short flashing of the respective green LED.

In some cases it may happen that the CNIP LED shows continuous ORANGE. This indicates that the CNIP configuration server that has been entered cannot be accessed. This is relevant only if an LNS network is being used.

The installation can be checked by taking the following steps:

- 1) The LEDs SMI1, SMI2, SMI3, SMI4 should be Off. If an LED lights up red, check the connections and the power supply of devices for this channel.
- 2) A long button push (more than 3 seconds) on the "Channel" button activates button mode. Further short button pushes change the active SMI channel (sequence: 1-2-3-4-all). The "UP/DOWN/BUS" button allows all connected SMI devices to be driven. A short button push on the "Channel" button takes you to the next channel. A long button push (more than 3 seconds) on the "Channel" button exits button mode.

For easy commissioning we recommend connection to the Ethernet network using the "100 base T" interface.

For direct connection between a PC and the LON SMI Controller please use a crossover cable for Ethernet.

## 6 Configuration

### 6.1 IP configuration

It is preferable that the configuration is performed using the "100 base T" interface. The device IP address is factory-set to **192.168.1.111**.

Before you can address the standard IP address you must set it up in your computer, providing your computer has as IP address for a subnet that differs from 192.168.1.xxx.

To do this, open a "command tool" and enter the following route instruction:

- 1) Windows START -> Execute
- 2) Command.com
- 3) Route add 192.168.1.111 %COMPUTERNAME%

Alternatively you can add an IP address for the same subnet to your local TCP/IP settings:  
Windows START -> Network connection -> LAN connection -> Properties -> Internet protocol (TCP/IP) -> Properties -> Extended

The IP addresses of the LON SMI Controller must not be identical to those for other devices on the network.

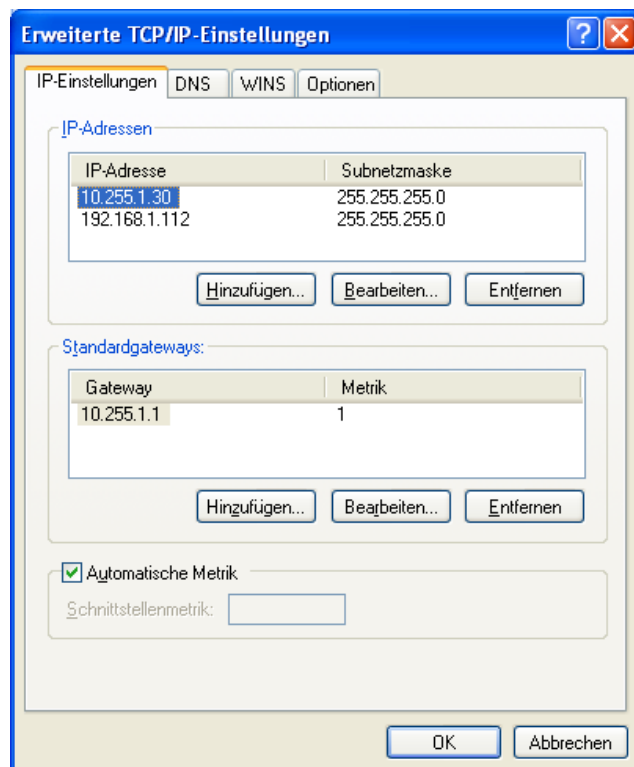


Fig. 6.1: Setting the IP address of your computer in the system control

## 6.2 Installing the configuration tool

A pre-requirement for installing the configuration tool is an operating system that supports Java JRE 1.4.x. Windows XP, Windows 2000, Linux (release 10.x) are platforms that have been tested for this.

Before installing the configuration tool, please install on your **Windows platform Java(TM) 2 Runtime Environment, Standard Edition 1.4.2\_12** (<http://java.sun.com/j2se/1.4.2/download.html> -- J2SE v 1.4.2\_12 JRE) or a more recent version.

After this, perform the setup "irc\_install\_xxxx" and follow the instructions in the installation program.

We recommend the configuration tool is used over the "100 base T" interface (Ethernet)!

Alternatively the configuration tool can communicate over the RS232 interface, but its functionality will be restricted. For this the "Java Communication Extension" is necessary. Please use the installation supplied. Open a "command tool" and perform the following instruction in the respective directory

- 1) Windows START -> Execute
- 2) Command.com
- 3) java -jar comm\_install.jar

Alternatively it is sufficient to double click on this file. However no acknowledgement is supplied in this case.

## 6.3 Installation und starting the IRC Project Manager and IRC Configurator

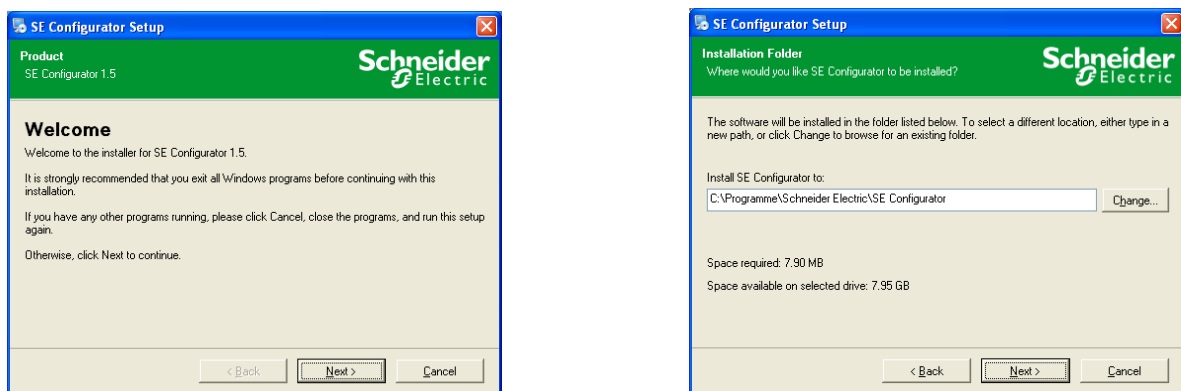


Fig. 6.3.1 und 6.3.2: Installation of the SE Configurator Software and selection of a project directory

For installing and managing of projects the SE Configurator software uses the Project Manager. This tool is needed to create new projects and to open and manage existing projects.

## Configuration

At the installation process of the SE Configurator Software, the routine saves the SE Configurator in a selected directory. If there is a SE Configurator software still existing at the computer, it is advisable to use the same directory to update the previous version. Otherwise existing links may activate old and limited software functions.

After starting the Project Manager you can initialize the first project at the register **Project/Create New** and giving a related project name. At **Project/Open** you can open this project and do the requested product selection with a click on  (see arrow).

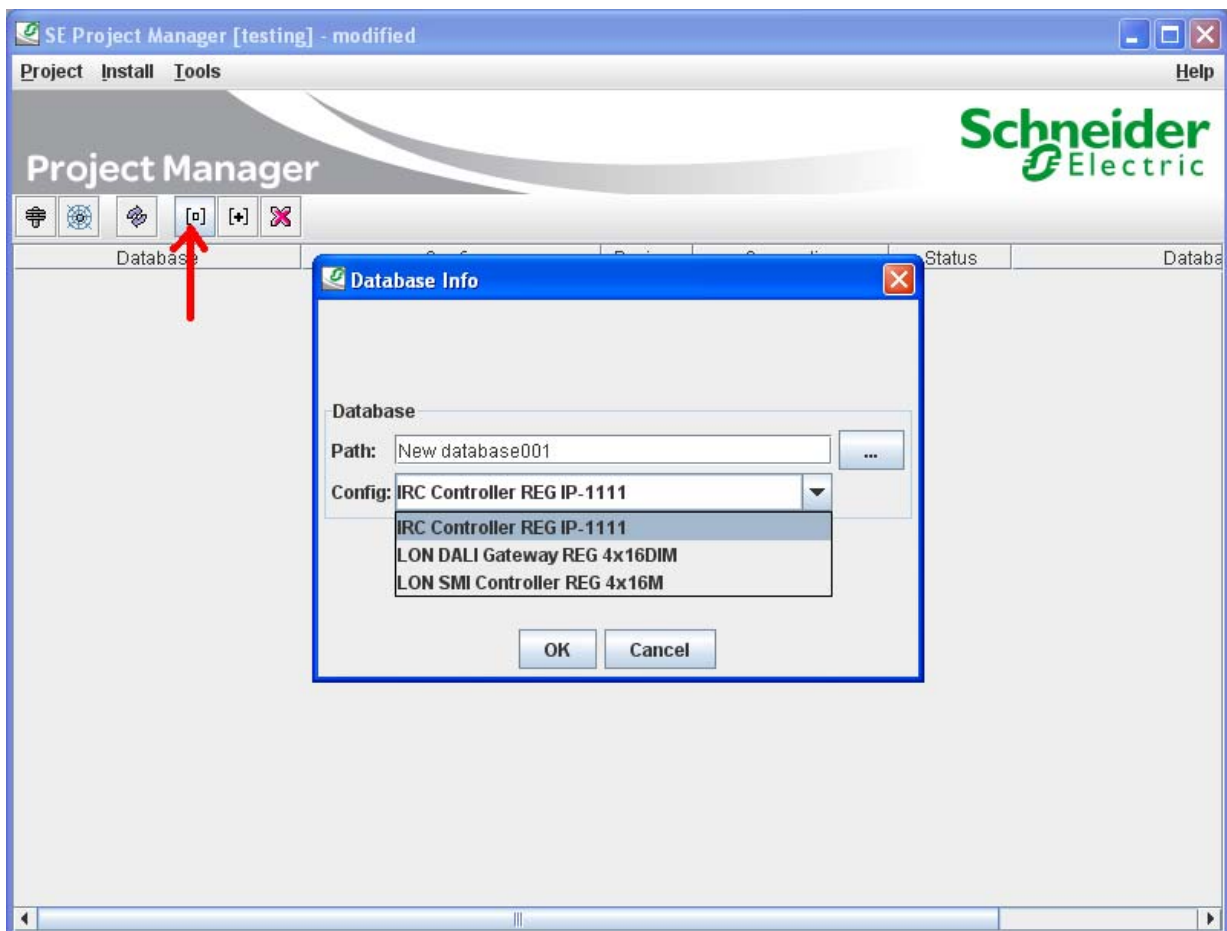


Fig. 6.3.3: Project start and related product selection

When initializing a new project, a copy of the folder ,template' will be created. It contains all available samples and standard configurations.

Each project contains and manages only **one** SMI Controller. Additional Controllers have to be integrated into new projects. The Project Manager implements the function to copy existing projects and therefore use as templates. Later on the SE Configuration Manager allows the function 'Save copy as' to create copies of the current running project.

The configuration of the device and related project at the Project Configurator can be activated by a double click on the designated device.

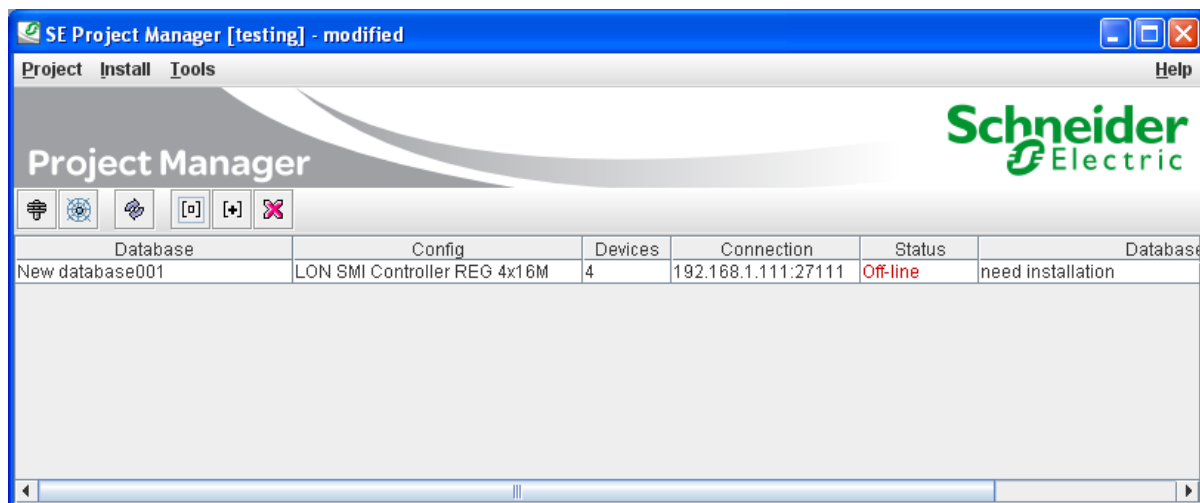


Fig. 6.3.4: Start of the SE Configurator

Each project is protected by a password.  
New projects are protected by the user name and password:

**User name: admin**  
**Password: admin**

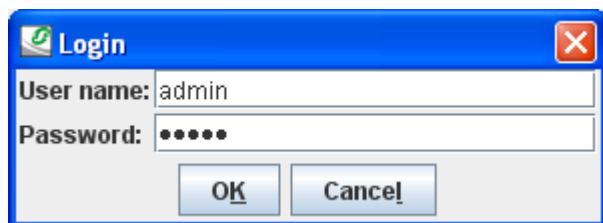
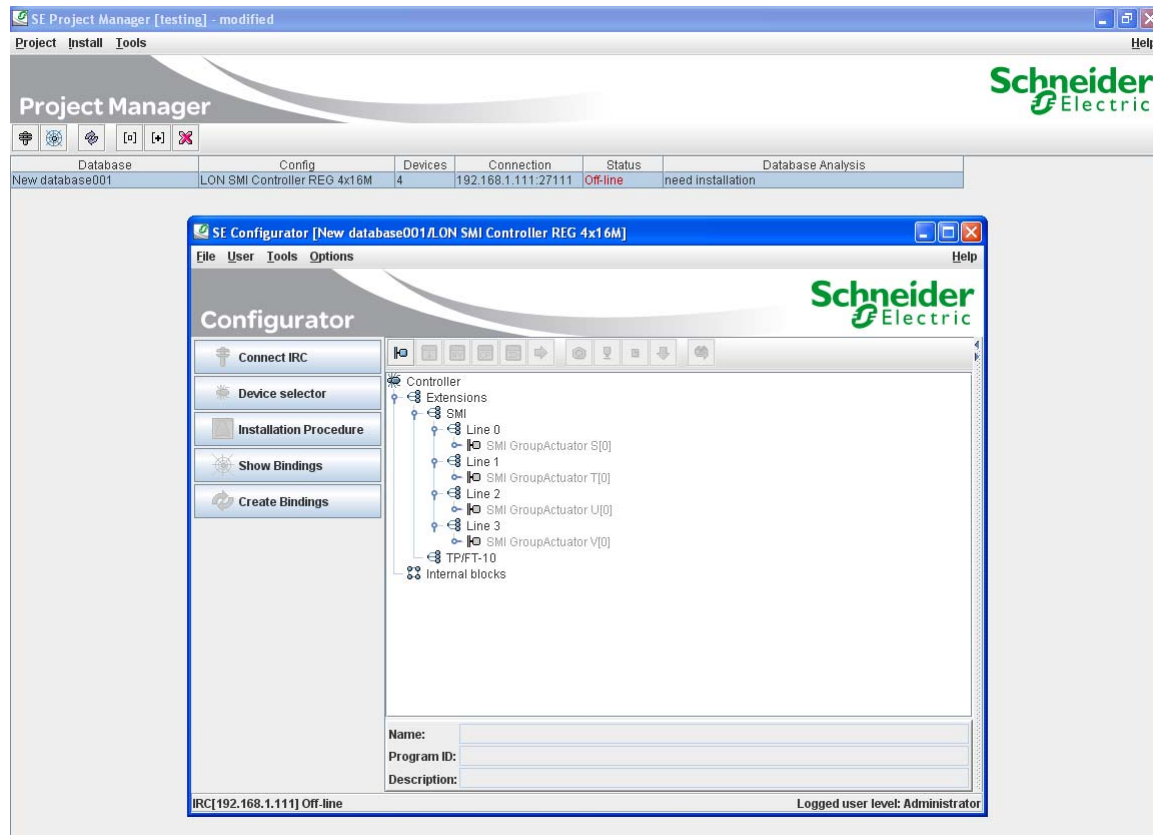


Fig. 6.3.5: Input of the user password

The SE Configurator now starts for a configuration of the requested project. Now user name and password can be changed at the register 'User' - 'Change password'.



## Configuration



**Fig. 6.3.6: Current Project configuration**

At the status line the current ,online' or ,offline' status of the Controller can be detected. Only the ,online' -modus signals access of the Project Configurator to the Controller.

The current project view (Fig. 6.3.6) shows the 'virtual' devices in a tree structure. These 'virtual' devices are divided into their related device interfaces. They are a copy of the physical devices at the SMI and TP/FT-10 interfaces and their internal functions.

At the SMI line you will find the ,virtual' device ,SMI Group Actuator' including 16 ,Actuator'-objects related to the ,Blind Actuator'–profile at LonMark. One actuator-object corresponds to one SMI group. At each SMI channel only one ,SMI Group Actuator' is possible.

At the category ,Internal blocks' basically controller functions can be found. Due to the uniform structure, these functions are shown as ,virtual' devices, each containing multiple function object of the same type: e.g. each ,Blind Controller' contains 4 functional blocks related to the LonMark-profile ,Sunblind Controller' for controlling sunblinds.

When closing the SE Configurator and Project Manager the project data always has to be saved with the function „save“. Otherwise new configurations get lost.

## 7 Creating a project

To create a new database configuration or to edit the current configuration, switch to the "Device Selector".

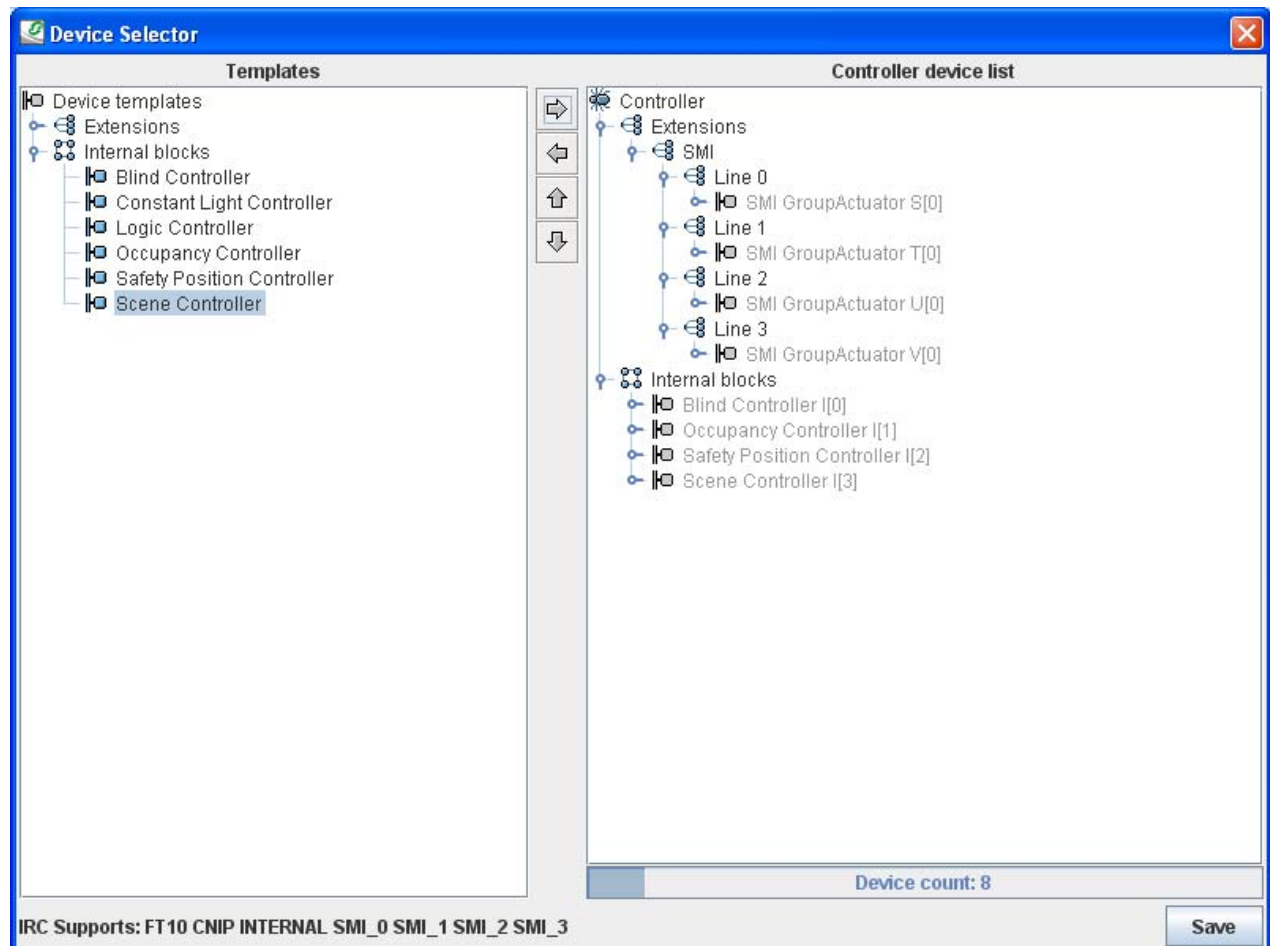


Fig. 7.1: Creating a device configuration (function) using the "Device Selector"

Down the left side of the "Device Selector" you will find the device templates. Use "drag&drop" or the arrow symbols to move the device templates into the "device list" or to remove devices that have been created (warning: This changes the LNS interface so that it is no longer compatible with an existing LNS interface. In this case the program ID should be modified to suit, see IRC configuration). For all interfaces, insert the devices that are connected to the device. When the "Device Selector" is exited (either with the "Save" button or the "Close" button), the selected project configuration is saved in the database.

Creating a project

7.1 Configuration of the function objects

Select "Configure" in the context menu (right mouse button) to call up the configuration view. In general you will find the configuration parameters listed here in tabular form.

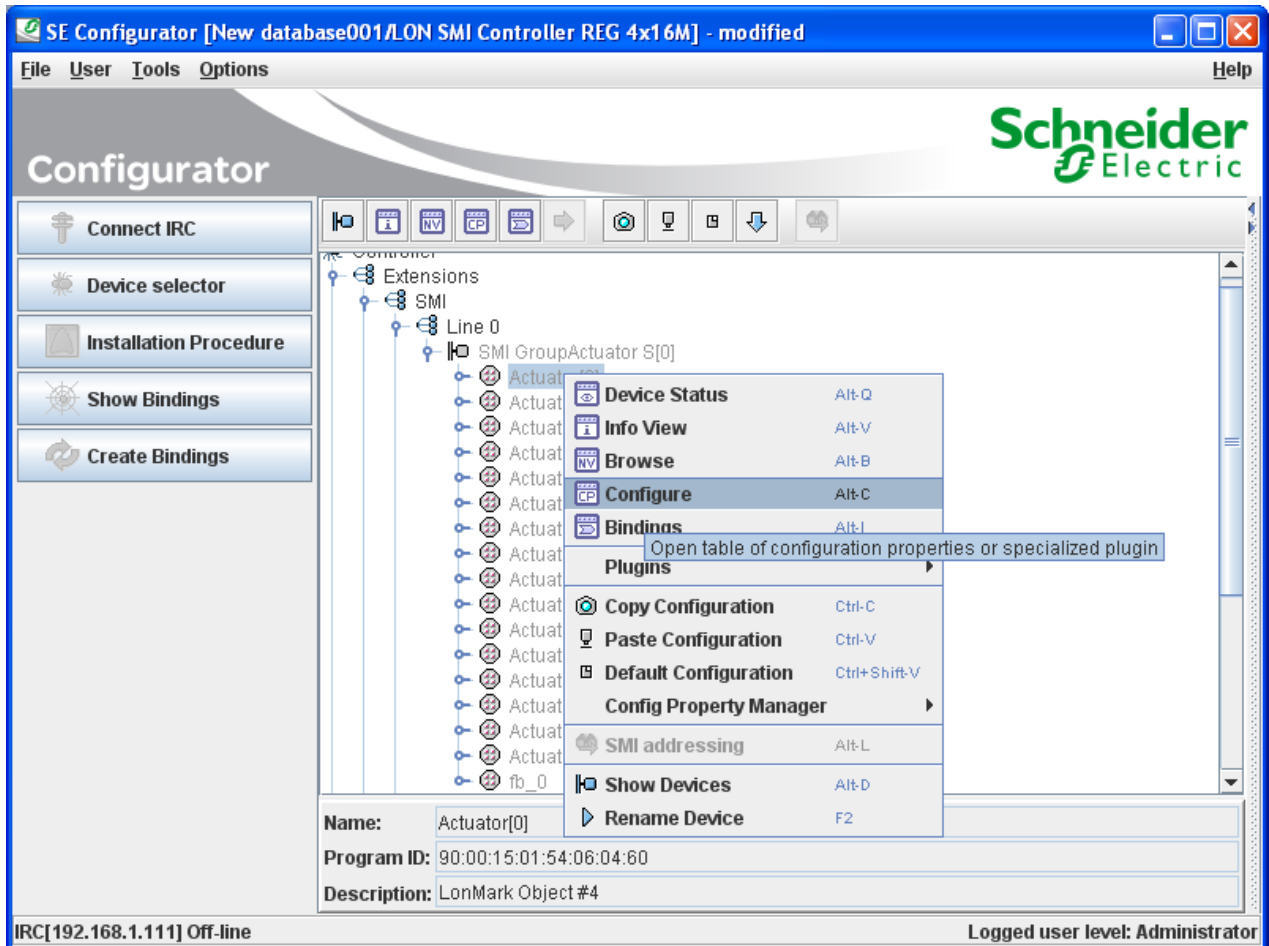


Fig. 7.1.1: Context menu for the function objects

Creating a project

Name	Value	Edit value
Panel turn pulses [units]	2000	2000
Panel turn model	LS_LINEAR	LS_LINEAR
Drive-down panel angle [degrees]	-75.0	-75.0
Drive-up panel angle [degrees]	75.0	75.0
Bottom panel angle [degrees]	0.0	0.0
Stop command	SC_STOP	SC_STOP
Default sunblind command [function, setting, rota...]	SET_STOP 0.0 0.0	SET_STOP 0.0 0.0
Maximum start-up delay [seconds]	0.0	0.0
Safety position command [function, setting, rotati...]	SET_UP 100.0 360.0	SET_UP 100.0 360.0
Working position [function, setting, rotation] [0]	SET_NUL 0.0 0.0	SET_NUL 0.0 0.0
Working position [function, setting, rotation] [1]	SET_NUL 0.0 0.0	SET_NUL 0.0 0.0
Working position [function, setting, rotation] [2]	SET_NUL 0.0 0.0	SET_NUL 0.0 0.0
Working position [function, setting, rotation] [3]	SET_NUL 0.0 0.0	SET_NUL 0.0 0.0
Maximum receive time [seconds]	300.0	300.0

Fig. 7.1.2: Configuration parameters of the function objects

Enter the desired value in the "New value" field. Quit the input with "Return" or the "!" symbol and the value will be written to the device, provided it is "online". Otherwise the configurations will be written at installation. The configuration can also be written to the device using the device context menu: "Info View" -> "Service" -> "Write CP file". The command "Read CP file" allows the current device configuration to be read.

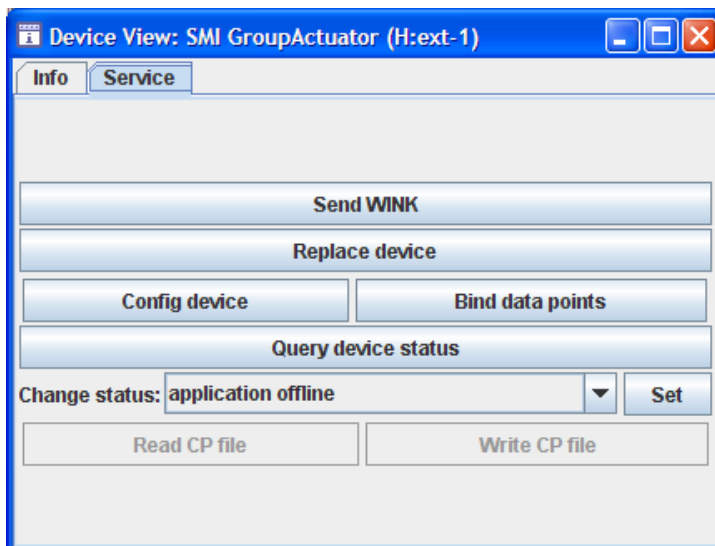


Fig. 7.1.3: Service menu for the device view

More complex devices can be configured using special views. You will find a description below of the device functions for which a special configuration view exists.

## 7.2 LON control panels

Appendix C contains a list of the devices that are supported by the LON SMI Controller.

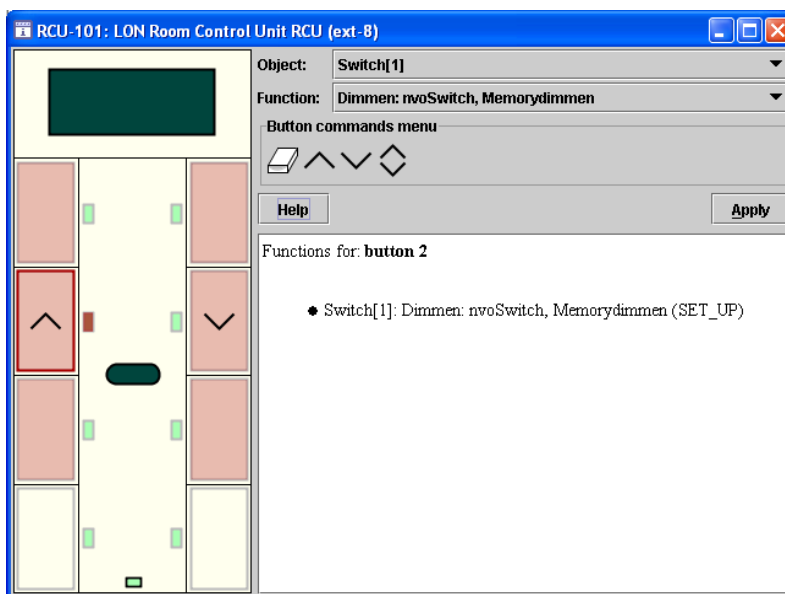
The configuration regarding control panels consists largely in the selection of operating functions and the assignment of control buttons and LEDs. For button controls the following function objects are available at the devices: Switch, Scene Panel, Occupancy Sensor. There follows a short description of these function objects. More detailed information can be found in the device documentation for the respective control panel.

Use the "Plug-Ins" context menu for any object to select the application module to be used. The respective view will open.

### 7.2.1 Switch Object

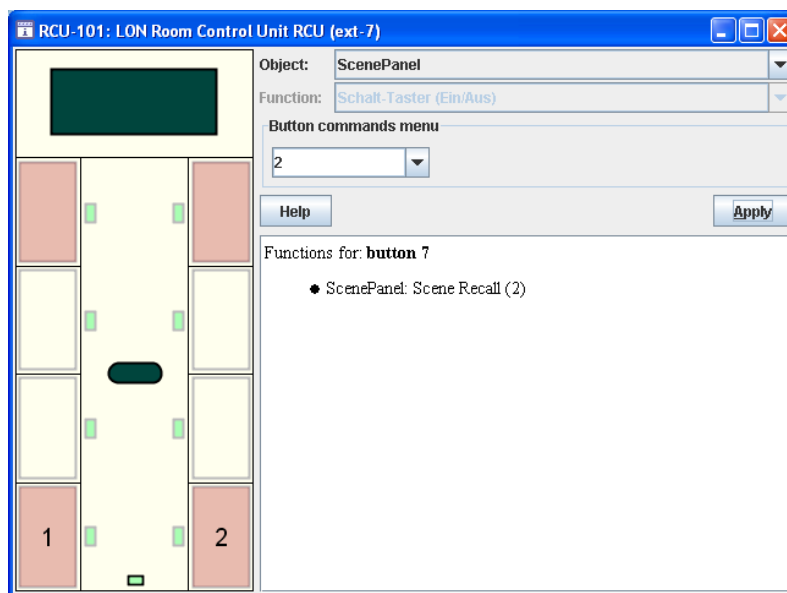
First select the function. Select the desired button function from the symbol menu and use "drag&drop" to move this to the desired button. The selection can be removed using the "Eraser" symbol or can be overwritten by other symbols.

The selection of the LED is performed by double clicking on the LED symbol. After they have been assigned, the selected buttons are highlighted in red.



## 7.2.2 Scene Panel Object

For configuration of scene calls, first highlight the desired button and then enter the respective scene number into the "Button commands menu" field. Press the "Enter" button to load the value. To delete an existing scene call, highlight the respective button; the configured scene will appear in the "Button commands menu" field. Delete this value, so that the field no longer shows an entry. Then press the "Enter" button to delete the existing value.



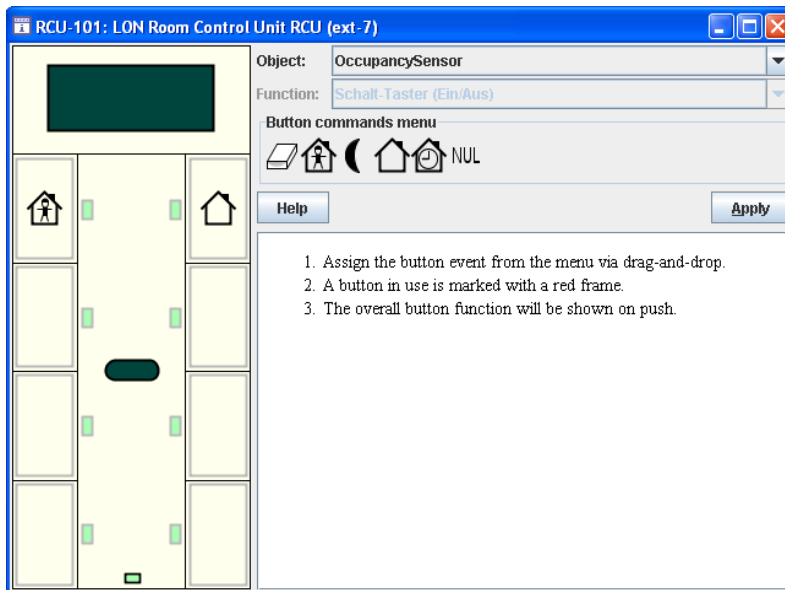
## 7.2.3 Occupancy Panel Object

Select the desired button function from the symbol menu and use "drag&drop" to move this to the desired button. The selection can be removed using the "Eraser" symbol or can be overwritten by other symbols.

The following functions are available for selection:

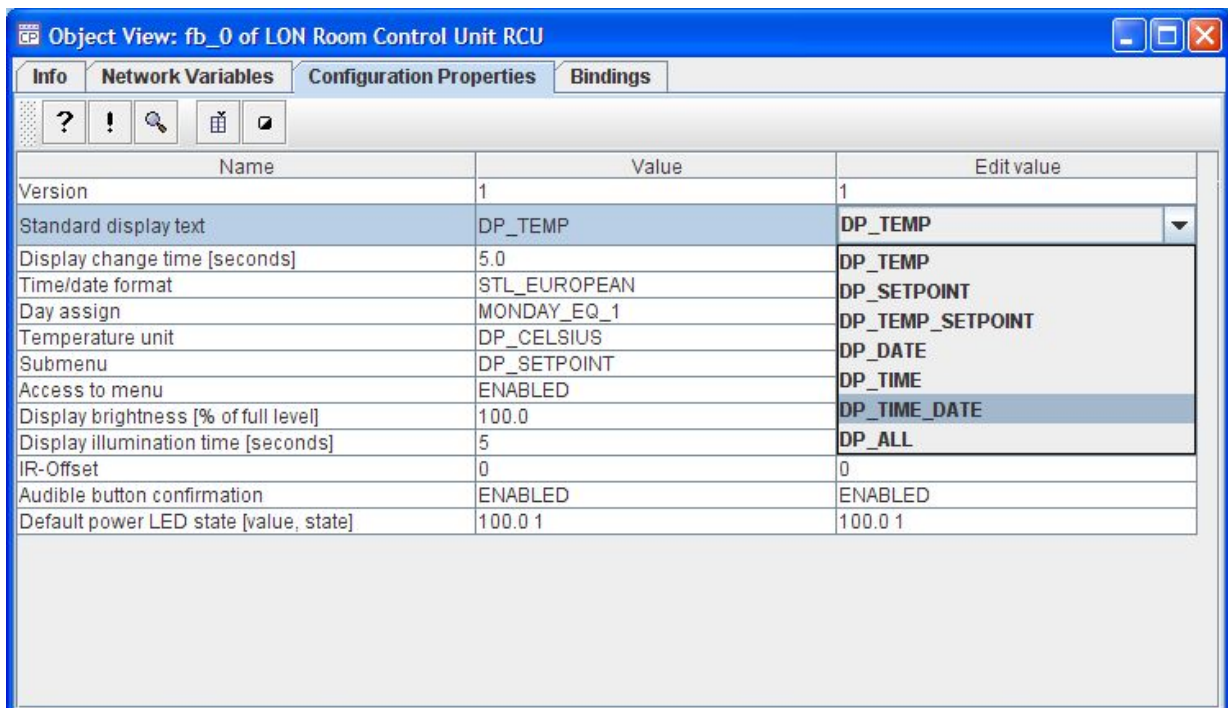
- Occupied (OC\_OCCUPIED)
- Unoccupied (OC\_UNOCCUPIED)
- Standby (OC\_STANDBY)
- Bypass (OC\_BYPASS)
- Invalid value (OC\_NUL)

Creating a project



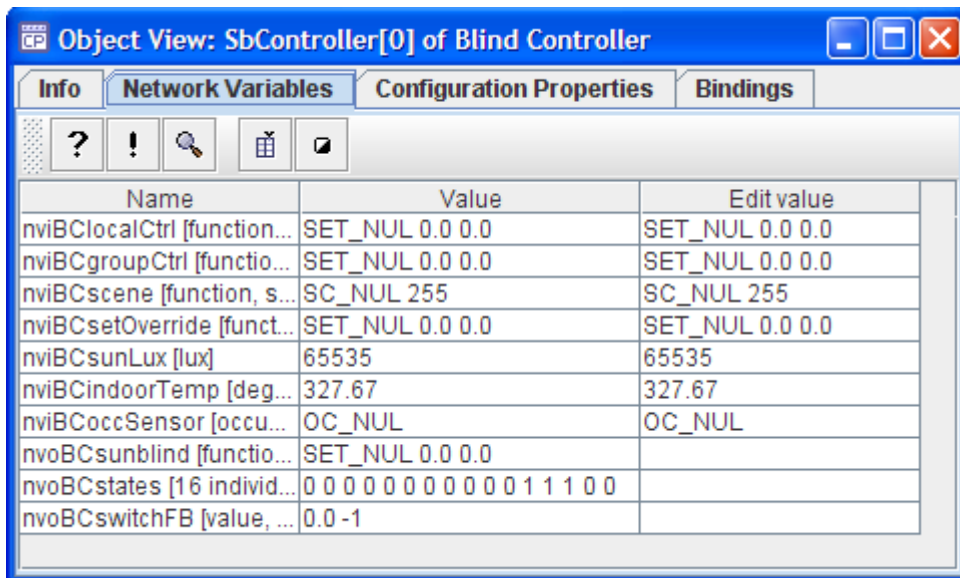
7.2.4 fb\_0-Object


The configuration parameters for general device functions, in particular those for setting display behaviour common to multiple objects, can be found in the context menu of the fb\_0-Object (-> Configure). The detailed description of the configuration parameters can be found in the device documentation for the respective (control) device.




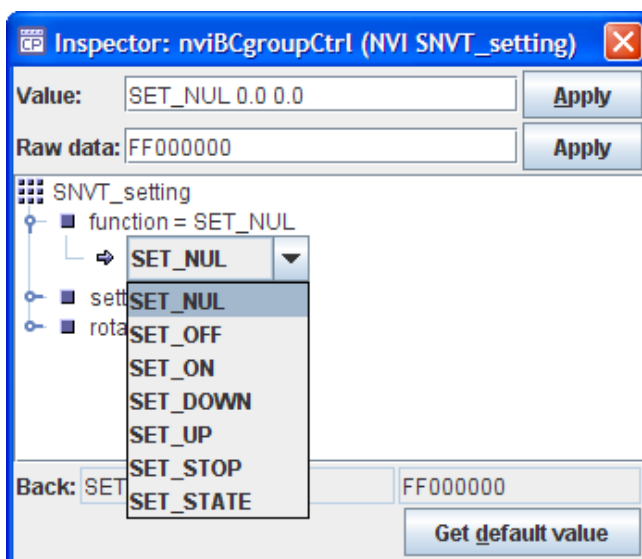
### 7.3 Simulation and test

You can describe and read the network variables directly in order to test their function. Select the item "Browse" in the context menu of the respective function object. The variables view will open.



First activate the "Polling"  option. You will then see the current values of the variables in the "Value" column.

Under the "New Value" item you can set the network input variables. You can also use the value inspector for this .





## 7.4 Saving a project

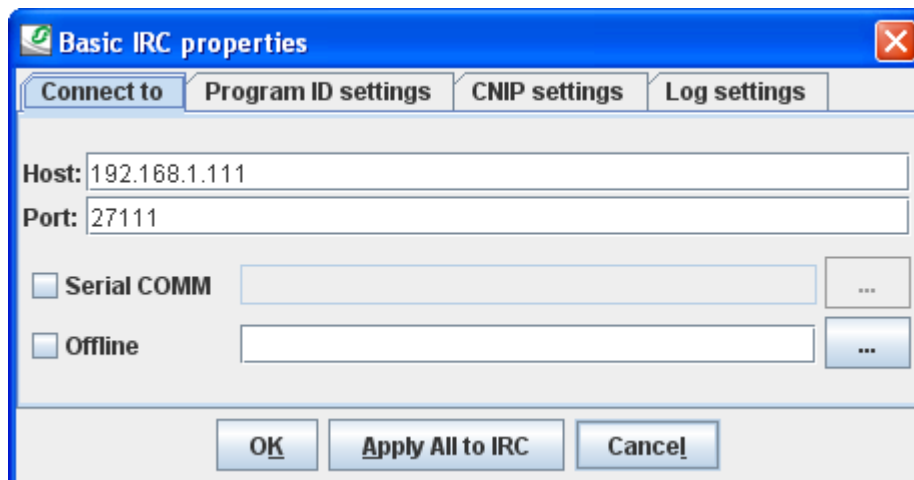
The current configuration can be saved with File -> Save Database. It is absolutely essential to save project changes during editing and when quitting the configuration tool. File -> "Save Database Copy as..." allows a copy of the current project database to be made. This copy can be the basis for configuration of another device.

## 7.5 Basic configuration

The LON SMI Controller is factory-set to a standard configuration (IP address: 192.168.1.111). This is also the standard setting of the SE Configurator, so that the device can be addressed during first commissioning.

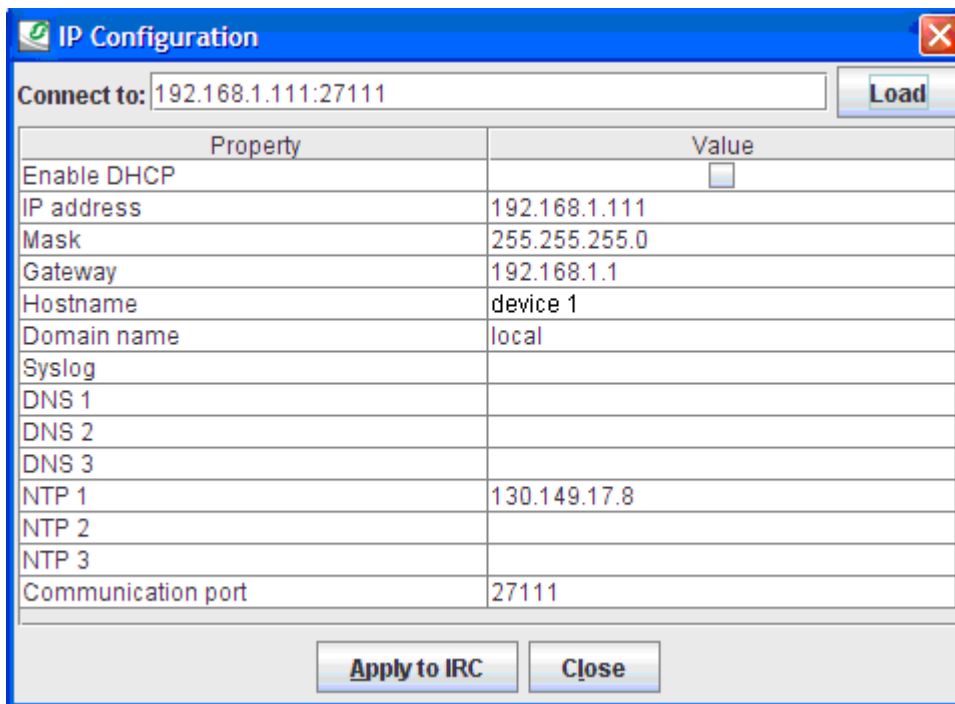
If the device is directly connected to the computer (not through a switch or hub) a crossover network cable should be used for the connection. Note: Be sure always only to connect a device with the standard address, since otherwise address conflicts can arise.

The IP address of the device with which it will be addressed by the SE Configurator is entered in "Options -> IRC configuration -> Connect to".



## 7.6 Configuration of the network address

Each device requires a unique IP address for binding into an IP network. To set the IP address, go to the menu "Options → IP Configuration".



Property	Value
Enable DHCP	<input type="checkbox"/>
IP address	192.168.1.111
Mask	255.255.255.0
Gateway	192.168.1.1
Hostname	device 1
Domain name	local
Syslog	
DNS 1	
DNS 2	
DNS 3	
NTP 1	130.149.17.8
NTP 2	
NTP 3	
Communication port	27111

Fig. 7.6: IP settings

The "Load" button is used to read the current device settings from the device.

Assigning an individual address can be done either dynamically using the "Enable DHCP" setting or from static addresses assigned by the network administrator. In the latter case please enter the IP address, the netmask and the default gateway. Host name and domain name are not required in the current firmware version.

Up to 3 domain name servers can be entered. The DNS server entries are currently not used. In many DHCP configurations however it may be necessary to enter a specific host name. Please contact your system administrator and ask about the necessary DHCP settings.

Under **NTP (Network Time Protocol)** up to three servers for synchronising the internal clock can be entered.

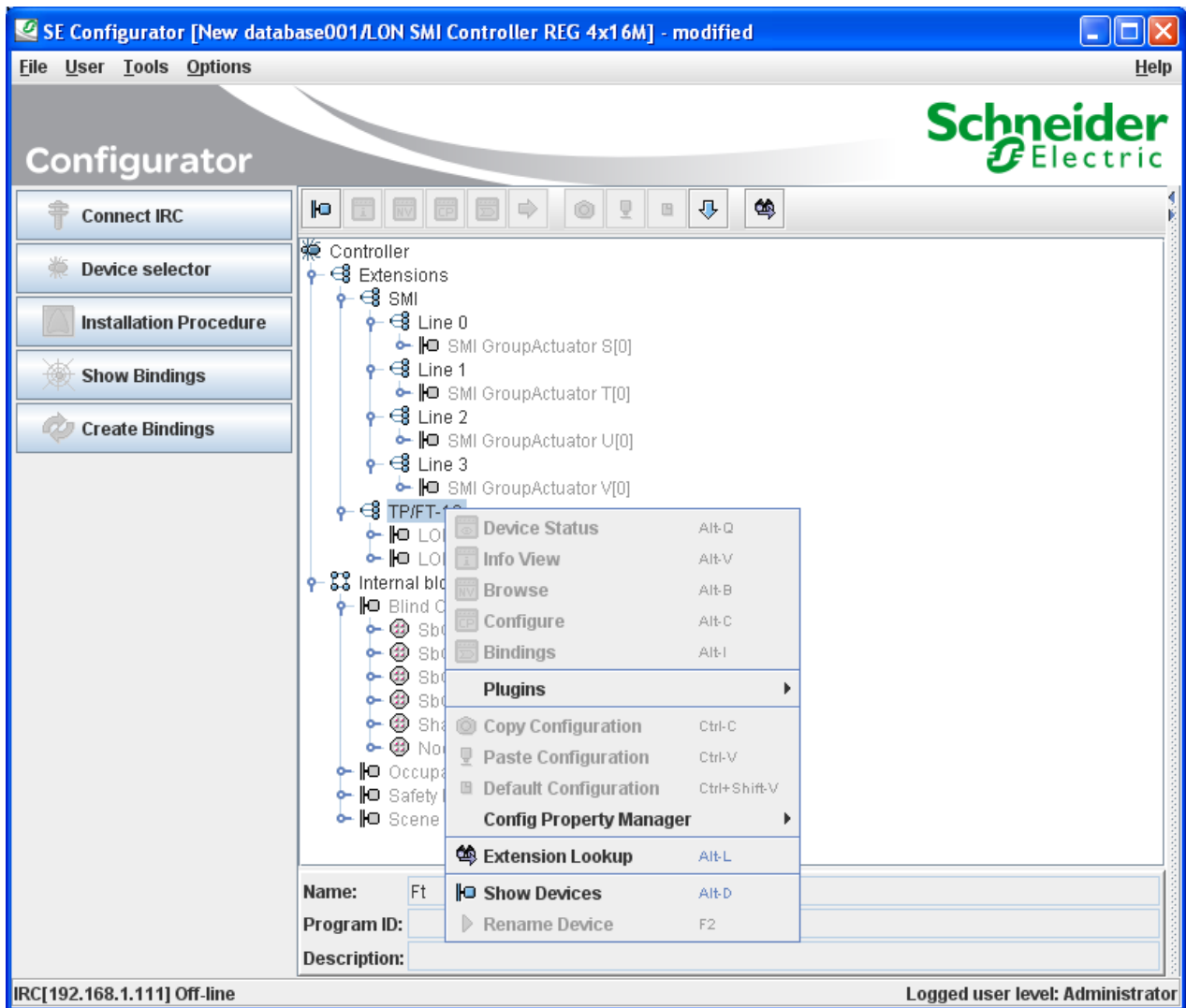
Enter the standard communication port (27111) in the "Communication port" field. In exceptional cases this communication port may be blocked by a firewall. In this case, please enter a free communication port.

After input of the IP address the status "online" should henceforth be displayed in the status bar.

Creating a project

## 7.7 Addressing the extension modules at TP/FT 10

Select "Extension Lookup" in the context menu (right mouse button) for the TP/FT 10 line, to address the devices connected to the TP/FT line.



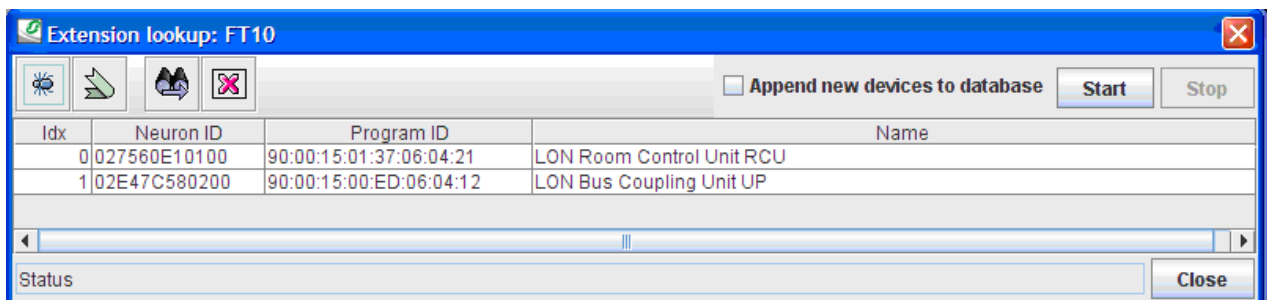
In this process you can bind in devices that previously had been created. To do this proceed as follows:

- 1) Select "Start".
- 2) Wait until the user message "Waiting for service pin (1 minute)" appears.
- 3) Actuate the "service pin" on the first LON device on the TP/FT 10 line.

### Creating a project

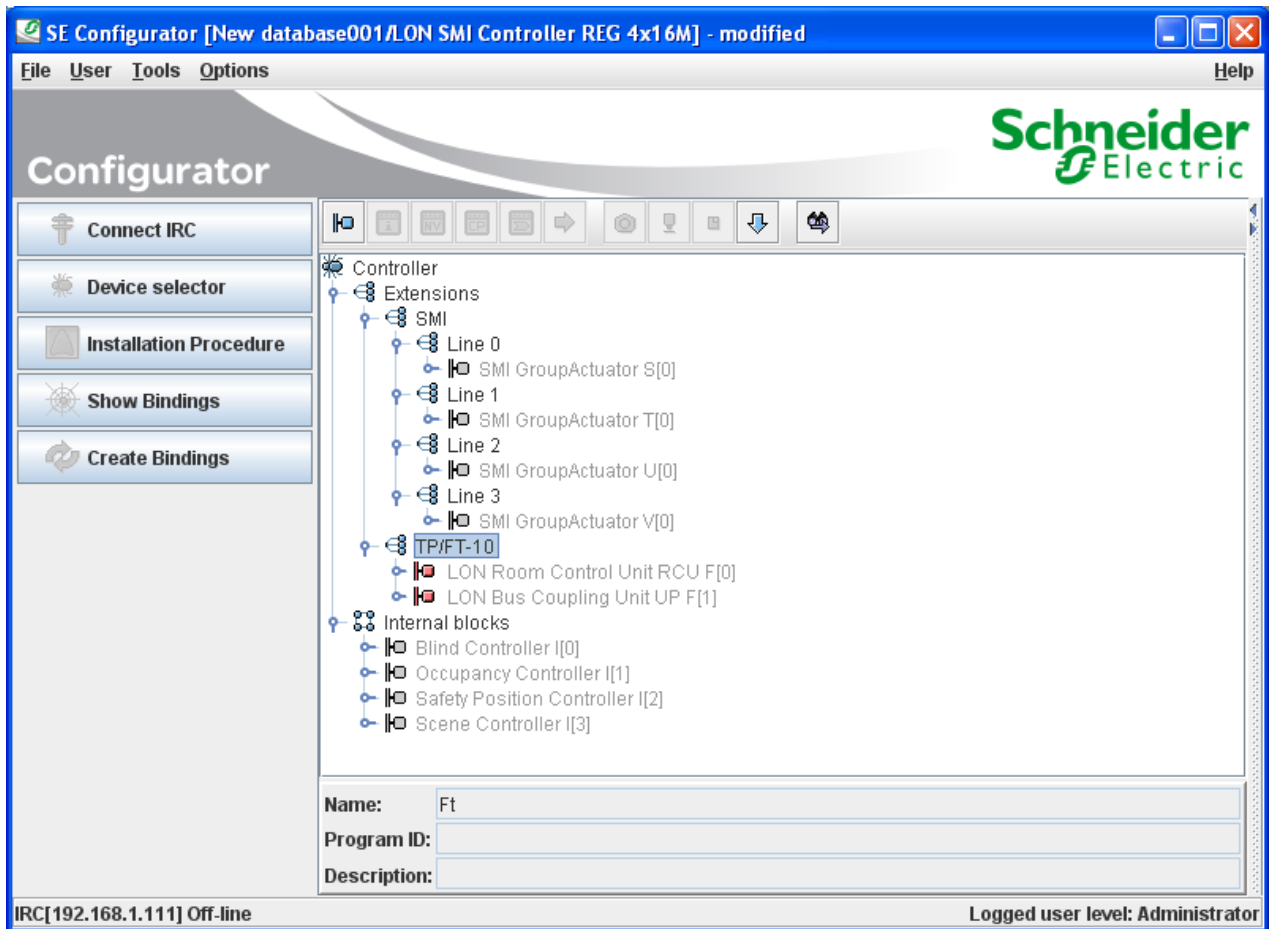
---

- 4) When the device has been successfully identified, both the neuron ID and the program ID will be shown in the list.
- 5) If the program ID does not match the anticipated program ID, the correct application must be loaded into the device that was found. The application is loaded under "Options" -> "Info View" -> "Service".
- 6) Repeat steps 2 and 5 for all LON devices connected to the TP/FT 10.



- 7) After you have addressed all the devices, end the process with "Stop", close the window and return to the project view. The addressed devices now appear brown in the tree view.

## Creating a project



## 7.8 Installation procedure

After you have created all the devices in the database and have addressed the TP/FT 10 line or IRC line, the project database must be loaded into the LON SMI Controller. Amongst other things, this generated the "virtual devices" and also creates the network variables.

Start the installation process with "Tools" → "Installation Procedure". The "virtual devices" are then loaded into the device. Depending on the size of the database, this process may take several minutes. On completion the program reports the successful configuration.

Creating a project

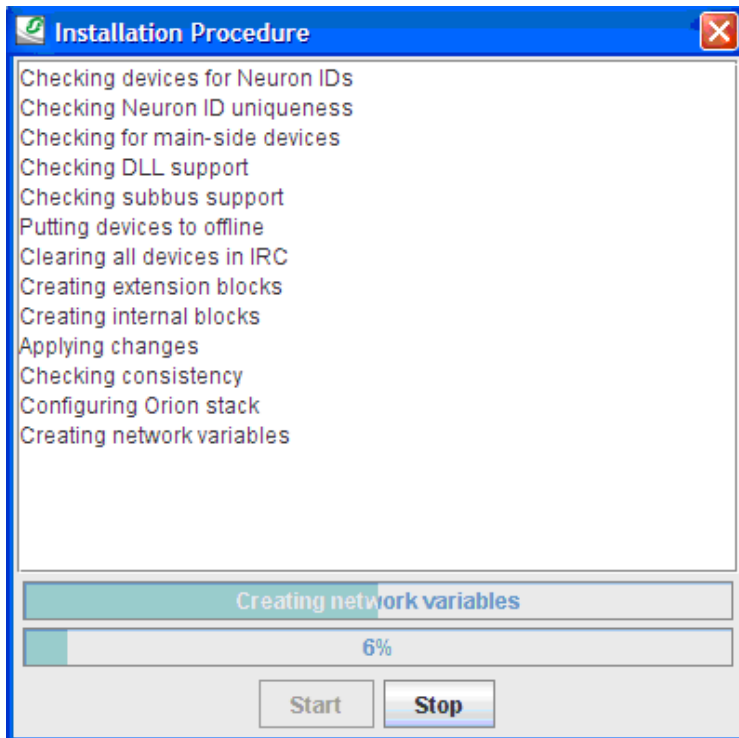


Fig. 7.8.1: Writing the configuration into the device

After the configuration is finished, the parameters will be written into the device.

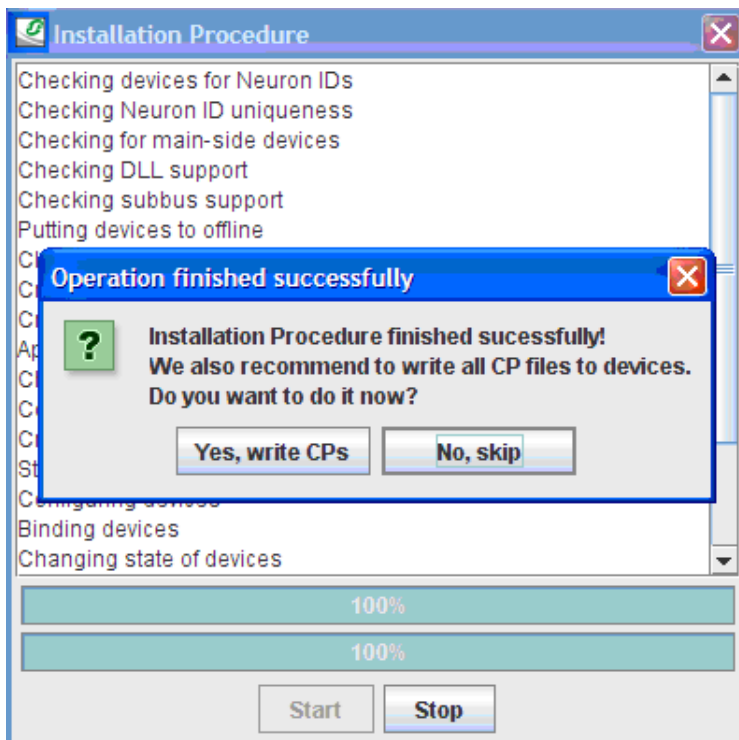


Fig. 7.8.2: Writing the configuration into the device

Creating a project

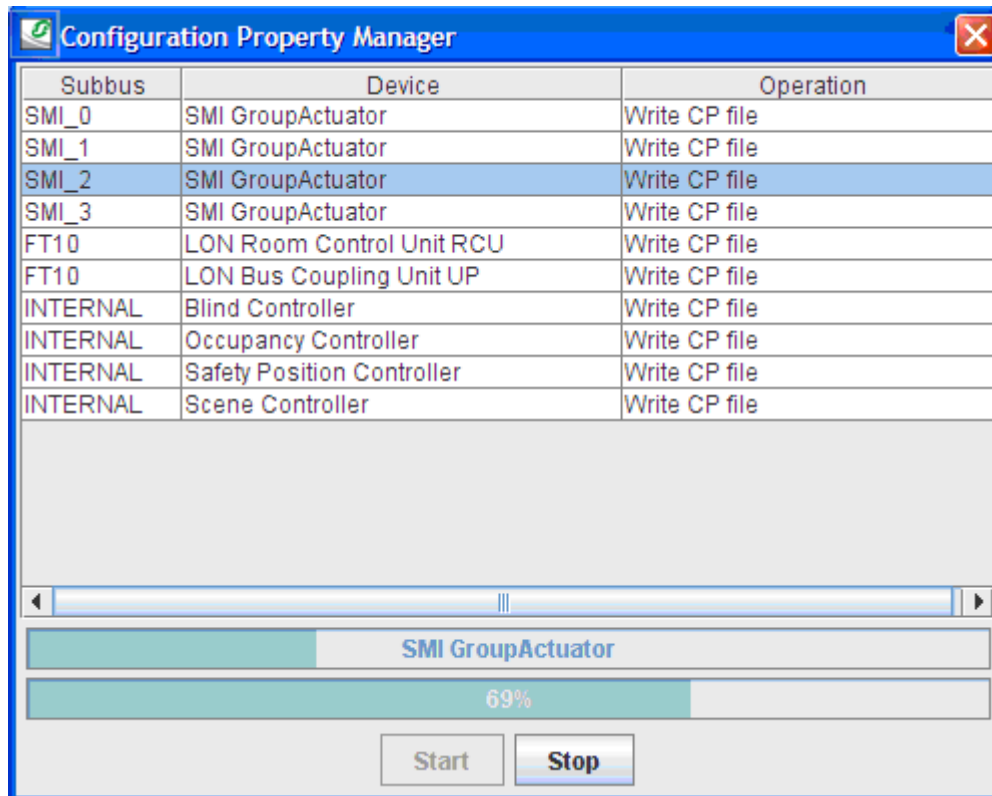


Fig. 7.8.3: Writing the configuration into the device

## 7.9 Commissioning the SMI lines

For commissioning the SMI lines the device must be "online" (see status bar). The SMI devices must be connected and in operation.

Highlight the SMI line to be configured (line 0, line 1, ...). Then select " SMI addressing" in the context menu (right mouse button).

Creating a project

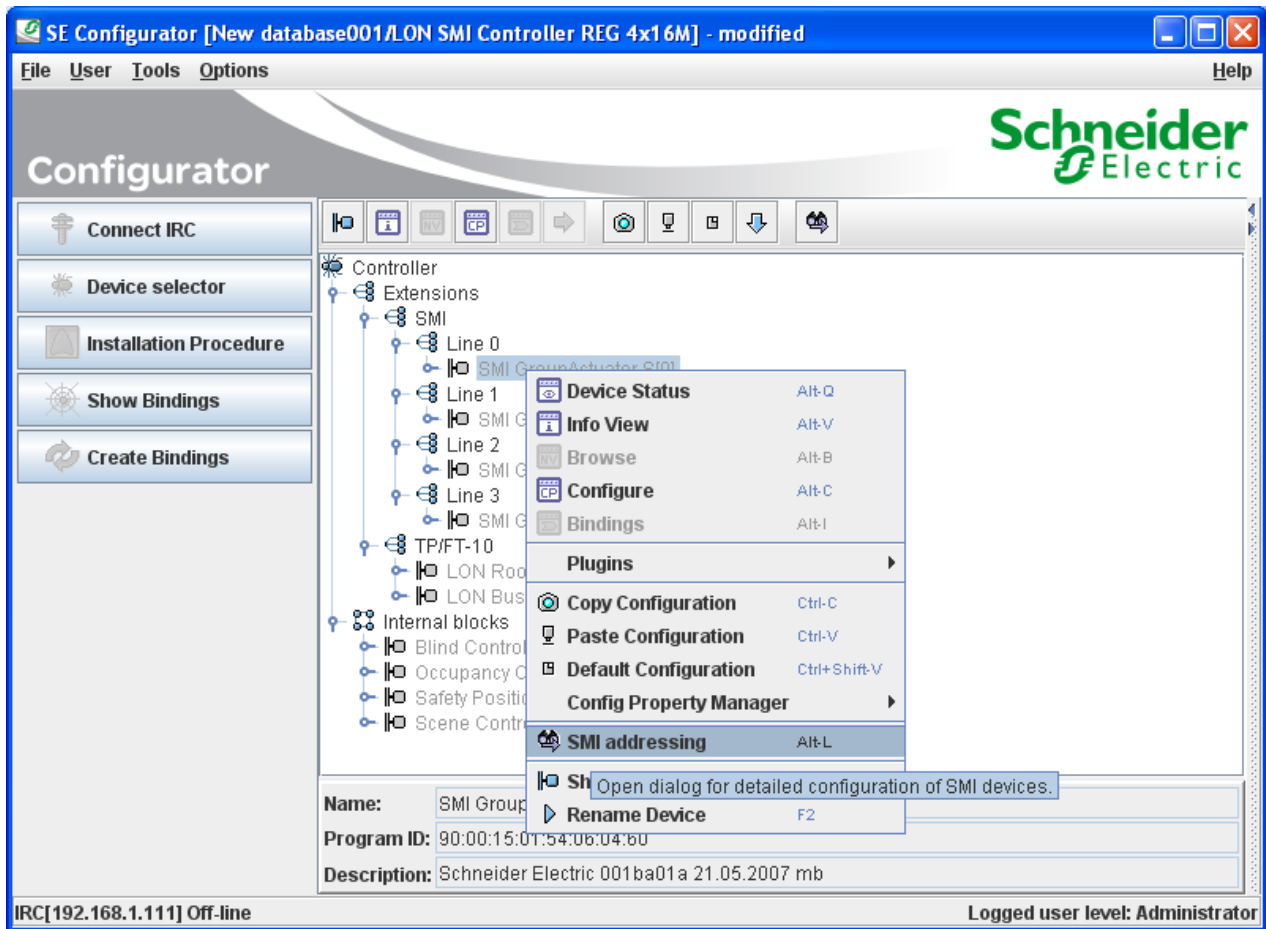



Fig. 7.9: SMI line context menu

### 7.9.1 Addressing the SMI devices

The commissioning window for the selected SMI line opens. Click on the symbol "Start SMI line

scanning" . This starts the search process for connected SMI devices. The devices that were found, together with devices contained in the database (e.g. from a previous scan of the SMI line) will be listed, and their device type and status displayed. A name for each device can be edited in the "Name" column. This will be saved as a configuration parameter.



## Creating a project

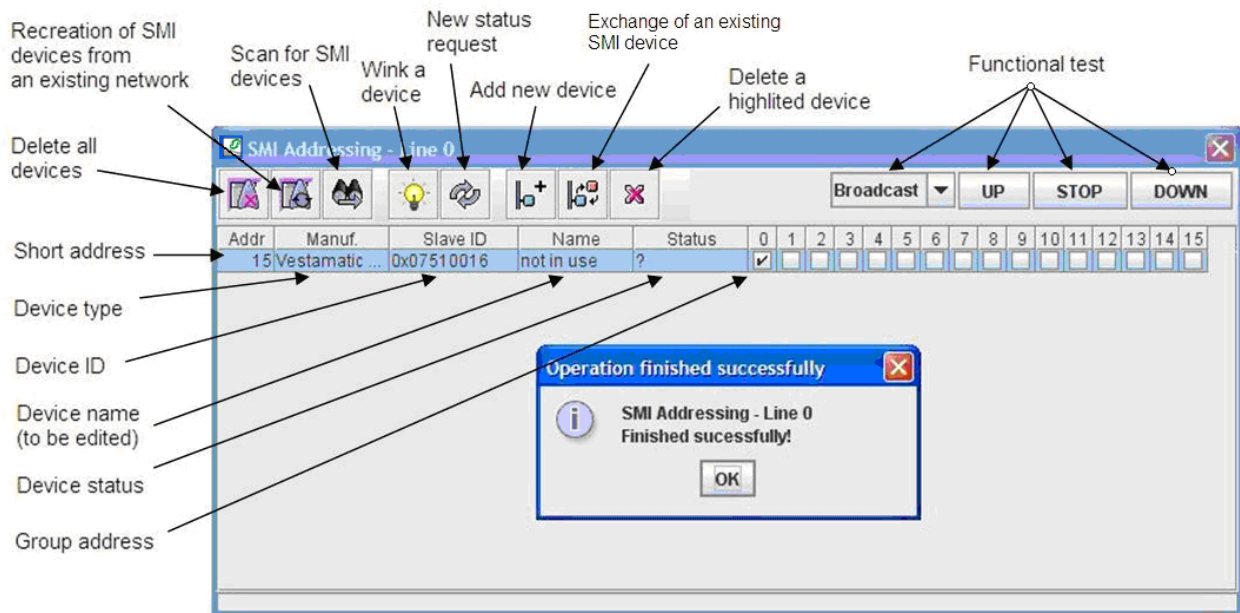


Fig. 7.9.1: SMI addressing

### 7.9.1.1 Wink a SMI device

The physical position of the SMI devices can be found by the "Wink" command. The selected devices move and thus could be identified.

### 7.9.1.2 Assigning the group address

The assigning of motors to one of the 16 available groups is performed by activating the checkbox 0...15. Each device can be assigned to only one group. The selected group corresponds to the object index of the "SMI Group Actuator".

### 7.9.1.3 Direct actuation of SMI devices

For test purposes the groups can be controlled individually or collectively by a "broadcast" to all devices.

### 7.9.1.4 Network Recovery Function

If the LON SMI Controller is inserted into an existing SMI network with motors that have already been configured (e.g. on exchanging the controller), you can press the "Recover" button in the

## Creating a project

---

"SMI addressing" menu to read the configuration of the entire SMI network into the device database. This contains all the address information stored in the motor.

### 7.9.1.5 Exchanging a SMI device.

To exchange a SMI device, proceed as follows:

- 1) Installing a new SMI device
- 2) Highlight the device to be exchanged (this should be shown in the status field as "?") and press the "Replace" button. The device will now be replaced with the unconfigured device. Note that for this exchange process only one unconfigured SMI device is connected.
- 3) Repeat steps 1 and 2 for further SMI devices to be exchanged.

If several unconfigured SMI devices are found, the process will crash. Therefore make sure that only one unconfigured SMI device is connected, or configure these devices using the standard addressing procedure and remove the defective devices from the database.

### 7.9.1.6 Manually exchanging a SMI device

It is not absolutely necessary to use the SE Configurator when exchanging one SMI device. You can use the device buttons for this. The pre-requirement however is that this allows only a defective device to be exchanged for an unconfigured device. Other cases require the use of the SE Configurator. If there are several defective SMI devices, it will always be the first short address that is replaced with the new SMI device.

Proceed as follows:

- 1) Install the exchange device
- 2) Perform a long button push on "Channel" to switch into manual mode. Then perform a short button push to select the respective SMI channel.
- 3) Test the connection by pressing the "UP/DOWN/BUS" button
- 4) Press the "Progr" button for longer than 3s.
- 5) When the exchange has been successfully completed, the LED of the respective SMI channel lights up orange (from version 0.1.3.) and the SMI device that was found "waves" for one cycle. There is no reaction otherwise. (e.g. if more than one unconfigured device was found on the line).
- 6) Perform a long button push on "Channel" to switch out of manual mode.

## 7.10 Creating binding links

This function is required only if the LON SMI Controller is not bound into a LON network!

Until now you have set up connected devices with their associated function objects on the LON SMI Controller. The application of the LON SMI Controller is set up by creating a binding link between the function objects and the desired overall function. The function objects contain

## Creating a project

---

network variables for communication with other function objects. Network variables "nviYYxxx" are input variables and "nvoYYxxx" are output variables. The connections between the input variables and the output variables are created by "binding". There can only ever be one input variable and one output variable of the same type bound to each other. The type information can be found as the "tool type" by pointing the cursor to the network variables.

The bindings between the object variables can be created using the Binding Editor.

Proceed as follows to create and edit bindings:

- 1) When "Show Bindings" is selected, the Binding Window opens.
- 2) Use the "Create New Binding" button to select the desired binding type. (Currently only the binding type "Binding between two data points" is supported.) A binding template will open with wild cards for the network variables to be bound.
- 3) In the left hand window, select from the project view the first network variable, drag it to a free position in the binding template and drop it.
- 4) Then select the corresponding network variable and drag this to another free position in the binding template. The variables must both be of the same type. Network variables of differing types cannot be bound to each other.
- 5) For further binding links, repeat steps 2 to 4.
- 6) Once all binding links have been generated you can load them to the LON SMI Controller by pressing the "Create Bindings" button.

Creating a project

Open the binding menu

Write bindings into the controller

Create new bindings

Delete bindings

Rename bindings

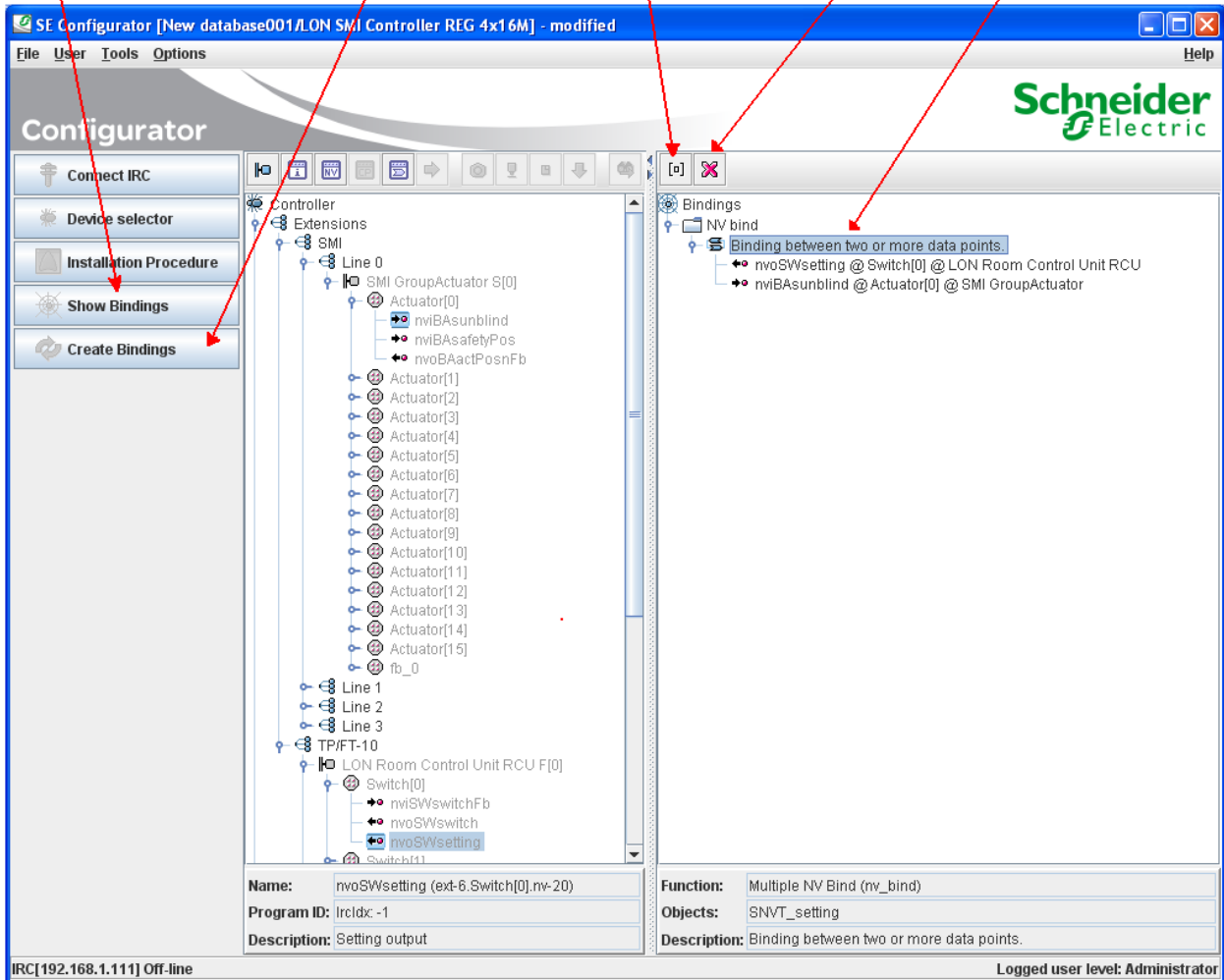


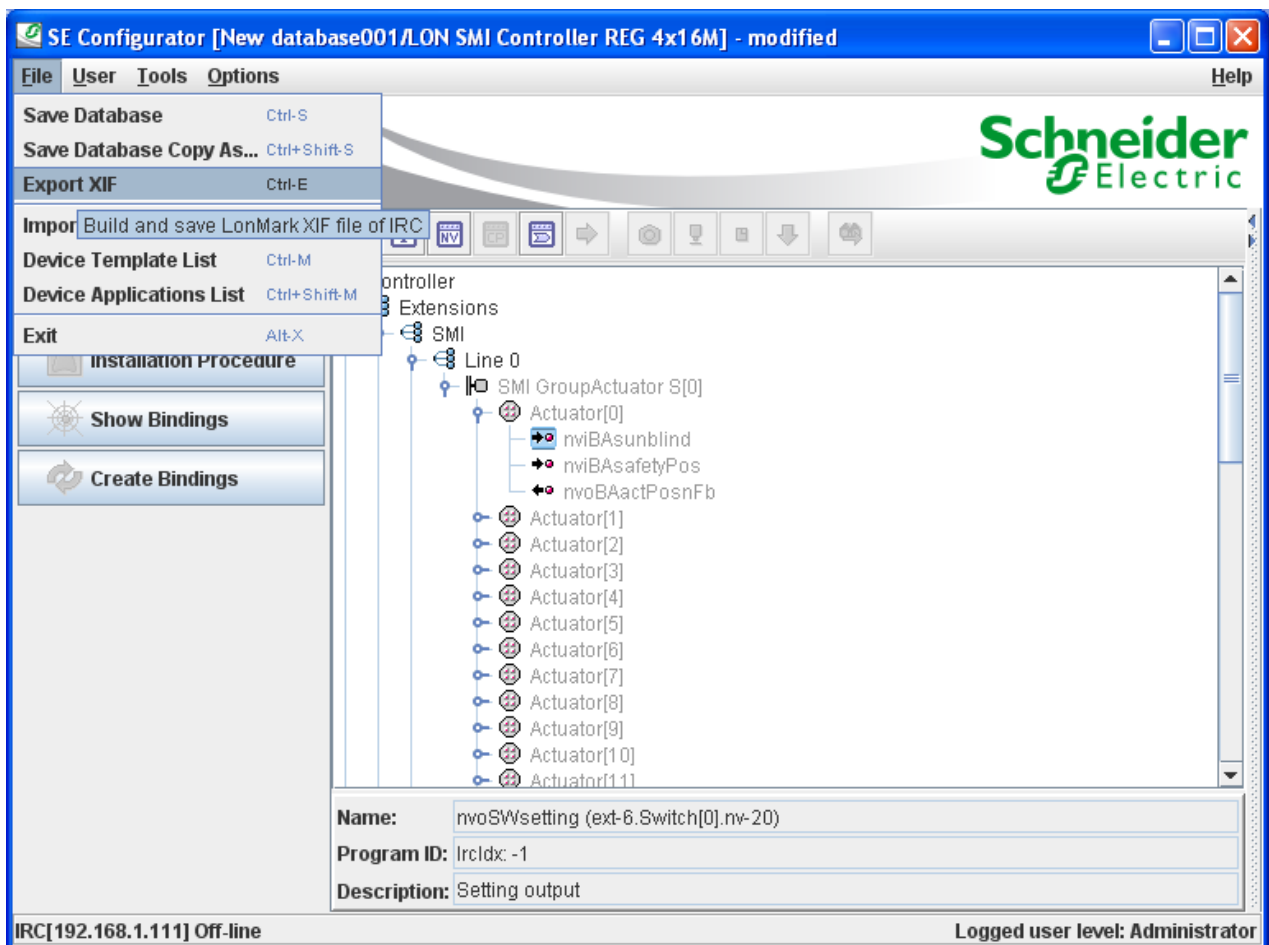
Fig. 7.10: Creating binding links in stand-alone mode

Creating a project

## 7.11 Application in a LON network

### 7.11.1 Creating a LON template (XIF)

Pre-requirement for an implementation of the LON SMI Controller into a LON network is the availability of a device template (XIF). After a project has been created **and loaded into the device**, a device template (XIF) can be created. To do this, select "File => Export XIF" in the folder and enter the desired path for saving.



This XIF file can be used to integrate the controller into a LON network management tool. It should be noted that configuration within this firmware version can only be performed by use of the configuration tool.

### 7.11.2 Program ID settings

For binding into a LON network further settings must be performed beforehand.

The LON SMI Controller can be bound into a LON network either using EIA-852 (Lon over IP) or using EIA-709 (TP/FT). The setting of the respective parameters is performed under "Options" => "IRC Configuration" => "Program ID settings" -> "Main side".

The setting "CNIP" indicates that the binding into the LON network is performed using the "100 Base T" connection. The setting "FT10" indicates that the binding into the LON network is performed using the TP/FT 10 connection. The program ID is automatically selected in accordance with the interface that is selected, since the transceiver settings are coded into it. For further management of the program ID the "version" can be modified to suit. This modification is necessary if an existing "template" already integrated into the LON network must be retrospectively changed.

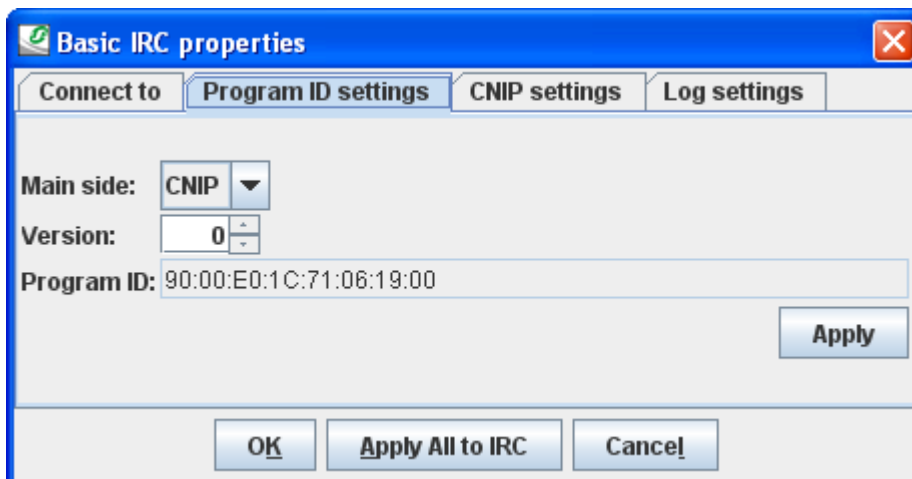


Fig. 7.11.2: Basic settings

### 7.11.3 Structure of an IP channel

IP is an acronym for Internet Protocol. IP is part of the TCP/IP protocol family (Transmission Control Protocol/Internet Protocol). The IP is the general program basis used for a worldwide exchange of computer messages by the Internet and within many LANs (Local Area Networks) and WANs (Wide Area Networks).

A LonWorks/IP channel is a communication medium which conveys IP packets that contain LonTalk packets. If the computer on which the LON commissioning tool is running is connected to a LONWORKS/IP channel, it must have an IP network interface (such as an Ethernet network card or a modem with PPP software) which it can use for communication with the physical network (extract from the LonMaker manual 1-11, 1-12).

## Creating a project

---

Note: It is essential to have a configuration server for setting up an IP channel. A configuration server is not a component of the LON SMI Controller. We refer at this point to the use of the configuration servers from Echelon and Loytec.

First create a LONWORKS/IP interface on the computer on which the LON commissioning tool is installed. To create a LONWORKS/IP interface, proceed as follows:

- 1) Point to Settings in the Windows Start menu, then select the control panel. Windows system control will open.
- 2) Double click on the system control application LNS IP Configuration (LonWorks/IP Channels) Windows panel control.
- 3) Click on Add. The dialog field Add an IP Device will open.
- 4) Input a unique name for the computer and check whether the displayed IP address tallies with your network card. Leave the port address as 1628.
- 5) Click on OK. The dialog field Add an IP Device will close.
- 6) Click on OK. The LNS IP configuration system control application will close.

Further information can be found in the Help file for the LNS IP configuration system control application.

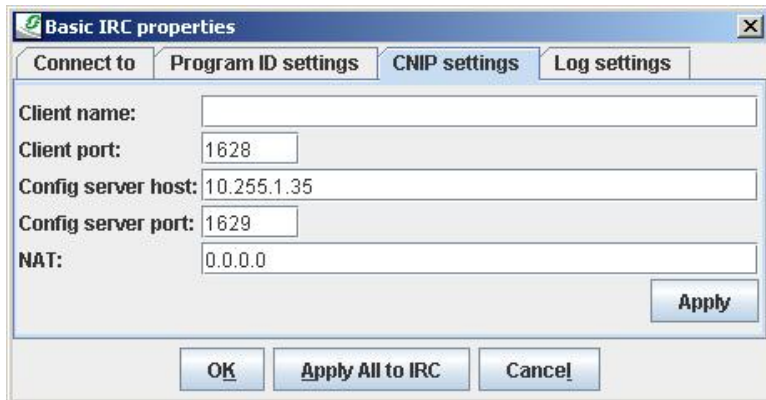
Define the LONWORKS/IP channel and the devices in the configuration server. The procedure to use depends on the configuration server in question. In this respect we refer to the respective data sheets issued by Echelon and Loytec.

The LonMaker computer is not fully commissioned on the LONWORKS/IP channel until you execute the LonMaker tool. Further information can be found in the Help file for the Loytec configuration server or in the user manual for the i.LON configuration server.

Creating a project

### 7.11.4 CNIP settings

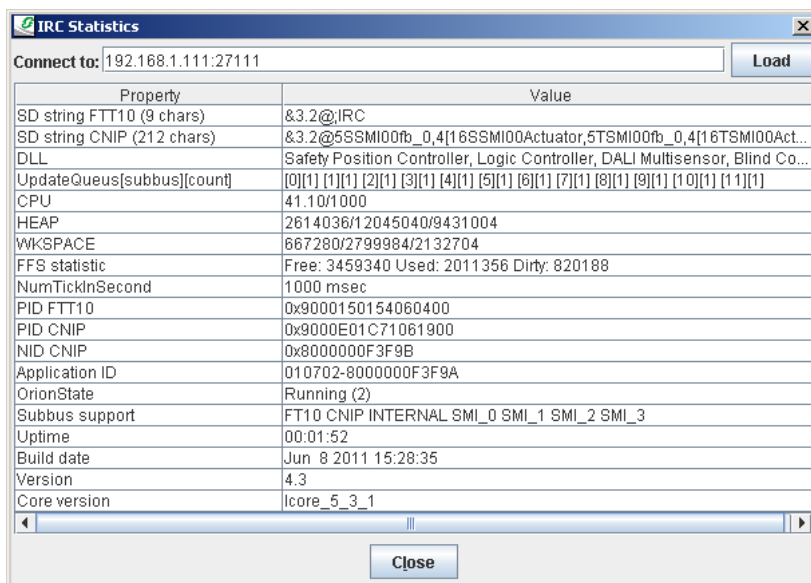
The participants on the IP channel are managed by a "configuration server". Enter under "CNIP settings" the name under which you will log in this device at the configuration server, and enter in "Config server host" the IP address of the configuration server. The port settings should be made as listed below.



### 7.12 Tools

You will find the following service programs in the "Tools" folder:

- "Check connector": Shows whether the device is online.
- "IRC devices": Shows the current configuration of the device, in case it is "online".
- "IRC Statistics": Shows general information on the device. 'Version' shows the currently loaded firmware version.





## Creating a project

- "Clear IRC Tables": Deletes all configuration entries in the device. If a project requires all existing devices to be deleted, please first use this function.
- "Upload IRC Firmware": Loads new firmware. Loading the firmware deletes the configuration data. The device configuration (Installation Procedure) must be renewed (as it must after LNS installations).

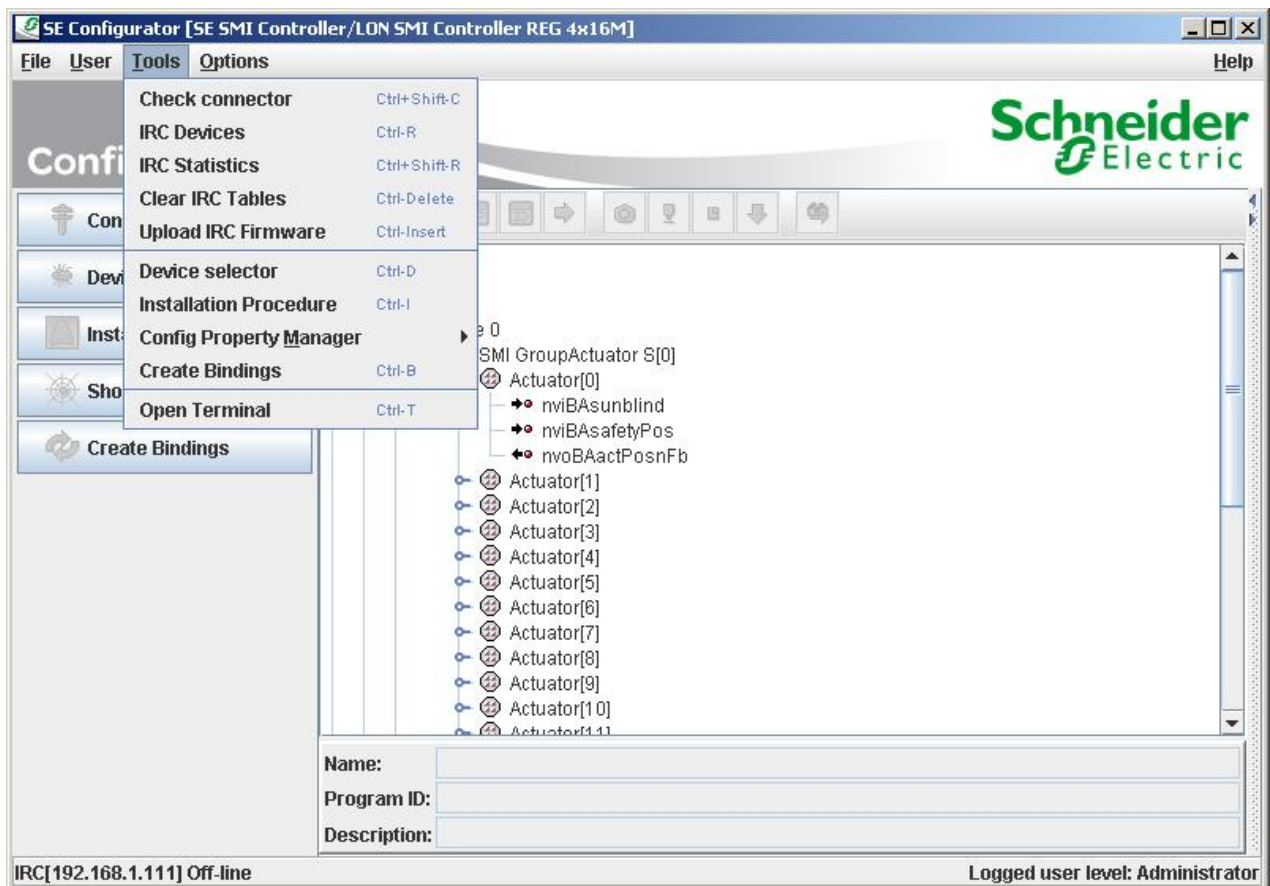


Fig. 7.12: Service programs

## 7.13 Configuration using the Web browser

The major settings can also be performed using the integral Web server. The following Web browsers are supported: Firefox 1.5 and Internet Explorer from version 6.0.

Enter the IP address of the device (default 192.168.1.111).

Note that in this firmware version it may take a few seconds to generate the pages, since these are created dynamically.

You can then navigate through the various subjects:

Creating a project

7.13.1 IP SETTING

Set the new IP configuration of the device here: IP address, netmask and gateway - all other settings are optional. The new settings become active only when the device is restarted.

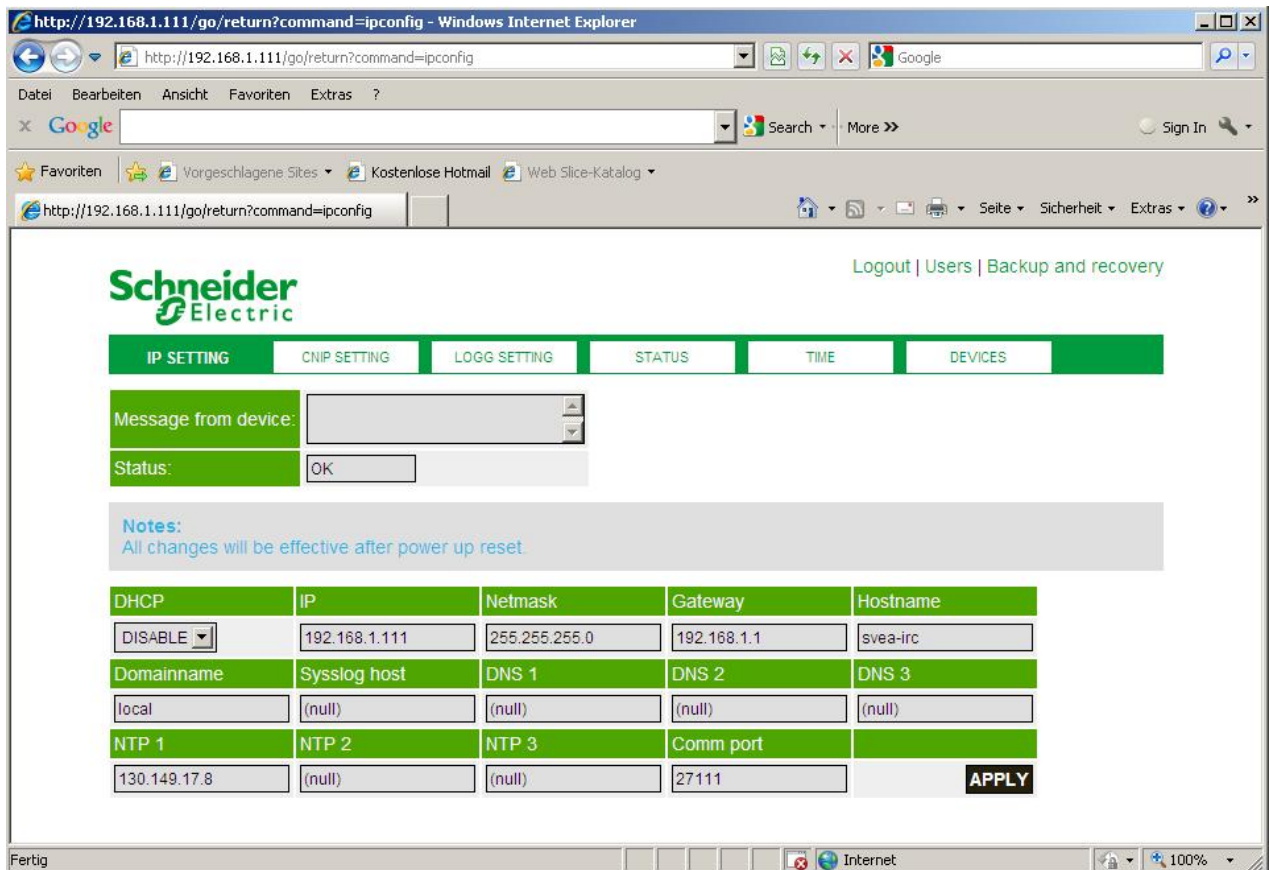


Fig. 7.13.1: Setting the IP configuration: IP address, netmask, gateway ...

### 7.13.2 CNIP SETTING

If the device is used in a LON over IP network, the device is logged on to a "Configuration Server". Enter here the IP address of the "Configuration Server".

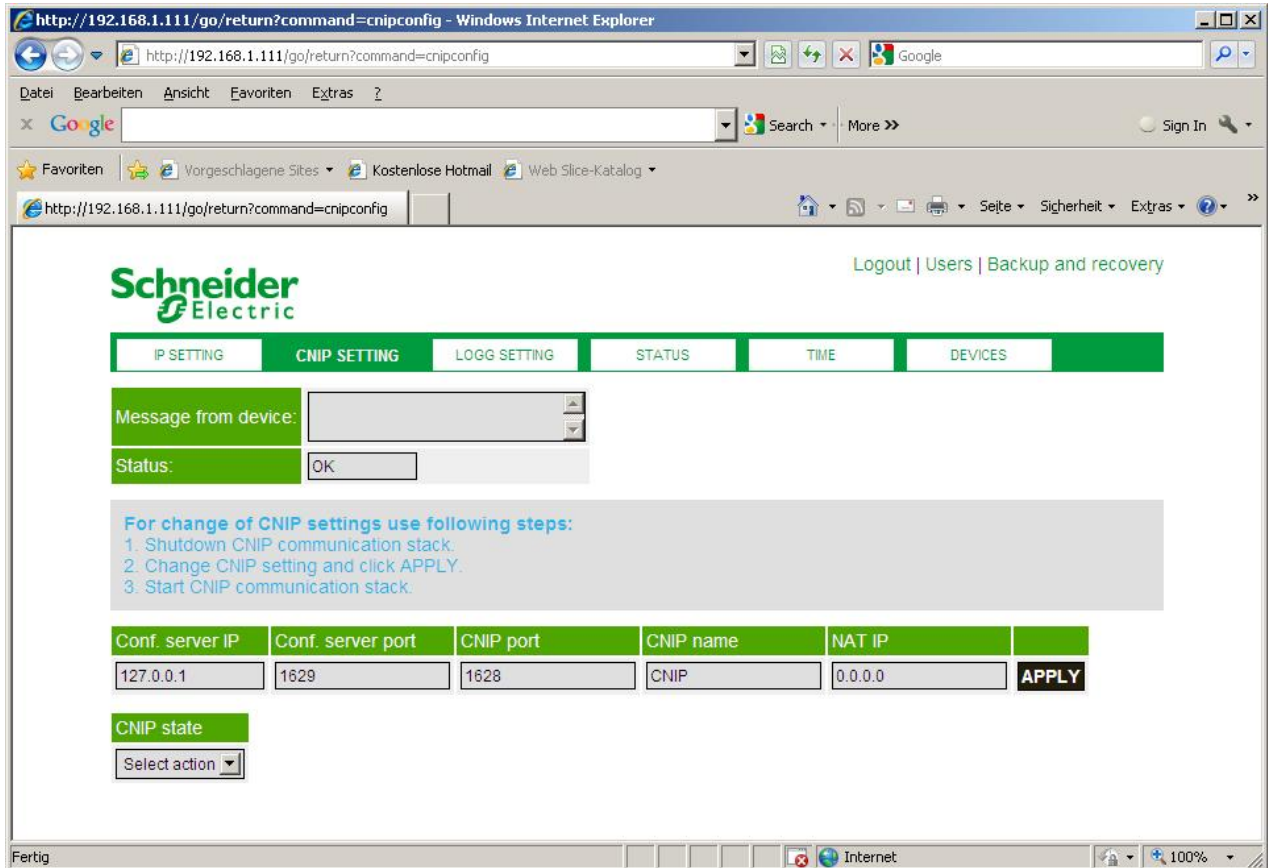


Fig. 7.13.2: Setting the IP address of the configuration server when a device is used in a LON over IP network

Creating a project

### 7.13.3 LOG SETTING

System and debug information can be saved temporarily in the local file system. Two files are available for this purpose; these are described successively as ring memories. Select here the sort of information that you wish to save.

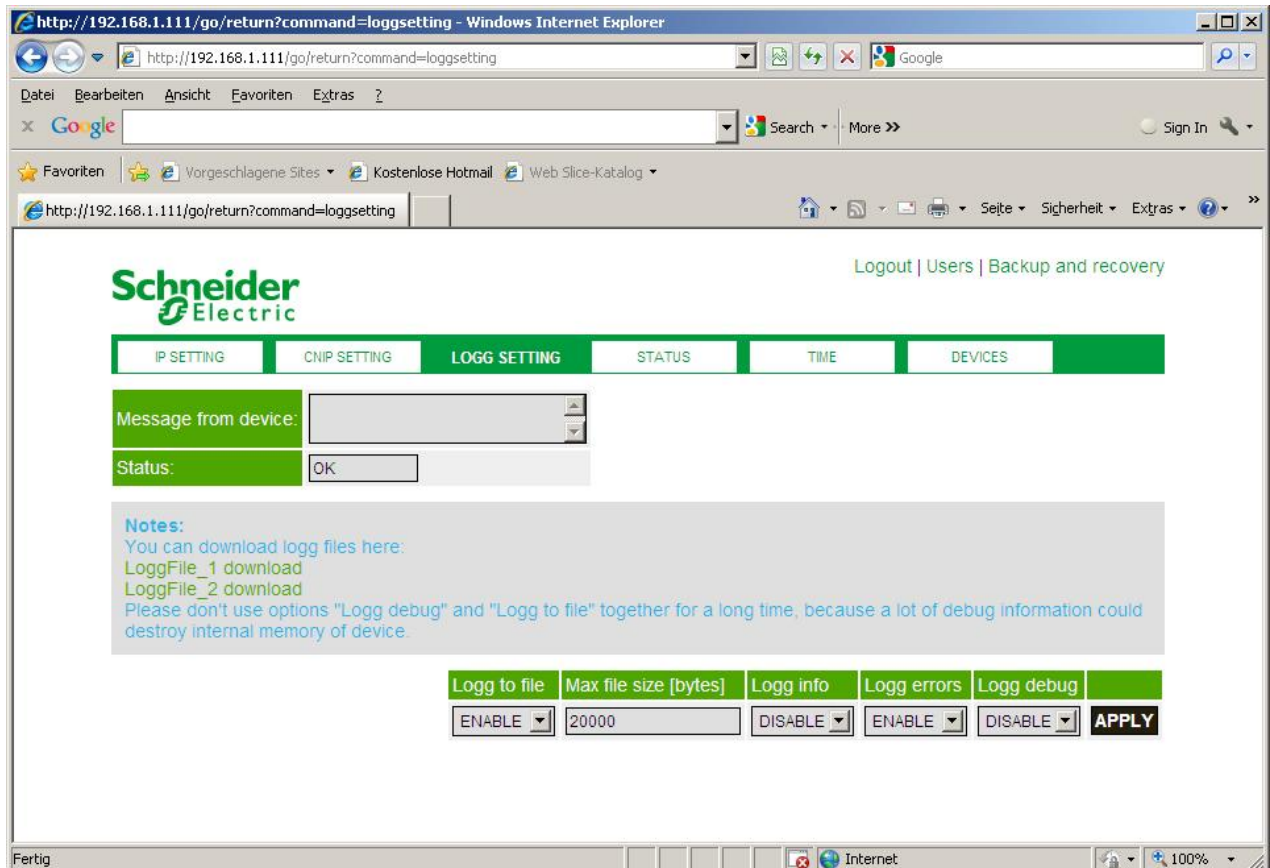


Fig. 7.13.3: Setting the log function. There are two files available; these are described as ring memories. In the standard setting only system information is saved (the recommended setting).

Creating a project

### 7.13.4 TIME

If you have not entered an NTP server from which the system time can be obtained, you can here manually set the system time.

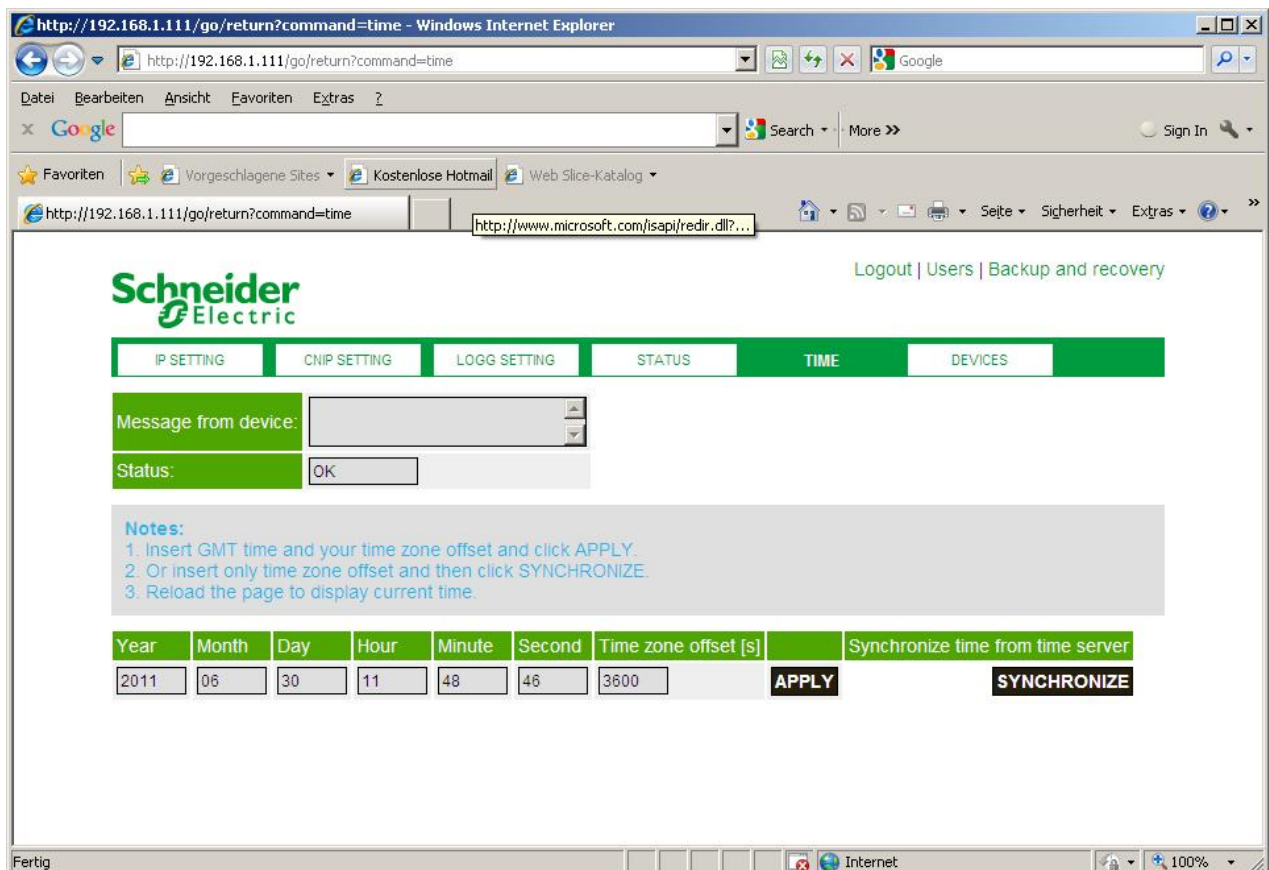


Fig. 7.13.4: Setting the time manually

Creating a project

### 7.13.5 LIST DEVICES

In this list you will find the "virtual" devices that have been configured. Press the "Get Status" button to obtain a view of the status of the "virtual" devices (see Fig. 7.13.5.1).

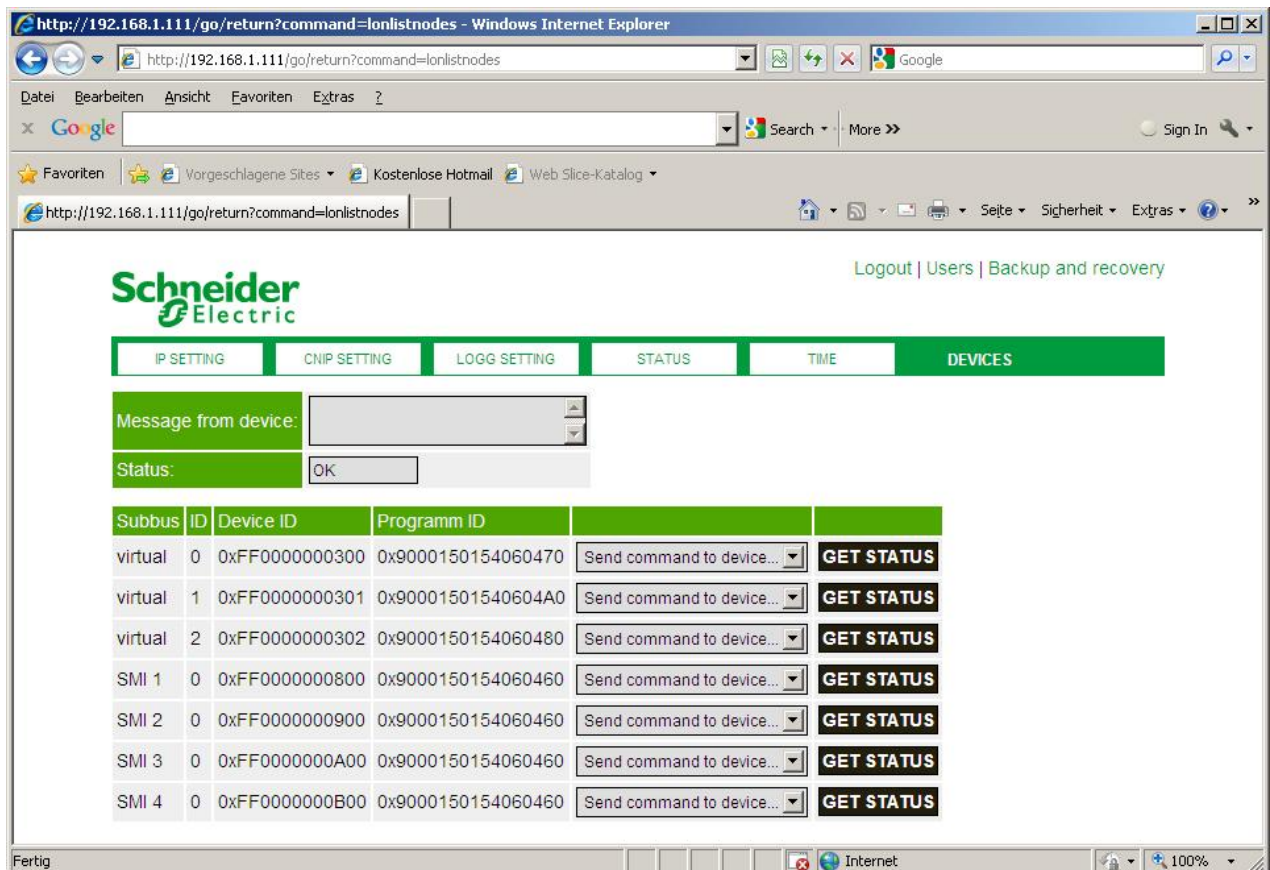
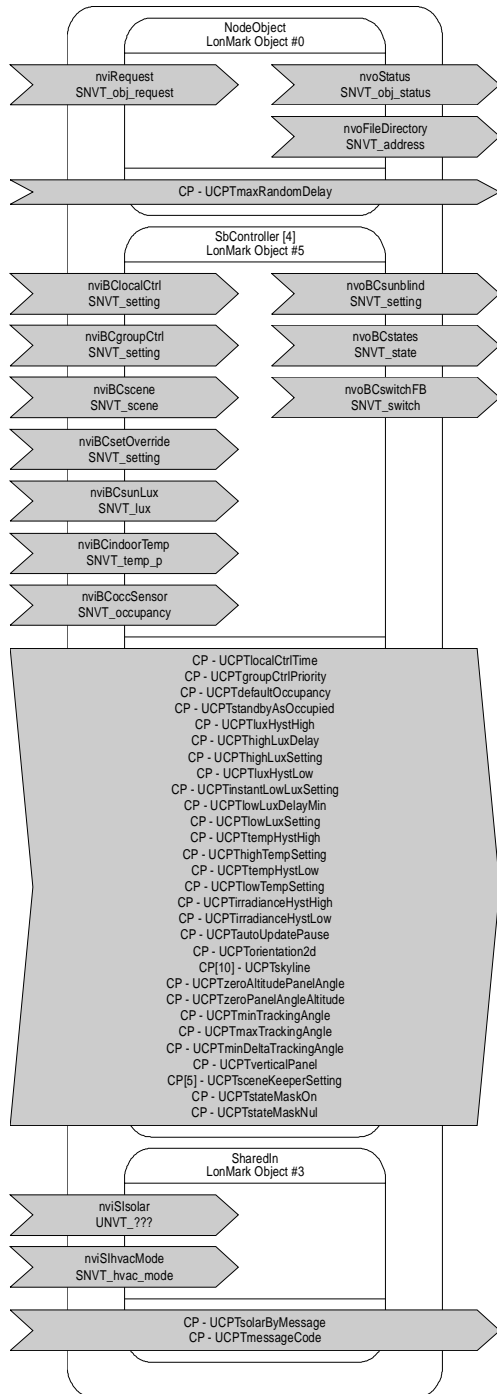


Fig. 7.13.5.1: List of the virtual devices

**8 Appendix A: Description of the function objects**

**8.1 LonMark®-object Blind Controller**



### 8.1.1 Introduction

The ‚Blind Controller’ object allows control of ‘Sunblind Actuator’ objects. The ‚Blind Controller Object’ realises these 2 basic functions:

1. Priority control (override, group control, local and scene control, internal automatic)
2. Automatic control (glare shield control, shading by environment, slat tracing, irradiance control, temperature control)
3. Local control

### 8.1.2 Priority Control

Network input variables `nviBClocalCtrl[i]`, `nviBCscene[i]`, `nviBCgroupCtrl[i]` and `nviBCoverride[i]` manage the transmission of commands via `nvoBCsunblind[i]` in following scheme (1 means highest priority):

Priority	Function	NV
1	Safety function	(not used, only for compatibility with former ‘Blind Actuators’)
2	Global Control	<code>nviBCoverride[i]</code>
3	Group Control	If <code>UCPTgroupCtrlPriority[i] = GCP_PRIORITY_3</code> , <code>nviBCgroupCtrl[i]</code> works with priority 3.
4	Local control and scene control	<code>nviBClocalCtrl[i]</code> <code>nviBCscene[i]</code>
5	Internal automatic control	Internal automatic via internal parameters (Glare shield control, temperature control, irradiance control)  If <code>UCPTgroupCtrlPriority[i] = GCP_PRIORITY_5</code> , external automatic via <code>nviBCgroupCtrl[i]</code> is active while <code>nviBCgroupCtrl[i]</code> not equal to <code>SET_NUL</code> , otherwise work internal automatic

In case of Prio 2 or Prio 3 control all network variable input with lower priority are disabled unless prioritised network variable input has been released via `SET_NUL` command. Only if `SCPTlocalCtrlTime[i]` not equal to 0, local control blocks lower priority for this time. When time expires, local control is released and lower priority re-enabled.

After release a valid command, an input variable with lower priority is then transmitted as valid command onto `nvoBCsunblind[i]` (except of relative commands).

By default, Group Control is of priority 3 (`UCPTgroupCtrlPriority[i] = GCP_PRIORITY_3`). The priority could be decreased to 5 in case a group command should be allowed to be oversteered by a local command via `nviBCsunblind[i]`.

Protection against simultaneous operation in the whole building `UCPTmaxRandomDelay` configures a randomized delay of operation. `UCPTmaxRandomDelay` delays the reactions to all network input variables (except of `nviBClocalCtrl[i]` and `nviBCscene[i]`) updates.

In case of internal automatic control any command via `nviBClocalCtrl[i]` or `nviBCscene[i]` can override the automatic for the time configured within `SCPTlocalCtrlTime[i]`. With `SET_NUL` command the override function is being cancelled.



### Example for UCPTgroupCtrlPriority[i] = CP\_PRIORITY\_5:

SCPTlocalCtrlTime[i] = 5minutes.

nviBClocalCtrl[i] is set to SET\_STATE 100% 0

nvoBCsunblind[i] is set to SET\_STATE 100% 0 too.

During the next 5 minutes the internal automatic is blocked. This means: nviBCgroupCtrl[i] is ignored for 5 minutes, occupancy control and HVAC control too.

### 8.1.3 Internal automatic Control

The internal automatic is set by the following functions:

Name of automatic function	Subfunctions	NV	Notes
Shading by environment	-	nviSIsolar (SharedIn)	Subject to charges
Occupancy control HVAC control	Glare shield	nviSIsunLux (SharedIn) nviBCsunLux[i]	Only if nviBCoccupancy[i] = OC_OCCUPIED, OC_BYPASS or OC_STANDBY by UCPTstandbyAsOccupied sets to TRUE.
	Room temperature	nviBCindoorTemp[i]	Only if nviBCoccupancy[i] = OC_UNOCCUPIED or OC_STANDBY by UCPTstandbyAsOccupied sets to FALSE and nviSIhvacMode[i] is not equal to HVAC_COOL.
	Irradiance control	nviSIsunLux (SharedIn) nviBCsunLux[i]	Only if nviBCoccupancy[i] = OC_UNOCCUPIED or OC_STANDBY by UCPTstandbyAsOccupied sets to FALSE
Slat tracing	-	nviBCgroupCtrl[i] nviBCoverride[i] nviBCscene[i] nviBClocalCtrl[i]	Only if .function=SET_STATE and .rotation>=360 is processed.
Accepting external automatic as internal automatic	-	nviBCgroupCtrl[i]	Only if UCPTgroupCtrlPriority[i] = CP_PRIORITY_5 otherwise external automatic has higher priority.

Automatic control is disabled, when nviBCoccupancy[i] is not equal to OC\_OCCUPIED, OC\_UNOCCUPIED, OC\_STANDBY or OC\_BYPASS.

UCPTautoUpdatePause defines the minimum time for the next automatic command and protects the blinds against overload.

### 8.1.4 Occupancy control

The following table shows the occupancy controlled modes:  
External automatic sends the commands to controller via nviBCgroupCtrl[i].

nviBCgroupCtrl[i].function	not equal to SET_NUL		equal to SET_NUL		
	UCPTgroupCtrlPriority[i]	CP_PRIORITY_3	CP_PRIORITY_5	CP_PRIORITY_3	CP_PRIORITY_5
nviBCoccupancy[i].state					
OC_OCCUPIED OC_BYPASS OC_STANDBY by UCPTstandbyAsOccupied sets to TRUE	External automatic	External automatic	Glare shield control	Glare shield control	
OC_UNOCCUPIED OC_STANDBY by UCPTstandbyAsOccupied sets to FALSE.	External automatic	HVAC with irradiance control and temperature control	HVAC with irradiance control and temperature control	HVAC with irradiance control and temperature control	
The rest	External automatic	External automatic	Internal automatic is off	Internal automatic is off	

After, reset' value of UCPTdefaultOccupancy will be adopted. If nviBCgroupCtrl[i].function = SET\_NUL and nviBCoccupancy[i] = OC\_OCCUPIED, glare shield is active.  
OC\_STANDBY operates as described in UCPTstandbyAsOccupied.  
OC\_BYPASS operates equal to OC\_OCCUPIED.

### 8.1.5 Glare shield control

Is active only if nviBCoccupancy[i].state = OC\_OCCUPIED, OC\_BYPASS or OC\_STANDBY (only if UCPTstandbyAsOccupied is set to TRUE).

Operation mode 'glare shield' requires bases on solar brightness value. This value is provided by nviSolar (shared and if nviBCluxLevel[i] is not bound, overrides it) or by nviBCluxLevel[i] (if bound, has higher priority as nviSolar, otherwise can be overridden by new update of nviSolar).

If the luminance is exceeding the threshold defined within UCPTluxHystHigh the blinds will be driven into the position configured in UCPThighLuxSetting. A delay can be defined within UCPThighLuxDelay. This delay avoids multiple operation which might annoy the occupant.

In case of luminance lower than defined within UCPTluxHystLow, the blind will move into the position defined within UCPTinstantLowLuxSetting (for example slats in horizontal position). After the time configured in UCPTlowLuxDelay has been expired the blind will move into position configured in UCPTlowLuxSetting. If luminance is in an intermediate value, the blinds won't drive by automatic.

### 8.1.6 HVAC support

HVAC support is active only if nviBCoccupancy[i].state = OC\_UNOCCUPIED or OC\_STANDBY (only if UCPTstandbyAsOccupied is set to FALSE).

nviSIhvacMode allows to distinguish between heating and cooling mode of the HVAC system.

### **nviSIhvacMode = HVAC\_COOL = cooling mode**

In cooling mode blind will operate to minimize the irradiance into the building.

If the luminance is exceeding the threshold defined within UCPTirradianceHystHigh the blinds will be driven into the position configured in UCPTHighTempSetting. A delay can be defined within UCPTHighLuxDelay.

In case of luminance lower than defined within UCPTirradianceHystLow the blind will move into the position defined within UCPTlowTempSetting. If luminance is in an intermediate value, blind won't drive by automatic.

nviBCindoorTemp[i] doesn't have any influence on this mode.

### **nviSIhvacMode not equal HVAC\_COOL =heating mode (HVAC\_HEAT, HVAC\_NULL...)**

In heating mode the blinds are controlled according to maximize the room temperature nviBCindoorTemp[i] within the gap defined by UCPTtempHystLow and UCPTtempHystHigh.

These values should analogue to the setpoints defined within the ,space comfort controller':

- UCPTtempHystLow = unoccupied\_heat (e.g.. 21°C)
- UCPTtempHystHigh = occupied\_cool (e.g. 23°C)

Overheating case:

If the space temperature exceeds the setpoint UCPTtempHystHigh and irradiance UCPTirradianceHystHigh, the blinds are moved into position UCPTHighTempSetting.

Undercooling case:

In case of lower space temperature blinds move into position UCPTlowTempSetting.

If no valid value nviBCindoorTemp[i] is available the behaviour is equal to overheating case.

#### **8.1.7 Slat tracing (not supported)**

Slat tracing gets active always if .function = SET\_STATE and .rotation >= 360° is processed.

Target is to trace the slat according to the position of the sun:

- Shading at maximum transparency
- Maximize daylight entry on the ceiling

The position of the slat in relation to the sun position angle may be configured by UCPTzeroAltitudePanelAngle and UCPTzeroPanelAngleAltitude. These parameters are calculated by a special Plug In depending on the following mechanical data:

- Slat width
- Distance between two slats
- Slat height (concave)
- Minimum of overlap
- Minimum of angle of reflection

#### **8.1.8 State monitoring**

For monitoring the state of this object there are two network variable outputs:

- nvoBCstate (SNVT\_state)

Appendix A: Description of the function objects

- nvoBCswitchFB (SNVT\_switch)

**nvoBCstate:**

- .bit0 not used
- .bit1 nviBCoverride activated
- .bit2 Override via nviBCgroupCtrl
- .bit3 Override timer is running triggered by a command on nviBClocalCtrl
- .bit4 Break after automatic command
- .bit5 Random delay for new automatic command is active
- .bit6 Delay after exceeding the irradiation setpoint is active
- .bit7 Delay after undershooting the irradiation setpoint is active
- .bit8 Glare shield is active (UCPTHighLuxSetting)
- .bit9 Slat tracing is active
- .bit10 HVAC support is active (UCPTHighTempSetting)
- .bit11 Sun position above environment envelope
- .bit12 Occupied
- .bit13 Occupancy value invalid
- .bit14 Change to glare shield
- .bit15 Maintenance mode

**nvoBCswitchFB**

NvoBCswitchFB value (state,vlaue)	Condition
0,0	If nvoBCstate[i] bit by bit AND conjunct with UCPTstateMaskOn[i] gives result zero and bit by bit AND conjunct with this UCPTstateMaskNul[i] gives the result equal zero.
1,100	If nvoBCstate[i] bit by bit AND conjunct with this UCPTstateMaskOn[i] gives result not equal zero.
-1,0	If nvoBCstate[i] bit by bit AND conjunct with UCPTstateMaskOn[i] gives result zero and bit by bit AND conjunct with this UCPTstateMaskNul[i] gives the result not equal zero.

**8.1.9 Behaviour after reset**

UCPTmaxRandomDelay and UCPTdefaultOccupancy are used to configure the behaviour of ,Blind controller' after reset.

### 8.1.10 Node Object (LonMark Object #0)

Function	Network Variable	Dir	Type
Object request	nviRequest	IN	SNVT_obj_request
Object status	nvoStatus	OUT	SNVT_obj_status
Neuron address	nvoFileDirectory	OUT	SNVT_address

Function	Configuration Parameter	Type
Maximum random delay	UCPTmaxRandomDelay	SNVT_time_sec

#### nviRequest – Object request

---

<b>Type</b>	SNVT_obj_request
<b>Range</b>	.object_id: <0.0; 65535.0> .object_request: RQ_NUL, RQ_NORMAL, RQ_DISABLED, RQ_UPDATE_STATUS, RQ_SELF_TEST, RQ_UPDATE_ALARM, RQ_REPORT_MASK, RQ_OVERRIDE, RQ_ENABLE, RQ_RMV_OVERRIDE, RQ_CLEAR_STATUS, RQ_CLEAR_ALARM, RQ_ALARM_NOTIFY_ENABLED, RQ_ALARM_NOTIFY_DISABLED, RQ_MANUAL_CTRL, RQ_REMOTE_CTRL, RQ_PROGRAM, RQ_CLEAR_RESET, RQ_RESET
<b>Default</b>	0 RQ_NORMAL [ID, request]
<b>Description</b>	Not supported.

Appendix A: Description of the function objects

**nvoStatus – Object status**

<b>Type</b>	SNVT_obj_status
<b>Range</b>	.object_id: <0.0; 65535.0> .invalid_id: <0.0; 1.0> .invalid_request: <0.0; 1.0> .disabled: <0.0; 1.0> .out_of_limits: <0.0; 1.0> .open_circuit: <0.0; 1.0> .out_of_service: <0.0; 1.0> .mechanical_fault: <0.0; 1.0> .feedback_failure: <0.0; 1.0> .over_range: <0.0; 1.0> .under_range: <0.0; 1.0> .electrical_fault: <0.0; 1.0> .unable_to_measure: <0.0; 1.0> .comm_failure: <0.0; 1.0> .fail_self_test: <0.0; 1.0> .self_test_in_progress: <0.0; 1.0> .locked_out: <0.0; 1.0> .manual_control: <0.0; 1.0> .in_alarm: <0.0; 1.0> .in_override: <0.0; 1.0> .report_mask: <0.0; 1.0> .programming_mode: <0.0; 1.0> .programming_fail: <0.0; 1.0> .alarm_notify_disabled: <0.0; 1.0> .reset_complete: <0.0; 1.0> .reserved2: <0.0; 0.0>
<b>Default</b>	0 [ID, status flags]
<b>Description</b>	Not supported yet.

**nvoFileDirectory – Neuron address**

<b>Type</b>	SNVT_address
<b>Range</b>	<16384.0; 64767.0>
<b>Default</b>	0 [16-bit address value]
<b>Description</b>	For reading SCPT data from object.

**UCPTmaxRandomDelay – Maximum random delay**

<b>Type</b>	SNVT_time_sec
<b>Range</b>	<0.0; 6553.5>
<b>Default</b>	0.0
<b>Description</b>	Maximum time between receiving an processing any input network variable (expect nviBClocalCtrl and nviBCscene[i], there are processed immediately)

Appendix A: Description of the function objects

8.1.11 SbController (LonMark Object #5)

Function	Network Variable	Dir	Type
Setting control	nviBClocalCtrl	IN	SNVT_setting
Setting control	nviBCgroupCtrl	IN	SNVT_setting
Scene control	nviBCscene	IN	SNVT_scene
Setting control	nviBCsetOverride	IN	SNVT_setting
Illumination	nviBCsunLux	IN	SNVT_lux
Temperature	nviBCindoorTemp	IN	SNVT_temp_p
Occupancy	nviBCoccSensor	IN	SNVT_occupancy
Setting control	nvoBCsunblind	OUT	SNVT_setting
State vector	nvoBCstates	OUT	SNVT_state
Switch	nvoBCswitchFB	OUT	SNVT_switch

Function	Configuration Parameter	Type
Local control time	UCPTlocalCtrlTime	SNVT_time_min
Group control priority	UCPTgroupCtrlPriority	UNVT_gpc_prio
Default occupancy state	UCPTdefaultOccupancy	SNVT_occupancy
Standby as occupied	UCPTstandbyAsOccupied	UNVT_boolean
Lux hysteresis high	UCPTluxHystHigh	SNVT_lux
High lux delay	UCPThighLuxDelay	SNVT_time_sec
High lux setting	UCPThighLuxSetting	SNVT_setting
Lux hysteresis low	UCPTluxHystLow	SNVT_lux
Instant low lux setting	UCPTinstantLowLuxSetting	SNVT_setting
Low lux delay	UCPTlowLuxDelayMin	SNVT_time_min
Low lux setting	UCPTlowLuxSetting	SNVT_setting
Temperature hysteresis high	UCPTtempHystHigh	SNVT_temp_p
High temperature setting	UCPThighTempSetting	SNVT_setting
Temperature hysteresis low	UCPTtempHystLow	SNVT_temp_p
Low temperature setting	UCPTlowTempSetting	SNVT_setting
Irradiance hysteresis high	UCPTirradianceHystHigh	SNVT_lux
Irradiance hysteresis low	UCPTirradianceHystLow	SNVT_lux
Auto update pause	UCPTautoUpdatePause	SNVT_time_min
Orientation 2D	UCPTorientation2d	UNVT_dir_2d
Skyline	UCPTskyline[10]	UNVT_dir_2d
Zero altitude panel angle	UCPTzeroAltitudePanelAngle	SNVT_angle_deg
Zero panel angle altitude	UCPTzeroPanelAngleAltitude	SNVT_angle_deg
Minimum tracking angle	UCPTminTrackingAngle	SNVT_angle_deg
Maximum tracking angle	UCPTmaxTrackingAngle	SNVT_angle_deg
Tracking angle send on delta	UCPTminDeltaTrackingAngle	SNVT_angle_deg
Vertical panel	UCPTverticalPanel	UNVT_boolean
Scene keeper setting	UCPTsceneKeeperSetting[5]	UNVT_setting
State mask on	UCPTstateMaskOn	SNVT_state
State mask nul	UCPTstateMaskNul	SNVT_state

**nviBClocalCtrl – Setting control**

<b>Type</b>	SNVT_setting
<b>Range</b>	.function: SET_NULL, SET_OFF, SET_ON, SET_DOWN, SET_UP, SET_STOP, SET_STATE .setting: <0.0; 100.0> .rotation: <-359.98; 360.0>
<b>Default</b>	SET_NULL 0.0 0.0 [function, setting, rotation]
<b>Description</b>	Control input, priority 4

Appendix A: Description of the function objects

---

**nviBCgroupCtrl – Setting control**

---

<b>Type</b>	SNVT_setting
<b>Range</b>	.function: SET_NUL, SET_OFF, SET_ON, SET_DOWN, SET_UP, SET_STOP, SET_STATE .setting: <0.0; 100.0> .rotation: <-359.98; 360.0>
<b>Default</b>	SET_NUL 0.0 0.0 [function, setting, rotation]
<b>Description</b>	External automatic input

**nviBCscene – Scene control**

---

<b>Type</b>	SNVT_scene
<b>Range</b>	.function: SC_NUL, SC_RECALL .scene_number: <1; 6>
<b>Default</b>	SC_NUL 255 [function, scene number]
<b>Description</b>	Scene trigger input

**nviBCsetOverride – Setting control**

---

<b>Type</b>	SNVT_setting
<b>Range</b>	.function: SET_NUL, SET_OFF, SET_ON, SET_DOWN, SET_UP, SET_STOP, SET_STATE .setting: <0.0; 100.0> .rotation: <-359.98; 360.0>
<b>Default</b>	SET_NUL 0.0 0.0 [function, setting, rotation]
<b>Description</b>	Control input, priority 2

**nviBCsunLux – Illumination**

---

<b>Type</b>	SNVT_lux
<b>Range</b>	<0.0; 65535.0>
<b>Default</b>	0 [lux]
<b>Description</b>	Outdoor brightness input

**nviBCindoorTemp – Temperature**

---

<b>Type</b>	SNVT_temp_p
<b>Range</b>	<-273.17; 327.67>
<b>Default</b>	327.67 [degrees Celsius]
<b>Description</b>	Indoor temperature input



Appendix A: Description of the function objects

---

**nviBCoccSensor – Occupancy**

---

<b>Type</b>	SNVT_occupancy
<b>Range</b>	OC_NUL, OC_OCCUPIED, OC_UNOCCUPIED, OC_BYPASS, OC_STANDBY
<b>Default</b>	OC_NUL [occupancy code names]
<b>Description</b>	Occupancy input

**nvoBCsunblind – Setting control**

---

<b>Type</b>	SNVT_setting
<b>Range</b>	.function: SET_NUL, SET_OFF, SET_ON, SET_DOWN, SET_UP, SET_STOP, SET_STATE .setting: <0.0; 100.0> .rotation: <-359.98; 360.0>
<b>Default</b>	SET_NUL 0.0 0.0 [function, setting, rotation]
<b>Description</b>	Controller output

**nvoBCstates – State vector**

---

<b>Type</b>	SNVT_state
<b>Range</b>	.bit0: <0.0; 1.0> .bit1: <0.0; 1.0> .bit2: <0.0; 1.0> .bit3: <0.0; 1.0> .bit4: <0.0; 1.0> .bit5: <0.0; 1.0> .bit6: <0.0; 1.0> .bit7: <0.0; 1.0> .bit8: <0.0; 1.0> .bit9: <0.0; 1.0> .bit10: <0.0; 1.0> .bit11: <0.0; 1.0> .bit12: <0.0; 1.0> .bit13: <0.0; 1.0> .bit14: <0.0; 1.0> .bit15: <0.0; 1.0>
<b>Default</b>	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 [16 individual bit values]
<b>Description</b>	Controller states output. See description above.

Appendix A: Description of the function objects

---

**nvoBCswitchFB – Switch**

---

<b>Type</b>	SNVT_switch
<b>Range</b>	.value: <0.0; 100.0> .state: <-1.0; 1.0>
<b>Default</b>	0.0 -1 [value, state]
<b>Description</b>	Programmable feedback

**UCPTlocalCtrlTime – Local control time**

---

<b>Type</b>	SNVT_time_min
<b>Range</b>	<0.0; 65535.0>
<b>Default</b>	0
<b>Description</b>	The time period, a local control request is valid and the controller is bypassed. If sets to 0, local control is valid while nviBClocalCtrl not equal to SET_NUL.

**UCPTgroupCtrlPriority – Group control priority**

---

<b>Type</b>	UNVT_gpc_prio
<b>Range</b>	GCP_PRIORITY_3, GCP_PRIORITY_5
<b>Default</b>	GCP_PRIORITY_3
<b>Description</b>	GCP_PRIORITY_3: nviBCgroupCtrl may overrides local settings GCP_PRIORITY_5: local settings may override nviGroupCtrl for the time configured as UCPTlocalCtrlTime

**UCPTdefaultOccupancy – Default occupancy state**

---

<b>Type</b>	SNVT_occupancy
<b>Range</b>	OC_NUL, OC_OCCUPIED, OC_UNOCCUPIED, OC_BYPASS, OC_STANDBY
<b>Default</b>	OC_NUL [occupancy code names]
<b>Description</b>	Occupancy state, adopt at power-on or reset.

**UCPTstandbyAsOccupied – Standby as occupied**

---

<b>Type</b>	UNVT_boolean
<b>Range</b>	FALSE, TRUE
<b>Default</b>	FALSE
<b>Description</b>	The OC_STANDBY command is interpreted as OC_OCCUPIED.

Appendix A: Description of the function objects

---

**UCPTluxHystHigh – Lux hysteresis high**

---

<b>Type</b>	SNVT_lux
<b>Range</b>	<0.0; 65535.0>
<b>Default</b>	35000 [lux]
<b>Description</b>	The upper lux level for the hysteresis.

**UCPTHighLuxDelay – High lux delay**

---

<b>Type</b>	SNVT_time_sec
<b>Range</b>	<0.0; 6553.5>
<b>Default</b>	20.0 [seconds]
<b>Description</b>	The time period, the high lux level must exist.

**UCPTHighLuxSetting – High lux setting**

---

<b>Type</b>	SNVT_setting
<b>Range</b>	.function: SET_NUL, SET_OFF, SET_ON, SET_DOWN, SET_UP, SET_STOP, SET_STATE .setting: <0.0; 100.0> .rotation: <-359.98; 360.0>
<b>Default</b>	SET_STATE 100.0 -45.0 [function, setting, rotation]
<b>Description</b>	This setting value is transmitted to output when the current lux value exceeds lux hysteresis high for the UCPTHighLuxDelay time.

**UCPTluxHystLow – Lux hysteresis low**

---

<b>Type</b>	SNVT_lux
<b>Range</b>	<0.0; 65535.0>
<b>Default</b>	15000 [lux]
<b>Description</b>	The lower lux level for the hysteresis.

**UCPTInstantLowLuxSetting – Instant low lux setting**

---

<b>Type</b>	SNVT_setting
<b>Range</b>	.function: SET_NUL, SET_OFF, SET_ON, SET_DOWN, SET_UP, SET_STOP, SET_STATE .setting: <0.0; 100.0> .rotation: <-359.98; 360.0>
<b>Default</b>	SET_STATE 127.5 0.0 [function, setting, rotation]
<b>Description</b>	This setting value is transmitted to output without delay when the current lux value falls below lux hysteresis low.

Appendix A: Description of the function objects

---

**UCPTlowLuxDelayMin – Low lux delay**

---

<b>Type</b>	SNVT_time_min
<b>Range</b>	<0.0; 65535.0>
<b>Default</b>	60
<b>Description</b>	The time period, the low lux level must exist.

**UCPTlowLuxSetting – Low lux setting**

---

<b>Type</b>	SNVT_setting
<b>Range</b>	.function: SET_NUL, SET_OFF, SET_ON, SET_DOWN, SET_UP, SET_STOP, SET_STATE .setting: <0.0; 100.0> .rotation: <-359.98; 360.0>
<b>Default</b>	SET_OFF 0.0 0.0 [function, setting, rotation]
<b>Description</b>	This setting value is transmitted to output when the current lux value falls below lux hysteresis low for the UCPTlowLuxDelayMin.

**UCPTtempHystHigh – Temperature hysteresis high**

---

<b>Type</b>	SNVT_temp_p
<b>Range</b>	<-273.17; 327.67>
<b>Default</b>	23.0 [degrees Celsius]
<b>Description</b>	The high temperature level for hysteresis.

**UCPThighTempSetting – High temperature setting**

---

<b>Type</b>	SNVT_setting
<b>Range</b>	.function: SET_NUL, SET_OFF, SET_ON, SET_DOWN, SET_UP, SET_STOP, SET_STATE .setting: <0.0; 100.0> .rotation: <-359.98; 360.0>
<b>Default</b>	SET_STATE 100.0 -45.0 [function, setting, rotation]
<b>Description</b>	This setting value is transmitted to output when the current temperature value exceeds temperature hysteresis high and actual brightness exceeds UCPTirradianceHystHigh.

**UCPTtempHystLow – Temperature hysteresis low**

---

<b>Type</b>	SNVT_temp_p
<b>Range</b>	<-273.17; 327.67>
<b>Default</b>	21.0 [degrees Celsius]
<b>Description</b>	The low temperature level for hysteresis.

Appendix A: Description of the function objects

---

**UCPTlowTempSetting – Low temperature setting**

---

<b>Type</b>	SNVT_setting
<b>Range</b>	.function: SET_NUL, SET_OFF, SET_ON, SET_DOWN, SET_UP, SET_STOP, SET_STATE .setting: <0.0; 100.0> .rotation: <-359.98; 360.0>
<b>Default</b>	SET_OFF 0.0 0.0 [function, setting, rotation]
<b>Description</b>	This setting value is transmitted to output when the current temperature value falls below temperature hysteresis low.

**UCPTirradianceHystHigh – Irradiance hysteresis high**

---

<b>Type</b>	SNVT_lux
<b>Range</b>	<0.0; 65535.0>
<b>Default</b>	7500 [lux]
<b>Description</b>	The upper energy flux level for the hysteresis in lux, calculated from W/m <sup>2</sup> .

**UCPTirradianceHystLow – Irradiance hysteresis low**

---

<b>Type</b>	SNVT_lux
<b>Range</b>	<0.0; 65535.0>
<b>Default</b>	5000 [lux]
<b>Description</b>	The lower energy flux level for the hysteresis in lux, calculated from W/m <sup>2</sup> .

**UCPTautoUpdatePause – Auto update pause**

---

<b>Type</b>	SNVT_time_min
<b>Range</b>	<0.0; 65535.0>
<b>Default</b>	0 [minutes]
<b>Description</b>	Pause for automatic commands after any command provided by internal automatic.

**UCPTorientation2d – Orientation 2D**

---

<b>Type</b>	UNVT_dir_2d
<b>Range</b>	.altitude: <-90.0; 90.0> .azimuth: <-359.98; 360.0>
<b>Default</b>	0.0 0.0 [altitude, azimuth]
<b>Description</b>	Orientation 2D

#### **UCPTskyline[10] – Skyline template (not supported)**

---

<b>Type</b>	UNVT_dir_2d
<b>Range</b>	.altitude: <-90.0; 90.0> .azimuth: <-359.98; 360.0>
<b>Default</b>	0.0 0.0 [altitude, azimuth]
<b>Description</b>	Determines the skyline by points of the neighboring buildings.

#### **UCPTzeroAltitudePanelAngle – Zero altitude panel angle (not supported)**

---

<b>Type</b>	SNVT_angle_deg
<b>Range</b>	<-359.98; 360.0>
<b>Default</b>	-45.0 [degrees]
<b>Description</b>	If the sun is a zero altitude (at sunrise and sunset), this panel angle is adopted.

#### **UCPTzeroPanelAngleAltitude – Zero panel angle altitude (not supported)**

---

<b>Type</b>	SNVT_angle_deg
<b>Range</b>	<-359.98; 360.0>
<b>Default</b>	45.0 [degrees]
<b>Description</b>	If the sun is at this altitude, the panel angle 0° is adopted.

#### **UCPTminTrackingAngle – Minimum tracking angle (not supported)**

---

<b>Type</b>	SNVT_angle_deg
<b>Range</b>	<-359.98; 360.0>
<b>Default</b>	-45.0 [degrees]
<b>Description</b>	Minimum panel angle in tracking mode.

#### **UCPTmaxTrackingAngle – Maximum tracking angle (not supported)**

---

<b>Type</b>	SNVT_angle_deg
<b>Range</b>	<-359.98; 360.0>
<b>Default</b>	0.0 [degrees]
<b>Description</b>	Maximum panel angle in tracking mode.

#### **UCPTminDeltaTrackingAngle – Tracking angle send on delta (not supported)**

---

<b>Type</b>	SNVT_angle_deg
<b>Range</b>	<-359.98; 360.0>
<b>Default</b>	5.0 [degrees]
<b>Description</b>	The minimum tracking angle change required to update the output network variable.

### UCPTverticalPanel – Vertical panel (not supported)

---

<b>Type</b>	UNVT_boolean
<b>Range</b>	FALSE, TRUE
<b>Default</b>	FALSE
<b>Description</b>	Tracking angle is calculated for vertical panels

### UCPTsceneKeeperSetting[5] – Scene keeper setting

---

<b>Type</b>	UNVT_setting
<b>Range</b>	.function: SET_NO_MESSAGE, SET_NUL, SET_OFF, SET_ON, SET_DOWN, SET_UP, SET_STOP, SET_STATE .setting: <0.0; 100.0> .rotation: <-359.98; 360.0>
<b>Default</b>	SET_NO_MESSAGE 0.0 0.0 [function, setting, rotation]
<b>Description</b>	Values, that are transmitted to output when a scene is recalled. When SET_NO_MESSAGE is set, output will be not changed.

### UCPTstateMaskOn – State mask on

<b>Type</b>	SNVT_state
<b>Range</b>	.bit0: <0.0; 1.0> .bit1: <0.0; 1.0> .bit2: <0.0; 1.0> .bit3: <0.0; 1.0> .bit4: <0.0; 1.0> .bit5: <0.0; 1.0> .bit6: <0.0; 1.0> .bit7: <0.0; 1.0> .bit8: <0.0; 1.0> .bit9: <0.0; 1.0> .bit10: <0.0; 1.0> .bit11: <0.0; 1.0> .bit12: <0.0; 1.0> .bit13: <0.0; 1.0> .bit14: <0.0; 1.0> .bit15: <0.0; 1.0>
<b>Default</b>	0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 [16 individual bit values]
<b>Description</b>	This mask is bit by bit AND-conjunct with the object status (nvoBCstates[i]). If nvoBCstate[i] bit by bit AND conjunct with this UCPTstateMaskOn[i] gives result not equal zero, field .state in nvoBCswitchFb[i] is set to 1 and field .value to 100%, otherwise see UCPTstateMaskNul[i].

Appendix A: Description of the function objects

---

**UCPTstateMaskNul – State mask nul**

**Type** SNVT\_state

**Range** .bit0: <0.0; 1.0>  
.bit1: <0.0; 1.0>  
.bit2: <0.0; 1.0>  
.bit3: <0.0; 1.0>  
.bit4: <0.0; 1.0>  
.bit5: <0.0; 1.0>  
.bit6: <0.0; 1.0>  
.bit7: <0.0; 1.0>  
.bit8: <0.0; 1.0>  
.bit9: <0.0; 1.0>  
.bit10: <0.0; 1.0>  
.bit11: <0.0; 1.0>  
.bit12: <0.0; 1.0>  
.bit13: <0.0; 1.0>  
.bit14: <0.0; 1.0>  
.bit15: <0.0; 1.0>

**Default** 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 [16 individual bit values]

**Description** This mask is bit by bit AND-conjunct with the object status (nvoBCstates[i]).  
If nvoBCstate[i] bit by bit AND conjunct with UCPTstateMaskOn[i] gives result zero and bit by bit AND conjunct with this UCPTstateMaskNul[i] gives the result not equal zero, field .state in nvoBCswitchFb[i] is set to -1, otherwise is field .state set to 0 and field .value to 0%.

**8.1.12 SharedIn (LonMark Object #3)**

Function	Network Variable	Dir	Type
	nviSIsolar	IN	reserved
HVAC mode	nviSIhvacMode	IN	SNVT_hvac_mode

Function	Configuration Parameter	Type
Solar without binding	UCPTsolarByMessage	UNVT_enabled
Identification code for explicit messages	UCPTmessageCode	UNVT_message_code



Appendix A: Description of the function objects

---

**nviSIsolar – Sun data**

---

<b>Type</b>	UNVT_solar
<b>Range</b>	.brightness: 0;1;...;63000 = 0;2;...;126000 lux .elevation: 0 .. 89° .azimuth: 0 .. 359°
<b>Default</b>	.brightness: 0 .elevation: 0 .azimuth: 0
<b>Description</b>	Sun data.

**nviSIhvacMode – HVAC mode**

---

<b>Type</b>	SNVT_hvac_mode
<b>Range</b>	HVAC_NUL, HVAC_AUTO, HVAC_HEAT, HVAC_MRNG_WRMUP, HVAC_COOL, HVAC_NIGHT_PURGE, HVAC_PRE_COOL, HVAC_OFF, HVAC_TEST, HVAC_EMERG_HEAT, HVAC_FAN_ONLY, HVAC_FREE_COOL, HVAC_ICE, HVAC_MAX_HEAT, HVAC_ECONOMY, HVAC_DEHUMID, HVAC_CALIBRATE, HVAC_EMERG_COOL, HVAC_EMERG_STEAM
<b>Default</b>	HVAC_HEAT [HVAC mode names]
<b>Description</b>	HVAC mode.

**UCPTsolarByMessage – Solar without binding – not supported in this version**

---

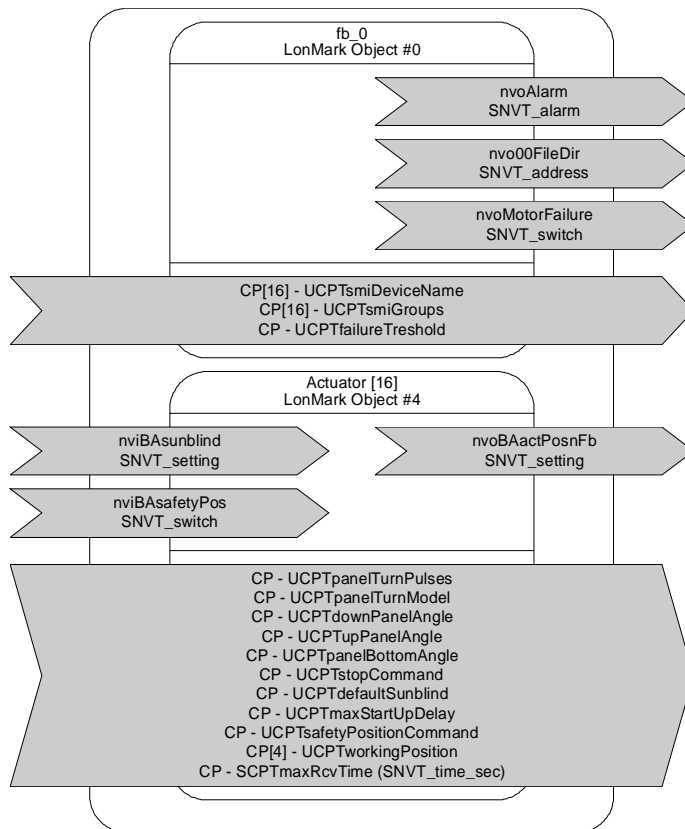
<b>Type</b>	UNVT_enabled
<b>Range</b>	DISABLED, ENABLED
<b>Default</b>	DISABLED
<b>Description</b>	Solar is send/received as broadcast message without network variable connection.

**UCPTmessageCode – Identification code for explicit messages – not supported in this version**

---

<b>Type</b>	UNVT_message_code
<b>Range</b>	<0.0; 62.0>
<b>Default</b>	45
<b>Description</b>	Identification code for explicit messages without network variable connection. This code has to be the same for transmitter and receiver.

## 8.2 LonMark®-object SMI Actuator



### 8.2.1 fb\_0 Object

Table: functions, parameters and variables of the Node-Object

Function	Network variables	Type
Alarm	nvoAlarm	SNVT_alarm
Motor Failure	nvoMotorFailure	SNVT_switch
Address for DMA	nvo00FileDir	SNVT_address
Function	Configuration parameter	Type
Device name	UCPTsmiDeviceName	UNVT_str_asc15
Allocation of devices to the group	UCPTsmiGroups	SNVT_state
Threshold for alarm	UCPTfailureThreshold	SNVT_lev_cont

## SMI motor faults and fault messages

If a drive fault is detected by a device on the SMI cable, then this is notified via the network variable `nvoLampFailure = {100,0 1}` and the "L-Fail" LED.

The network variable `nvoAlarm` can also be written at the same time to provide detailed information on the nature of the fault. This contains the following data:

`nvoAlarm.location` : Mounting location of the SMI Controller as a 6 byte location ID.  
`nvoAlarm.object_id` : object\_id of the SMI Actuator object having a fault.  
`nvoAlarm.alarm_type` : AL\_NO\_CONDITION = Alarm removed;  
AL\_WARNING = Fault proportion below the critical threshold;  
AL\_ERROR = Fault proportion above the critical threshold;  
`nvoAlarm.value[0]` : Group address of the newly affected SMI motors.  
`nvoAlarm.value[1]` : Index of the newly affected SMI motor (255 = not yet determined)  
`nvoAlarm.value[2]` : Device status; 1 = Status not OK; 2 = Motor failure;  
245 = SMI cable occupied for too long;  
254 = SMI device does not answer  
`nvoAlarm.value[3]` : Proportion of faults in the affected group  
in 0...200 -> 0...100% (0 when not yet determined)  
`nvoAlarm.alarm_limit[0]` : Alarm counter, counts the emitted messages. Begins at zero  
after  
be  
255 messaged. If `nvoAlarm` is cyclically polled then this value can  
be  
used to define whether alarm messages are recorded.

The internal realtime clock served by an SNTP can be used to provide the messages at the **nvoAlarm** output with a timestamp of the actual time. The internal clock has an accuracy of  $\pm 1\%$ .

When all motors in a group function once again, the alarm is removed using `nvoAlarm.alarm_type = AL_NO_CONDITION`.

The alarm types can be influenced using the parameters **UCPTfailureTreshold**.

All devices together, a group, and an individual device are tested cyclical. This makes the collective fault message via **nvoMotorFailure**. A group fault message occurs with **nvoAlarm.value[2] = 255**. Up to 3 minutes can pass until the index of the affected device is displayed. The group fault messages can be suppressed by setting **UCPTmode.bit0 = 1**.

## Network variable details:

### nvoAlarm – Object status output

---

<b>Type</b>	SNVT_alarm
<b>Range</b>	.location[6]: 0x00 ... 0xff (Location string) .object_id: 1 ... 16 .alarm_type: AL_NO_CONDITION, AL_WARNING; AL_ERROR; AL_FATAL_ERROR .priority_level: PR_LEVEL_0 .index_to_SNVT: 0 .value[0]: 0 ... 15 (DALI group address) .value[1]: 0 ... 64; 255 (DALI shortaddress) .value[2]: 0 ... 255 (device status) .value[3]: 0 ... 200 (0 ... 100% proportion of affected devices) .year: -1 ... 3.000 .month: 0 ... 12 .day: 0 ... 31 .hour: 0 ... 23 .minute: 0 ... 59 .second: 0 ... 59 .milisecond: 0 ... 999 .alarm_limit[0]: 0 ... 255 (alarm number, distinguishing poll characteristic) .alarm_limit[1]: 0 .alarm_limit[2]: 0 .alarm_limit[3]: 0
<b>Default</b>	All elements = 0
<b>Description</b>	This output can be logged to provide exact details of lamp faults. The interpretation of the values is described above.

### nvo00FileDir – Address of the configuration parameter

---

<b>Type</b>	SNVT_address
<b>Range</b>	0x0000 ... 0xffff
<b>Default</b>	0x0000
<b>Description</b>	Is required exclusively for internal functionality.

### nvoMotorFailure – Motor failure collective message

---

<b>Type</b>	SNVT_switch
<b>Range</b>	.value: 0; 100 % .state: 0; 1
<b>Default</b>	.value = 0 .state = 0
<b>Description</b>	This output emits {100,1} when at least one motor is recognised as faulty. Details of the fault can be taken from nvoAlarm. Fault-free SMI hardware is indicated by {0,0}.

Appendix A: Description of the function objects

---

**Configuration parameters**

**UCPTsmiDeviceName – SMI device names**

---

<b>Type</b>	UNVT_str_asc_15
<b>Range</b>	Ascii
<b>Default</b>	not in use
<b>Description</b>	Individual name for each SMI device. (do not modify!)

**UCPTsmiGroups – SMI groups**

---

<b>Type</b>	SNVT_state
<b>Range</b>	0, 1
<b>Default</b>	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
<b>Description</b>	SMI group information for internal management. (do not directly modify!)

**UCPTfailureThreshold - Failure limit**

---

<b>Type</b>	SNVT_lev_cont
<b>Range</b>	0.0 ... 100.0 % [0.5 %]
<b>Default</b>	0.0 %
<b>Description</b>	If the number of faulty lamps in a group is more than the percent value specified here, then a fault message instead of a warning is output.

**8.2.2 Object SMIActuator**

Table: functions, parameters and variables of the SMIActuator-Object

Function	Network variable	Type
Sunblind setting input	nviBASunblind	SNVT_setting
Actual position feedback	nvoBAactPosnFb	SNVT_setting
Safety position trigger, priority 1	nviBASafetyPos	SNVT_switch
Function	Configuration parameter	Type
Panel turn pulses	UCPTpanelTurnPulses	SNVT_count
Panel time to angle	UCPTpanelTimeToAngle	UNVT_lin_sin
Drive-up panel angle	UCPTdownPanelAngle	SNVT_angle_deg
Drive-down panel angle	UCPTupPanelAngle	SNVT_angle_deg
Panel angle at bottom	UCPTpanelBottomAngle	SNVT_angle_deg
Stop command	UCPTstopCommand	UNVT_stop_cmd
Default sunblind command	UCPTdefaultSunblind	UNVT_setting
Maximum start-up delay	UCPTmaxStartUpDelay	SNVT_time_sec
Safety position command	UCPTsafetyPositionCommand	SNVT_setting
Working position	UCPTworkingPosition	SNVT_setting
Maximum receive time	SCPTmaxRcvTime	SNVT_time_sec

Appendix A: Description of the function objects

---

**Network variable details:**

**nviBAsunblind - Sunblind setting input**

---

<b>Type</b>	SNVT_setting
<b>Range</b>	.function: SET_OFF, SET_DOWN, SET_UP, SET_STOP, SET_STATE .setting: 0 .. 100 % .rotation: -359.98° .. 360.00°
<b>Default</b>	.function = SET_NUL .setting = 0 .rotation = 0
<b>Description</b>	Sunblind setting input

**nviBAsunblind - Sunblind setting input**

---

<b>Type</b>	SNVT_setting
<b>Range</b>	.function: SET_OFF, SET_DOWN, SET_UP, SET_STOP, SET_STATE .setting: 0 .. 100 % .rotation: -359.98° .. 360.00°
<b>Default</b>	.function = SET_NUL .setting = 0 .rotation = 0
<b>Description</b>	Actual position feedback

**nviBAafetyPos - Safety position trigger, priority 1**

---

<b>Type</b>	SNVT_switch
<b>Range</b>	.value: 0 .. 100 % .state: 0, 1
<b>Default</b>	.value = 0 .state = 0
<b>Description</b>	Safety position trigger, priority 1

**Configuration parameters**

**UCPTpanelTurnPulses - Panel turn pulses**

---

<b>Type</b>	SNVT_count
<b>Range</b>	0 .... 65535
<b>Default</b>	135
<b>Description</b>	Amount of pulses for a total turn of the panels (slats).

## Appendix A: Description of the function objects

---

### UCPTpanelTimeToAngle - Panel time to angle

---

<b>Type</b>	UNVT_lin_sin
<b>Range</b>	LS_LINEAR, LS_SINUS, LS_VESTAMATIC
<b>Default</b>	LS_SINUS
<b>Description</b>	Determined the dependency between the panel turn time and the panel angle. Choose LS_SINUS if the panel angle is controlled by ropes at the edges.

### UCPTdownPanelAngle - Drive-up panel angle

---

<b>Type</b>	SNVT_angle_deg
<b>Range</b>	-359.98 ... 360.00 degrees [0.02 degrees]
<b>Default</b>	-75.00 degrees
<b>Description</b>	Angle of the panels when the blind is lowered. (0° = horizontal position)

### UCPTupPanelAngle - Drive-down panel angle

---

<b>Type</b>	SNVT_angle_deg
<b>Range</b>	-359.98 ... 360.00 degrees [0.02 degrees]
<b>Default</b>	75.00 degrees
<b>Description</b>	Angle of the panels when the blind is raised. (0° = horizontal position)

### UCPTpanelBottomAngle - Panel angle at bottom

---

<b>Type</b>	SNVT_angle_deg
<b>Range</b>	-359.98 ... 360.00 degrees [0.02 degrees]
<b>Default</b>	75.00 degrees
<b>Description</b>	Angle of the panel when bottom is reached. (0° = horizontal position)

### UCPTstopCommand - Stop command

---

<b>Type</b>	UNVT_stop_cmd
<b>Range</b>	SC_STOP, SC_NEXT, SC_OPPOSITE
<b>Default</b>	SC_STOP
<b>Description</b>	Defines which command stops. (Needed if the blind switch cannot send SET_STOP)

### UCPTdefaultSunblind - Default sunblind command

---

<b>Type</b>	UNVT_setting
<b>Range</b>	.function: SET_NO_MESSAGE, SET_NUL, SET_OFF, SET_ON, SET_DOWN, SET_UP, SET_STOP, SET_STATE; .setting: 0.0 ... 100.0 % of full level [0.5 % of full level]; .rotation: -359.98 ... 360.00 degrees [0.02 degrees]
<b>Default</b>	SET_STOP 0.0 0.00
<b>Description</b>	The command the sunblind actuator adopts at power-on or reset.

---

Appendix A: Description of the function objects

---

**UCPTmaxStartUpDelay - Maximum start-up delay**

---

<b>Type</b>	SNVT_time_sec
<b>Range</b>	0.0 ... 6553.5 seconds [0.1 seconds]
<b>Default</b>	0.0 seconds
<b>Description</b>	The maximum random time by which the default values is delayed after start-up. (Avoids electrical switching peaks)

**UCPTsafetyPositionCommand - Safety position command**

---

<b>Type</b>	SNVT_setting
<b>Range</b>	.function: SET_NULL, SET_OFF, SET_ON, SET_DOWN, SET_UP, SET_STOP, SET_STATE; .setting: 0.0 ... 100.0 % of full level [0.5 % of full level]; .rotation: -359.98 ... 360.00 degrees [0.02 degrees]
<b>Default</b>	SET_UP 100.0 360.00
<b>Description</b>	The value to reach the safety position.

**UCPTworkingPosition - Working position**

---

<b>Type</b>	SNVT_setting
<b>Range</b>	.function: SET_NULL, SET_OFF, SET_ON, SET_DOWN, SET_UP, SET_STOP, SET_STATE; .setting: 0.0 ... 100.0 % of full level [0.5 % of full level]; .rotation: -359.98 ... 360.00 degrees [0.02 degrees]
<b>Default</b>	SET_NULL 0.0 0.00
<b>Description</b>	The device successively stops at these positions via { SET_UP 0 0 }{ SET_DOWN 0 0 }. It stops also at position [0] via SET_OFF and at position [1] via SET_ON. Position [0] and [j] are taken after calibration. If no valid position was found the endposition will be adopted.

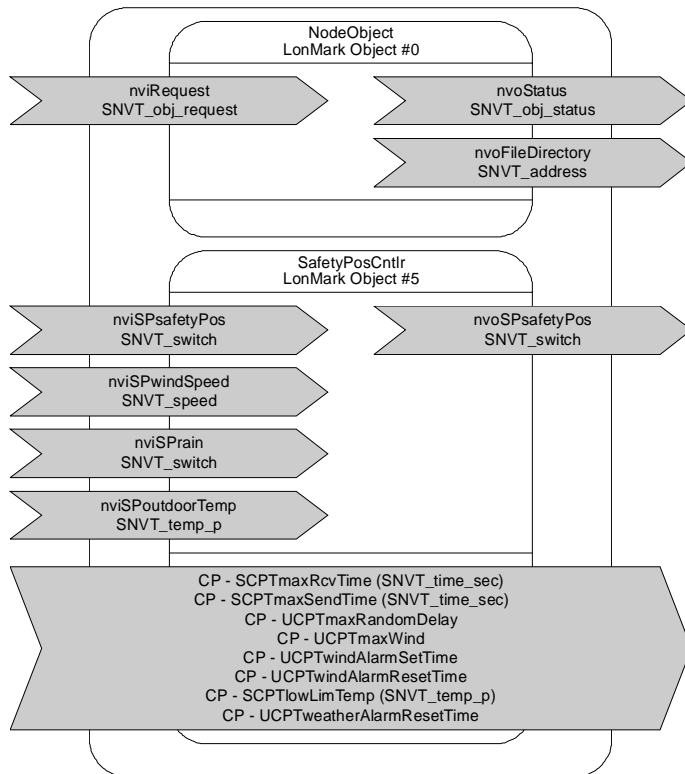
**SCPTmaxRcvTime - Maximum receive time**

---

<b>Type</b>	SNVT_setting
<b>Range</b>	0.0 ... 6553.5 seconds [0.1 seconds]
<b>Default</b>	300.0 second
<b>Description</b>	The maximum period of time that may expire with no updates on the associated input network nviBASafetyPos variables before the object goes into heartbeat failure mode. A zero value disables.



### 8.3 LonMark®-object Safety Position



#### 8.3.1 Introduction

The output nvoSPsafetyPos[i] is sent cyclically within the time configured in SCPTmaxSendTime. The input nviSPsafetyPos[i] can also be connected to a threshold value i. e. a wind switch. Please note that this input is not equipped with a timer (like the nviSPwind[i]). This has the result that commands to this input are executed immediately.

##### 8.3.1.1 Wind alarm

Via the input nviBCwind[i] the device receives the current wind speed. If the current wind speed is higher than the value defined in UCPTmaxWind the controller waits for the time UCPTwindAlarmSetTime. Wind alarm is activated when no new wind speed lower than UCPTmaxWind is received during this time.

During an activated wind alarm the current wind speed has to be below UCPTmaxWind for the time defined in UCPTwindAlarmResetTime to release the wind alarm.

### 8.3.1.2 Ice alarm

There are two possibilities to protect the blinds from damage through ice.

1. Connect an ice detector to the nviSPrain[i] and let the nviSPoutdoorTemp[i] unconnected. The nviSPoutdoorTemp[i] has to be set to an invalid value,
2. Connect a rainfall sensor to the nviSPrain[i] and an outdoor temperature sensor to the nviSPoutdoorTemp[i].

The rainfall sensor gives the information whether the blinds are wet or dry. After rainfall or an ice alarm the blinds are regarded as dry only if there is no rain/ice for the time configured in UCPTweatherAlarmResetTime.

Ice alarm is activated when the blinds are wet and the outdoor temperature goes below the value defined in SCPTlowLimTemp. The ice alarm is only released when the temperature is higher than SCPTlowLimTemp for the time configured in UCPTweatherAlarmResetTime.

If no rainfall sensor is connected the blinds are regarded as wet. If there is an invalid value at nviSPoutdoorTemp the blinds are driven into the safety position when rainfall is notified at nviSPrain[i]. To release the ice alarm there mustn't be ice or rain for the time defined in UCPTweatherAlarmResetTime.

The following scheme shows examples of possible temporal events connected to an ice alarm:

Rainfall (nviSPrain active)					
Frost (nviSPoutdoorTemp < SCPTlowLimTemp)					
Drying time – Time until the blinds change from wet to dry. (delay defined as UCPTweatherAlarmResetTime active)					
Dewing time – Time until the blinds change from frosted to wet. (delay defined as UCPTweatherAlarmResetTime active)					
Wet blinds (internal state)					
Ice alarm (nvoSPsafetyPos active)					

Rainfall (nviSPrain active)										
Frost (nviSPoutdoorTemp < SCPTlowLimTemp)										
Drying time – Time until the blinds change from wet to dry. (delay defined as UCPTweatherAlarmResetTime active)										
Dewing time – Time until the blinds change from frosted to wet. (delay defined as UCPTweatherAlarmResetTime active)										
Wet blinds (internal state)										
Ice alarm (nvoSPsafetyPos active)										

Appendix A: Description of the function objects

Rainfall (nviSPrain active)										
Frost (nviSPoutdoorTemp < SCPTlowLimTemp)										
Drying time – Time until the blinds change from wet to dry. (delay defined as UCPTweatherAlarmResetTime active)										
Dewing time – Time until the blinds change from frosted to wet. (delay defined as UCPTweatherAlarmResetTime active)										
Wet blinds (internal state)										
Ice alarm (nvoSPsafetyPos active)										

All bound input variables have to be updated within SCPTmaxRcvTime. A critical value is adopted if this does not happen within this time.

### 8.3.2 fb\_0 Object (LonMark Object #0)

Function	Network Variable	Dir	Type
Object request	nviRequest	IN	SNVT_obj_request
Object status	nvoStatus	OUT	SNVT_obj_status
Neuron address	nvoFileDirectory	OUT	SNVT_address

#### nviRequest – Object request

<b>Type</b>	SNVT_obj_request
<b>Range</b>	.object_id: <0.0; 65535.0> .object_request: RQ_NUL, RQ_NORMAL, RQ_DISABLED, RQ_UPDATE_STATUS, RQ_SELF_TEST, RQ_UPDATE_ALARM, RQ_REPORT_MASK, RQ_OVERRIDE, RQ_ENABLE, RQ_RMV_OVERRIDE, RQ_CLEAR_STATUS, RQ_CLEAR_ALARM, RQ_ALARM_NOTIFY_ENABLED, RQ_ALARM_NOTIFY_DISABLED, RQ_MANUAL_CTRL, RQ_REMOTE_CTRL, RQ_PROGRAM, RQ_CLEAR_RESET, RQ_RESET
<b>Default</b>	0 RQ_NORMAL [ID, request]
<b>Description</b>	Not supported

**nvoStatus – Object status**

---

<b>Type</b>	SNVT_obj_status
<b>Range</b>	.object_id: <0.0; 65535.0> .invalid_id: <0.0; 1.0> .invalid_request: <0.0; 1.0> .disabled: <0.0; 1.0> .out_of_limits: <0.0; 1.0> .open_circuit: <0.0; 1.0> .out_of_service: <0.0; 1.0> .mechanical_fault: <0.0; 1.0> .feedback_failure: <0.0; 1.0> .over_range: <0.0; 1.0> .under_range: <0.0; 1.0> .electrical_fault: <0.0; 1.0> .unable_to_measure: <0.0; 1.0> .comm_failure: <0.0; 1.0> .fail_self_test: <0.0; 1.0> .self_test_in_progress: <0.0; 1.0> .locked_out: <0.0; 1.0> .manual_control: <0.0; 1.0> .in_alarm: <0.0; 1.0> .in_override: <0.0; 1.0> .report_mask: <0.0; 1.0> .programming_mode: <0.0; 1.0> .programming_fail: <0.0; 1.0> .alarm_notify_disabled: <0.0; 1.0> .reset_complete: <0.0; 1.0> .reserved2: <0.0; 0.0>
<b>Default</b>	0 [ID, status flags]
<b>Description</b>	Not supported

**nvoFileDirectory – Neuron address**

---

<b>Type</b>	SNVT_address
<b>Range</b>	<16384.0; 64767.0>
<b>Default</b>	0 [16-bit address value]
<b>Description</b>	Not supported

### 8.3.3 SafetyPosCntlr (LonMark Object #5)

Function	Network Variable	Dir	Type
Switch	nviSPsafetyPos	IN	SNVT_switch
Linear velocity	nviSPwindSpeed	IN	SNVT_speed
Switch	nviSPrain	IN	SNVT_switch
Temperature	nviSPoutdoorTemp	IN	SNVT_temp_p
Switch	nvoSPsafetyPos	OUT	SNVT_switch

Function	Configuration Parameter	Type
Maximum receive time	SCPTmaxRcvTime	SCPTmaxRcvTime
Maximum send time	SCPTmaxSendTime	SCPTmaxSendTime
Maximum random delay	UCPTmaxRandomDelay	SNVT_time_sec
Maximum wind speed	UCPTmaxWind	SNVT_speed
Wind alarm set time	UCPTwindAlarmSetTime	SNVT_time_sec
Wind alarm reset time	UCPTwindAlarmResetTime	SNVT_time_min
Low limit temperature	SCPTlowLimTemp	SCPTlowLimTemp
Weather alarm reset time	UCPTweatherAlarmResetTime	SNVT_time_min

#### nviSPsafetyPos – Switch

<b>Type</b>	SNVT_switch
<b>Range</b>	.value: <0.0; 100.0> .state: <-1.0; 1.0>
<b>Default</b>	0.0 0 [value, state]
<b>Description</b>	Input to activate the safety position with the highest priority.

#### nviSPwindSpeed – Linear velocity

<b>Type</b>	SNVT_speed
<b>Range</b>	<0.0; 6553.5>
<b>Default</b>	0.0 [meters/second]
<b>Description</b>	Input for a wind sensor.

#### nviSPrain – Switch

<b>Type</b>	SNVT_switch
<b>Range</b>	.value: <0.0; 100.0> .state: <-1.0; 1.0>
<b>Default</b>	0.0 0 [value, state]
<b>Description</b>	Input for a rainfall sensor.

#### nviSPoutdoorTemp – Temperature

<b>Type</b>	SNVT_temp_p
<b>Range</b>	<-273.17; 327.67>
<b>Default</b>	0.0 [degrees Celsius]
<b>Description</b>	Input for an outdoor temperature sensor.

Appendix A: Description of the function objects

---

**nvoSPsafetyPos – Switch**

---

<b>Type</b>	SNVT_switch
<b>Range</b>	.value: <0.0; 100.0> .state: <-1.0; 1.0>
<b>Default</b>	0.0 0 [value, state]
<b>Description</b>	Safety position trigger output

**SCPTmaxRcvTime – Maximum receive time**

---

<b>Type</b>	SCPTmaxRcvTime
<b>Range</b>	<0.0; 6553.5>
<b>Default</b>	300.0 [seconds]
<b>Description</b>	The maximum period of time that may expire with no updates on the associated input network variables before the object goes into heartbeat failure mode. A zero value disables. Used for following inputs, if they are bound, or after an update: nviSPoutdoorTemp, nviSPsafetyPos, nviSPrain, nviSPwindSpeed.

**SCPTmaxSendTime – Maximum send time**

---

<b>Type</b>	SCPTmaxSendTime
<b>Range</b>	<0.0; 6553.5>
<b>Default</b>	60.0 [seconds]
<b>Description</b>	The maximum period of time between consecutive transmissions of the current value

**UCPTmaxRandomDelay – Maximum random delay**

---

<b>Type</b>	SNVT_time_sec
<b>Range</b>	<0.0; 6553.5>
<b>Default</b>	0.0
<b>Description</b>	Maximum time between receiving and processing global commands. (Avoids electrical switching peaks)

**UCPTmaxWind – Maximum wind speed**

---

<b>Type</b>	SNVT_speed
<b>Range</b>	<0.0; 6553.5>
<b>Default</b>	14.0 [metres/second]
<b>Description</b>	The maximum wind speed allowed before alarm is activated.

Appendix A: Description of the function objects

---

**UCPTwindAlarmSetTime – Wind alarm set time**

---

<b>Type</b>	SNVT_time_sec
<b>Range</b>	<0.0; 6553.5>
<b>Default</b>	2.0 [seconds]
<b>Description</b>	The maximum wind speed must be exceeded for this time period before the alarm state is activated.

**UCPTwindAlarmResetTime – Wind alarm reset time**

---

<b>Type</b>	SNVT_time_min
<b>Range</b>	<0.0; 65535.0>
<b>Default</b>	30 [minutes]
<b>Description</b>	The current wind speed must be below the maximum wind speed for this time period before the alarm state is deactivated.

**SCPTlowLimTemp – Low limit temperature**

---

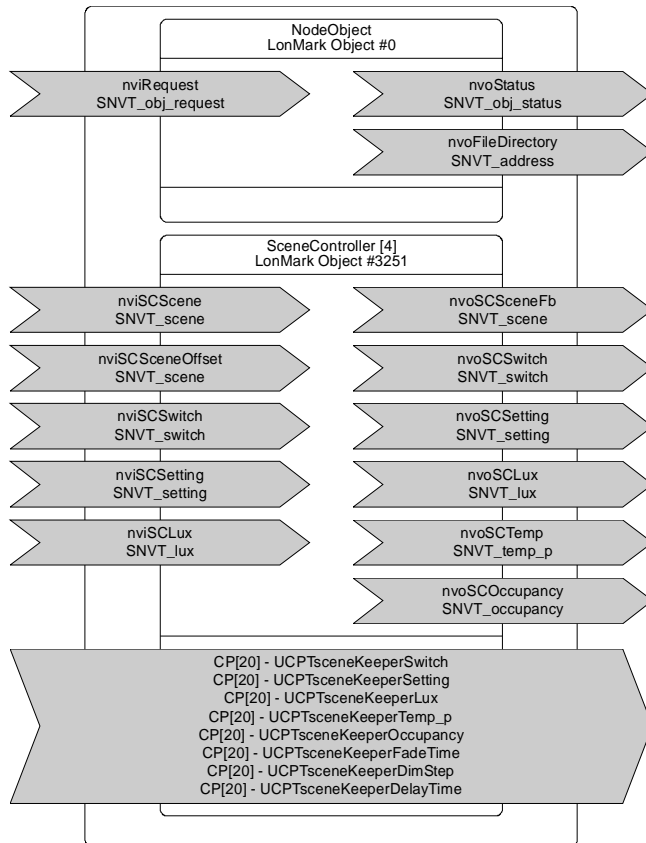
<b>Type</b>	SCPTlowLimTemp
<b>Range</b>	<-273.17; 327.67>
<b>Default</b>	3.0 [degrees Celsius]
<b>Description</b>	The limit for the outdoor temperature to activate an alarm.

**UCPTweatherAlarmResetTime – Weather alarm reset time**

---

<b>Type</b>	SNVT_time_min
<b>Range</b>	<0.0; 65535.0>
<b>Default</b>	0 [minutes]
<b>Description</b>	All alarm conditions must have passed for this time period before the alarm state is deactivated.

## 8.4 LonMark®-object Scene Controller



### 8.4.1 Introduction

#### Recalling Scenes

After the Scene Controller has been enabled by nviSCsetting[i] (SET\_ON) the stored scene settings can be recalled by nviSCscene[i].function = SC\_RECALL and the corresponding scene number. The settings are then propagated via the outputs nvoSCswitch[i], nvoSCsetting[i], nvoSCLux[i], nvoSCTemp[i] and nvoSCOccupancy[i]. Scenes can be delayed by use of the UCPTsceneKeeperDlyTime[i] property.

If an output shall not change when a new scene is recalled, the corresponding parameter has to be set at an SW\_HOLD and/or SET\_NO\_MESSAGE values under this particular scene number.



Appendix A: Description of the function objects

---

Scene settings that shall be propagated when the controller is turned off have to be stored under scene number 20. Next time the controller is turned on the last enabled scene is recalled.

### Storing Scenes

There are two ways to configure the scene controller memory:

1. The scene memory can be configured directly by use of the UCPTsceneKeeperXXX[i][j] property.
2. For lighting scenes, the current values of the nviSCswitch[i] and nviSClux[i] input can be stored in the scene memory unit corresponding to the given scene number by a learn command (nviSCscene[i].function = SC\_LEARN). A long pulse (e. g. initiated by hold of a make-contact element) usually causes this command.

### Cross-fading

The UCPTsceneKeeperFadeTime[i] property determines the time between two consecutively transmitted dim commands and UCPTsceneKeeperDimStep[i] defines the step value for cross-fading between two scenes. The cross-fading function is only provided for dimming actuators.

### 8.4.2 fb\_0Object (LonMark Object #0)

Network variable	Type	Dir	Description	Default value
nviRequest	SNVT_obj_request	IN		-
nvoStatus	SNVT_obj_status	OUT		-
nvoFileDirectory	SNVT_address	OUT		-

#### nviRequest – Object request

---

<b>Type</b>	SNVT_obj_request
<b>Range</b>	.object_id: <0.0; 65535.0> .object_request: RQ_NUL, RQ_NORMAL, RQ_DISABLED, RQ_UPDATE_STATUS, RQ_SELF_TEST, RQ_UPDATE_ALARM, RQ_REPORT_MASK, RQ_OVERRIDE, RQ_ENABLE, RQ_RMV_OVERRIDE, RQ_CLEAR_STATUS, RQ_CLEAR_ALARM, RQ_ALARM_NOTIFY_ENABLED, RQ_ALARM_NOTIFY_DISABLED, RQ_MANUAL_CTRL, RQ_REMOTE_CTRL, RQ_PROGRAM, RQ_CLEAR_RESET, RQ_RESET
<b>Default</b>	0 RQ_NORMAL [ID, request]
<b>Description</b>	Not supported

**nvoStatus – Object status**

---

<b>Type</b>	SNVT_obj_status
<b>Range</b>	.object_id: <0.0; 65535.0> .invalid_id: <0.0; 1.0> .invalid_request: <0.0; 1.0> .disabled: <0.0; 1.0> .out_of_limits: <0.0; 1.0> .open_circuit: <0.0; 1.0> .out_of_service: <0.0; 1.0> .mechanical_fault: <0.0; 1.0> .feedback_failure: <0.0; 1.0> .over_range: <0.0; 1.0> .under_range: <0.0; 1.0> .electrical_fault: <0.0; 1.0> .unable_to_measure: <0.0; 1.0> .comm_failure: <0.0; 1.0> .fail_self_test: <0.0; 1.0> .self_test_in_progress: <0.0; 1.0> .locked_out: <0.0; 1.0> .manual_control: <0.0; 1.0> .in_alarm: <0.0; 1.0> .in_override: <0.0; 1.0> .report_mask: <0.0; 1.0> .programming_mode: <0.0; 1.0> .programming_fail: <0.0; 1.0> .alarm_notify_disabled: <0.0; 1.0> .reset_complete: <0.0; 1.0> .reserved2: <0.0; 0.0>
<b>Default</b>	0 [ID, status flags]
<b>Description</b>	Not supported

**nvoFileDirectory – Neuron address**

---

<b>Type</b>	SNVT_address
<b>Range</b>	<16384.0; 64767.0>
<b>Default</b>	0 [16-bit address value]
<b>Description</b>	Not supported

### 8.4.3 SceneController (LonMark Object #3251) [4]

Network variable	Type	Dir	Description	Default value
nviSCScene	SNVT_scene	IN	<p><b>Scene trigger input</b></p> <p>This input triggers a scene (SC_RECALL) or loads the scene-preset memory with current values (SC_LEARN). Memory units for 20 scenes are provided.</p> <p>By SC_RECALL, the scene settings stored under the chosen .scene_number are recalled. The recall command can be delayed by the time defined in UCPTsceneKeeperDlyTime[i].</p> <p>An SC_LEARN command stores the current values of nviSCswitch[i] and nviSCLux[i] in the scene memory unit corresponding to the given .scene_number. Values nviSCswitch[i] and nviSCLux[i] are stored only if they are bound.</p> <p>A scene number zero does not cause any control action (only needed for default before commissioning/at reset).</p>	<p>.function = SC_RECALL</p> <p>.scene_number = 0</p>
nviSCSceneOffset	SNVT_scene	IN	<p>Value within nviSCSceneOffset[i].scene_number will be added to nviSCScene[i].scene_number.</p> <p>nviSCSceneOffset[i].scene_number &gt;= 20 leads to Scene 0.</p>	<p>.function = SC_RECALL</p> <p>.scene_number = 255</p>
nviSCSwitch	SNVT_switch	IN	<p><b>Direct control input</b></p> <p>Updates of this input are directly passed to the nvoSCswitch[i] output. This input overrides other inputs and ongoing fades/delays. Thus, scene settings can be modified e. g. manually.</p>	<p>.value = 0</p> <p>.state = -1</p>
nviSCSetting	SNVT_setting	IN	<p><b>Controller enabling/disabling input</b></p> <p>Used to turn the controller on and off. A SET_ON command recalls the last scene. When the controller is turned off (SET_OFF), the scene stored in memory unit no. 20 is propagated without any configured delays.</p>	<p>.function = SET_ON</p> <p>.setting = 0</p> <p>.rotation = 0</p>
nviSCLux	SNVT_lux	IN	<p><b>Illumination level input</b></p> <p>Input for an illumination value [lux], which is stored in the scene memory when nviSCscene[i] receives an SC_LEARN command and .nviSCscene[i] is bound.</p>	0
nvoSCSceneFb	SNVT_scene	OUT	<p><b>Scene feedback output</b></p> <p>Propagates the current state of the scene controller to the network.</p>	<p>.function = SC_RECALL</p> <p>.scene_number = 0</p>
nvoSCSwitch	SNVT_switch	OUT	<p><b>Switch output</b></p> <p>Provides the value of the UCPTsceneKeeperSwitch[i][j] scene memory for an actuator (e. g. a lamp actuator), whenever a scene change is initiated.</p>	<p>.value = 0</p> <p>.state = -1</p>
nvoSCSetting	SNVT_setting	OUT	<p>Provides the value of the UCPTsceneKeeperSetting[i][j] scene memory for a controller (e. g. a sunblind controller). If sunblind are controlled, information about their position (.setting) and panel angle (.rotation) can be stored in the scene memory.</p>	<p>.function = SET_NUL</p> <p>.setting = 0</p> <p>.rotation = 0</p>
nvoSCLux	SNVT_lux	OUT	<p><b>Illumination level output</b></p> <p>Propagates the illumination level of the UCPTsceneKeeperLux[i][j] scene memory.</p>	0
nvoSCTemp	SNVT_temp_p	OUT	<p><b>Temperature output</b></p>	327.67 °C

Appendix A: Description of the function objects

Network variable	Type	Dir	Description	Default value
			Propagates the temperature value of the UCPTsceneKeeperTemp[i][j] scene memory [°C].	(undefined)
nvoSCOoccupancy	SNVT_occupancy	OUT	<b>Occupancy state output</b> Propagates the occupancy state defined in the UCPTsceneKeeperOccupancy[i][j] scene memory.	OC_NUL

Configuration Properties

Network variable	Type	Description	Default value
CP	UCPTsceneKeeperSwitch[20]	<b>Scene keeper switch</b> Provides direct access to the scene memory to configure SNVT_switch values for every scene.  If the switch output shall not change when a new scene is recalled, this parameter has to be set at an HOLD value (.function = HOLD = 0), which is not propagated.	.value = 0 .function = -1
CP	UCPTsceneKeeperSetting[20]	<b>Scene keeper setting</b> Provides direct access to the scene memory to configure SNVT_setting values for every scene.  If the setting output shall not change when a new scene is recalled, this parameter has to be set at an SET_NO_MESSAGE value (.function = SET_NO_MESSAGE), which is not propagated.	.function = SET_NUL .setting = 0 .rotation = 0
CP	UCPTsceneKeeperLux[20]	<b>Scene keeper lux</b> Provides direct access to the scene memory to configure illumination levels for every scene.  If the illumination level output lux shall not change when a new scene is recalled, this parameter has to be set at an undefined value (0), which is not propagated.	0
CP	UCPTsceneKeeperTemp_p[20]	<b>Scene keeper temperature</b> Provides direct access to the scene memory to configure temperatures [°C] for every scene.  If the temperature output shall not change when a new scene is recalled, this parameter has to be set at an undefined value (327.67 °C), which is not propagated.	327.67 °C (undefined)
CP	UCPTsceneKeeperOccupancy[20]	<b>Scene keeper occupancy</b> Provides direct access to the scene memory to configure occupancy states for every scene.  If the occupancy state output shall not change when a new scene is recalled, this parameter has to be set at an undefined value (OC_NUL), which is not propagated.	OC_NUL
CP	UCPTsceneKeeperFadeTime[20]	<b>Time for fading scenes</b> If scene <i>i</i> is recalled, this time is used to reach requested value for this scene. Applied on nvoSCswitch[i]. Value should be larger than 500ms	0
CP	UCPTsceneKeeperDimStep[20]	<b>Scene keeper dim step</b> Sets the step value of nvoSCswitch[i].value for cross-fading.	3.5 %
CP	UCPTsceneKeeperDelayTime[20]	<b>Scene keeper delay</b> Defines the time between recall and performance of the corresponding scene. Only affects the nvoSCswitch[i] output.	0 (disabled)

### nviSCscene[i] – Scene trigger input

---

<b>Type:</b>	SNVT_scene
<b>Valid Range:</b>	.function: SC_RECALL, SC_LEARN .scene_number: 1 .. 20 :
<b>Default Value:</b>	.function = SC_RECALL .scene_number = 0
<b>Description:</b>	<p>This input triggers a scene (SC_RECALL) or loads the scene-preset memory with current values (SC_LEARN). Memory units for 20 scenes are provided. By SC_RECALL, the scene settings stored under the chosen .scene_number are recalled. The recall command can be delayed by the time defined in UCPTsceneKeeperDlyTime[i].</p> <p>An SC_LEARN command stores the current values of nviSCswitch[i] and nviSClux[i] in the scene memory unit corresponding to the given .scene_number.</p> <p>A scene number zero does not cause any control action (only needed for default before commissioning/at reset).</p>

### nviSCswitch[i] – Direct control input

---

<b>Type:</b>	SNVT_switch
<b>Valid Range:</b>	.value: 0 .. 100 % .state: 0, 1, -1
<b>Default Value:</b>	.value = 0 .state = -1
<b>Description:</b>	<p>Updates of this input are directly passed to the nvoSCswitch[i] output. This input overrides other inputs and ongoing fades/delays. Thus, scene settings can be modified e. g. manually.</p>

### nviSCsetting[i] – Controller enabling/disabling input

---

<b>Type:</b>	SNVT_setting
<b>Valid Range:</b>	.function: SET_ON, SET_OFF :
<b>Default Value:</b>	.function = SET_ON .setting = 0 .rotation = 0
<b>Description:</b>	<p>Used to turn the controller on and off. A SET_ON command recalls the last scene. When the controller is turned off (SET_OFF), the scene stored in memory unit no. 20 is propagated without any configured delays.</p>

#### nviSClux[i] – Illumination level input

---

<b>Type:</b>	SNVT_lux
<b>Valid Range:</b>	0 .. 65,534 lux
<b>Default Value:</b>	0
<b>Description:</b>	Input for an illumination value [lux], which is stored in the scene memory when nviSCscene[i] receives an SC_LEARN command.

#### nviSCsceneOffset[j] – Scene offset input

---

<b>Type:</b>	SNVT_scene
<b>Valid Range:</b>	.function: SC_RECALL .scene_number: 1 .. 20
<b>Default Value:</b>	.function: SC_RECALL .scene_number = 255
<b>Description:</b>	The pending .scene_number value at this input is added to the .scene_number value at nviSCswitch. If the sum is an invalid value (sum > 20) the result will be 0.

#### nvoSCswitch[i] – Switch output

---

<b>Type:</b>	SNVT_switch
<b>Valid Range:</b>	.value: 0 .. 100 % .state: 0, 1, -1
<b>Default Value:</b>	.value = 0 .state = -1
<b>Description:</b>	Provides the value of the UCPTsceneKeeperSwitch[i][j] scene memory for an actuator (e. g. a lamp actuator), whenever a scene change is initiated.

#### nvoSCsceneFb[i] – Scene feedback output

---

<b>Type:</b>	SNVT_scene
<b>Valid Range:</b>	.function: SC_RECALL, SC_LEARN .scene_number: 1 .. 20
<b>Default Value:</b>	.function = SC_RECALL .scene_number = 0
<b>Description:</b>	Propagates the current state of the scene controller to the network.

Appendix A: Description of the function objects

---

**nvoSCsetting[i] – Setting output**

---

<b>Type:</b>	SNVT_setting
<b>Valid Range:</b>	.function SET_OFF, SET_ON, SET_DOWN, SET_UP, SET_STOP, : SET_STATE, SET_NUL 0 .. 100 % .setting: -359.98° .. +360.00° .rotation :
<b>Default Value:</b>	.function = SET_NUL .setting = 0 .rotation = 0
<b>Description:</b>	Provides the value of the UCPTsceneKeeperSetting[i][j] scene memory for a controller (e. g. a sunblind controller). If the blinds are controlled, information about their position (.setting) and panel angle (.rotation) can be stored in the scene memory.

**nvoSClux[i] – Illumination level output**

---

<b>Type:</b>	SNVT_lux
<b>Valid Range:</b>	0 .. 65,534 lux
<b>Default Value:</b>	0
<b>Description:</b>	Propagates the illumination level of the UCPTsceneKeeperLux[i][j] scene memory.

**nvoSCtemp[i] – Temperature output**

---

<b>Type:</b>	SNVT_xxx (Default: SNVT_temp_p)
<b>Valid Range:</b>	-273.17 °C .. +327.66 °C
<b>Default Value:</b>	327.67 °C (undefined)
<b>Description:</b>	Propagates the temperature value of the UCPTsceneKeeperTemp[i][j] scene memory [°C].

**nvoSCoccupancy[i] – Occupancy state output**

---

<b>Type:</b>	SNVT_occupancy
<b>Valid Range:</b>	OC_OCCUPIED, OC_UNOCCUPIED, OC_BYPASS, OC_STANDBY, OC_NUL
<b>Default Value:</b>	OC_NUL
<b>Description:</b>	Propagates the occupancy state defined in the UCPTsceneKeeperOccupancy[i][j] scene memory.

## Configuration Properties

### UCPTsceneKeeperSwitch[i][j] – Scene keeper switch

---

<b>Type:</b>	SNVT_switch
<b>Valid Range:</b>	.value: 0 .. 100 % .state: 0, 1, -1
<b>Default Value:</b>	.value = 0 .state = -1
<b>Description:</b>	Provides direct access to the scene memory to configure SNVT_switch values for every scene. If the switch output shall not change when a new scene is recalled, this parameter has to be set at an undefined value (.state = -1), which is not propagated.

### UCPTsceneKeeperSetting[i][j] – Scene keeper setting

---

<b>Type:</b>	UNVT_setting
<b>Valid Range:</b>	.function: SET_OFF, SET_ON, SET_DOWN, SET_UP, SET_STOP, SET_STATE, SET_NO_MESSAGE, SET_NUL .setting: 0 .. 100 % .rotation: -359.98° .. +360.00°
<b>Default Value:</b>	.function = SET_NUL .setting = 0 .rotation = 0
<b>Description:</b>	Provides direct access to the scene memory to configure UNVT_setting values for every scene. If the setting output shall not change when a new scene is recalled, this parameter has to be set at an undefined value (.function = SET_NUL), which is not propagated.

### UCPTsceneKeeperLux[i][j] – Scene keeper lux

---

<b>Type:</b>	SNVT_lux
<b>Valid Range:</b>	0 .. 65,534 lux
<b>Default Value:</b>	0
<b>Description:</b>	Provides direct access to the scene memory to configure illumination levels for every scene. If the illumination level output lux shall not change when a new scene is recalled, this parameter has to be set at an undefined value (0), which is not propagated.



#### UCPTsceneKeeperTemp[i][j] – Scene keeper temperature

---

<b>Type:</b>	SNVT_temp_p
<b>Valid Range:</b>	-273.17 °C .. +327.66 °C
<b>Default Value:</b>	327.67 °C (undefined)
<b>Description:</b>	Provides direct access to the scene memory to configure temperatures [°C] for every scene. If the temperature output shall not change when a new scene is recalled, this parameter has to be set at an undefined value (327.67 °C), which is not propagated. Attention: If the type of <code>nvoSCtemp[i]</code> has been changed, the type of this parameter has to be adjusted as well.

#### UCPTsceneKeeperOccupancy[i][j] – Scene keeper occupancy

---

<b>Type:</b>	SNVT_occupancy
<b>Valid Range:</b>	OC_OCCUPIED, OC_UNOCCUPIED, OC_BYPASS, OC_STANDBY, OC_NUL
<b>Default Value:</b>	OC_NUL
<b>Description:</b>	Provides direct access to the scene memory to configure occupancy states for every scene. If the occupancy state output shall not change when a new scene is recalled, this parameter has to be set at an undefined value (OC_NUL), which is not propagated.

#### UCPTsceneKeeperFadeTime[i][j] – Scene keeper fade time for nviSCSwitch

---

<b>Type:</b>	UNVT_time_msec
<b>Valid Range:</b>	100 .. 65,534 ms
<b>Default Value:</b>	0 (disabled)
<b>Description:</b>	Cross-fading time for the change from one scene to another at <code>nviSCSwitch[i]</code> .

#### UCPTsceneKeeperDimStep[i][j] – Scene keeper dim step

---

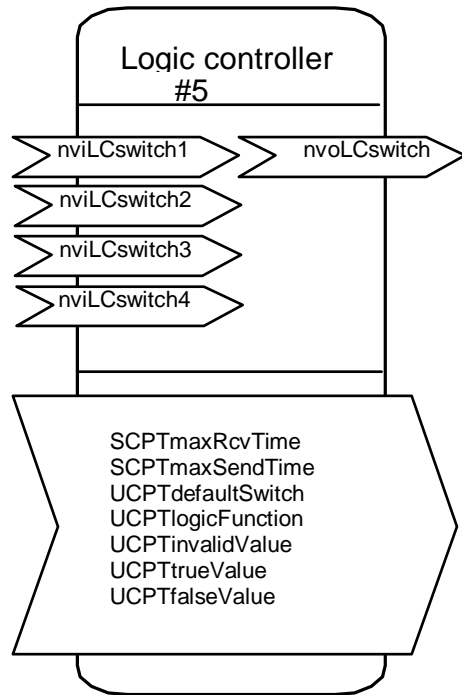
<b>Type:</b>	SNVT_lev_cont
<b>Valid Range:</b>	0 .. 100 %
<b>Default Value:</b>	3.5 %
<b>Description:</b>	Sets the step value of <code>nvoSCswitch[i].value</code> for cross-fading.

**UCPTsceneKeeperDelayTime[i][j] – Scene keeper delay**

---

<b>Type:</b>	SNVT_time_sec
<b>Valid Range:</b>	0 .. 6,553.4 s
<b>Default Value:</b>	0 (disabled)
<b>Description:</b>	Defines the time between recall and performance of the corresponding scene. Only affects the <code>nvoSCswitch [i]</code> output.

### 8.5 LonMark®-object Logic controller (#) switch



Inputs/outputs	Type / SNVT index	Value range	Default value	Description
nviLCswitch1	SNVT_switch	default	{-1,0}	Switch input 1.
nviLCswitch2	SNVT_switch	default	{-1,0}	Switch input 2.
nviLCswitch3	SNVT_switch	default	{-1,0}	Switch input 3.
nviLCswitch4	SNVT_switch	default	{-1,0}	Switch input 4.
nvoLCswitch	SNVT_switch	default	{-1,0}	Switch output.

Configuration	Type / CPT index	Value range	Default value	Description
SCPTmaxSendTime	SNVT_time_sec	0 .. 6553 s in 1s	0 s (Off)	The maximum interval of time that can be sent using the variable nvoLWswitch .
LcmaxRcvTime	SNVT_time_sec	0 .. 6553 s in 1s	0 s (Off)	The maximum interval of time between two update of a variable. When this time has been exceeded, the output nvoLCswitch takes the status "invalid" (UCPTinvalidValue).
UCPTdefaultSwitch	SNVT_switch	default	{ 0, -1 }	Value taken by the nviLCswitch after a reset.
LclogicFunction	UNVT_logic_fnc	LF_AND, LF_OR, LF_XOR, LF_NXOR, LF_NAND, LF_NOR, LF_OVRIDE, LF_TRSHLD		Setting the logic function. Account is taken only of those variables which have one value at variance from (0,-1).
UCPTinvalidValue	UNVT_switch_cfg	.function:	{SW_NUL, 0}	Value for the output if the result of the logic is invalid. The result is invalid if 1) no variable has received an update, 2) LcmaxRcvTime has been exceeded for a variable.
UCPTonValueSW	UNVT_switch_cfg		{SW_NUL, 100}	Value for the output if the result of the logic is TRUE.
UCPToffValueSW	UNVT_switch_cfg		{SW_NUL, 0}	Value for the output if the result of the logic is FALSE.