L-ROC

L-ROC™ Room Automation Controller

User Manual

LOYTEC electronics GmbH

Contact

LOYTEC electronics GmbH
Blumengasse 35
1170 Vienna
AUSTRIA/EUROPE
support@loytec.com
http://www.loytec.com

Version 2.1.16

Document № 88085409

Contents

Abbreviations

100Base-T	100 Mbps Ethernet network with RJ-45 plug
Aggregation	1) Collection of several CEA-709 packets into a single CEA-852 packet 2) Calculating an aggregate value (min, max, average) from several values
AST	Alarming, Scheduling, Trending
BACnet	Building Automation and Control Network
BBMD	BACnet Broadcast Management Device
BDT	Broadcast Distribution Table
BOOTP	Bootstrap Protocol, RFC 1497
CA	Certification Authority
CAT	Composite Automation Type (L-STUDIO)
CEA-709	Protocol standard for LONWORKS networks
CEA-852	Protocol standard for tunneling CEA-709 packets over IP channels
CN	Control Network
COV	change-of-value
CR	Channel Routing
CS	Configuration Server that manages CEA-852 IP devices
DA	Data Access (Web service)
DHCP	Dynamic Host Configuration Protocol, RFC 2131, RFC 2132
DIF, DIFE	Data Information Field, Data Information Field Extension
DL	Data Logger (Web service)
DNS	Domain Name Server, RFC 1034
DST	Daylight Saving Time
EEP	EnOcean Equipment Profile
GMT	Greenwich Mean Time
IP	Internet Protocol
IP-852	logical IP channel that tunnels CEA-709 packets according CEA-852
LAN	Local Area Network
LSD Tool	LOYTEC System Diagnostics Tool
LWEB-802	LOYTEC Visualization Technology
LWEB-900	LOYTEC Building Management System
MAC	Media Access Control
MD5	Message Digest 5, a secure hash function, see Internet RFC 1321
M-Bus	Meter-Bus (Standards EN 13757-2, EN 13757-3)
MIB	Management Information Base
MS/TP	Master/Slave Token Passing (this is a BACnet data link layer)
NAT	Network Address Translation, see Internet RFC 1631
NV	Network Variable
OPC	Open Process Control
OPC UA	OPC Unified Architecture
PEM	Privacy Enhanced Mail
PLC	Programmable Logic Controller

RCD ..... Room Control Device
RNI ..... Remote Network Interface
RSTP ..... Rapid Spanning Tree Protocol (Standard IEEE 802.1D-2004)
RTT ..... Round-Trip Time
RTU ..... Remote Terminal Unit
SCPT ..... Standard Configuration Property Type
SFB ..... Special Function block (L-STUDIO)
SSH ..... Secure Shell
SL ..... Send List
SMTP ..... Simple Mail Transfer Protocol
SNMP ..... Simple Network Management Protocol
SNTP ..... Simple Network Time Protocol
SSL ..... Secure Socket Layer
STP ..... Spanning Tree Protocol (Standard IEEE 802.1D)
TLS ..... Transport Layer Security
UCPT ..... User-defined Configuration Property Type
UI ..... User Interface
UNVT ..... User-defined Network Variable Type
UTC ..... Universal Time Coordinated
VIF, VIFE ..... Value Information Field, Value Information Field Extension
WLAN ..... Wireless LAN
XML ..... eXtensible Markup Language
YSP ..... Year Shade Progression

1 Introduction

1.1 Overview

The L-ROC Room Controller provides the basis for a revolutionary room automation system based on IP, which seamlessly integrates with native BACnet/IP networks and LonMark Systems at the controller level. Together with the L-STUDIO software, flexible room solutions can be created and adapted to changing requirements during the project with little effort. Integral parts of the L-ROC system are a web-based room operation via an LWEB-802/ 803 dashboard and the automatic generation of graphics for the L-VIS Touch Panel for local operation.

LROC hardware models

The LROC-102 room controllers are DIN-rail mountable and can be extended by L-IOB I/O modules. They provide six communication ports for integrating CEA-709, BACnet MS/TP, KNX, Modbus RTU, M-Bus or SMI channels.

The LROC-400 room controllers provide all common interfaces and a large number of physical I/Os for room automation projects. KNX devices are integrated via the built-in KNX TP1 or the KNXnet/IP interface. DALI lamps and DALI sensors are connected to the DALI interface with an integrated DALI power supply. Up to 16 SMI sunblind motors connect to the SMI interface. Belimo valves connect to the MP-Bus interface. BACnet MS/TP devices connect to the RS-485 interface, which can also be configured as a Modbus RTU interface to connect MODBUS devices like energy meters or ekey finger scanner for access control. L-STAT thermostats connect to the dedicated L-STAT interface. The EXT interface can connect 16 more SMI sunblind motors through the LSMI-800 interface or M-Bus meters through the L-MBUS20 interface. EnOcean devices connect to the EnOcean interface through an external antenna. Dual Ethernet ports allow daisy chaining of L-ROC controllers in a ring topology and provide BACnet/ IP, LON/ IP, MODBUS/ IP, KNXnet/ IP and OPC communication. Optionally the L-ROC can communicate via wireless LAN through the LWLAN-800 wireless adapter connected to the USB port. 24 relay outputs, 8 TRIAC outputs, 8 analog outputs, 10 universal inputs and 2 digital inputs connect various physical inputs and outputs. Our room automation library provides pre built function modules for all lighting, heating, cooling, ventilation, sunblinds and access control via finger scanners. Built-in SSL encryption ensures secure operation of the room automation system.

Flexible Room Concept for Room Automation

A room segment is the smallest individually controllable entity in the L-ROC System. The L-ROC Room Controller provides a set of functions for every room segment including:

- lighting control with constant light controller

- sunblind control with angle adjustment
- temperature control for heating, cooling, and ventilation
- occupancy detection
- window monitoring and window contact

Depending on the model an L-ROC Room Controller can control between 8 and 16 room segments. Based on the various room segment types, larger buildings can be modeled in a hierarchical manner. Areas are built with an area manager by combining multiple room controllers. A floor manager manages multiple areas in one floor. Depending on the architecture, the building can be split up into areas and floors as needed.

Area/Floor Managers are responsible for handling functions needed for corridor, staircase, and bathroom lighting, or even ventilation. Floor managers facilitate the data communication between the floors and handle floor relevant functions e.g. processing meter data.

Rooms can be created in any size by moving, installing, or removing partition walls. The resulting logical connections between the L-ROC Room Controllers will be built automatically. All graphical user interfaces and network connections are automatically generated and adapted respectively.

AST™ for every Room Segment

L-ROC provides a set of functions for Alarming, Scheduling, and Trending (AST™) for every room segment. Each room segment can be operated entirely independently. The AST™ functions are fully available to higher-level systems through BACnet/ IP and web services (L-WEB System). Distributed schedulers can be efficiently managed and changed with LWEB-900.

Inter Room Communication through redundant IP Network

The L-ROC Room Controllers are equipped with two Ethernet ports. It can be configured to use the internal switch to interconnect the two ports in a bus or ring topology.

Using the internal switch, a daisy chained line topology of up to 20 devices can be built, which reduces costs for network installation. The IP switch also allows the setup of a redundant Ethernet installation (ring topology), which increases reliability. The redundant Ethernet topology is enabled by the Rapid Spanning Tree Protocol (RSTP), which is supported by most managed switches.

Integrated L-WEB Room Operation

L-ROC controllers provide graphical user interfaces for room operation directly via an IP connection to the user, without the need for an additional web server. Graphic projects are distributed among the L-ROC Room Controllers and can be accessed by LWEB-802/ 803 from any PC workstation, smart phone, or tablet PC running Android or iOS.

Integration of the L-STAT Network Thermostat

Depending on the model, 8 to 16 L-STAT thermostats can be integrated via the L-STAT interface per L-ROC Room Controller. In addition to the attractive, modern design and intuitive operation, L-STAT provides a range of features to individually increase the room comfort.

Internal sensors measure temperature, humidity, dew point, occupancy, and the CO_2 content of the air. There is also the possibility to control room functions via an IR remote control. Standard pushbuttons and external temperature sensors can be integrated through additional inputs. A built-in NFC chip (Near Field Communication) provides the ability to direct mobile devices to the URL of the respective room website.

Connection to Higher-Level Systems

Higher-level systems can seamlessly integrate L-ROC Room Controllers via BACnet/IP, LonMark IP-852, or web services (OPC).

All these protocols are simultaneously available. It is possible to integrate the L-ROC Room Controller in a BACnet Operator Workstation and at the same time L-ROC will communicate with other CEA-709 devices on the IP-852 channel. Moreover, a higher-level SCADA or ERP System (Facility Management) gets information directly from the L-ROC Room Controller by using web services based on OPC XML-DA or OPC UA.

Full LWEB-900 Support

The L-WEB System uses web services to communicate with the L-ROC System. All device and operating parameters of every single L-ROC Room Controller are automatically synchronized with the LWEB-900 SQL database. Controllers can be replaced from the database with a backup without user interaction.

I/O Integration via Plug and Play

The L-ROC Room Controllers can automatically integrate additional physical I/Os by using L-IOB I/O Modules. Up to 2 L-IOB I/O Modules can be connected through LIOB-IP. All I/Os can be used by the L-ROC application and are also available via the web interface of L-ROC. All configurations of the L-IOB Modules are stored on the L-ROC and loaded on demand into the L-IOB I/O Modules. Exchanging I/O modules is done without any configuration effort requiring only a few quick configuration steps.

Application scenarios

The L-ROC system is used for:

- Room Automation
- HVAC control
- Light control
- Sunblind control
- Room operation panels
- Energy monitoring
- Security and access

1.2 L-ROC Models

The L-ROC hardware models cover different usage scenarios:

LROC-102: This DIN-rail mountable device can be extended by L-IOB models to create a customized I/O interface. It is used when the I/O configuration does not fit one of the LROC-400 models. Another recommended scenario is to provide a dedicated controller for building and floor management which can be installed into a switching cabinet.

LROC-400: This is the base model for a 8-segment controller. It is intended to control up to 8 windows axis. It is mounted near the controlled area in the ceiling or in the floor.

LROC-401: This variant is used when only intelligent actuators/sensors (DALI, SMI, MPBUS, KNX) are used.

LROC-402: This variant is used for HVAC applications when lighting and sunblind is implemented outside the LROC system.

LDALI-PLC4: This model is actually a programmable DALI controller with 4 channels. It is compatible with the L-ROC system and can be used for light-heavy applications, like entry halls, conference rooms or hallways.

This section provides an overview of the different L-ROC models in Table 1. This table identifies the different features of those models. Models that possess a certain feature have a check mark (✓) in the respective column. If a feature is not available in the particular model, the column is left blank.

L-ROC models with the router option have a CEA-709 router or a BACnet router, respectively. The LROC-102 has both a CEA-709 router between FT and IP and a BACnet router between MS/TP and IP.

Model / Features	LROC-102	LROC-400	LROC-401	LROC-402	LDALI-PLC4
CEA-709 Router	✓				
CEA-709 RNI	✓				
CEA-709 (FT)	✓				
CEA-852 (IP)	✓	✓	✓	✓	✓
BACnet Router	✓	✓	✓	✓	✓
BACnet MS/TP	✓	✓[3]	✓[3]	✓[3]	✓[3]
BACnet IP	✓	✓	✓	✓	✓
BBMD	✓	✓	✓	✓	✓
Modbus RTU	✓	✓[3]	✓[3]	✓[3]	✓[3]
Modbus IP	✓	✓	✓	✓	✓
L-STAT interface		✓	✓	✓	
M-Bus	✓[4]	✓[4]	✓[4]	✓[4]	
DALI channels		**1**	**1**		**4**
SMI	✓[4]	✓	✓		✓[4]
KNX TP1	✓[4]	✓	✓	✓	
KNX IP	✓	✓	✓	✓	✓
EnOcean	✓[5]	✓	✓	✓	✓[5]
OPC XML-DA	✓	✓	✓	✓	✓
OPC UA	✓	✓	✓	✓	✓
OPC Client	✓	✓	✓	✓	✓
SNMP	✓	✓	✓	✓	✓
PLC (IEC 61499)	✓	✓	✓	✓	✓
PLC (IEC 61131, planned)					✓
LIOB Connect	✓				
LIOB FT	✓				
LIOB IP	✓	✓	✓	✓	✓
LCD Display	✓	✓	✓	✓	✓
USB	✓	✓	✓	✓	✓
Ethernet Switch/Hub	✓	✓	✓	✓	✓
WLAN	✓[5]	✓[5]	✓[5]	✓[5]	✓[5]
SSH, HTTPS, Firewall	✓	✓	✓	✓	✓

1 This model can be configured to have either FT or IP active for CEA-709.

Model / Features	LROC-102	LROC-400	LROC-401	LROC-402	LDALI-PLC4

2 This model can be configured to have either MS/TP or IP active for BACnet.
3 Modbus RTU can only be used, if BACnet MS/TP is not active on this model.
4 M-Bus, SMI and KNX TP1 can be used alternatively only on this model. To operate these protocols an expansion module is needed and must be ordered separately. LSMI-804 can be connected independently.
5 To operate these protocols an expansion module is needed and must be ordered separately.
6 These models require a separate license that needs to be purchased to use this feature.

Table 1: Available features in L-ROC models.

On L-INX models without the router option and on all L-GATE models, certain ports can only be used alternatively. On models with CEA-709 this means either as CEA-709 FT or as CEA-852 IP (see note 1 in Table 1 On models with BACnet this means either as BACnet MS/TP or as BACnet IP (see note 2 in Table 1). Some BACnet models have a restriction on Modbus RTU and BACnet MS/TP as they share the same port. On those models Modbus RTU can only be used, if BACnet MS/TP is disabled (see note 3 in Table 1).

1.3 Scope

This document covers L-ROC devices with firmware version 6.4 or later and describes specific functions of those device models. Basic device operations are covered in the LOYTEC Device User Manual [1] and data point configuration is covered by the L-INX Configurator User Manual [2].

2 What's New in L-ROC

2.1 New in L-ROC 2.1.16

The SR06LCD control panels now support manual occupancy.

2.2 New in L-ROC 2.1.14

This version does not provide functional changes. The manual now contains documentation on year shade progression modelling.

2.3 New in L-ROC 2.1.12

A new controller for motorized windows has been added. It support night purge applications, manual operation and various weather and operational alarms

The sunblind module can now use the multi-façade weather station blocks. Every sunblind module can be assigned one of 16 façade directions per façade group. This allows to configure sunblind modules in corner segments to react to different wind levels.

The weather station can be extended by multi-façade modules for wind, sun and solar irradiation. This allows to implement weather or solar models to distribute calculated sensor data to different façade directions. Every façade group supports up to 16 independent values. For complex applications, multiple façade groups can be created.

The CLC can now be enabled/disabled temporarily. The light module can now be used without ballasts in the first light module.

The HVAC module now aggregates air flows and calculates the air balance for central monitoring. Additionally, it delivers the maximum damper position for optimizing an air handling unit.

The sensor blocks now automatically adapt to the SensorHub block. If the sensor hub is present, sensor data is sent to the sensor hub, else it is sent directly to the controllers.

2.4 New in L-ROC 2.1.10

Support for LDALI-BM2 has been added. Buttons can be programmed to trigger light and sunblind operation functions.

The coRoutedAO block now support HEATCOOL for 2-pipe systems.

HMI BACnet string datapoints are now not-commandable, as this is not supported by the firmware.

L-STAT models can be extended with a CO2 color addon that adjusts the display color based on the CO2 level.

Sunblind HMI is simplified, so that rotation and position can be ignored by SET_STATE which makes creating position and rotation controls easier.

The wall feedback calculation has been improved so that it also works for non-adjacent rooms. Furthermore, first and last segments as well as segment/room count is available for easier debugging.

2.5 New in L-ROC 2.1.8

The weather station blocks received their own trends. The standard segment in the start solution now does not contain the weather blocks anymore to prevent recording weather data multiple times in every L-ROC.

The weather station blocks received a bi-directional BACnet interface. This means that an OWS read the weather data from the L-ROC system or could provide external weather data to the L-ROC system.

The light switch block now also allows DALI-based dimming which removes the visible lamp value steps during dimming.

2.6 New in L-ROC 2.1.7

A block for sending central fire alarms has been added.

2.7 New in L-ROC 2.1.6

2.7.1 Support for LDALI-MS2

The LROC library now supports the LDALI-MS2 DALI Multisensor. It supports occupancy, lux value, temperature and humidity.

Temperature and humidity can be used to calculate the dewpoint temperature at the ceiling and is then used to control supply cool temperature or can close the cool valve.

2.7.2 CLC Improvements

The lighting controller now support manual-only modes. The CLC needs to be activated manually then.

The CLC now also supports a fixed algorithm for special lighting situations.

The CLC now can be enabled/disabled from the HMI. It also supports a leave mode where the CLC is disabled for a configurable time period. If no occupancy is detected at the end of the leave time, it stays off.

2.7.3 Sunblind Improvements

The sunblind shading controller now allows to select different hold times for glare start and glare stop conditions to lower the amount of sunblind movements.

2.8 New in L-ROC 2.1.5

2.8.1 Year Shade Progression

The Year Shade Progression (YSP) feature allows to calculate shading based on a 3D model. The LROC uses a 3D model in DXF format to calculate shading for every sunblind actuator or sunblind zone. Complex shading scenarios with multiple buildings in tightly occupied areas can be easily coped with.

The Year Shade Progession function requires LROC/LINX firmware 6.4.0 or later.

2.9 New in L-ROC 2.1.4

2.9.1 General Changes

The versioning of this manual has been changed to be identical to the L-ROC library version.

2.9.2 Routed IO

The L-ROC library now supports the Routed IO feature which allows to determine the mapping between HVAC, light and sunblind controllers and their physical I/Os using parameterization. This new feature is described in section 4.2.

2.9.3 HVAC

The optimum start function has been improved by allowing it to be disabled and/or time-limited separately for heating and cooling.

2.9.4 Lighting

The clRemoteChan logic block allows to integrate L-DALI group objects via OPC.

2.9.5 New Graphics Design

Two designs (LDesignDark, LDesignLight) have been added to the HVAC, Lighting and Sunblind symbols.

2.10 New in L-ROC 6.2.0 (2.1.2)

This section describes the major changes and new features. For a full list of changes refer to the Readme file.

This is the first release of the manual, so this section is empty.

3 Hardware Installation

3.1 Common for all models

3.1.1 Enclosure

The enclosure of the product and its terminal layout are shown on the installation sheet found in the product's box.

3.1.2 Product Label

The product label on the side of device contains the following information (for an example see Figure 1):

- Device order number with bar-code (e.g. LROC-102),
- serial number with bar-code (Ser#).

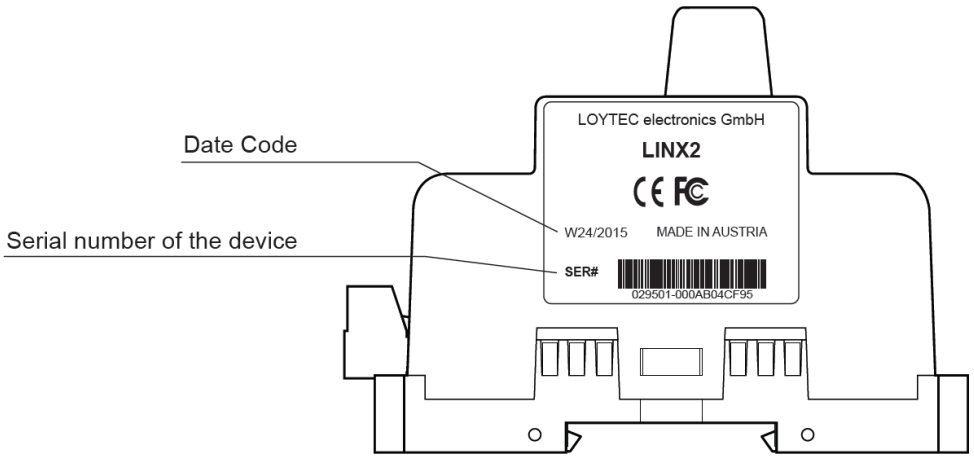


Figure 1: Example product label.

Unless stated otherwise, all bar codes are encoded using "Code 128". An additional label is also supplied with the device for documentation purposes. The specific contents of the product label are shown on the installation sheet found in the product's box.

3.2 DIN-Rail models (LROC-10x)

3.2.1 Mounting

The device comes prepared for mounting on DIN rails following DIN EN 50 022. The device can be mounted in any position. However, an installation place with proper airflow must be selected to ensure that the device's temperature does not exceed the specified range (see Chapter 14).

3.3 Ceiling/Floor models (LROC-40x)

3.3.1 Mounting

The device is intended to be mounted in the L-ROC mounting box LBOX-ROC1 or LBOX-ROC2. The device can be mounted in any position. However, an installation place with proper airflow must be selected to ensure that the device's temperature does not exceed the specified range (see Chapter 14).

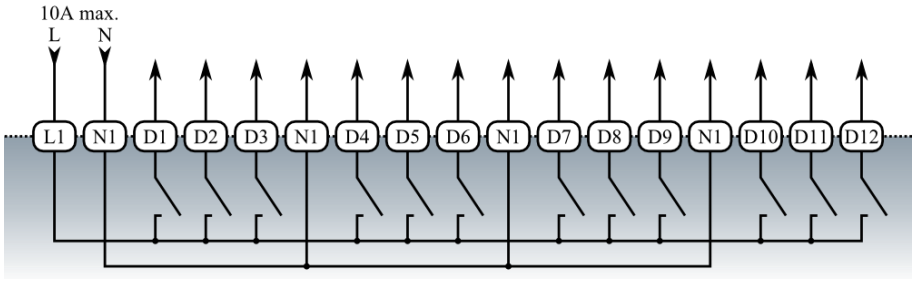
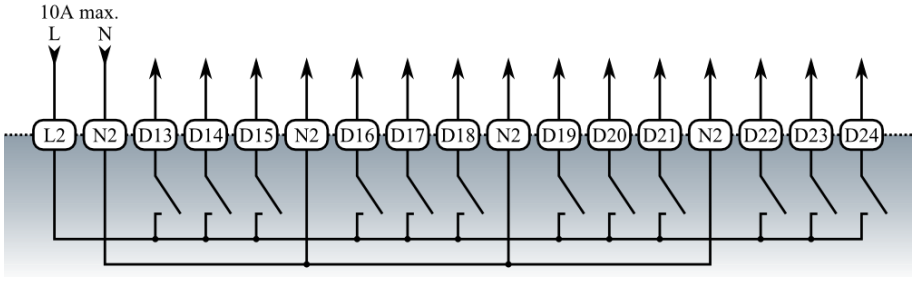
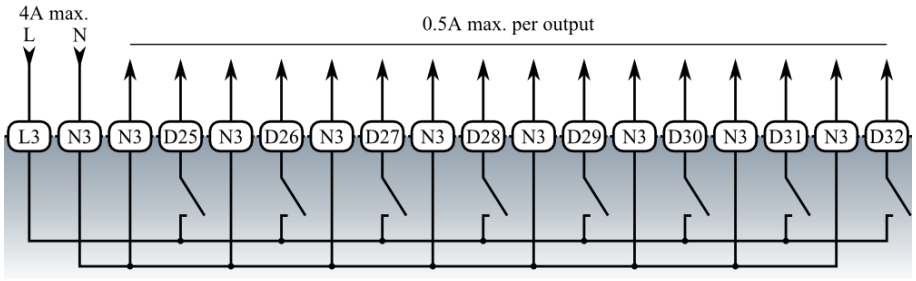
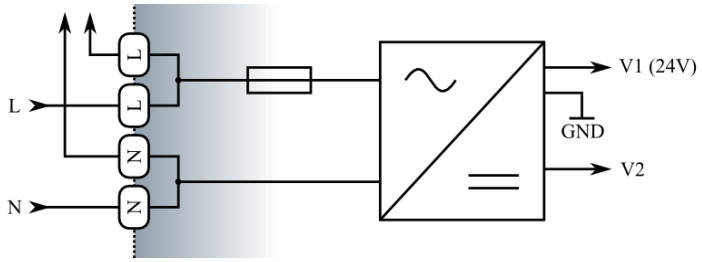
3.3.2 Wiring

The recommended wiring of the terminals is shown in Table 2.

Warning: The LROC-40x labels indicate 24V AC/DC which is wrong. The LROX-40x models can only be operated with 24V DC.

DALI and SMI are only available when using mains power.

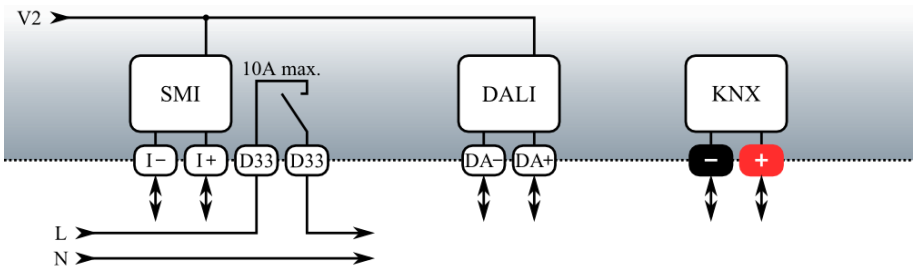
Mains power and 24V DC supply must not be used together.

Digital Output L1 (Relay)	
Digital Output L2 (Relay)	
Digital Output L3 (Triac)	
ⓘ There is no interconnection between L1, L2 and L3 or N1, N2 and N3 inside the device!	
Power Supply Mains	

SMI

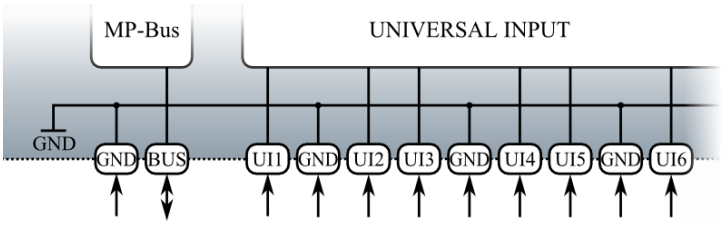
DALI

KNX



MP-Bus

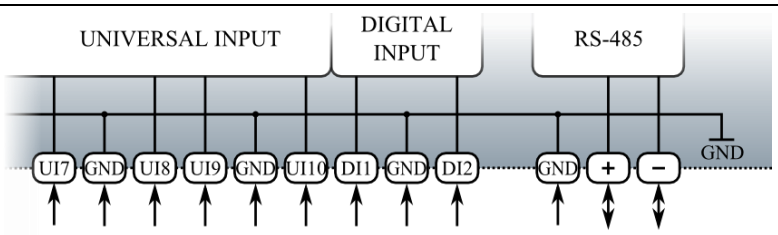
UI1-UI6



UI7-UI10

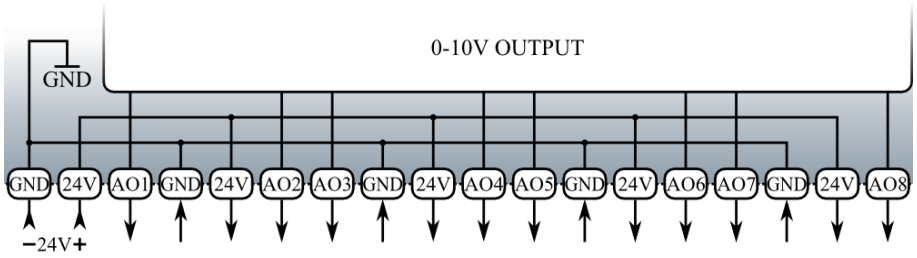
DI1-DI2

RS-485



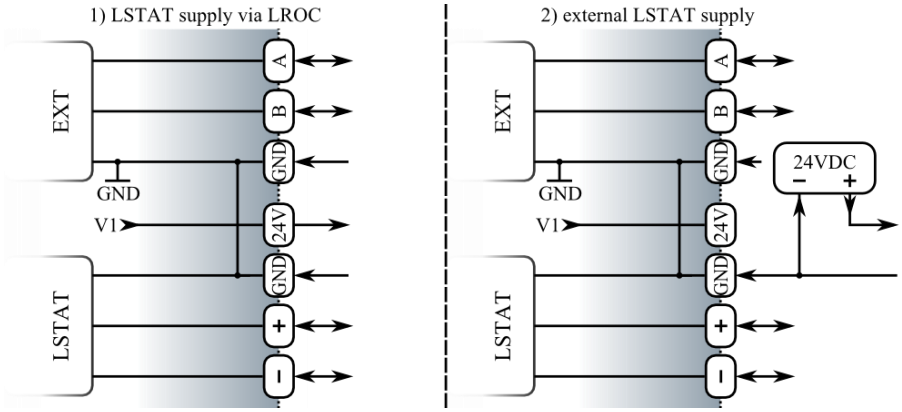
24VAC/DC Input

AO1-AO8



LSTAT

EXT



ⓘ For option 1) the guaranteed LSTAT suppy power must be calculated according to Table 98.

Ethernet Daisy Chain

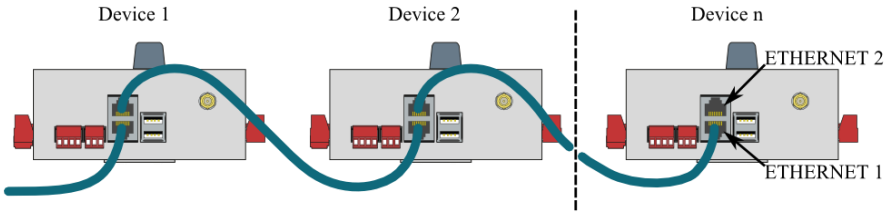


Table 2: LROC-400/401/402 Wiring Diagrams

3.4 LED signals

Available LEDs and their location on the respective device model can be found on the product's installation sheet. The installation sheet can be found in the product's box.

Device models can have different sets of LEDs. If an LED is present, it behaves like described in the next sections.

3.4.1 Power LED

The power LED lights up green when power is supplied to the power terminals.

3.4.2 Status LED

The device is equipped with a red status LED (see product installation sheet). This LED is normally off. During boot-up the status LED is used to signal error conditions (red). If the fall-back image is executed the status LED flashes red once every second.

3.4.3 OPC LED

The OPC Server LED illuminates green when at least one OPC client is connected to the OPC server. The LED flickers on OPC XML-DA traffic activity.

3.4.4 PLC LED

The three-color PLC LED indicates the state of the PLC kernel and the PLC program (see product installation sheet). Table 3 shows the different LED patterns and their meaning. This LED is not available on the L-INX models 10X, 20X. On L-INX models with a green LED (12X, 15X, 22X) an alternative pattern is used.

Behavior	Description	Comment
GREEN permanent	The PLC program has been stopped	The PLC kernel is running, there is an PLC program, but the program execution was stopped.
GREEN flashing slow at 1 Hz	Normal condition, PLC program running	The PLC kernel and program were successfully loaded. The PLC program is executed.
ORANGE or GREEN flashing fast in an on-on-pause pattern	I/O driver disabled	The I/O driver is disabled, that is that no updates from or the PLC program were handled.
RED or GREEN flashing fast	CPU overload	CPU load exceeded 80%. Modify the PLC program to reduce CPU load in order to guarantee normal system operation (for example, reduce the cycle time of the program).
OFF	No PLC program	No PLc program is loaded or a problem occurred while starting the PLC kernel.

Table 3: PLC LED Patterns

3.4.5 FT Activity LED

The FT port on the device has a three-color LED (green, red, and orange, see product installation sheet). Table 4 shows different LED patterns of the port and their meaning.

Behavior	Description	Comment
GREEN flashing fast	Traffic	
GREEN flashing at 1Hz	The OPC node or LINX-101's router port is unconfigured	On the LINX-101 this LED only stops flashing if both, node and router, are commissioned.
RED permanent	Port damaged	
RED flashing fast	Traffic with high amount of errors	
RED flashing at 1 Hz (all ports)	Firmware image corrupt	Please upload new firmware.
ORANGE permanent	Port disabled	e.g., using LSD Tool
ORANGE flashing fast	Traffic on port configured as management port	e.g., using LSD Tool

Table 4: CEA-709 Activity LED Patterns.

3.4.6 MSTP Activity LED

The MS/TP port has a three-color MSTP Activity LED (see product installation sheet). Table 5 shows the different LED patterns of the port and their meaning. A permanent color reflects a state. Flicker is for 25 ms when there is activity on the MS/TP data link layer.

Behavior	Description	Comment
GREEN permanently, flicker off	Multi-Master, token ok, flicker when traffic	Normal condition on a multi-master MS/TP network.
ORANGE flicker	Sole master, flicker when traffic	Normal condition on a single-master MS/TP network.
RED permanent, flicker GREEN	Token lost state, flicker when transmit attempt	Cable might be broken.
RED flash fast	Transmission or receive errors	This indicates bad cabling.

Table 5: MS/TP Activity LED Patterns.

3.4.7 Modbus LED

The Modbus LED is a three-color LED. Table 6 shows the different LED patterns of the port and their meaning. Flicker is for 25 ms when there is activity on the bus.

Behavior	Description	Comment
GREEN flicker	Traffic, transmission successful	Normal condition.
RED flicker	Traffic, transmission unsusccesful	Device not responding or other network error.

Table 6: Modbus Activity LED Patterns.

3.4.8 EXT LED

The EXT port has a three-color LED (see product installation sheet), which displays link and traffic information on the protocol enabled on the port (M-Bus or KNX TP1). Table 7 shows the different LED patterns of the port and their meaning. A permanent color reflects a state. Flicker is for 25 ms when there is activity on the attached network. The LGATE-951 has two EXT ports that conform both to the described behavior.

Behavior	Description	Comment
GREEN permanently, flicker off	LKNX-300 attached, flicker when traffic	Normal condition on a KNX TP1 network.
GREEN flash fast	M-Bus traffic	Normal condition on an M-Bus network.
RED permanent	LKNX-300 not attached or port damaged	Check if cabling to LKNX-300 is bad.
RED flash fast	M-Bus transmission or receive errors	This indicates bad cabling. Check if L-MBUS is connected. Check if M-Bus devices are operational.

Table 7: EXT port LED Patterns.

3.4.9 Ethernet Link LED

The Ethernet Link LED lights up green whenever an Ethernet cable is plugged-in and a physical connection with a switch, hub, or PC can be established.

3.4.10 Ethernet Activity LED

The Ethernet Activity LED flashes green whenever a packet is transmitted or received or when a collision is detected on the network cable.

3.4.11 Wink Action

If the CEA-709 device receives a wink command on any of its network ports, it shows a blink pattern on the CNIP and the CEA-709 activity LEDs. The CEA-709 activity and the CNIP LED turn green/orange/red (each 0.15 s). This pattern is repeated six times. After that, the CNIP LED flashes orange six times if the wink command was received on the IP channel or the CEA-709 activity LED flashes orange six times if the wink command was received on the CEA-709 channel. After that the device's LEDs resume their normal behavior.

3.4.12 Network Diagnostics

The CEA-709 device provides simple network diagnostics via its CEA-709 activity LED:

If the LED does not light up at all, this port is not connected to any network segment or the connected network segment currently shows no traffic.

If the LED is flashing green, the network segment connected to this port is ok.

If the LED is flashing red, a potential problem exists on the network segment connected to this port. This state is referred to as overload condition.

A port overload condition occurs if

- the average bandwidth utilization of this port was higher than 70 %, or
- the collision rate was higher than 5 %, or
- more than 15 % CRC errors have occurred on a port with a power-line transceiver, or more than 5 % on a port with a transceiver other than power-line, or
- the device was not able to process all available messages.

For a deeper analysis of the reason for the overload condition, it is recommended to use a protocol analyzer (e.g., LOYTEC's LPA) or a similar tool. The exact reason of the overload condition can also be determined with the LSD Tool.

3.5 Dial Button

The dial button can be used to switch the device back to factory default state. Press the dial button and power-cycle the device. Keep the button pressed until the port LEDs illuminate orange permanently. Release the button within five seconds from that time on to reset the device to factory defaults. Alternatively, the device can be switched back to factory defaults over the LCD display.

4 L-ROC Concepts

4.1 General Concepts

4.1.1 Introduction

The L-ROC library implements an integrative room automation solution covering the following application areas:

- HVAC (Heating, Ventilation, Air Conditioning)
- Lighting
- Sunblind
- Security
- Fire Damper Monitoring (Addon)

The L-ROC systems supports flexible room, dynamic zone assignment, central function interfaces with a unique communication system which is presented throughout this chapter.

The design on an L-ROC application is split into different hierarchy levels which are chosen to support reusing common elements like floors or areas:

- The building level is the top-level of the LROC-system.
- A building consists of one or more floors.
- A floor consists of one or more areas. Areas are typically limited by tenant areas or fire protection zones.
- An area consists of one or more room segments.
- A room segment contains the sensor interfaces and one or more control modules for each trade, for example, 1 HVAC controller, 2 sunblind controllers and 2 light controllers
- Each control module can control one or more actuators

The following functions are provided on the building, floor and area level:

- Weather station (typically building)
- Heating/Cooling system interface (changeover, statistics)
- HVAC, light and sunblind central overrides
- Security zones

There are two possible layouts for an LROC-Project

- A high building (sky scraper style) with a low floor area is used in the regular way.
- Wide-spread buildings (campus style) with a large floor area but low floor count can be used with floors and areas swapped.

Figure 2 shows the type layouts. The skyscraper building has four floors with 4 areas each. In the campus style building the floor levels are used for the building parts, and the area levels are used for the floors.

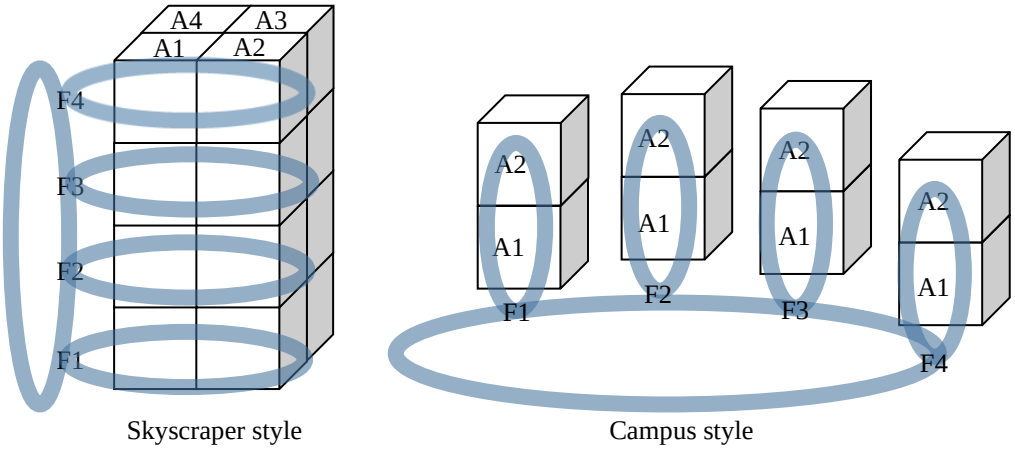


Figure 2: L-ROC projects layouts

Table 8 shows the communication limits of an L-ROC project. Exceeding these limits might result in performance degradation, communication timeouts or application failure.

Property	Limit
Maximum programmable devices in one L-STUDIO project	400
Maximum floors per building	20
Maximum areas per floor	20
Maximum segments per area	100
Maximum control modules per segment[1]	10
Maximum control modules per area[2]	400
Maximum segments per room	100
Maximum aggregation blocks per area/floor/building	16
Maximum L-ROC devices per area	20
Maximum area/floor/building couplers per device	20
Maximum segment couplers in LROC-102	16
Maximum segment couplers in LROC-400	8
Maximum segment couplers in LROC-401	16
Maximum segment couplers in LROC-402	8
Additional segments per device with LROC_SEG8 license	8

Table 8: Communication limits.

4.1.2 Building (Top Level)

The building level is the top-most level in the L-ROC hierarchy. It is represented by the `BuildingCoupler` block within the application. The building coupler has the following functions:

- Establish and monitor communication to the floor couplers
- Weather station interface
- Interfaces to supply systems (heating, cooling, metering, ...)
- Central building overrides
- Statistics

The building coupler provides a building ID parameter. It should be set to a short form of the building project, e.g. `"LT"` for LOYTEC Tower.

The following guidelines should be checked for each project:

Dedicated hardware for building manager: As the building coupler provides central data to the entire L-ROC system, it should be implemented on a separate hardware and should be powered independently. A hardware running the building coupler is called the building manager. While the LROC system works without central data, some alarm functions (e.g. wind data for sunblinds) would assume that the alarm is active for safety reasons. Therefore it is important to ensure that power supply and communication to the building manager is as reliable as possible.

Uninterruptable power supply: The building manager should be powered by an independent and battery-backed power supply.

[1] A control module is a standard, unmodified HVAC, light, sunblind or security control module from the L-ROC library. If the control module has been modified, application performance needs to be evaluated before deployment.

[2] Given that the zone IDs are single-digit letters (A-Z).

Monitoring: The building manager should be monitored in the BMS with high priority. Communication loss to the building manager needs to be handled as soon as possible.

4.1.3 Floors

The floor is the second level in the L-ROC hierarchy. It is represented by the `FloorCoupler` block within the application. The floor coupler has the following functions:

- Establish and monitor communication to the building coupler
- Establish and monitor communication to the area couplers
- Interfaces to supply systems (heating, cooling, metering, ...)
- Central floor overrides
- Statistics

The floor coupler provides a floor ID parameter. It should be set to a short string containing the floor number, e.g. `"F01"` for Floor 1.

Hint: *If you have 10 floors ore more, use leading zeroes to make lists containing the floor ID look more uniform.*

The following guidelines should be checked for each project:

Floor coupler placement: A floor coupler can be implemented in several ways, each having their own advantages and disadvantages, as listed in Table 9.

Coupler location	**Pros**	**Cons**
Own hardware	Independent on tenant area power supply Floor Manager can be used to implement area managers Can use other hardware models	More devices
Building manager	Less devices Floor communication is internal and not dependent on external components	Not suited for for than 10 floors Loss of floor functions on building manager failure
Normal L-ROC	Less devices	More L-ROC types in project Floor communication fails on outage of a tenant area Interdependency of tenant areas

Table 9: Floor coupler locations.

Uninterruptable power supply: The floor manager should be powered by an independent and battery-backed power supply.

Monitoring: The floor managers should be monitored in the BMS with high priority. Communication loss to the floor managers needs to be handled as soon as possible. The L-ROC ring system will continue to work if one floor managers fails. The failure of two floor managers will result in communication loss to the floors between the two failing floor managers.

4.1.4 Areas

The area is the third level in the L-ROC hierarchy. It is represented by the `AreaCoupler` block within the application. The area coupler has the following functions:

- Establish and monitor communication to the floor coupler
- Establish and monitor communication to the segment couplers

- Interfaces to supply systems (heating, cooling, metering, ...)
- Central area overrides
- Statistics

The area coupler provides a area ID parameter. It should be set to a short string containing the area number, e.g. "A01" for Floor 1.

Hint: *If you have 10 areas or more, use a leading zero to make lists containing the area ID look more uniform.*

The following guidelines should be checked for each project:

Area coupler placement: An area coupler can be implemented in several ways, each having their own advantages and disadvantages, as listed in Table 9.

Coupler location	**Pros**	**Cons**
Own hardware	Independent on tenant area power supply Can use other hardware models	More devices
Floor or building manager	Less devices Area communication is internal and not dependent on external components	Not suited for for than 10 areas Loss of area functions on floor/building manager failure
Normal L-ROC	Less devices	More L-ROC types in project Area communication fails on outage of a specific L-ROC

Table 10: Floor coupler locations.

Uninterruptable power supply: The area manager should be powered by an independent and battery-backed power supply.

Monitoring: The area managers should be monitored in the BMS with high priority. Communication loss to the area managers needs to be handled as soon as possible. The L-ROC ring system will continue to work if one area managers fails. The failure of two areas managers will result in communication loss to the areas between the two failing area managers.

4.1.5 Rooms

A room is a set of room segments which can be created by parameterization during runtime. Therefore, there is no room object in the logic program. The `idRoom` parameter of the segment coupler defines to which room the room segment belongs. It should be set to a short string defining the room designation, e.g. `100`, `R100` or `E1-100`. Choosing a long name can limit the number of supported segments per area.

Note: Room names can contain the following characters: `0-9, a-z, A-Z, - _ /` *The following characters are not allowed:* `. { }`

Rooms can be formed within an area and can include segments on different L-ROC devices and can be discontiguous. It is however recommended to place adjacent room segments to adjacent places in the L-ROC segment chain, as control messages have to pass the L-ROC devices in between.

Rooms can be reconfigured at runtime. However, it can take a couple of seconds for the zones to reform and share sensor and control data. A room is always included all trades (in contrast to a zone). Therefore a room segment can be assigned only one room ID, while it can be part of many zones.

Room segments can have a flexible wall at their right side in the segment ring. The wall can be switch on/off by a favorite data point. In case of a flexible room, then suffix `_{N}` is added to the room number, where {N} is an increasing number starting from zero. When using flexible wall, avoid having room IDs ending with `_0`, `_1`, `_2` and so on.

4.1.6 Zones

While rooms collect room segments to a single entity, zones can split a room into smaller parts or can also span several rooms.

Zones are defined by the zone controllers, for example HVAC, Sunblind, and Lighting. Every controller module has its own zone ID, so that different zones can be formed for each trade. The zone namespaces are distinct for each trade, so the HVAC, Lighting and Sunblind zones can all be `A` without interfering each other. This is also the default zone ID.

Zone IDs should be kept short, best one or two letters or digits.

There are two zone types which can be configured:

- *Room zones*: These zones are automatically limited by the room containing the zone. The names can be kept short, e.g. `A`. Therefore, zones with the same names in different rooms are distinct zones. Room zones must not start with a `*` or `#` character.
- *Area zones*: These zones are not limited by room borders. Area zone names start with a `*` or `#` character. Their IDs need to be distinct within an area. An application of an area zone would be an open space office. It would use area zones for HVAC but room zones for lighting. Area zone ID also need to be kept as short as possible.

Figure 3 shows an example using area and room zones. The HVAC module uses the zone ID `#HV1` and spans the rooms 100 to 102. In Room 100, the light zones are split into `A` and `B`, while the sunblinds are joined into zone `A`. Rooms 101 and Rooms 102 have use both `A` for lighting and sunblinds, but as these are room zones, these don't mix up.

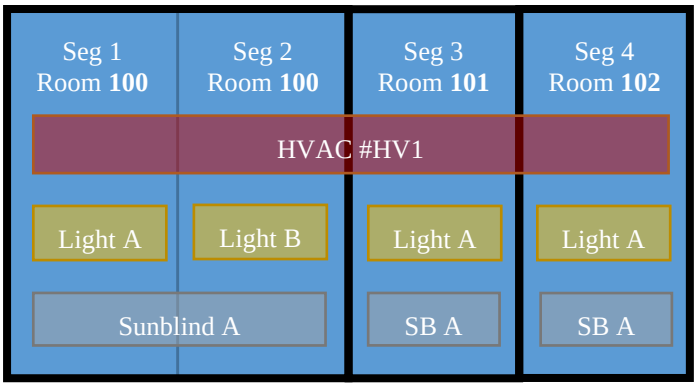


Figure 3: Zoning example.

Zone IDs can be reconfigured at runtime. However, it can take a couple of seconds for the zones to reform and share sensor and control data.

4.1.7 Room Segments

A room segment is the basic building block of the L-ROC system. It contains the controllers for HVAC, sunblinds and lighting, as well as the sensor integration.

The core block of a room segment is the `SegmentCoupler` block. It is responsible for connecting the segments to each other and to the area coupler. It also conducts the forming of zones and the addressing between the controllers within the segment.

A room segment is identified by its segment ID, which should be an integer value. It needs to be unique within the area.

A room segment consists of the `SegmentCoupler` block, the sensor blocks and the controller blocks.

The room segment defines the room functions, sensor equipment, available room operation panels and HMI devices. Most of the engineering effort in an L-ROC project will be done on room segment level.

A room segment can be defined in different ways:

- *Windows axis*: In this case, the room segment is similar to the smallest divisible room element. This is the most flexible approach.
- *Partial*: In some projects, a windows axis is not suitable, for example if HVAC and sunblind planning results in different segmentation.
- *Entire Room*: A room segment is identical to a room. This fits projects without flexibility requirements.

When planning an L-ROC project, listing the room segment types is one of the first steps. This list results in the number of room segment types and the number of instances of each type.

An L-ROC project should contain a small set of room segment types. If the L-ROC device types all can use the same segment types, all engineering effort collapses into a few L-ROC device types.

To simplify engineering, the following guidelines should be followed:

- *Unify similar segment types*: If two room segment types appear frequently and they differ only in the number of light or sunblind zones, there should be only one type with the maximum of zone controllers. Those segments with less zones can leave the additional zones unused.
- *Include optional modules*: If some room segment types have additional modules, e.g. a fan coil unit, consider including it in a common segment type. It can remain uncommissioned in segments lacking this unit.
- *Keep segment types uniform*: If there are special units that are only mounted rarely, they should be either implemented in extra segments or on the device type level. It is better to keep the standard segment types as reusable as possible to reduce the number of needed device types.

Actuators can be added to the controller block or on the segment level. The following guidelines should be used for choosing the right way:

- *Add actuator to zone controller*: If the actuator is used in most of the zones, add it to the controller and do not use the actuator in the instances where it is not present
- *Add actuator to segment type*: If the actuator is used rarely, add it to the segment types where is could occur. If the actuator is used very rarely, create a new segment type which is only used in the rare occurrence.

Sensors that are optionally not mounted, should be added to the segment type in any case. They consume very little resources, so they can be left uncommissioned when they are not equipped in a room segment.

Some typical building constellations require one of the following patterns and segment assignments

Corner segments: These segments are situated at the corner of a building and need to be modelled by two segments with different orientations. There are two special requirements for corner segments:

- Typically there are twice as much sunblinds
- Additional light bands and sunblind controls on the RCD.

The first approach for corner segments is shown in Figure 4. The segments 3-6 overlap in a rotated way. This allows to assign all sunblinds and to create four different light zones. The valves can be connected as needed and left open if there are less physical valves than controller outputs. The light assignments are made in a way allowing the windows bands to be faded in CLC mode. A parameterization example for a corner segment is found in Table 11. Table 12 shows some possible light zones of a corner segment.

Of course a corner segment can also be designed as a separate segment type. This however increases the number of device types and the number of HMI projects to be created. The increased engineering effort must be compared to the slight overhead of the approach presented above.

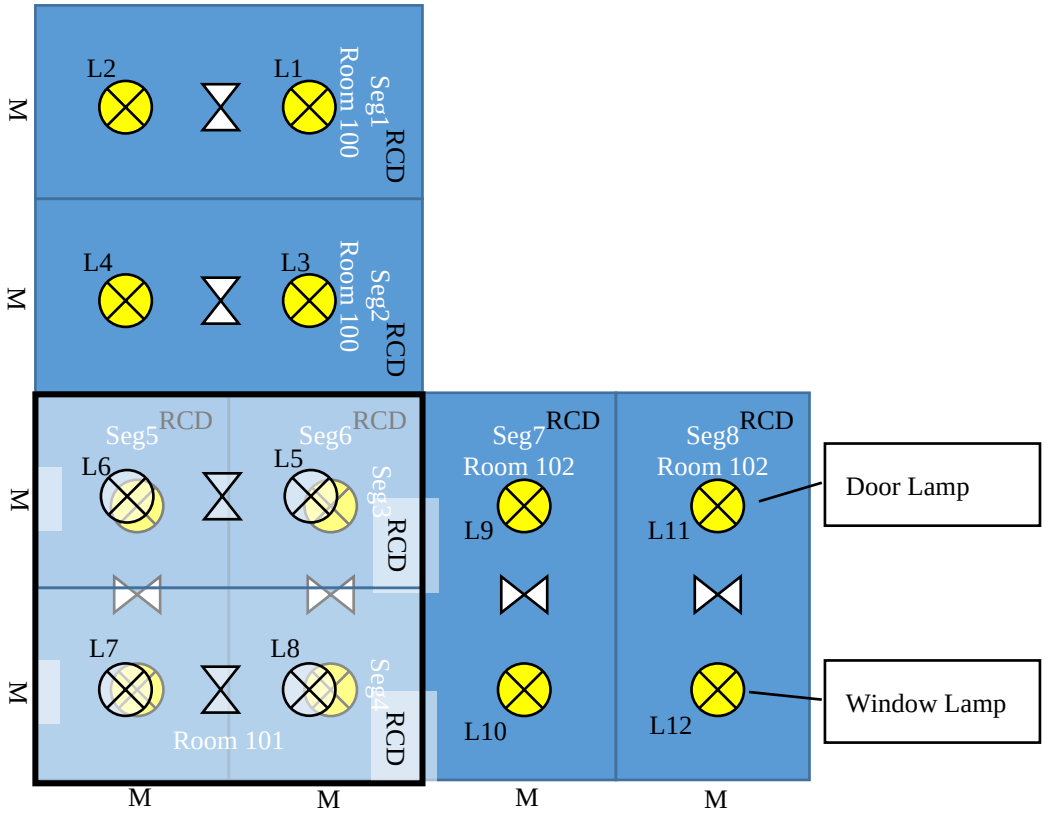


Figure 4: Corner segments using overlapping standard segments.

Segment	1	2	**3**	**4**	**5**	**6**	7	8
Room ID	100	100	**101**	**101**	**101**	**101**	102	102
HVAC zone	A	A	**A**	**A**	**A**	**A**	A	A

Segment	1	2	**3**	**4**	**5**	**6**	7	8
Orientation[3]	W	W	**W**	**W**	**S**	**S**	S	S
Light zone	A	A	**A**	**B**	**C**	**D**	A	A
Sunblind zone	A	A	**A**	**A**	**B**	**B**	A	A
Valve assignment	✓	✓	✓	✖	✓	✖	✓	✓
Door Lamp	L1	L3	**L5**	✖	✖	✖	L9	L11
Windows Lamp	L2	L4	✖	**L7**	**L6**	**L8**	L10	L12
Sunblind assignment	✓	✓	✓	✓	✓	✓	✓	✓

Table 11: Parameterization and actuator assignment of a corner segment

Seg 3 L5	Seg 4 L7	Seg 5 L6	Seg 6 L8	Pattern	Result
A	A	A	A	A A A A	One light zone
A	B	A	B	A A B B	2 “horizontal” zones
A	B	B	A	B A B A	2 “vertical” zones
A	B	B	B	B A B B	1 large window zone + 1 small door zone
A	B	A	A	A A B A	1 large door zone + 1 small window zone
A	C	B	B	B A C B	3 diagonal zones
A	B	C	D	C A B D	4 distinct zones

Table 12: Zoning examples for a corner segment.

[3] W for West and S for South

4.1.8 Couplers & Managers

This section discusses the correct planning of couplers and managers in an L-ROC project. The model presented in this section is balanced between hardware costs and reliability. Changes to the proposed use cases must be evaluated for each individual case.

The rationale behind these use cases are:

- L-ROC communication limits are not exceeded.
- Devices with central couplers shall be powered independently from tenant areas
- Communication between floors and areas should be either within a device or implemented in a redundant Ethernet ring.
- Tenant areas should be independent of each other. Therefore common couplers should be outside tenant areas.

If couplers of different types are mixed in one device, care must be taken to use distinct IDs for each element. For example, if floor and areas are used on the same device, they must use distinct names, like `F01`, `A01`, `A02` and so on. In the case of a single manager device, the area names also need to be distict, e.g. `A13` for floor 1, area 3 and `A31` for floor 3 area 1.

4.1.8.1 Skyscraper Use Case

This model is suited for high buildings. The model assumes 20 identical floors (max) and 2 tenant areas on each floor, each having 60 room segments. A higher building should be split into separate projects.

This model results in a total of 2400 room segments and 341 (20 * (2*8 + 1) + 1) L-ROC controllers.

The building manager should be a dedicated hardware mounted in a switching cabinet, e.g. an LROC-102. It can be expanded by LIOB-Models if it needs physical interfaces with the supply systems. The weather station is also connected to the L-ROC, either via physical inputs or via communication (BACnet, MODBUS ...). The building manager should be connected directly to a central Ethernet switch.

Each floor should have its own floor manager which implements the floor coupler and the 4 area couplers. This device should be mounted outside the tenant areas in a switching cabinet. It would be well-suited for connecting to metering devices or interfaces HVAC changeover valves.

Each tenant area is equipped with 8 LROC-400, each controlling up to 8 room segments. These are connected in a ring to a managed Ethernet switch.

Variant 1: *If the aisles and general rooms shall be automated by the L-ROC system, another LROC-400 should be planned for these functions. It is not adviced to mix standard segments and special rooms to avoid overhead due to additional L-ROC device types needing their own engineering. It is better to have N standard device types and one non-standard device types than having N non-standard device types.*

Variant 2: *The area manager can be shifted into an extra L-ROC within the tenant area. This L-ROC then can also be used for aisles and general rooms. The drawback of this solution is that the tenant areas are then interdependent. If more than one tenant area is offline, also the areas in between are disconnected from the floor manager.*

Variant 3: *The floor coupler is put into the first device of a tenant area. Therefore, no extra device per floor is needed. It would reduce the necessary device count to 321 (20* (2*8) + 1) L-ROC devices. This variant is not recommended, as the entire building depends on all floors to be powered.*

Variant 4: *It is possible to use a manager device every 2 floors, including 2 floor managers and 8 area managers. This reduces the device count to 331 (20*(2*8) + 20/2 + 1) L-ROC devices, but creates an interdependency of floor pairs.*

4.1.8.2 Campus Use Case

This model is suited for wide-area buildings with central supply. This model assumes 5 areas with 8 floors each. Each floor in each area has 60 room segments. Therefore, the total is again 4800 room segments. This implementation requires 326 (= 5 * (8*8+ 1) + 1) L-ROC controllers.

Note that floor and area couplers are swapped in this use case.

The building manager should be a dedicated hardware mounted in a switching cabinet, e.g. an LROC-102. It can be expanded by LIOB-Models if it needs physical interfaces with the supply systems. The weather station is also connected to the L-ROC, either via physical inputs or via communication (BACnet, MODBUS ...). The building manager should be connected directly to a central Ethernet switch.

Each area should have its own (floor) manager which implements the floor coupler and 8 area couplers (responsible for the 8 floors). The manager should be powered independently.

Each floor in each area is equipped with 8 LROC-400 or 9 LROC-400 when general rooms are to be automated. These are connected in a ring to a managed Ethernet switch.

Variant 1: *The area couplers can also be put into each floor. Then the general functions can be put on the area manager (responsible for a floor).*

4.1.8.3 Small Building Use Case

This model is suitable if the total number of areas is 10 or less. The use case assumes that each area has 60 room segments. This implementation required 81 (10 * 8 + 1) L-ROC controllers. This model can be used up to about 1000 room segments.

The building, floor and area couplers are all put into a single device which also interfaces the supply systems and the weather station. This building manager should use a separate power circuit and should be directly connected to a central Ethernet switch.

Each area includes 8 LROCs in an Ethernet ring.

The small building use case can be shrunk or extended as needed, as long as the number of segments in an area remain within its limit.

As all managers are concentrated on one device, the coupler IDs must be made distinct, w.g. `Fx` for floor x and `Axy` for area y on floor x.

4.2 Routed IO

4.2.1 Routed IO Concept

To facilitate designs contradicting the type-based approach, the Routed IO functions can be used to:

- Reassign room functions to different zones at run time.
- Support a variable amount of actuators in a zone
- Add additional actuators at run time.
- Mix sensors of different technologies.

Using Routed IO has the following advantages:

- One can reassign a controller output (e.g. Heat%, or Lamp value) to a different I/O object of the same type.
- One can dynamically assign existing I/O objects to existing controllers.
- One can use a diffrerent number of actuators per controller without creating a new devices type.

Routed IO also has some disadvantages and limits.

- The assignments can be only done within a device. It cannot be used to attach a sensor or an actuator to a different device.
- The connection between sensor/actuator and controller has to be done using additional parameters which increases commissioning effort.
- The actuator feedback has to be manually enabled.

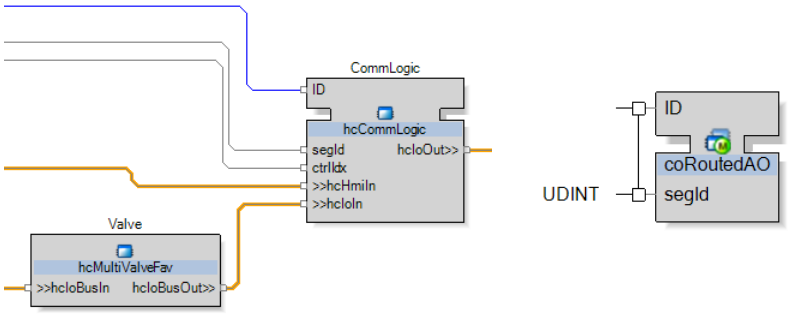


Figure 5: Connected IO vs Routed IO

Figure 5 shows the difference between conventional IO and Routed IO. In the former case, the IO logic block is directly connected to the controller. In the Routed IO case, the block can be put anywhere in the application and can be mapped to any suitable controller on the same device.

5 Central Building Functions

5.1 Weather Station

The LROC system can be interfaced to weather stations. The weather station logic provides the following features:

- Automatic distribution to all LROC devices
- Supported data points:
  - Outdoor temperature
  - Outdoor humidity
  - Outdoor dew point
  - Outdoor air pressure
  - Outdoor brightness (4 segments)
  - Solar irradiance (expandable to multiple facades)
  - Precipitation
  - Frost detection
  - Wind speed and direction (expandable to multiple facades)

The weather station is implemented in the *Loytec.Automation* library.

The weather station sender blocks are used to interface a weather station to the LROC system. The weather station blocks are placed into the building manager, as shown in Figure 6. The weather station is divided into several submodules.

The weather station modules contain Favorites and BACnet objects which are connected by a bi-directional connection. So weather data can be either provided by favorites linking to an existing analog or Modbus weather station or provided by an BACnet OWS.

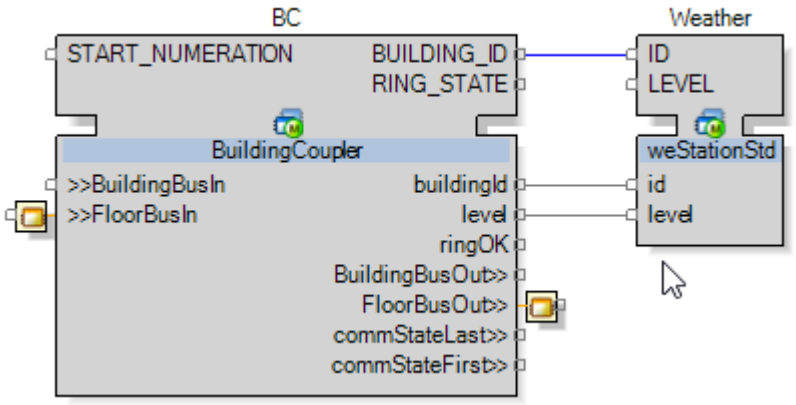


Figure 6: Weather station connection

The locic block *weWeatherStationStd* can be used for smaller projects and as a template for larger projects. It provides one instance of each *weStationAir*, *weStationSun*, *weStationWind* and *weStationPrecipitation* modules, as shown in Figure 7.

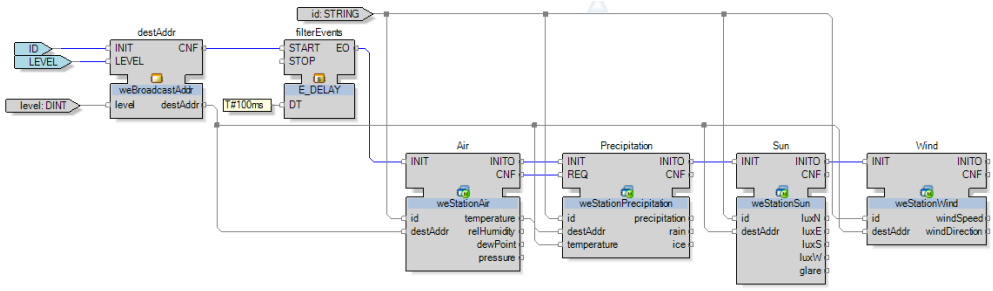


Figure 7: Standard weather station.

5.1.1 Air Data

The *weStationAir* module gathers temperature, humidity, dew point and air pressure.

Favorite	Type	Description
SensorFailure	Binary	Can be used to indicate sensor failure. Ignored if left open.
Temperature	Analog	Outdoor air temperature
RelHumidity	Analog	Outdoor relative humidity
Dewpoint	Analog	Outdoor dewpoint
Pressure	Analog	Outdoor air pressure

Table 13: Favorite Data Points of *weStationAir*

The favorite data points of the weStationAir module are shown in Table 13. If the SensorFailure input is not used, the sensors are assumed to be always operational.

Air data is distributed using the *WEA_AIR* serialize client command with a cycle of 60 seconds.

Currently, the LROC library does not process SensorFailure, RelHumidity, Dewpoint and Pressure data.

As air data is normally constant and independent of direction, there is usually only one *weStationAir* instance in the weather station.

The following functions use the data points from the air data interface:

- Temperature
    - HVAC: MinHeating, Summercompensation
    - Sunblinds: Thermal module

5.1.2 Sun Data

The *weStationSun* module gathers outdoor illumination data from a segmented brightness sensor. Ideally, there should be 4 segments, but the sun data logic works also with one to three segments.

Favorite	Type	Description
SensorFailure	Binary	Can be used to indicate sensor failure. Ignored if left open.
LuxNorth	Analog	Lux in north direction
LuxEast	Analog	Lux in east direction
LuxSouth	Analog	Lux in south direction
LuxWest	Analog	Lux in west direction
Glare	Binary	Output of glare condition

Table 14: Favorite Data Points of *weStationSun*

The favorite data points of the weStationAir module are shown in Table 14. If the SensorFailure input is not used, the sensors are assumed to be always operational.

Sun data is distributed using the *WEA_SUN* serialize client command with a cycle of 60 seconds.

Currently, the LROC library does not process SensorFailure input.

As sun data is normally constant over the building site and independent of direction, there is usally only one *weStationSun* instance in the weather station.

The weStationSun module contains a glare predictor which works on the measured segmented brightness. The calculation needs at least 2 different segments. If the minimum brightness level is greater than 500 Lux, then minimum, average and maximum Lux values over all segments are calculated. A glare condition is indicated when

- The minimum level is above 500 Lux (sanity check) AND
    - The average level is above a threshold OR
    - The quotient between the maximum and the minimum lux level is above a threshold.

The thresholds of the glare detection can be parameterized and have a hysteresis to avoid asserting the glare signal too frequently. The parameters of the glare detection are shown in Table 15.

Parameter View	Datapoint	Type	Default	Description
Glare Minimum Average Lux	weSunGlareHystMin	Real	35000	Minimum average brightness to deactivate glare signal
Glare Maximum Average Lux	weSunGlareHystMax	Real	40000	Maximum average brightness to activate glare signal
Glare Minimum Quotient	weSunGlareQuotientMin	Real	3	Minimum quotient to deactivate glare signal
Glare Maximum Quotient	weSunGlareQuotientMax	Real	4	Maximum quotient to activate glare signal

Table 15: Glare Detection Parameters

The following functions use the data points from the sun data interface:

- Lux Levels
  - Sunblinds: Shading activation
  - Fault detection

5.1.3 Precipitation Data

The *weStationPrecipitation* module receives precipitation data and generates a rain and frost alarm signal.

Favorite	Type	Description
SensorFailure	Binary	Can be used to indicate sensor failure. Ignored if left open.
PrecipitationBinary	Binary	Binary precipitation sensor
PrecipitationAnalog	Analog	Analog precipitation sensor

Table 16: Favorite Data Points of *weStationPrecipitation*

The favorite data points of the *weStationPrecipitation* module are shown in Table 16. If the SensorFailure input is not used, the sensors are assumed to be always operational. The binary and analog sensors can be used mutually. Any analog value greater than zero is considered as precipitation.

Precipitation data is distributed using the *WEA_PRECIP* serialize client command with a cycle of 60 seconds.Addtionally the commands *AL_ICE* and *AL_RAIN* are send every 15 seconds.

Currently, the LROC library does not process SensorFailure input.

As precipitation data is normally constant and independent of direction, there is usally only one *weStationPrecipitation* instance in the weather station.

The *weStationPrecipitation* module contains a frost indicator which takes outdoor temperature and precipitation into consideration. Note that the frost indicator needs to be parameterized properly and provides only an estimate.

The frost indicator works as follows: The precipitation signal is extended by the *weIcePrecipDelay* parameter. If the extended precipitation signal occurs while the outdoor temperature is below *weIceTempLimit*, the frost signal is asserted. The frost signal remains asserted until the outdoor temperature rises above *weIceTempReset*. Currently, the frost indication is not saved persistently.

The alarm delay parameters delay and extend the rain and ice alarms to lower the number of transitions.

Table 17 shows the parameters of the *weStationPrecipitation* module.

Parameter View	Datapoint	Type	Default	Description
Ice Temp Limit	weIceTempLimit	Real	3°C	Temperature below which frost alarm is asserted upon precipitation
Ice Temp Reset	weIceTempReset	Real	6 °C	Temperature above which frost alarm is released.
Ice Temp Delay	weIcePrecipDelay	Real	12h	Delay for extending the precipitation delay
Rain Alarm Delay On	weRainAlarmDelayOn	Real	1 s	Time after precipitation to rain alarm
Rain Larm Delay Off	weRainAlarmDelayOff	Real	60 s	Timer after end of precipitation to rain alarm off
Ice Alarm Delay On	weIceAlarmDelayOn	Real	1 s	Time after precipitation to ice alarm
Ice Alarm Delay Off	weIceAlarmDelayOff	Real	600 s	Timer after end of precipitation to ice alarm off

Table 17: Precipitation Parameters

The following functions use the data points from the precipitation data interface:

- Rain
  - Sunblinds: Rain alarm function
- Ice
  - Sunblinds: Ice alarm function

5.1.4 Wind Data

The *weStationWind* module receives wind speed and direction and generates a wind alarm signal. The wind module is suited for smaller or compact buildings.

For larger buildings with complex shapes and many façade directions, several wind sensors for a façade direction must be used together. As the assignment and calculation of the effective wind speed per façade is highly project dependent, this must be done programmatically for such projects. The algorithm for calculating the effective wind speed for a façade direction must be provided by the planning authority, it is not part of the LROC library.

The LROC library supports multiple façade directions. Each segment can be assigned an orientation ID to which the wind data can be sent. The weather station in a complex project can be built to calculate an effective wind speed out of the available sensors for each orientation.

Favorite	Type	Description
SensorFailure	Binary	Can be used to indicate sensor failure. Ignored if left open.
WindSpeed	Analog	Wind speed
WindDirection	Analog	Wind direction

Table 18: Favorite Data Points of *weStationWind*

The favorite data points of the *weStationWind* module are shown in Table 18. If the SensorFailure input is not used, the sensors are assumed to be always operational. The wind direction is not considered by the built-in wind function, but can be used to calculate a virtual wind speed for complex wind scenarios.

Wind data is distributed using the *WEA_WIND* serialize client command with a cycle of 15 seconds. Currently, the LROC library does not process SensorFailure input.

Table 17 shows the parameters of the *weStationPrecipitation* module.

Parameter View	Datapoint	Type	Default	Description
Wind Orientation	weWindOrientation	String	*	Orientation address

Table 19: Wind Parameters

The following functions use the data points from the wind data interface:

- Wind
    - Sunblinds: Wind alarm function

Figure 8 shows an example for a more complex wind scenario: The building has 6 façade orientations: North (N), East (E), South (S), West (W), Interior 1 (I1) and Interior 2 (I2). Interior 1 will have a lower effective wind speed, but Interior 2 has a higher effective wind speed, because of the nozzle effect. There are 5 Sensors in the example, North (SN), East (SE), South (SS), West (SW) and Roof (SRoof).

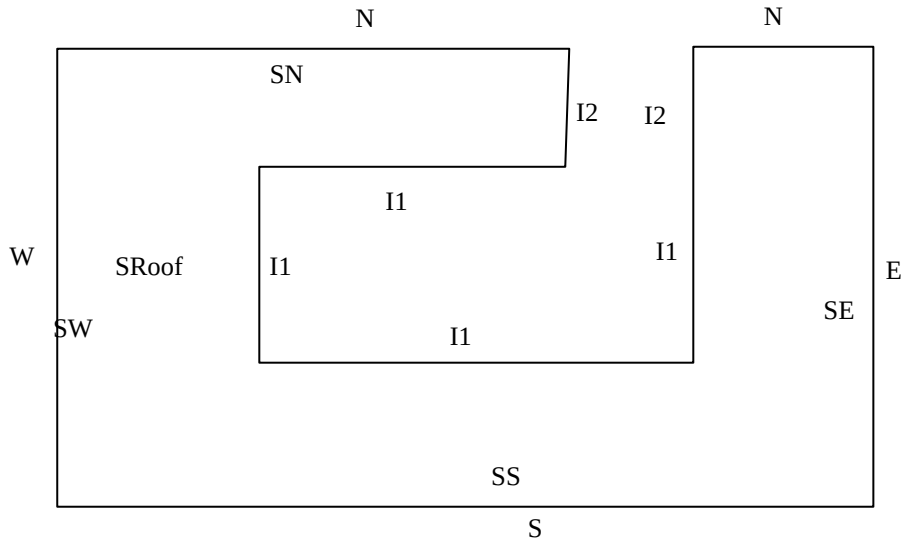


Figure 8: Complex wind scenario

Here the task is to calculate 6 effective wind speeds out of the 5 wind speeds and wind directions.

A safe way is to calculate minimum of all sensors and to send it to all façade directions. However this can lead to acceptance problems.

```
windN  := MIN(SN,SE,SS,SW,SRoof);
windE  := MIN(SN,SE,SS,SW,SRoof);
windS  := MIN(SN,SE,SS,SW,SRoof);
windW  := MIN(SN,SE,SS,SW,SRoof);
windI1 := MIN(SN,SE,SS,SW,SRoof);
windI2 := MIN(SN,SE,SS,SW,SRoof);
```

A more aggressive way to calculate the effective wind speeds would be to take only the façade sensor into consideration, but have a conservative approach for the nozzle-like area.

```
windN  := SN;
windE  := SE;
windS  := SS;
windW  := SW;
windI1 := MIN(SN,SE,SS,SW,SRoof);
windI2 := SRoof;
```

Also the wind direction could be taken into consideration. However, these decisions need to be taken upon the wind planning of the building.

In any way, the resulting effective wind speeds has to be provided to 6 instances of the `weStationWindSender` block, as outlined in Figure 9. Please note the hard-coded orientation. If a weather station is tailored to a project, the addressing strings can be hardcoded. The `WindExampleCalc` block would receive the weather data, do the above calculations and then provide the effective wind speeds to the sender blocks. The wind direction is omitted for clarity.

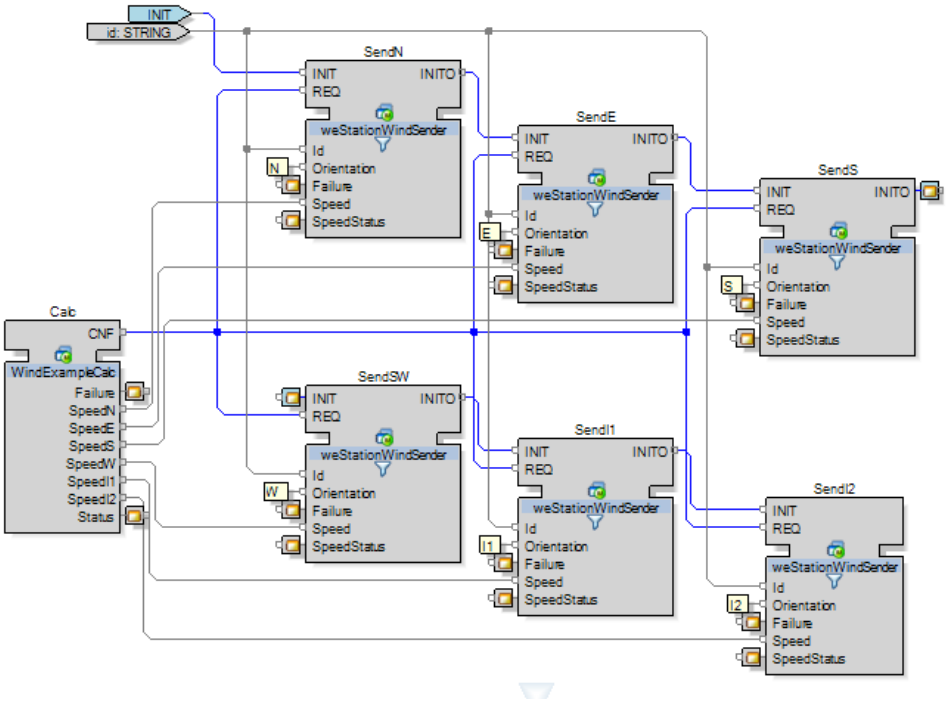


Figure 9: Complex wind sensor example.

5.1.5 Multi-Façade Functions

Starting with L-ROC 2.1.12, the L-ROC library contains functions to send wind speed, lux level and solar irradiation per façade element. Every façade group has 16 values and façade groups can be named to create more than 16 façade elements.

Currently the sunblind actuator module can use the multi-façade functions to make engineering of corner segments easier. The façade direction is a property of the sunblind controller, so the sunblind controllers of a corner segment can be put into different façade directions.

As the values and directional assignments of the façade elements depend on the project and/or wind model, the library only provides core blocks that have to be adapted to a particular project.

All multi-façade blocks have 16 favorite inputs + one SensorFailure input. These are gathered and sent at once to the control functions. The values are also trended in the weather station block.

The block weStationWindMulti, weStationSunMulti and weStationSolarIrradianceMulti have the parameters shown in Table 20: Weather Station Multi-Façade ParametersTable 20.

Parameter Name	**Datapoint**	**Type**	**Default**	**Description**
Zone ID	ZoneId	String	*	Target zone ID
Override Address	Override Address	String	“”	Alternate target address, leave empty for default.
Façade Group	FacadeGroup	String	“”	Façade Group Name (4 letters). Can be left empty when using only one façade group.
Heartbeat Time	HeartBeatTime	Analog	0	Heartbeat monitor

Table 20: Weather Station Multi-Façade Parameters

Zone ID (ZoneId)

This parameter defines the target Zone IDs. The default of "*" addresses all zone IDs.

Override Address (Override Address)

The module usually determines the right destination address from its coupler. This parameter allows to override the ID for special cases.

Façade Group (FacadeGroup)

The façade group name defines the façade group. It should be limited to 4 characters. If there is only one façade group, it can be left empty.

Heartbeat Time (HeartBeatTime)

The weather station can monitor a heartbeat on the weather data. This parameter defines the maximum interval between the sensor updates. If the updates don't arrive on time, the sensor is considered unreliable and the controllers go to their configured protection modes. A value of 0 disables the heartbeat monitor.

6 Sensor Functions

6.1 Sensor Introduction

The LROC system uses sensors to capture space temperature, occupancy, CO2 level, illumination level and similar values. These values are automatically sent to the HVAC, light and sunblind zones in which these sensors are located. The zones calculate an aggregated value of all available sensors. Sensor data also can be collected on area, floor or building level for reporting to the building management system.

Figure 10 shows an example of using a temperature and an illumination sensor in three segments. The HVAC zone A covers all three segments, so the temperature used for HVAC control is the average of the three sensors, in this case 21°C. The light is split into two zones. Light zone A covers two segments and uses the average light level (350 Lux) for constant light control, while light zone B consists only of one segment and uses its sole sensor (500 Lux) for constant light control. If the zone assignment is changed, the aggregation automatically adapts to the new zoning.

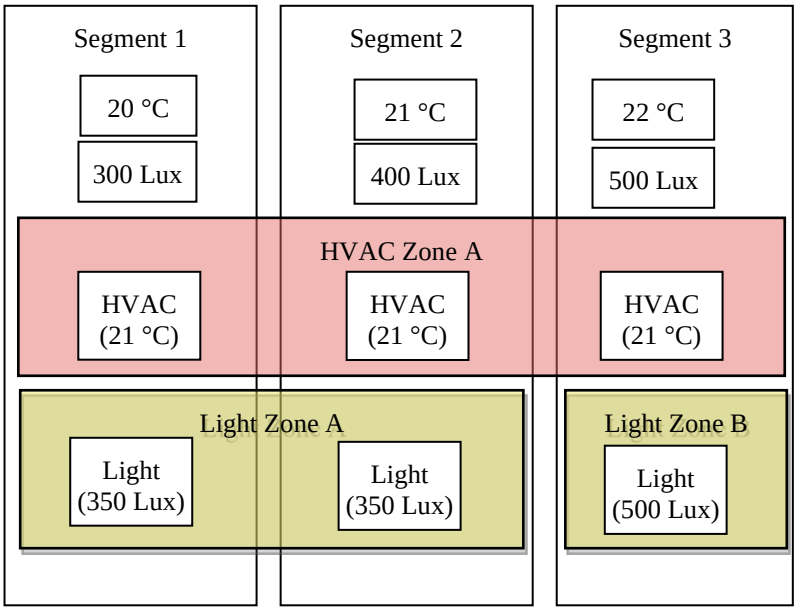


Figure 10: Sensor zone aggregation

The LROC sensor infrastructure is available to every room segment and provides the following features.

Zone aggregration: All sensors in a zone are aggregated to a single value which is used by the zone master as input to its controllers. Each trade (HVAC, Light, Sunblinds …) has its own aggregation so that they can be zoned independently. The aggregation is also configurated automatically, so that there is no extra effort necessary when changing zones.

The aggregation function depends on the sensor type, for example temperatures are averaged. The following sections describe the exact function of each sensor type

Multiple sensors per segment: The sensor library allows to chain several sensors within a room segment. For example if an aisle segment has 6 occupancy sensors, they can be chained, so that only one room segment is needed for the aisle. The chained sensors appear as a single sensor to the room segment. For this function the 61499 adapters of the sensor need to be connected. The last sensor in the chain then provides the aggregated value to the room segment.

Supported technologies: The LROC library provides ready-to-use sensors for common technology, like DALI multisensors or L-STAT room operation panels. It also provides a sensor block which encapsulates the sensor logic and can be easily extended by an arbitrary supported LROC datapoint, e.g. a KNX room panel.

6.2 Temperature Sensors

Temperature sensors measure the space temperature. They are used to HVAC and sunblind thermal control, as well as for software-based dewpoint calculation.

The aggregate value within a zone is the average value of all functional sensors.

Technically, they send their temperature value (REAL) and validity (BOOL) using the SPACETEMP command to their room segment.

The following sensors are supported natively:

Data point favorites (coLiobTemp): This sensor provides a data point favorite which can be linked to another data point, for example a LIOB I/O terminal. This sensor type can be chained.

L-STAT: L-Stat devices automatically send their temperature measurement to their room segment.

6.3 Humidity Sensors

Humidity sensors measure the relative humidity. They are used for software-based dewpoint calculation.

The aggregate value within a zone is the maximum value of all functional sensors.

Technically, they send their relative humidity value (REAL) and validity (BOOL) using the AIR_RELHUMID command to their room segment.

The following sensors are supported natively:

L-STAT: L-Stat devices automatically send their humidity measurement to their room segment.

6.4 CO2 Sensors

CO_2 sensors measure the CO_2 concentration (ppm). They are used for air quality control.

The aggregate value within a zone is the maximum value of all functional sensors.

Technically, they send their air quality value (REAL) and validity (BOOL) using the IAQ_CO2 command to their room segment.

The following sensors are supported natively:

L-STAT: L-Stat devices automatically send their humidity measurement to their room segment.

6.5 Dewpoint Sensors

Dewpoint sensors are not distributed as other sensors but directly connected to the HVAC modules to increase safety. Dewpoint sensors typically are binary sensors which assert a signal just before condensation occurs. When using chilled ceilings or other cooling actuators with cold media, it is highly recommended to use a dewpoint sensor for avoiding condensation and resulting damage.

The following sensors are supported natively:

Data point favorites (hcDewpointFav): This block is connected to the HVAC IO bus. It provides a favorite data point which can be connected to a LIOB I/O terminal.

6.6 Occupancy Sensors

Occupancy sensors detect occupancy, typically with PIR or more rarely radar signals. But every technical system which can indicate occupancy can be used, for example a button, using a room operation panel or running a LWEB-803 room panel on a PC which registers key presses and mouse movements. Occupancy sensors are common to all trades.

The aggregate value within a zone is the maximum value of all functional sensors, so if a single segment is occupied, the entire zone is occupied.

Technically, they send the occupancy state (REAL, 2=UNOCCUPIED, 5=OCCUPIED) and validity (BOOL) using the OCC_SENSOR command to their room segment.

The following sensors are supported natively:

Data point favorites (coLiobOccupancy): This block is connected to the HVAC IO bus. It provides a favorite data point which can be connected to a LIOB I/O terminal.

L-STAT: L-Stat devices with occupancy sensors automatically send their occupancy measurement to their room segment.

DALI (coDaliMultisensor): This block creates a DALI multisensor object containing an occupancy and an illumination sensor. Once commissioned on the DALI bus, the sensor values are automatically sent to their room segment.

L-WEB (coLWebOccupancy): This block can be used in an LWEB-803 project. When the user operates keyboard or mouse of the device, it automatically sends an occupancy event to the room segment. The “PC active” favorite of this block has to be connected to the “PC Active” data point in the LWEB project.

6.7 Illumination Sensors

Illumination sensors measure the light level within a room. They are used by the light controller.

The aggregate value within a zone is the average value of all functional sensors.

Technically, they send the light level (REAL) and validity (BOOL) using the LUX command to their room segment.

The following sensors are supported natively:

Data point favorites (coLiobLux): This sensor provides a data point favorite which can be linked to another data point, for example a LIOB I/O terminal. This sensor type can be chained.

DALI (coDaliMultisensor): This block creates a DALI multisensor object containing an occupancy and an illumination sensor. Once commissioned on the DALI bus, the sensor values are automatically sent to the room segment.

6.8 Window Contacts

Window contact sensors determine whether windows, door or similar components are opened. They are used by HVAC control, sunblind control and the security system.

The aggregate value within a zone is the sum value of all functional sensors, so it is possible to count the number of open windows.

Technically, they send the number of open windows (INT), closed windows (INT) and a list of open window contact names (STRING) using the WINCONT command to their room segment.

The following sensors are supported natively:

Data point favorites (coLiobWinContact): This sensor provides a data point favorite which can be linked to another data point, for example a LIOB I/O terminal. This sensor type can be chained.

Enocean (coEnoceanWindowContact): This block creates an Enocean windows contact device. Once commissioned, the sensor values are automatically sent to the room segment.

6.9 Sensor Core Functions

The sensor core functions allow to easily create new sensors for special data point types or sensors that require special processing.

The sensor core functions come in two flavors:

- A simple sensor core block connects a single sensor and distribute its value to the room segment. Figure 11 shows the interface of a simple sensor. Only the sensor value, sensor state and segment ID has to be provided.

- An aggregation sensor core block allows chaining several sensors. Figure 12 shows how the sensor core block is to be used. The *IO* block imports the real sensor data.

The *outputConnected* block is used to determine whether this is the last block in the chain. The *INST_VAR* input has to have the same name as the output adapter. If not, the *disableSend* variable disables sending the value in this sensor block. Only the last block in the chain will send the aggregated values to the room segment. The sensor logic block is connected to the adapter *coTempIn* and *coTempOut* which are used to chain the sensors together.

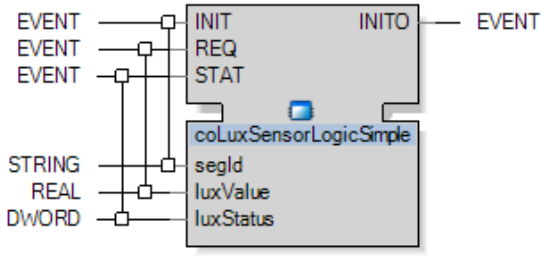


Figure 11: Simple sensor core block

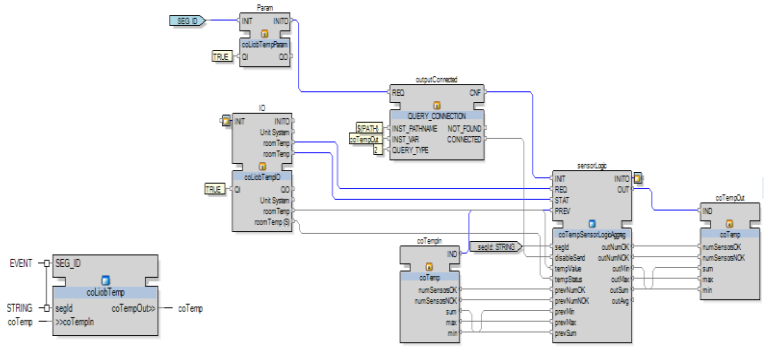


Figure 12: Aggregation sensor block

The following core sensors blocks are provided:

- Illumination sensor: coLuxSensorLogic(Simple|Aggreg)
- Occupancy sensor: coOccSensorLogic(Simple|Aggreg)
- Temperature sensor: coTempSensorLogic(Simple|Aggreg)
- Window contact: coWindowContactSensorLogic(Simple|Aggreg)

6.10 Routed IO Sensors

6.10.1 Introduction

Using the routed IO concept presented in section 4.2, one can put sensors arbitrarily in segments, device types or even onto a device instance. These values are routed to the configured segment and are processed there.

Note: For the samve value, Routed IO sensors and conventional sensors cannot be used at the same time.

6.10.2 Sensor Hub

In order to use the Routed IO sensor blocks, the receiving segment needs a SensorHub block.

The sensor hub receives the routed IO sensor data, aggregates it and distributes it into the segment. This is the reason why it cannot be use in parallel to conventional sensors. The two sensors would send out their data concurrently without aggregating them.

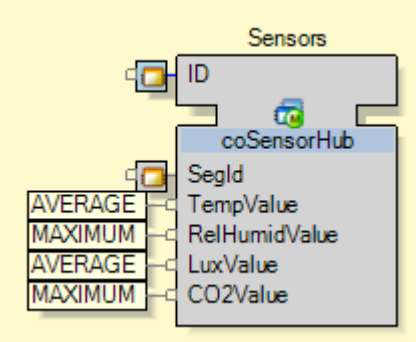


Figure 13: Sensor Hub

As shown in Figure 13, one can select the aggregation function for each sensor type (MINIMUM, AVERAGE, and MAXIMUM).

The sensor hub supports up to 8 attached sensors of each type. Sensor assignments exceeding this limit will be ignored.

The sensor hub automatically gathers data from all connected sensors. If a sensor stops delivering data (e.g. because of parameterization), it is removed from the sensor hub.

Starting from L-ROC 2.1.12, the conventional sensor block automatically detect the presence of the sensor hub and send their sensor values to the sensor hub instead.

6.10.3 Analog Inputs

The block coRoutedAI is used to provide a routable analog sensor.

It provides a favorite which can be linked with the physical I/O, e.g. one of the universal inputs of the device (UI1, UI2 …)

Parameter Name	Datapoint	Type	Default	Description
RIO Segment ID	rioSegId	Int	0	Segment ID
RIO UI Type	rioUIType	Enum	UNASSIGNED (1)	Sensor type
RIO UI Scaling	rioUIScale	Real	1.0	Value Scaling
RIO UI Offset	rioUIOffset	Real	0.0	Value Offset

Table 21: coRoutedAI Parameters

Table 21 shows the parameters of the coRoutedAI type.

RIO Segment Id (rioSegId)

This number defines the segment ID of the connected segment. If this value is 0, the segId input of the block is used, if it is non-zero, it overrides the value of the segId input. The segment ID must be used on the device, else the values are ignored.

RIO UI Type (rioUIType)

This parameter selects the semantics of the input:

- UNASSIGNED (1): Value is ignored
- SPACE_TEMP (2): Indoor temperature

- REL_HUMID (3): Indoor relative humidity
- CO2 (4): Indoor CO_2 concentation
- LUX (5): Indoor brightness
- FLOW_FB_SUP (6): Supply air flow feedback
- FLOW_FB_EXH (7): Exhaust air flow feedback

RIO UI Scaling (rioUIScale)
RIO UI Offset (rioUIOffset)

These two parameters define a linear scaling of the sensor value:

```
segmentValue := sensorValue * rioUIScane + rioUIOffset;
```

6.10.4 Digital Inputs

The block coRoutedDI is used to provide a routable digital sensor.

It provides a favorite which can be linked with the physical I/O, e.g. one of the digital inputs of the device (DI1, DI2 …)

Parameter Name	Datapoint	Type	Default	Description
RIO Segment Id	rioSegId	Int	0	Segment ID
RIO DI Type	rioUIType	Enum	UNASSIGNED (1)	Sensor type
RIO DI Invert	rioDIInvert	Bool	inactive	Invert value

Table 22: coRoutedDI Parameters

RIO Segment Id (rioSegId)

This number defines the segment ID of the connected segment. If this value is 0, the segId input of the block is used, if it is non-zero, it overrides the value of the segId input. The segment ID must be used on the device, else the values are ignored.

RIO DI Type (rioDIType)

This parameter selects the semantics of the input:

- UNASSIGNED (1): Value is ignored
- WINDOW (2): Window contact
- OCCUPANCY (3): Occupancy sensor
- DEWPOINT (4): Dewpoint sensor

RIO DI Invert (rioDIInvert)

When set to active, the input value is inverted.

6.10.5 DALI Multisensor

The coRoutedDaliMs block implements a routed DALI multisensor. It can be used when multisensors are not available in a regular grid so that segments have variable numbers of sensors.

Parameter Name	Datapoint	Type	Default	Description
RIO Segment Id	rioSegId	Int	0	Segment ID
RIO Disable Lux Value	DisableLux	Bool	inactive	Do not use lux value
RIO Disable Occupancy	DisableOccupancy	Bool	inactive	Do not use occupancy

Table 23: coRoutedDaliMs Parameters

Table 23 shows the parameters of the routed DALI multisensor.

RIO Segment Id (rioSegId)

This number defines the segment ID of the connected segment. If this value is 0, the segId input of the block is used, if it is non-zero, it overrides the value of the segId input. The segment ID must be used on the device, else the values are ignored.

RIO Disable Lux Value (DisableLux)

When set to active, the lux sensor value is not propagated to the segment.

RIO Disable Occupancy Value (DisableOccupancy)

When set to active, the occupancy sensor value is not propagated to the segment.

6.10.6 DALI Multisensor V2

The coRoutedDaliMsV2 block implements a routed DALI multisensor. It can be used when multisensors are not available in a regular grid so that segments have variable numbers of sensors.

Parameter Name	Datapoint	Type	Default	Description
RIO Segment Id	rioSegId	Int	0	Segment ID
RIO Disable Lux Value	DisableLux	Bool	Enable Lux	Do not use lux value
RIO Disable Occupancy	DisableOccupancy	Bool	Enable Occupancy	Do not use occupancy
RIO Disable Temperature	DisableTemp	Bool	Enable Temperature	Do not use temperature
RIO Disable Rel. Humidity	DisableRelHumid	Bool	Enable Rel. Humidity	Do not use rel. humidity

Table 24: coRoutedDaliMs Parameters

Table 23 shows the parameters of the routed DALI multisensor.

RIO Segment Id (rioSegId)

This number defines the segment ID of the connected segment. If this value is 0, the segId input of the block is used, if it is non-zero, it overrides the value of the segId input. The segment ID must be used on the device, else the values are ignored.

RIO Disable Lux Value (DisableLux)

When set to active, the lux sensor value is not propagated to the segment.

RIO Disable Occupancy Value (DisableOccupancy)

When set to active, the occupancy sensor value is not propagated to the segment.

RIO Disable Temperature (DisableTemperature)

When set to active, the temperature sensor value is not propagated to the segment.

RIO Disable Rel. Humidity (DisableRelHumidity)

When set to active, the relative humidity sensor value is not propagated to the segment.

7 HVAC Functions

7.1 HVAC Overview

The L-ROC library provides a versatile HVAC control system. It covers not only room control, but also manages the communication between room control and the central heating/cooling components.

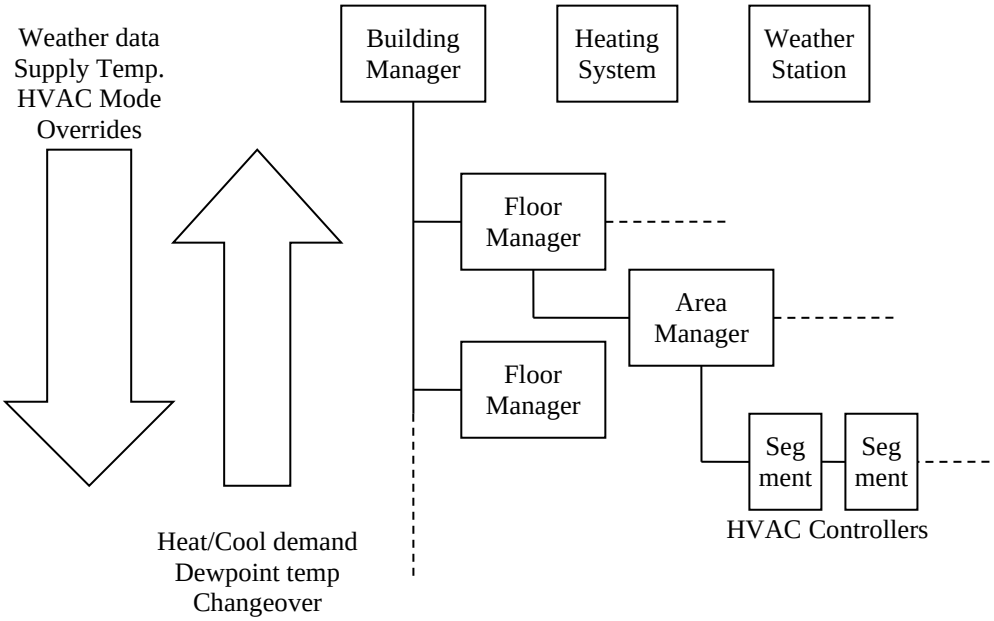


Figure 14: L-ROC HVAC Components

Figure 14 shows the main components of the L-ROC HVAC System:

The HVAC controller is responsible for comfort and energy-efficiency. It supports various heating/cooling systems, like radiators, chilled ceilings and fancoil devices. It provides different energy/occupancy setpoints, occupancy control, PI control, and energy holdoff functions. The HVAC controllers can be joined according to the L-ROC segment system and zoned where necessary. The HVAC controller also reports energy demand to its area manager.

Building, floor and area managers contain one or more circuit servers. They implement central functions which can be used at area, floor or building level. For example, the area manager can be connected to a changeover valve which switches the area between heating/cooling modes. It decides upon the information it gets from the HVAC controllers.

The heating/cooling system typically interfaces to the building manager. It receives aggregate valve positions to optimize its supply temperature. In addition the heating system can provide the heat/cool supply temperatures to the room controllers.

The weather station provides data like outdoor temperature to the room controllers. This information is automatically transmitted to every segment in the system.

The LROC system supports multiple heating/cooling systems to which the HVAC controllers can be assigned to. For example a radiator-based heating system and a fancoil-based cooling system can be easily integrated.

7.2 OPC/BACNET Interface

The HVAC module contains an OPC/BACnet interface for BMS integration and LVIS/LWEB connectivity. The OPC and BACnet data points are connected with global connections and can be used simultaneously. Table 25 shows the OPC and BACnet data points used for controlling the HVAC controller.

Feedback data points used for display have an “Fb” suffix. The other data points are used to send commands to the controller.

The OPC data points are found within the “User Registers” folder, while the BACnet data points are found within the “BACnet Port” folder.

Datapoint	Type	Description
hcZoneId	String	Zone ID write
hcZoneIdFb	String	Zone ID feedback
mode	Multistate	User HVAC mode write
modeFb	Multistate	User HVAC mode feedback
hvacModeFb	Multistate	Effective HVAC Mode feedback
spExt	Analog	External setpoint write
spExtFb	Analog	External setpoint feedback
spOffset	Analog	Setpoint offset write
spOffsetFb	Analog	Setpoint offset feedback
spDisplayFb	Analog	Displayed setpoint feedback (heat/cool average)
spEffectiveFb	Analog	Effective setpoint feedback
coolValueFb	Analog	Cool value output
heatValueFb	Analog	Heat value output
tempRoomFb	Analog	Room temperature
spCoolFb	Analog	Cool setpoint feedback
spHeatFb	Analog	Heat setpoint feedback
windowOpenFb	Analog	Number of open windows
fanSpeed	Analog	Fan speed write
fanSpeedFb	Analog	Fan speed feedback
fanAutoMode	Analog	Deprecated
fanAutoModeFb	Analog	Deprecated
occFb	Multistate	Occupancy feedback
occMan	Multistate	Manual occupancy write
occManFb	Multistate	Manual occupancy feedback
spOffsetIncr	Analog	Incremental setpoint offset change
fanSpeedIncr	Multistate	Incremental fan speed change
fanStageSelFb	Multistate	Selected fan stage feedback
fanStage	Multistate	Fan stage write
fanStageFb	Multistate	Effective fan stage feedback
flapStage	Multistate	Flap stage write
flapStageFb	Multistate	Flag stage feedback
dewpointFb	Binary	True if dewpoint alarm is active
circuitIdFb	String	Circuit ID to which this heating/cooling controller belongs to.
flapStageIncr	Analog	Incremental flag stage change
cmdFlowFb	Analog	Relative flow setpoint (%)
flowLocalSupplyFb	Analog	Supply air flow (segment)
flowLocalSpSupplyFb	Analog	Supply air flow setpoint (segment)
flowLocalExhaustFb	Analog	Exhaust air flow (segments)
flowLocalSpExhaustFb	Analog	Exhaust air flow setpoint (segment)
flowZoneSupplyFb	Analog	Supply air flow (zone)
flowZoneSpSupplyFb	Analog	Supply air flow setpoint (zone)
flowZoneExhaustFb	Analog	Exhaust air flow (zone)
flowZoneSpExhaustFb	Analog	Exhaust air flow setpoint (zone)
flowZoneNumSupOk	Analog	Number of operative supply VAV units

Datapoint	Type	Description
flowZoneNumSupNok	Analog	Number of supply VAV units with errors
flowZoneNumExhOk	Analog	Number of operative exhaust VAV units
flowZoneNumExhNok	Analog	Number of exhaust VAV units with errors
co2Fb	Analog	CO_2 concentration
relHumidFb	Analog	Relative humidity
dewPointTempFb	Analog	Dew point temperature
flowOverrideMode	Enum	Zone flow mode override
flowOverrideValue	Analog	Flow setpoint (%) in override mode
masterFb	Binary	Indicates that this is the master module for the HVAC zone

Table 25: HVAC OPC/BACnet Interface

7.2.1 Zoning

The *hcZoneId* data point is used to set the zone ID for the controller. The current value of the zone ID parameter is reflected in the *hcZoneIdFb* data point.

The *circuitIdFb* data point shows the assigned heating/cooling circuit. It can be used for display purposes and cannot be changed over the HMI interface.

The *masterFb* data point indicated whether this HVAC module is the master module of its HVAC zone

7.2.2 HVAC Modes

The *mode* data point allows changing the manual HVAC mode. It is used when the user shall be able to select between heating and cooling modes manually, for example if a dedicated air conditioning system is used. Meaningful HVAC values for user-interfaces are *AUTO (0)*, *HEAT (1)*, *COOL (3)* and *OFF (6)*.

The *modeFb* data point indicates the resulting mode, while the *hvacModeFb* indicates the state of the heating/cooling system. This can be used to indicate that the used has selected an improper mode. For example if *hvacModeFb* indicates that only cooling is possible, an attempt so select heating will result in the *OFF* state. In case that heating and cooling is enabled the *hvacModeFb* data point will indicate *AUTO* and the user can freely select between heating and cooling (or automatic mode).

7.2.3 Setpoint Selection

The *spExt* data point allows overriding the internal setpoint logic with an absolute external setpoint. To disable the external setpoint feature, it needs to be written to zero. A value greater than 0 enables the external setpoint. The *spExtFb* data point indicates that an external setpoint is used.

The *spOffset* data point allows shifting the setpoint array within limits. The current setpoint offset is reflected in the *spOffsetFb* data point.

The *spOffsetIncr* data point is used to add a constant to the setpoint shift. It is a convenience data point for removing the need for calculations in the HMI. For example: To lower the setpoint by -1 degrees, write -1 once on the data point. The value will be only evaluated once.

The *spDisplayFb* data point is calculated by the average occupied heating and cooling setpoints. It is the center of the deadband of the priority array and does not change its value when switching between heating and cooling modes. The *spEffectiveFb* data point displays the current active setpoint. It changes upon occupancy and mode changes.

The *spCoolFb* data point indicates the effective setpoint in cool mode and the *spHeatFb* data point indicates the effective setpoint in heat mode.

7.2.4 Sensor Feedback

The *tempRoomFb* data point reads the current average zone temperature.

The *windowOpenFb* data point reads the number of open windows within zone.

The *dewpointFb* data point indicates an active dew point alarm in the zone which prevents cooling.

The *co2Fb* data point displays the maximum CO2 concentration in the zone.

The *relHumidFb* data point displays the maximum relative humidity in the zone.

The *dewPointTempFb* data point displays the highest dewpoint temperature in the zone.

7.2.5 Control Feedback

The data points *heatValueFb* and *coolValueFb* display the output of the heating and cooling controller. They are typically used to control valves and can be used to display the amount of heating and cooling to the user.

7.2.6 Occupancy

The occupancy data points allow to set occupancy states manually and to display the current occupancy state.

The *occMan* data point is used to set the manual occupancy. It can be either state based or event based. If set to NULL (1), manual occupancy is undefined and not taken into account by the controller. In the states UNOCCUPIED (2), STANDBY (3) and OCCUPIED (5) the manual state is set to the select value. The BYPASS (4) state is evaluated only once upon writing that value and triggers manual occupancy for the amount of time configured in the HVAC controller.

The *occManFb* data point reflects the current manual occupancy state.

The *occFb* data point reads the current occupancy state of the controller which is composed of manual occupancy, scheduler occupancy and sensor occupancy.

7.2.7 Fan Control

The fan control data points allow controlling a fancoil or other fan. Both staged and analog fans are supported.

The *fanStage* data point selects either automatic mode *AUTO (-1)*, *OFF (0)* or 5 manual modes *STAGE 1 (1)* to *STAGE 5 (5)*. In automatic mode, the HVAC controller calculates the fan stage automatically, while the manual modes instruct the controller to use the selected mode. Note that turning off the fan can disable the heating actuator for safety reasons.

The *fanStageSelFb* data point reflects the fan setting for this HVAC zone.

The *fanStageFb* data point indicates the current fan stage. It can differ from *fanStageSelFb* due to controller decisions or protection conditions. It is also limited to *OFF (0)* or 5 manual modes *STAGE 1 (1)* to *STAGE 5 (5)*.

The *fanSpeed* data point is used to control an analog fan (0…100%). Negative values are interpreted as automatic modes (device dependent). The *fanSpeedFb* reads the current fan speed (0…100%).

The *fanSpeedIncr* data point is a convenience data point which is used to shift the fan stage or fan speed (depending on actuator) by the amount written to the data point. It is a used for removing the need for calculations in the HMI. Its value is only used once when written.

7.2.8 Flap Control

The flap control data points allow controlling the position and mode of a fancoil flap or louver. These elements are typically used to direct the air flow into a direction convenient for the user.

The *flapStage* data point selects an automatic *AUTO (-1)* mode, or one of 5 fixed modes *F1 (0)* to *F5 (4)*. As these modes are typically device-dependent, the names are kept generic intentionally.

The *flapStageIncr* data point is a convenience data point to add a constant to the current flap state for avoiding mathematic operations in the HMI. For example, writing +1 to *flapStageIncr* advances the flap stage by one step.

The *flapStageFb* data point reads the current flap stage setting.

7.2.9 Air Flow

The *cmdFlowFb* data point reads the calculated relative air demand for the zone. It is dependent on the occupancy state and CO2 level.

The *flowLocalSpSupplyFb* and *flowLocalSpExhaustFb* data points indicate the requested absolute supply and exhaust air flows for this segment.

The *flowLocalSupplyFb* and *flowLocalExhaustFb* data points read the current absolute supply and exhaust air flows for this segment.

The *flowZoneSpSupplyFb* and *flowZoneSpExhaustFb* data points indicate the requested absolute supply and exhaust air flows for this zone.

The *flowZoneSupplyFb* and *flowZoneExhaustFb* data points read the current absolute supply and exhaust air flows for this zone.

The *flowZoneNumSupOk*, *flowZoneNumSupNok*, *flowZoneNumExhOk* and *flowZoneNumExhNok* datapoints read the number of VAV supply or exhaust VAV modules in this zone which are OK or not OK (Nok). These values are used to indicate problems reported by the VAV devices.

The *flowOverrideMode* data point allows to override the automatic indoor air quality control with the following settings:

- NULL (1): Override disabled. Normal control operation is performed.
- AUTO (2): Overrides to use normal automatic operation.
- OFF (3): Shut down ventilation.
- UNOCC (4): Set ventilation to unoccupied flow setpoint.
- OCC_MIN (5): Set ventilation to minimum occupied flow setpoint.
- OCC_MAX (6): Set ventilation to maximum occupied flow setpoint.
- OVERRIDE (7): Set flow setpoint to the value of the *flowOverrideValue* data point.

7.3 Communication

The HVAC controllers have a built-in communication module. It automatically performs the following tasks:

- Joining HVAC controllers within the same zone and determining masters and slaves automatically
- Distribute weather data to the HVAC controller
- Distribute statistics information on HVAC modes and valve positions to the superordinating building level.

Parameter Name	Datapoint	Type	Default	Description
Zone ID	hcZoneId	String	"A"	Heating/Cooling zone ID
Default Zone Mode	hcZone DefaultMode	Enum	AUTO (0)	HVAC mode to select when zone is unoccupied for hcZoneDefaultDelay
Default Zone Delay	hcZone DefaultDelay	Real	0 s	Delay time between unoccupied state and default mode restoration
Circuit ID	hcCircuitId	String	""	Heating/cooling circuit ID for valve aggregation
Circuit Heat Weight	hcCircuit HeatWeight	Real	1.0	Weight of heat value for valve aggregation (default 1)
Circuit Cool Weight	hcCircuit CoolWeight	Real	1.0	Weight of cool value for valve aggregation (default 1)
Circuit Application Mode Override	hcCircuitAppl ModeOverride	Enum	NULL (-1)	Allows overriding the application mode. NULL for no override.
Zone Location	hcZoneLocation	String	""	Location of Heating/Cooling Zone
Reset Setpoint	hcZoneDefaultResetSP	Enum	OFF	Reset setpoint shifts
Zone Type	HcZoneType	String	2"	Zone type
Software Dewpoint Alarm	hwDewpointSwAlarm	Binary	Off	Enable/disable software dewpoint
Software Dewpoint Safety Margin	hcDewpointSwSafetyMargin	Analog	0 K	Safety Margin for software dewpoint
Software Dewpoint Hold Time	hcDewpointHoldTime	Analog	900 s	Minimum dewpoint alarm time.
Ceiling Temperature Offset	hcCeilTempOffset	Analog	2 K	Expected temperature between ceiling and space temperature

Table 26: HVAC Communication Parameters

Table 26 shows the parameters of the hcCommLogic block which define the communication behavior and default of the HVAC controller.

Zone ID (hcZoneId)

This parameter assigns a zone ID to the heating/cooling controller. Its default value ("A") is suitable for almost any single room control. Having different room climate zones within a single room doesn't work usually and wastes energy then.

It is possible to assign a global zone ID (e.g. "#A") to several rooms if they are thermally coupled.

Default Zone Mode (hcZoneDefaultMode)

The default mode of the HVAC zone selects one of the following modes:

AUTO (0): This mode allows selecting automatically between heating and cooling. It is suitable for 4-pipe or 2-pipe changeover systems which provide heating and cooling media.

HEAT (1): This mode only allows heating. It is suitable for 2-pipe systems.

COOL (3): This mode only allows cooling. It is suitable for 2-pipe systems.

OFF (6): This mode disables heating and cooling.

Default Zone Delay (hcZoneDefaultDelay)

The default mode delay parameter resets the HVAC controller to the mode selected by *hcZoneDefaultMode* after a manual mode override. If the parameter is not 0, it enables a timer when a user overrides the mode, e.g. by setting the mode to COOL to activate an A/C device. The timer starts when the room gets unoccupied. It is restarted in case the zone gets occupied and unoccupied again. A timing example for a zone with a user-controllable A/C-device is shown in Figure 15.

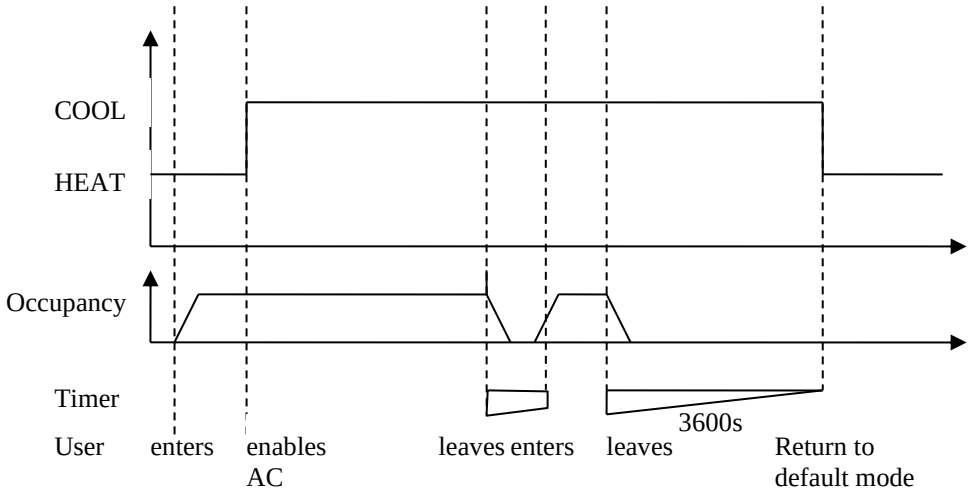


Figure 15: Example timing behavior of HVAC default mode (HEAT) and HVAC default delay (3600s).

Circuit ID (hcCircuitId)

The Circuit ID parameter assigns the HVAC controller to a HVAC circuit. This information is used to

- Aggregate HVAC controller information like heat or cool demand to the area, floor or building level
- Distribute heating/cooling system data like feed temperatures to the HVAC controllers.

The value of this parameter, e.g. HC1, needs to match the name of a heating circuit server in the next area, floor or building coupler.

If there a several independent heating/cooling systems in the building they can be addressed using this parameter.

Circuit Heat Weight (hcCircuitHeatWeight)
Circuit Cool Weight (hcCircuitCoolWeight)

Each HVAC controller can be weighted independently for heating and cooling aggregation. The weights have no fixed interpretation, but can be used to reflect the relative power of a radiator or fan coil unit.

The circuit weight parameters control how much a HVAC controller contributes to the aggregated values. The default value of 1.0 results in each controller having the same importance and influence on the heating system. For most purposes the default value is appropriate.

If a zone should not be considered for aggregation, the value 0.0 can be used. For example a technical room or store could be excluded from the HVAC aggregation.

Within an area each weighted value is multiplied with is weight before it is added to the total sum. For example, a standard L-ROC device with 16 segments will have a weight sum of 16 and a weighted heat output sum of 1600 if all segments are at full heat. The weighted normalized output will be in the range of 0 … 100%.

Circuit Application Mode Override (hcCircuitApplModeOverride)

This parameter overrides the automatic HVAC mode determination. The default value of -1 (NULL) disables the override function.

The following modes are supported:

- NULL (-1): No override
- AUTO (0): Automatic selection between heating and cooling.
- HEAT (1): Heating only.
- COOL (3): Cooling only.
- OFF (6): No operation.

In normal operation this parameter should be kept at the default level. It can be used for testing or emergency purposes, if for example the communication with the area manager is cut off.

Zone Location (hcZoneLocation)

The zone location parameter is for documentation purposes only. It can be set to a planning ID or location code to identify the HVAC zone. It has no influence on HVAC control.

Zone Type (hcZoneType)

The zone type parameter is for documentation purposes only. It is indented for sorting an LWEB-900 parameter view so that zones with similar parameters are nearby.

Reset Setpoint (hcZoneDefaultResetSP)

This function resets setpoint shifts, when the zone returns to its default mode. The following options are available.

- OFF (1): No reset.
- OFFSET (2): Only the setpoint offset is reset.
- EXTERNAL (3): Only the external setpoint is reset.
- BOTH (4): Setpoint offset and external setpoint are reset.

Software Dewpoint (hcDewPoint…)

The *swDewpointSwAlarm* data point enables a software based dewpoint calculation. It calculates the maximum dewpoint of all segments of the zone and compares it with the cool supply temperature. If the dewpoint temperature drops bellow the cool supply temperature (including the safety margin *hcDewpointSwSafetyMargin*), a software dewpoint alarm is active for *hcDewpointHoldTime* seconds.

In order to work, the following items must be ensured:

- The cooling system has to provide the supply temperature
- The user has to measure and enter the safety margin because the temperature and humidity measuring locations are typically apart from the cooling actuator.

Note: *We recommend using real hardware dewpoint sensors, as these are mounted at the right position and can determine condensation reliably and consistently. Although the software dewpoint calculation has several measures to compensate for communication problems, there is no guarantee that it provides the same safety as a hardware based solution. The customer has to validate the solution for every application.*

Ceiling Temperature Offset (hcCeilTempOffset)

The controller can track two temperature/humidity data sets, one for wall and one for ceiling measurements. In case there is only a ceiling temperature, the wall temperature can be guessed from the ceiling temperature. This parameter defines the expected offset (ceiling – wall). The possible combinations are shown ein Table 27.

Sensors	Temperature for control	Humidity for control	Temperature for dewpoint	Humidity for dewpoint
Wall	T_{Wall}	RH_{Wall}	T_{Wall}+offset	RH_{Wall}
Ceiling	T_{Ceil}-offset	RH_{Ceil}	T_{Ceil}	RH_{Ceil}
Wall+Ceiling	T_{Wall}	RH_{Wall}	T_{Ceil}	RH_{Ceil}

Table 27: Wall and ceiling temperature and relative humidity selection.

7.4 Room Control Functions

The room control functions implement various comfort and energy-efficiency functions. The following functions are implemented:

Occupancy: The occupancy function calculates an effective zone occupancy from scheduled occupancy, sensor occupancy and manual occupancy

Setpoint calculation: The setpoint calculation function determines an effective setpoint from occupancy and a configurable setpoint array.

Energy holdoff: The energy holdoff function disables the heating/cooling controller for some time in case a window has been opened.

Summer compensation: The summer compensation function raises cooling setpoints dependent on the outside air temperature for energy savings.

Minimum heating: The minimum heating functions allows to heat glass facades to prevent uncomfortable air flows.

Optimum start: The optimum start function allows to reach the scheduled setpoints on time by anticipating the heating/cooling ramp.

PI-Controller: The dual PI-controller calculates a heating and cooling signal which can be used for sequencing different actuators like valves and fancoil units.

7.4.1 Occupancy

The L-ROC library uses the following occupancy states:

- **NULL** (1): This value indicates that the occupancy source is deactivated and should not be considered in any calculation. A zone which has every occupancy source set to NULL will not heat or cool actively, except when it reaches the protection setpoints.
- **UNOCCUPIED** (2): In this state a zone is considered unoccupied. The heat and cool setpoints are far from occupied setpoints and it will take a considerable amount of time to reach the occupied setpoints.
- **STANDBY** (3): In this state the zone is heated or cooled to setpoints which allows reaching occupied temperatures in a reasonable amount of time. This occupancy state allows conserving energy by allowing the zone the heat up or cool down when occupancy is expected.
- **BYPASS** (4): This special value is less a state than a signal. If the heating controller receives this value it will set itself OCCUPIED for the time specified in the *hcOccManTimeout* parameter.
- **OCCUPIED** (5): This is the occupied (or comfort) state. It typically has a narrow deadband between heating and cooling mode.

The L-ROC HVAC controller processes occupancy states from the following sources:

Sensor occupancy: Occupancy can be determined by an occupancy sensor. In this use case, the zone will be typically be scheduled to the bypass occupancy and enters occupied state when the sensor registers occupancy. The zone will be considered occupied for the time given by the *hcOccSensorOffDelay* parameter after the sensor stopped registering occupancy.

Local Scheduler: Every segment contains a local scheduler which allows scheduling the zone according to the zone usage pattern. Schedulers can be managed centrally and hierarchically by the L-WEB Master Scheduler.

Manual occupancy: Manual occupancy enables an inhabitant to indicate occupancy via a button. The button will typically set manual occupancy to BYPASS, so that the HVAC controller will enter OCCUPIED state for the *hcOccManTimeout* period. If the button is pressed while the zone is in BYPASS mode, the bypass timer is reset to *hcOccManTimeout*.

External occupancy: The external occupancy source allows a building manangement system set occupancy states. That is, a BMS could set an unoccupied area to UNOCCUPIED state. External occupancy can be managed on each L-ROC hierarchy level.

Table 28 shows the calculation of the effective occupancy from the manual, scheduled and sensor occupancy.

Use case	Manual	Scheduler	Sensor	Effective
Sensor only	NULL	NULL	NULL UNOCCUPIED STANDBY BYPASS OCCUPIED	**NULL** **UNOCCUPIED** **STANDBY** **BYPASS** **OCCUPIED**
Scheduler with sensor occupancy	NULL	BYPASS	NULL UNOCCUPIED STANDBY BYPASS OCCUPIED	**STANDBY** **STANDBY** **OCCUPIED** **OCCUPIED** **OCCUPIED**
Scheduler without sensor occupancy	NULL	UNOCCUPIED STANDBY OCCUPIED	d.c.	**UNOCCUPIED** **STANDBY** **OCCUPIED**
Temporary occupancy	BYPASS	d.c.	d.c.	**OCCUPIED for *hcOccManTimeout* period**
Manual override	UNOCCUPIED STANDBY OCCUPIED	d.c.	d.c.	**UNOCCUPIED** **STANDBY** **OCCUPIED**

Table 28: HVAC Occupancy table (d.c. means don't care)

The typical occupancy use cases are:

Sensor only: Manual occupany will not be used (no room control panel) and the scheduler is not configured. The zone occupancy will be determined from the sensor occupancy alone.

Scheduler with sensor occupancy: The scheduled state BYPASS will schedule the zone for STANDBY mode, but sensor occupancy can raise the occupancy level to OCCUPIED.

Scheduler without sensor occupancy: The states UNOCCUPIED, STANDBY and OCCUPIED will set the zone into exactly these states. Sensor occupancy will be ignored.

Temporary occupancy: A manual occupancy value of BYPASS will put the zone into OCCUPIED state for the hcOccManTimeout period. It can be combined with scheduled occupancy.

Manual override: Using the states UNOCCUPIED, STANDBY or OCCUPIED for manual occupancy will override scheduled and sensor occupancy entirely. This is not a very typical use case but can be used to move occupancy control entirely to a room control panel.

Note: *The recommended occupancy scheme is using scheduled and sensor occupancy. This allows for most comfort and least user interaction. With LWEB-900, schedulers can be configured in a central, hierarchical way. For office rooms, LWEB-802/803 also can determine occupancy via keyboard and mouse events.*

Table 29 lists the parameters related the the occupancy control.

Parameter	Datapoint	Type	Default	Description
Manual Occupancy Timeout	hcOcc ManTimeout	Real	3600 s	Time for which manual occupancy is active
Sensor Occupancy Timeout	hcOcc SensorOffDelay	Real	900 s	Time until unoccupied after sensor occupancy falling edge

Table 29: Occupancy parameters

Manual Occupancy Timeout (hcOccManTimeout)

The manual occupancy timeout parameter determines how long the zone is in occupied state after the user signaled manual occupancy. The timer is restarted if the occupancy button is pressed during the timeout period. This value should not be set too small to avoid forcing the user to operate the button too often.

Sensor Occupancy Timeout (hcOccSensorOffDelay)

The sensor occupancy timeout determines how long the zone is considered occupied after the occupancy sensor dropped to unoccupied state. The default value of 15 minutes is appropriate for typical office rooms. For rooms with less motion the value should be raised to avoid switching between standby and occupied states too often.

7.4.2 Setpoint Calculation

The setpoint calculation function is responsible for switching between different room energy states.

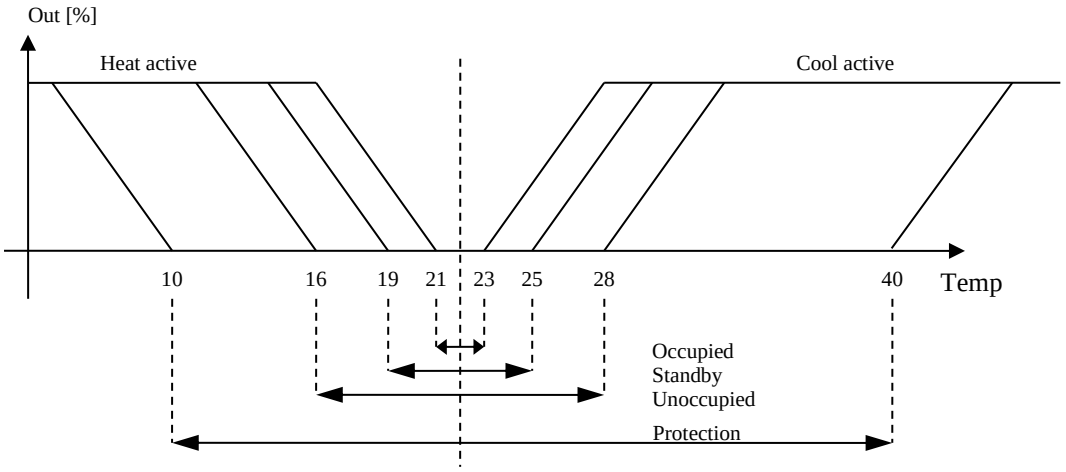


Figure 16 shows the default setpoints and the control output with varying temperature (shown without the time-dependent integral component).

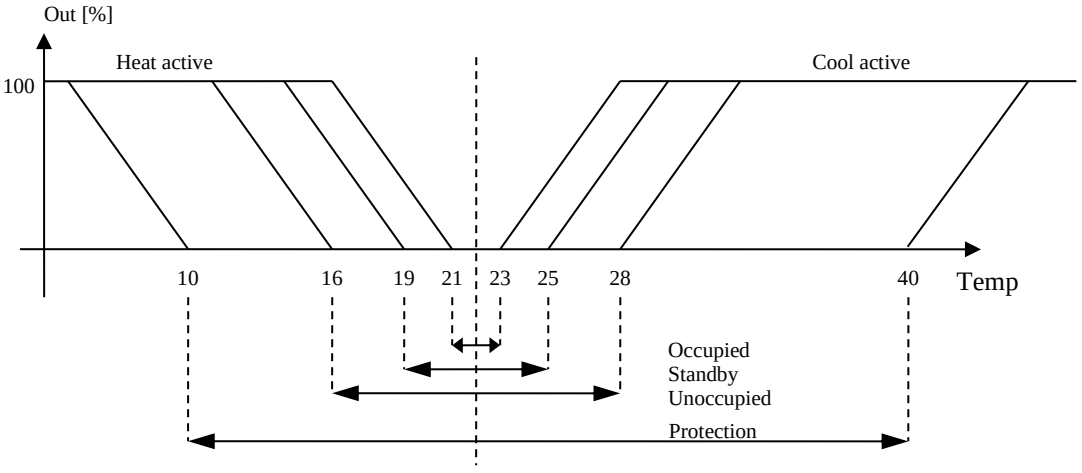


Figure 16: Setpoints and control output.

The setpoint logic considers the following inputs:

- The setpoint array
- The setpoint shift
- An external setpoint (for overriding the setpoint array)

The setpoint array should be choosen to be symmetrical in regard to the deadband average temperature. This avoids that the display setpoint changes when the occupancy state changes.

The following should be considered when changing the setpoint array:

- Lowering a heat setpoint by 1°C yields an energy saving of about 6%.

When displaying the absolute setpoint to the user, all heating/cooling setpoint pairs should have the same average temperature.

The setpoint offset shift allows a user to modify the standard setpoints within a defined range. Figure 17 shows the influence of the setpoint shift value on the effective setpoints. The standby and occupied setpoints are modified by the setpoint shift. Unoccupied and protection setpoints are not modified by the setpoint shift. In case the modified standby setpoints would cross the unoccupied setpoints, the unoccupied setpoints are adjusted accordingly.

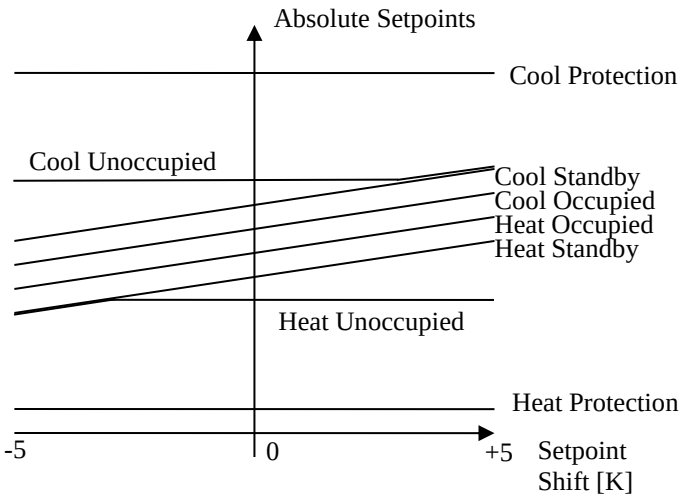


Figure 17: Setpoint shift

The external setpoint function allows setting an absolute setpoint via the user interface (instead of the relative setpoint shift). If the external setpoint is used, the occupied and heat setpoints are recalculated as follows:

Effective Heat SP = External SP - (Configured Cool SP – Configured Heat SP) / 2

Effective Cool SP = External SP + (Configured Cool SP – Configured Heat SP) / 2

This means that the spread between heat and cool setpoint is applied to the external setpoint (instead of the average of the configured setpoints).

The unoccupied and protection setpoints are not affected by the external setpoint parameter.

When using the external setpoint feature, the setpoint shift is applied additionally. For example, the occupied heat setpoint is 21°C, the occupied cool setpoint is 23°C, the external setpoint is set to 23°C and the setpoint shift is -1 °C. The resulting occupied heat setpoint is then 23 - (23 – 21)/2 – 1 = 21°C. The corresponding cool setpoint is 23 + (23-21)/2 -1 = 23°C.

Parameter View	Datapoint	Type	Default	Description
Occupied Cool Setpoint	hcSpCoolOccupied	Real	23	Cooling setpoint in occupied state
Standby Cool Setpoint	hcSpCoolStandby	Real	25	Cooling setpoint in standby state
Unoccupied Cool Setpoint	hcSpCoolUnoccupied	Real	28	Cooling setpoint in unoccupied state
Protection Cool Setpoint	hcSpCoolProtection	Real	40	Cooling setpoint in protection state (e.g. open window)
Occupied Heat Setpoint	hcSpHeatOccupied	Real	21	Heating setpoint in occupied state
Standby Heat Setpoint	hcSpHeatStandby	Real	19	Heating setpoint in standby state
Unoccupied Heat Setpoint	hcSpHeatUnoccupied	Real	16	Heating setpoint in unoccupied state
Protection Heat Setpoint	hcSpHeatProtection	Real	10	Heating setpoint in protection state (e.g. open window)
Setpoint Shift Range	hcSpOffsetRange	Real	6	Setpoint offset shift range

Table 30: HVAC Setpoint Parameters

Table 30 shows the available setpoint parameters:

Occupied Cool Setpoint (hcSpCoolOccupied)
Standby Cool Setpoint (hcSpCoolStandby)
Unoccupied Cool Setpoint (hcSpCoolUnoccupied)
Protection Cool Setpoint (hcSpCoolProtection)
Occupied Heat Setpoint (hcSpHeatOccupied)
Standby Heat Setpoint (hcSpHeatStandby)
Unoccupied Heat Setpoint (hcSpHeatUnoccupied)
Protection Cool Setpoint (hcSpHeatProtection)

These parameters define the setpoint array as described in Section 7.4.2. The occupied setpoints are in effect when the zone is in occupied state. The standby setpoints are in effect when the zone is in standby state. The unoccupied setpoints are in effect when the zone is in unoccupied state. The protection setpoints are in effect, when the energy holdoff function is triggered, e.g. when a window is opened.

The heat and cool setpoints of an energy state should have the same average value to avoid setpoint display ambiguities.

Setpoint Shift Range (hcSpOffsetRange)

The setpoint shift range defines the temperature range by which the user can modify the default setpoints. The default value of 6K permits changing the setpoint in a range of -3 to +3 K. The setpoint shift value is applied to the occupied and standby setpoints.

7.4.3 Energy Holdoff

The energy holdoff function prevents heating or cooling in case of open windows. If a window contact sensor indicates an open window, the HVAC controller will select the protection setpoints for this zone. Thus, the energy holdoff function works only if at least one window contact sensor is available in the HVAC zone. Table 31 shows the parameters of the energy holdoff function.

Parameter Name	Datapoint	Type	Default	Description
Energy Holdoff On Delay	hcEnergyHoldOff DelayOn	Real	30 s	Time between window open and energy holdoff alarm
Energy Holdoff Off Delay	hcEnergyHoldOff DelayOff	Real	300 s	Time between window close and energy holdoff alarm clear

Table 31: Energy Holdoff Parameters

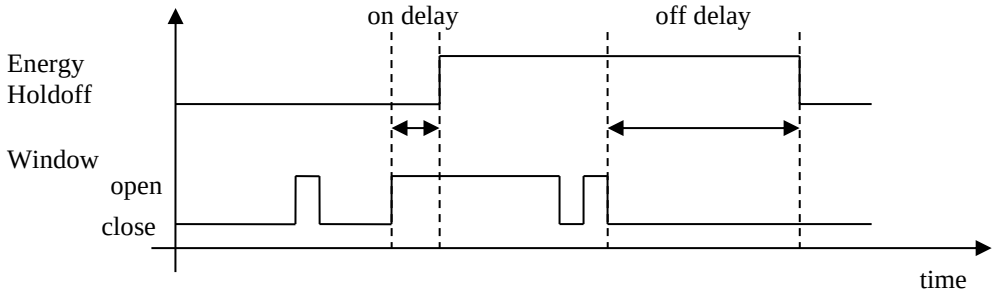


Figure 18: Energy holdoff timing.

Figure 18 shows the timing of the energy holdoff function. If a window is opened only for a short time, the energy holdoff function is not activated. Likewise, during energy holdoff, closing the window for a short time, retriggers the off timer.

Energy Holdoff On Delay (hcEnergyHoldOffDelayOn)

The energy holdoff on delay parameter determines how long the window has to be open to activate the energy holdoff mode. Setting this parameter larger than 0 seconds allows opening the window for a short time without disabling the HVAC controller.

Energy Holdoff Off Delay (hcEnergyHoldOffDelayOff)

The energy holdoff off delay parameter determines how long the window has to be closed to deactivate the energy holdoff mode. It allows the room to heat up for some time (e.g. from walls which to do not cool off that fast) befort the HVAC controller starts again.

7.4.4 Summer Compensation

The summer compensation function raises the standby and occupied setpoints on hot days. It improves energy efficiency due to less cooling demand. Further, as difference between indoor and outdoor temperature is lowered, comfort when entering or leaving the building is increased.

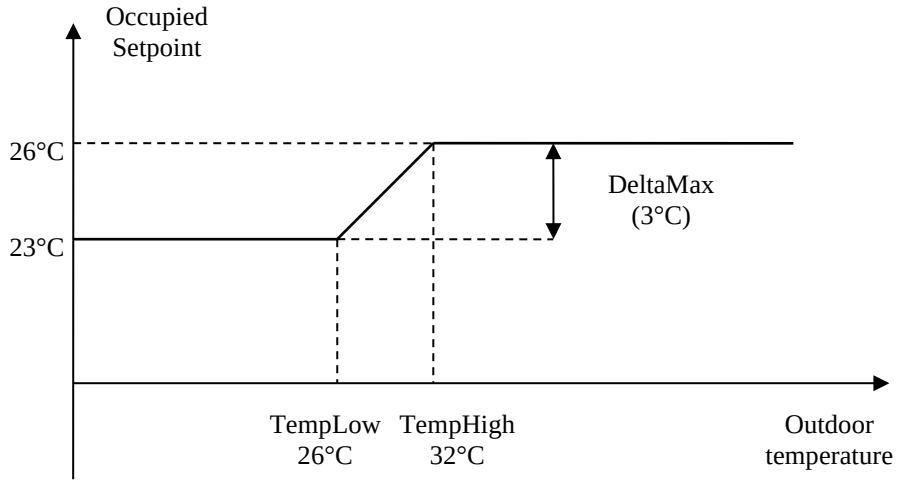


Figure 19: Summer Compensation.

Figure 19 shows the interaction between the outdoor temperature and the occupied cooling setpoint. The setpoint is raised by the hcSummerCompDeltaMax parameter within the outdoor temperature range from *hcSummerCompTempLow* to *hcSummerCompTempHigh*. Table 32 lists the summer compensation parameters.

Parameter	Datapoint	Type	Default	Description
Summer Compensation Active	hcSummerComp Active	Bool	FALSE	Enables summer compensation
Summer Compensation Low Temperature	hcSummerComp TempLow	Real	26 °C	Outdoor temperature at which summer compensation starts
Summer Compensation High Temperature	hcSummerComp TempHigh	Real	32 °C	Outdoor temperature for maximum summer compensation
Summer Compensation Delta Max	hcSummerComp DeltaMax	Real	3 K	Maximum cooling setpoint increase

Table 32: Summer Compensation parameters

Summer Compensation Active (hcSummerCompActive)

This binary parameter activates the summer compensation function.

Summer Compensation Low Temperature (hcSummerCompTempLow)

The summer compensation low temperature defines the temperature at which the summer compensation function starts to raise the occupied cool setpoint.

Summer Compensation High Temperature (hcSummerCompTempHigh)

The summer compensation high temperature defines the temperature at which the offset is raised by hcSummerCompDeltaMax degrees.

Summer Compensation Delta Max (hcSummerCompDeltaMax)

The summer compensation delta max parameter defines by which range the occupied cool setpoint is modified with raising outdoor air temperature.

7.4.5 Minimum Heating

The minimum heating function is used to provide a minimum warm air flow to glass facades. With minimum heating, the heating actuators are active, even if there is no actual heat demand. The minum heating function creates a warm airflow which heats the glass facade slightly. The warm interior surface prevents that cold air drops from the glass facade to the occupants. Figure 20 shows the minimum heating function. It is controlled by the outdoor air temperature and raises the minimum of the heating output with decreasing outdoor air temperature up to a configurable limit. The minimum heating function is only activated when the HVAC zone is in standby or occupied state.

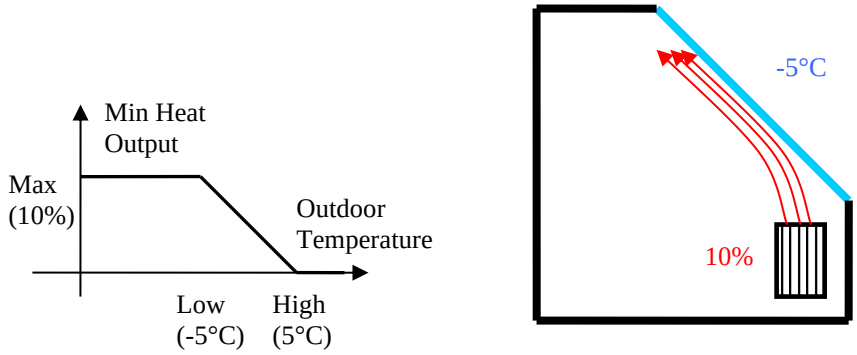


Figure 20: Minimum heating function

Parameter View	Datapoint	Type	Default	Description
Minimum Heating Low Temperature	hcMinHeatOutdoorTempLow	Real	-5 °C	Outdoor temperature at which minimum heat is fully applied
Minimum Heating High Temperature	hcMinHeatOutdoorTempHigh	Real	5 °C	Outdoor temperature at which minimum heat starts
Minimum Heating Max Limit	hcMinHeatLimitMax	Real	10%	Amount of heat value applied below hcMinHeatOutdoorTempLow

Table 33: Minimum Heating Parameters

Table 33 shows the available minimum heating parameters.

Minimum Heating Low Temperature (hcMinHeatOutdoorTempLow)

The minimum heating function reaches its max limit at this outdoor air temperature.

Minimum Heating High Temperature (hcMinHeatOutdoorTempHigh)

The minimum heating function starts raising the heat output minimum at this outdoor air temperature.

Minimum Heating Max Limit (hcMinHeatLimitMax)

The minimum heating max limit parameter determines the minimum heat output at the minimum heating low outdoor temperature.

7.4.6 Optimum Start

The optimum start function interacts with the occupancy scheduler. It observes the scheduled heating/cooling sequences, determines the heating/cooling ramp (K/h) and stores this value persistently. Heating and cooling ramps are stored separately. Next day, the heating sequence will start earlier so that the scheduled temperature is reached at the scheduled time.

In order to work, the scheduler must provide time-to-next and next-state information so that the controller can anticipate the heating/cooling ramp.

Table 34 shows the configuration parameters for the optimum start function.

Parameter Name	Datapoint	Type	Default	Description
Optimum Start Mode	hcOptStartMode	Enum	NORMAL	Optimum start mode
Optimum Start Heat Time Limit	hcOptStartMaxHeatHours	Real	0 h	Maximum preheat time
Optimum Start Cool Time Limit	hcOptStartMaxCoolHours	Real	0 h	Maximum precool time.

Table 34: Optimum Start Parameters

Optimum Start Mode (hcOptStartMode)

The *hcOptStartMode* parameter allows the following operation modes:

- NORMAL (1): Optimum start function is active.
- DISABLED (2): Optimum start function is disabled.

Optimum Start Heat Time Limit (hcOptStartMaxHeatHours)
Optimum Start Cool Time Limit (hcOptStartMaxCoolHours)

This parameters limit the calculated preheat/precool time. This is to prevent overly long preheat/precool times.

7.4.7 PI Controller

The PI controller is the core of the room comfort controller. It contains two separate PI controllers, one for heating and one for cooling. The PI controllers calculate heat and cool output from the measured room temperature and current setpoint.

Heating and cooling sequence have separate PI controllers, so that they can be parameterized differently.

Parameter Name	Datapoint	Type	Default	Description
PI Cool Proportional Gain	hcPICoolKp	Real	20	Proportional gain of heating controller
PI Cool Integral Time	hcPICoolTi	Real	300s	Integral gain of heating controller
PI Heat Proportional Gain	hcPIHeatKp	Real	20	Proportional gain of cooling controller
PI Heat Integral Time	hcPIHeatTi	Real	300s	Integral gain of cooling controller
PI Dead Zone	hcPIDeadZone	Real	0°C	Deadband for PI controllers to avoid control noise.
PI Pre Heat Level	hcPIPreHeatLevel	Real	100 %	Heat output during optimum start heat
PI Pre Cool Level	hcPIPreCoolLevel	Real	100 %	Cool output during optimum start cool

Table 35: PI Controller Parameters

Table 35 shows the available PI controller parameters. The control output is shown in Figure 21.

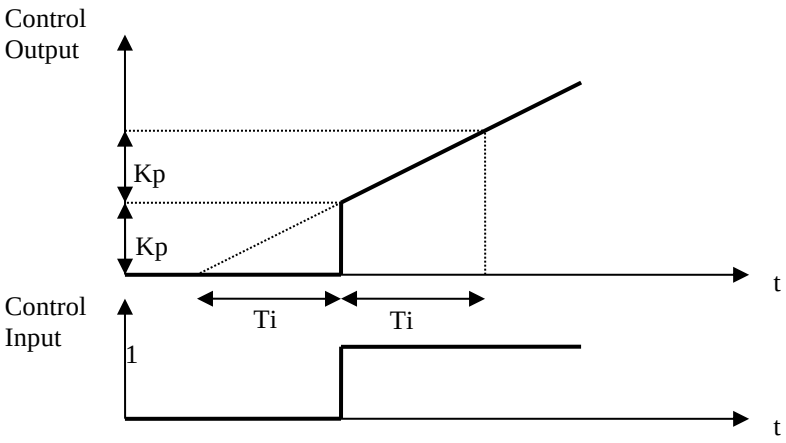


Figure 21: PI Controller step response

PI Cool Proportional Gain (hcPICoolKp)
PI Heat Proportional Gain (hcPIHeatKp)

The proportional gain parameters define the direct reaction to the controller input (which is the difference between the room temperature and the setpoint). If Kp is 20 then a input change by 1 °C results in a control output of 20%. The proportional band is calculated by 100%/Kp. In the example it requires a difference between the room temperature and the setpoint of 5 °C to reach 100% control output.

PI Cool Integral Time (hcPICoolTi)
PI Heat Integral Time (hcPIHeatTi)

The integral time determines (also known as reset time) determines the influence of the integral component to the controller output. In the step response it represents the time the integral component needs to reach the same amount as the proportional component.

PI Dead Zone (hcPIDeadZone)

The dead zone parameter defines a dead band in which the control input is considered zero. It is used to silence the controller at zero input. Usually, the default value of zero (no deadback) can be used for the HVAC controller, as there is a deadband between the heating and cooling temperature ranges anyway.

PI Parameter influence to	Response	Overshoot
Proportional Gain	Faster	More
Integral Time	Slower	Less

Table 36: PI Parameter Influence

Table 36 shows the influence of the PI parameters to the step response. For heating/cooling purposes, the proportional gain parameter should be set between 20 (5°C for 100% output) and 50 (2°C for 100% output). The integral time parameter should be adjusted to the time periods of the heating/cooling systems. Slower systems, such as floor heating have large time constants, while fan-coil based systems have short time constants.

PI Pre Heat Level (hcPIPreHeatLevel)
PI Pre Cool Level (hcPIPreCoolLevel)

The *hcPIPreHeatLevel* and *hcPIPreCoolLevel* is the cool controller output when the controller is preheating/cooling due to the optimum start logic. It can be set lower than 100% to avoid overshoot.

7.4.8 Indoor Air Quality Control

The HVAC module supports occupancy and CO2 based indoor air quality control (IAQ).

The IAQ controller calculates a relative air demand request from occupancy and CO2 level (0-100%). Every segment can have VAV I/O modules which translate the relative demand requests into absolute air demands (m³/h). The zone aggregates these requests (and measured values) and so that the air balance can be monitored in the HMI.

Parameter Name	Datapoint	Type	Default	Description
IAQ Occupied PPM Min	hcFlowIaqMin	Real	600 ppm	CO_2 level at which min. flow is demanded
IAQ Occupied Flow Min	hcFlowOccMin	Real	60%	Minimum occupied air flow demand.
IAQ Occupied PPM Max	hcFlowIaqMax	Real	1200 ppm	CO_2 level at which max. flow is demanded
IAQ Occupied Flow Max	hcFlowOccMax	Real	100%	Maximum occupied air flow demand
IAQ Unoccupied Flow	hcFlowUnocc	Real	20%	Unoccupied Air flow demand

Table 37: IAQ Controller Parameters

Table 37 shows the parameters of the IAQ controller.

IAQ Occupied PPM Min (hcFlowIaqMin)
IAQ Occupied Flow Min (hcFlowOccMin)

IAQ Occupied PPM Max (hcFlowIaqMax)
IAQ Occupied Flow Max (hcFlowOccMax)
IAQ Unoccupied Flow (hcFlowUnocc)

Figure 22 shows the parameterization of the IAQ controller. If the zone is unoccupied, the air demand is fixed by the *hcFlowUnocc* parameter.

For an occupied zone, the demand is calculated from the CO_2 level. If the CO_2 level is below *hcFlowIaqMin*, the controller demands relative air flow given by *hcFlowOccMin*. If the the CO_2 level is above *hcFlowIaqMax*, the controller demands relative air flow given by *hcFlowOccMax*. Between those points, the curve is extrapolated in a linear way.

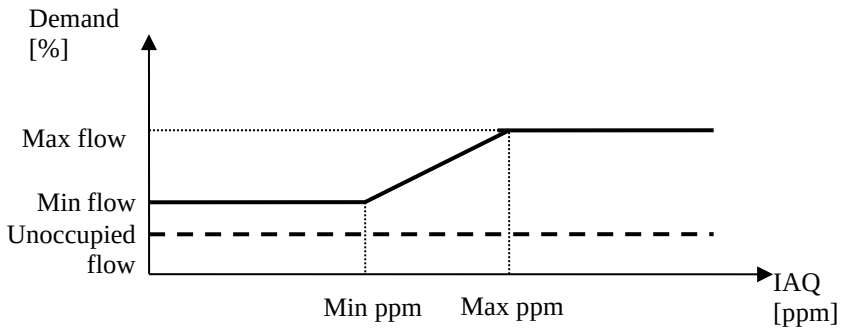


Figure 22: IAQ controller curve

7.5 Actuators

7.5.1 Multivalve Actuator

The multivalve I/O block is the standard L-ROC heating/cooling actuator. It supports a wide range range of applications, such as

- Radiators
- Ceiling actuators
- Floor actuators
- Fancoil valves

Table 38 lists the available favorites of the multivalve block. The favorites are to be connected to LIOB I/Os.

For analog valve outputs the following options are available:

- Thermal valves on TRIAC outputs (PWM)
- Analog outputs (0-10V)
- Binary outputs can be used on relay and TRIAC outputs.

The hcDrive… favorites are intended for floating valves. They signal the drive times for opening and closing the valve. The feedback inputs can be optionally used to indicate stop positions.

Favorite	Type	Description
hcAnalogHeat	Analog	Analog valve: Heat value (0-100%)
hcAnalogCool	Analog	Analog valve: Cool value (0-100%)
hcBinaryHeat	Binary	Binary valve: Heat valve (e.g. for fan/pump release)
hcBinaryCool	Binary	Binary valve: Cool valve (e.g. for fan/pump release)
hcSixWayValve	Analog	6-way valve: Valve output for 6-way valves (0-10V)

Table 38: Multivalve favorites

Table 39 lists examples how the outputs of the multivalve block can be used in different heating/cooling applications. The heating and cooling sequences can be used for the same actuator (e.g. 4-pipe chilled ceiling) or different actuators (e.g. radiator + fancoil). The binary valves can be used for pump or fan release signals. The binary signals are asserted when the analog value is greater than 1%. This is to prevent small control deviations to switch on a connected pump or fan device.

System	hcAnalog Heat	hcAnalog Cool	hcBinary Heat	hcBinary Cool	hcSixWay Valve
Radiator	Radiator	-	-	-	-
Changeover ceiling	Heat valve	-	Heat return	-	-
4-pipe ceiling	Heat valve	Cool valve	Heat return	Cool return	-
Radiator + Chilled ceiling	Radiator	Chilled Ceiling	Heat return	Cool return	-
4-pipe chilled ceiling with 6-way valve	-	-	-	-	0-10V
Radiator heat Fancoil cool	Radiator	Fancoil	-	-	-

Table 39: Multivalve application examples

Table 40 lists the parameters for the Multivalve actuator.

Parameter	Datapoint	Type	Default	Description
Valve Mode	hcMulti ValveMode	Enum	2 LINE (1)	The valve mode selects between 2 line, 4 line and crossover system
Valve Deadtime	hcMultiValve Deadtime	Real	360 s	Dead time for switching between heating and cooling valves
Valve Service Interval	hcMultiValve ServiceInterval	Real	30 d	hcMultiValveServiceIntervall
Valve Service Duration	hcMultiValve ServiceDuration	Real	180 s	Valve service cycle open time in seconds
Valve Output at Full Cool	hcMultiValve YCool100	Real	2	Output value for 6-way valve for 100% cool
Valve Output at Zero Cool	hcMultiValve YCool0	Real	5	Output value for 6-way valve for 100% cool
Valve Output at Zero Heat	hcMultiValve YHeat0	Real	7	Output value for 6-way valve for 0% heat
Valve Output at Full Heat	hcMultiValve YHeat100	Real	10	Output value for 6-way valve for 100% heat
Valve Location	hcMultiValve Location	String	""	Valve location
Valve Lock if Fancoil is Manual Off	hcMultiValve LockIfFanManOff	Enum	OFF (1)	Turns off valve when fancoil is manually off

Table 40: Multivalve parameters

Multivalve Mode (hcMultiValveMode)

The multivalve mode selects how the heat/cool outputs are used as shown in Table 41. In 2-pipe and 4-pipe mode, the heat and cool valves are used in the heat and cool sequences. 2-pipe and 4-pipe mode behave the same way.

In changeover mode, the heat/cool valves are used for both, heating and cooling output (and can be used simultaneously).

This parameter has no influence on the 6-way valve output.

Valve Mode	Operation	Heat Valve	Cool Valve	6way Valve
2-PIPE (1)	Heat 100%	100%	0%	10V
2-PIPE (1)	Cool 100%	0%	100%	2V
CHANGEOVER (2)	Heat 100%	100%	100%	10V
CHANGEOVER (2)	Cool 100%	100%	100%	2V
4-PIPE (3)	Heat 100%	100%	0%	10V
4-PIPE (3)	Cool 100%	0%	100%	2V

Table 41: Multivalve mode parameter (with default parameters)

Valve Deadtime (hcMultiValveDeadtime)

The deadtime parameter determines a how long heat and cool valves are closed when switching between heat and cool sequence.

Valve Service Interval (hcMultiValveServiceInterval)
Valve Service Duration hcMultiValveServiceDuration)

The service interval defines after which time the valves are opened for the service duration period. This is to allow flushing the pipe system from time to time. After the service interval, the heat and cool valve are each opened for the service duration period. For the service function to work, the heating system (e.g. pumps) has to be activated.

Valve Output at Full Cool (hcMultiValveYCool100)
Valve Output at Zero Cool hcMultiValveYCool0)
Valve Output at Zero Heat (hcMultiValveYHeat0)
Valve Output at Full Heat (hcMultiValveYHeat100)

These factors define the sequence of the 6-way valve output, as shown in Figure 23. The

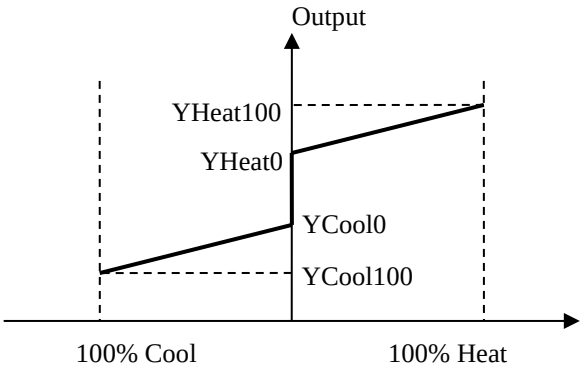


Figure 23: 6-way valve parameters

Multivalve Location (hcMultiValveLocation)

The multivalte location parameter stores a location identifier of the multivalve. It is for informational purposes only.

Multivalve Lock if Fan Manual Off (hcMultiValveLockIfFanManOff)

The lock if fan manual off parameter can be set to the following values:

- OFF (1): No locking
- HEAT (2): Lock only in heat sequence.
- COOL (3): Lock only in cool sequence.
- HEAT+COOL (4): Lock in heat and cool sequence.

This parameter locks the valve to 0% if the fan is manually disabled. This prevents that a fancoil has an open valve with an intentionally turned off fan.

7.5.2 Fancoil Actuator

The fancoil block allows to control up to 5 fan or heater devices which binary or analog inputs. A typical application is a 3-stage fan with an analog valve. Typical applications are:

- 3-stage fan with relays
- Analog fan with 0-10V input

Favorite	Type	Description
active	Binary	Signal if any fan stage is enabled
stage1	Binary	Enable fan stage 1
stage2	Binary	Enable fan stage 2
stage3	Binary	Enable fan stage 3
stage4	Binary	Enable fan stage 4
stage5	Binary	Enable fan stage 5
analog	Analog	Output for analog fan (0..100%)

Table 42: Fancoil favorites

Table 42 lists the Favorite data points which are used to connect the outputs of the fancoil actuator to LIOB or other datapoints.

The datapoint “active” is on when any fan state is enabled. The fan stage favorites stage1 … stage5 are exclusive, so in stage 3 only stage3 is enabled. The analog favorite is used for 0-10V outputs.

Parameter	Datapoint	Type	Default	Description
Fancoil Stages	hcFanCoil Stages	Analog	3	number of fan coil stages (1 ... 5)
Fancoil Min Stage Time Up	hcFanCoil MinStageTime Up	Analog	3 s	minimum time before switching up to the next stage
Fancoil Min Stage Time Down	hcFanCoil MinStageTime Down	Analog	0 s	minimum time before switching down to the previous stage
Fancoil Auto Start Delay Time	hcFanCoilAuto StartDelayTime	Analog	0 s	delay time before starting the fan in auto mode
Fancoil Mode	hcFanCoilMode	Multi-state	OFF (1)	Selects if fan is active in heat/cool mode
Fancoil Cool Stage 1	hcFanCoilCool Stage1	Analog	1 °C	Temperature difference for activating first cool stage
Fancoil Cool Stage N	hcFanCoilCool StageMax	Analog	3 °C	Temperature difference for activating last heat stage
Fancoil Cool Hysteresis	hcFanCoilCool Hysteresis	Analog	25%	Hysteresis in percent of stage width when switching down
Fancoil Heat Stage 1	hcFanCoilHeatStage1	Analog	1	Temperature difference for activating first heat stage
Fancoil Heat Stage N	hcFanCoilHeatStageMax	Analog	3	Temperature difference for activating last heat stage
Fancoil Heat Hysteresis	hcFanCoilHeat Hysteresis	Analog	25%	Hysteresis in percent of stage width when switching down

Table 43: Fancoil parameters

Table 43 shows the fancoil parameters.

Fancoil Stages

The fancoil stages parameters select how many stages a fancoil uses. Possible values are 1,2,3,4 or 5.

Fancoil Min Stage Time Up
Fancoil Min Stage Time Down

These parameters define how long the fancoil remains in the current state before going to the next stage. This can be used to distribute switching over some time or to allow cooling down the actuator when turning off.

Fancoil Auto Start Delay Time

This parameter defines how long the fancoil will remain turned off in automatic mode. This prevents blowing unconditioned air during the warm-up/cool-down phase of the fancoil actuator.

Fancoil Mode

This parameter defines the heating/cooling modes in which the fancoil actuator is active:

- OFF (1): The fancoil is off.
- HEAT (2): The fancoil is active in heat mode.
- COOL (3): The fancoil is active in cool mode
- HEAT+COOL (4): The fancoil is active in heat and in cool mode.

Fancoil Heat/Cool Stage 1
Fancoil Heat/Cool Stage N
Fancoil Heat/Cool Hysteresis

These parameters can be set independently in heat/cool mode and determine the sequence of the fancoil actuator. The fancoil stage is calculated by the difference between the space temperature and the setpoint. Figure 24 shows the stages and analog outputs over the delta temperature.

The Stage1 parameter determines at which delta temperature the first stage is activated. For the analog output it determines at which delta temperature starts rising.

The StageN parameter determines at which delta temperature the maximum stage is activated. For the analog output it determines at which delta temperature the analog output reaches 100%.

The hysteresis parameter determines the off-hysteresis in percent of the step-width. When switching up, the stages are aligned with the configured Stage1 and StageMax parameters. When switching down, the stage is deactivated the configured percentage below.

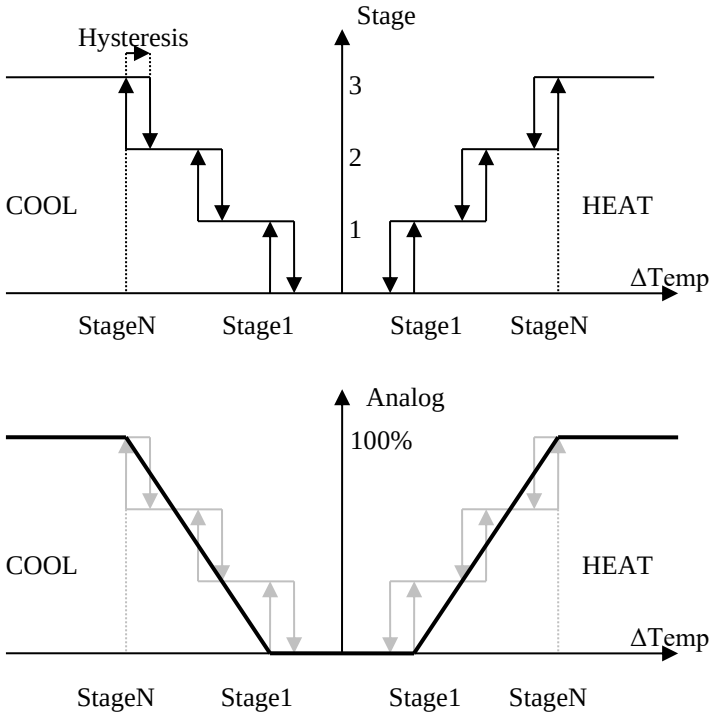


Figure 24: Fancoil sequence.

7.5.3 VAV Actuator

The VAV actuator translates the relative air demand from the IAQ controller into an absolute supply or exhaust air demand.

Care must be taken to balance supply and exhaust air in flexible buildings. If rooms are to be joined their supply and exhaust air demands need to remain balanced. This will automatically happen if every segment has a supply and exhaust VAV box. However care must be taken if some segments have only supply boxes while other have only exhaust boxes.

The HVAC controller automatically calculates summarized supply and exhaust air flows (in case air flow sensors are available), so that the balance can be easily monitored.

The VAV actuator consists of two identical damper blocks, *Supply* and *Exhaust*.

The following favorites are provided by each block

Favorite	**Type**	**Description**
hcFlowCmd	Analog	Requested air flow
hcFlowFbPct	Analog	Feedback from VAV box

Table 44: VAV actuator favorites

The hcFlowCmd is the value sent to the physical VAV box. This can be a real air flow or a relative air flow depending on the *hcFlowmaxNominal* parameter.

The hcFlowFbPct favorite is a feedback datapoint so that the HMI can display the real flow.

The following parameters are similar for both blocks:

Parameter	Datapoint	Type	Default	Description
VAV Nominal Flow	hcFlowmaxNominal	Analog	100	Nominal air flow at 100%

Table 45: VAV actuator parameters

VAV Nominal Flow (hcFlowmaxNominal)

This parameter defines the requested absolute air flow when the IAQ controller requests 100% air flow. Its default value (100) is used if the VAV equipment expects a relative air flow (100% output on 100% input). If the VAV equipment interface requires a physical air flow, e.g. 1000 m³/h, the value would be set to 1000 (1000 m³/h at 100% input).

7.6 Heating Server

The heating system interface provides functions to efficiently integrate room control with heating/cooling systems. It provides the following functions:

- Aggregation of maximum heat/cool request
- Reporting average heat/cool requests
- Reporting average temperature/setpoint deviations
- Demanding heat/cool power supply, e.g. to turn on pumps or boilers
- Provide supply feed temperature to the HVAC controllers
- Override HVAC modes, e.g. heat only or cool only
- Support changeover valves for 2-pipe systems
- Hierarchical external scheduler

The heating system interface can be used on every hierarchy level: area, floor or building level. For example, a changeover valve can be controlled per area, per floor or per building dependent on the design of the heating/cooling system.

The L-ROC system supports several heating/cooling systems in parallel. Each area/floor/building coupler needs to have as many heating system blocks as there are heating systems. For example, a building having a radiator heat and fancoil cool system needs two heating system logic blocks.

7.6.1 External I/O Interface

Favorite	Type	Description
IO/		
HeatAvailableInput	Binary	Indicates active heat supply
CoolAvailableInput	Binary	Indicates active cool supply
HeatRequestOutput	Binary	TRUE when heat supply is requested
HeatRequestInput	Binary	TRUE when cool supply is requested
HeatSupplyTemp	Real	Heating supply temperature
CoolSupplyTemp	Real	Cooling supply temperature

Table 46: Heating/cooling favorite data points

Table 46 shows the favorite data points of the heating server block.

HeatAvailableInput
CoolAvailableInput

These inputs can be used for physically integrating the heating/cooling system with the L-ROC system. If the inputs are TRUE, the corresponding energy sources are available.

HeatRequestOutput
CoolRequestOutput

These outputs indicate that at least one HVAC controller requests heating or cooling energy. It can be used to activate or release the heating system to normal operation.

HeatSupplyTemp
CoolSupplyTemp

These inputs are used to provide the L-ROC system with the supply heat or cool temperature.

External Scheduler

Favorite	Type	Description
ExtSched/IO/		
state	Enum	Current scheduled occupancy
nextState	Enum	Next scheduled occupancy
timeToNext	Minutes	Minutes to next scheduled occupancy state
priority	Real	Priority 0..15 (Low value is high priority)

Table 47: Heating/cooling system external scheduler favorite data points

Table 47 shows the favorite data points for the external scheduler feature. It allows setting occupancy states centrally for the building, a floor or an area. The occupancy states are encoded as follows:

- NULL (1): External scheduler off
- UNOCCUPIED (2): Unoccupied state
- STANDBY (3): Standby (economy) state
- BYPASS (4): Temporary occupancy
- OCCUPIED (5): Occupied state

The external states employ the current state data point (which is used for control), the nextState data point which is used for the optimum start algorithm, and the timeToNext data point which indicates when the next state will be valid.

The priority data point can be used to fine tune the external scheduler setpoints. It can have values from 0 to 15 and is similar to the BACnet priority (high values are low priority).

For example: If the external schedule input of the building is set to UNOCCUPIED (priority 6) and the external schedule input of an area is set to OCCUPIED (priority 3), this area will be in OCCUPIED state.

External Flow Override

The local air quality control can be overridden on building/floor/area level with the inputs shown in Table 48.

Favorite	Type	Description
ExtSched/IO/		
mode	Enum	Override mode
override	Percent	Setpoint when mode=OVERRIDE
priority	Real	Priority 0..15 (Low value is high priority)

Table 48: Flow override favorite data points.

The mode parameter supports the following values:

- NULL (1): No override
- AUTO (2): Normal operation
- OFF (3): Ventilation off
- UNOCC (4). Ventilation set to unoccupied setpoint
- OCC_MIN (5): Ventalation set to minimum occupied setpoint
- OCC_MAX (6): Ventilation set to maximum occupied setpoint
- OVERRIDE (7): Ventilation set to value taken from *override* favorite.

7.6.2 Communication

Heating servers talk to each other between the building, floor and area level when they belong to the same heating system. This can be configured by the heating circuit IDs. All data on HVAC modes and heat/cool demands is then automatically communicated.

Figure 25 shows a building with assigned heating system IDs. The segments report their data to HC1 which is aggregated by the heating server in their area manager. The area managers aggregate their data to the floor level of the HC1 circuit which is called HC1F. The floor managers then aggregate their data to the building level. The circuit names should be choosen to be short. It is necessary to add a suffix to the circuit names on the floor and building level to make the IDs unique in the system.

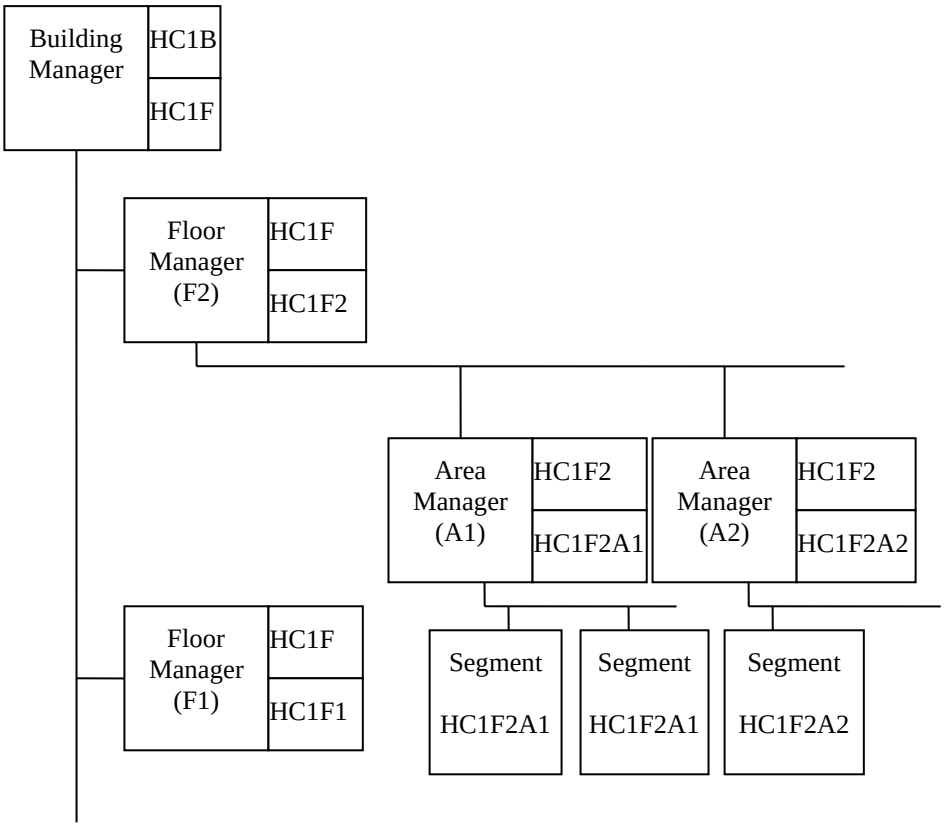


Figure 25: Heating Circuit IDs for heating circuit 1 (HC1).

Parameter (Aggregation/Param)	**Datapoint**	**Type**	**Default**	**Description**
Circuit ID Up	hsCircuit IdUp	String	""	ID of upstream circuit
Circuit ID Down	hsCircuit IdDown	String	""	ID of downstream circuit
Circuit Pipe System	hsCircuit PipeSystem	Multi-state	2PIPE (1)	Pipe system below this circuit server

Table 49: Circuit server parameters

Table 49 shows the basic parameters of the circuit server.

Circuit ID Up
Circuit ID Down

These IDs designate the upstream and downstream heating system. Figure 25 shows a system with assigned IDs. The upstream ID of a lower-level circuit server must match the downstream ID of the higher-level server for the system to work.

Circuit Pipe System

The pipe system parameter selects the pipe system below the circuit server. So for example on an area manager is defines whether the area is a 2-pipe or 4-pipe system. It mains enables the changeover function in the 2-pipe case.

7.6.3 Aggregation Function

The aggregation function calculates heat and cool demands from the underlying HVAC controllers. It is available on the area, floor and building level. The gathered information is used for:

- Trends and reporting
- Automatic changeover
- Heating/cooling system optimization

Parameter (Aggregation/Param)	Datapoint	Type	Default	Description
Aggregation Heat Weight	hsCircuit HeatWeight	Analog	1.0	Weighting factor for aggregated heat values (default 1)
Aggregation Cool Weight	hsCircuit CoolWeight	Analog	1.0	Weighting factor for aggregated cool values (default 1)

Table 50: Aggregation parameters

Table 50 shows the parameters of the aggregation function.

Aggregation Heat Weight (hsCircuitHeatWeight)
Aggregation Cool Weight (hsCircuitHeatWeight)

These weights allow adjusting the influence of this heating server in its superordinate heating server.

For example if all areas in a floor have the same weight factor, the floor aggregates will be the average of the area aggregates. If an area weight factor is 0.0, the area will not be accounted at all.

Value	Datapoint	Type	Description
Total Count	TotalCount	Analog	Number of heat/cool controllers
Heating Count	HeatingCount	Analog	Number of controllers in heat mode
Cooling Count	CoolingCount	Analog	Number of controllers in cool mode
Weighted Heat Temp Deviation	WeightedHeat TempDeviation	Analog	Average deviation from heat setpoint
Weighted Cool Temp Deviation	WeightedCool TempDeviation	Analog	Average deviation from cool setpoint
Weighted Heat Request	WeightedHeat Request	Analog	Average heat request
Weighted Cool Request	WeightedCool Request	Analog	Average cool request
Max Heat Request	MaxHeat Request	Analog	Maximum Heat Request
Max Cool Request	MaxCool Request	Analog	Maximum Cool Request

Table 51: Aggregation values

Table 51 shows the list of aggregation values.

Total Count (TotalCount)

This value indicates the number of HVAC controllers below the heating server.

Heating Count (HeatingCount)

This value reads the number of HVAC controllers in heat mode. A value of zero indicates no heat demand.

Cooling Count (CoolingCount)

This value reads the number of HVAC controllers in cool mode. A value of zero indicates no cool demand.

Weighted Heat Temp Deviation (WeightedHeatTempDeviation)

This value is a measure for heat demand and depends on the differences between the space temperatures and heat setpoints.

This value is the weighted average of the differences between the space temperatures and the setpoints if that difference is positive.

If a segment is warmer than the setpoint, it is not taken into consideration.

This value is independent on the PI control parameterization. Values greater than zero indicate heat demand.

Weighted Cool Temp Deviation (WeightedCoolTempDeviation)

This value is a measure for cool demand and depends on the differences between the space temperatures and cool setpoints.

This value is the weighted average of the differences between the setpoints and the space temperatures if that difference is positive.

If a segment is cooler than the setpoint, it is not taken into consideration.

This value is independent on the PI control parameterization. Values greater than zero indicate cool demand.

Weighted Heat Request (WeightedHeatRequest)

This value is the weighted average of the heat outputs of the PI controllers.

Values greater than zero indicate heat demand.

Weighted Cool Request (WeightedCoolRequest)

This value is the weighted average of the cool outputs of the PI controllers.

Values greater than zero indicate cool demand.

MaxHeatRequest (MaxHeatRequest)

This value is the maximum of the heat outputs of the PI controllers.

Values greater than zero indicate heat demand.

MaxCoolRequest (MaxCoolRequest)

This value is the maximum of the cool outputs of the PI controllers.

Values greater than zero indicate cool demand.

7.6.4 Changeover Function

The changeover functions implements automatic and manual changeover for 2-pipe systems. Figure 26 shows a changeover system which has a 4-pipe system on building and floor level. At area level, it employs a 2-pipe system. The decision between heat and cool mode can be met for every area.

The L-ROC system supports changeover valve at area, floor and building level.

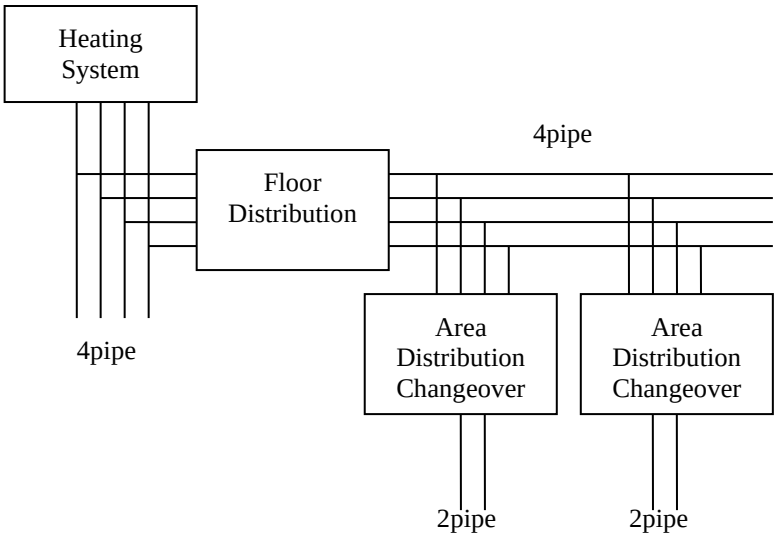


Figure 26: Changeover system

Table 52 shows the parameters for the changeover function.

Parameter (ChangeOver/Param)	Datapoint	Type	Default	Description
Circuit Changeover Mode	hsCircuit ChangeOver Mode	Multi state	AUTO_ TEMPDIFF (5)	Selects algorithm for change over
Circuit Changeover Delay	hsCircuit ChangeOver Delay	Analog	5s	Delay between switching between heat and cool mode
Circuit Changover Hysteresis	hsCircuit ChangeOver Hyst	Analog	0.1	Hysteresis for heat/cool changeovert (% or degrees)

Table 52: Changeover parameters

Circuit Changeover Mode

The mode parameter determines how the heat/cool decision is made:

MANUAL_OFF (1): Both heat/cool sequences are locked

MANUAL_HEAT (2): Changeover manually set to heat

MANUAL_COOL (3): Changeover manually set to cool

AUTO_DEMAND (4): The changeover decision is based on the average heat and cool demand decision. This mode is not recommended.

AUTO_TEMPDIFF (5): The changeover decision is based on the average difference between space temperature and setpoint.

In automatic modes, the changeover logic determines whether there are more HVAC controllers with heat or cool demand.

Circuit Changeover Delay

When changing between heat and cool mode, the application mode will be set to OFF for this amount of time. This allows for taking actuator and thermal times into consideration.

Circuit Changover Hysteresis

This hysteresis is applied to the selected automatic source to avoid having too many changeover events when heating and cooling demand is similar. A higher value makes changeover happen more unlikely.

7.6.5 Application Mode

The application mode function is responsible for selecting an application mode (auto/heat/cool/off) from the inputs and changeover logic. It automatically reports the resulting application mode to its subordinates (heating servers, HVAC controllers).

Table 53 shows the application mode parameters.

Parameter (ApplMode/Param)	Datapoint	Type	Default	Description
Circuit Mode Override	hsCircuitMode Override	Multi state	NULL (-1)	Override HVAC application mode. NULL to disable override
Circuit Mode Source	hsCircuitMode Source	Analog	AUTO_SLAVE (5)	Application mode source

Table 53: Application mode parameters

Circuit Mode Override (hsCircuitModeOverride)

The override parameter allows to manually set the application mode for this heating server and all subordinate heating servers and HVAC controllers.

It is the responsibility of the user to provide the correct application level.

Supported values are:

- NULL (-1): No override
- AUTO (0): Heating and cooling mode
- HEAT (1): Heating only
- COOL (3): Cooling only
- OFF (6): Heating and cooling turned off

Other values are currently not supported.

Circuit Mode Source (hsCircuitModeSource)

The circuit mode source parameter determines how the application mode is selected. The following options are available:

MAN_OFF (1): The application mode is set to OFF (6)

MAN_HEAT (2): The application mode is set to HEAT (1)

MAN_COOL (3): The application mode is set to COOL (3)

MAN_BOTH (4): The application mode is set to AUTO (0)

AUTO_SLAVE (5): The application mode is determined by the higher-level heating server.

AUTO_INPUT (6): The application mode is determined by the HeatAvailableInput and CoolAvailableInput favorite datapoints as shown in Table 54.

CoolAvailableInput	HeatAvailableInput	Mode
OFF	OFF	OFF (6)
OFF	ON	HEAT (1)
ON	OFF	COOL (3)
ON	ON	AUTO (0)

Table 54: HVAC mode mapping in AUTO_INPUT mode

The resulting application mode is indicated in the data points shown in Table 55.

Value	**Datapoint**	**Type**	**Description**
Circuit Application Mode	Circuit-Application-Mode	Multi state	Resulting HVAC mode
Circuit Heat Active	CircuitHeat-Active	Binary	Heating active
Circuit Cool Active	CircuitCool-Active	Binary	Cooling active

Table 55: Application mode values

7.6.6 Supply Temperature

The supply temperature feature provides the heating/cooling supply temperatures to the HVAC controllers.

Parameter (SupplyData/Param)	Datapoint	Type	Default	Description
Circuit Heat Supply Temp Source	hsCircuitHeat SupplyTempSource	Multi state	SLAVE (1)	Source of heat supply temperature
Circuit Cool Supply Temp Source	hsCircuitCool SupplyTempSource	Multi state	SLAVE (1)	Source of cool supply temperature

Table 56: Supply temperature parameters

Circuit Heat Supply Temp Source (hsCircuitHeatSupplyTempSource)
Circuit Cool Supply Temp Source (hsCircuitCoolSupplyTempSource)

These parameters select the source of the supply temperatures:

SLAVE (1): The heating server receives the supply temperature from its superordinate heating server.

INPUT (2): The heating server reads thes supply temperature from its HeatSupplyTemp and CoolSupplyTemp favorites and distributes them to its subordinate heating servers.

The resulting supply temperatures are displayed in the data points shown in Table 57.

Value	Datapoint	Type	Description
Heat Supply Temp	HeatSupplyTemp	Real	Heating supply temperatute
Cool Supply Temp	CoolSupplyTemp	Real	Cooling supply temperature

Table 57: Supply temperature values

8 Lighting Functions

8.1 Lighting Overview

The LROC library provides a solution for light applications. This includes room lighting, but can also be extended to staircases and special lighting applications in entry rooms or conference rooms.

The lighting application consists of the following components:

- The lighting application uses the brightness and occupancy sensors of the LROC segments.
- The lighting communication logic allows linking light controllers within rooms and also allows defining zones within rooms. Communication between the light controllers is defined by the controller parameterization and requires no further bindings.
- The lighting controller already provides manual mode, constant light control, automatic light and scene control. The controller can be extended by project-specific light controllers for special lighting demands.
- Various light actuators can be connected to the light controller. These can be DALI ballasts, favorites for I/Os or customized actuators, e.g. for KNX.

The light controller architecture is shown in Figure 27.

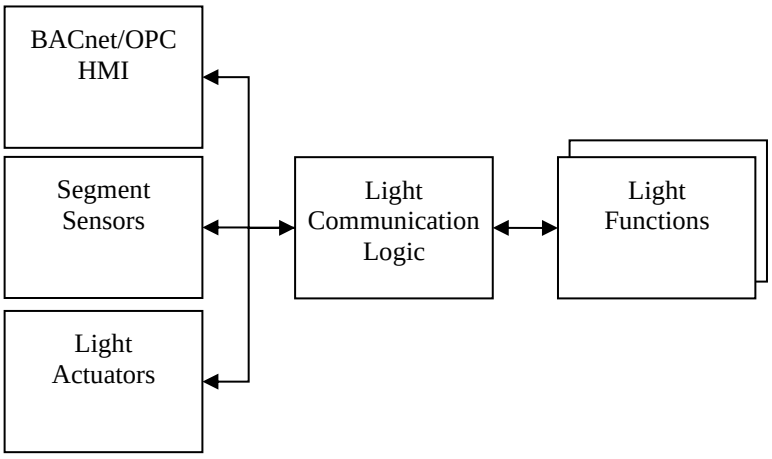


Figure 27: Light controller

If two or more light modules are configured to form a lighting zone, the following functions are automatically activated:

- The light controllers elect a master light module whose control output is used by all other (slave controllers). Usually the master light module is the first light module within the zone, leftmost in the L-ROC communication chain.

- All light modules, master and all slaves, accept HMI inputs. Slaves deliver their HMI inputs to the master module. If there are conflicts, the latest input is valid.

- Sensor input within the zone is aggregated. If any occupancy sensor reports occupancy, the entire zone is occupied. The light controller uses the average lux level from the available illumination sensor.

The zoning functions are depicted in Figure 28.

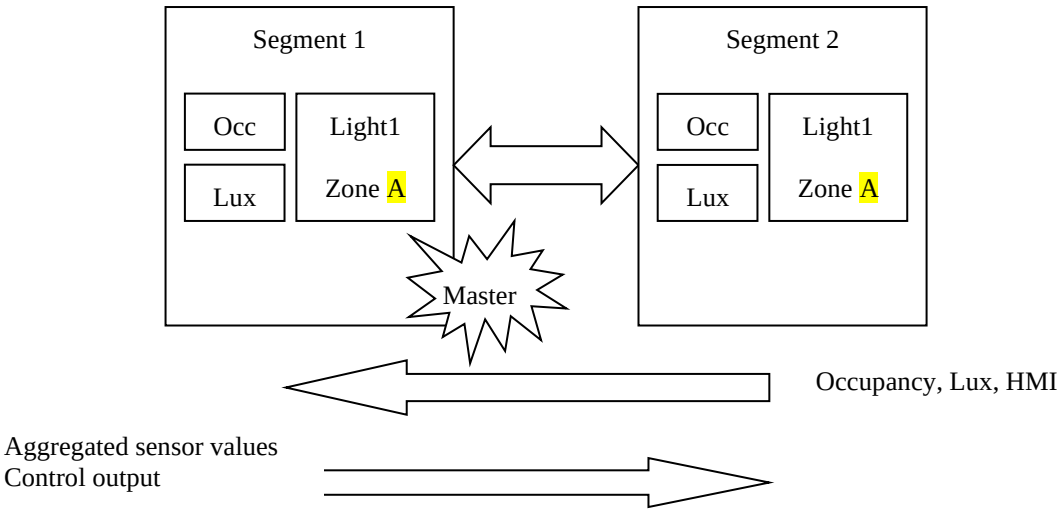


Figure 28: Light zone functions

8.2 OPC/BACNET Interface

Table 58 shows the data points available on the OPC and BACnet interface.

Datapoint	Type	Description
clZoneId	String	Zone ID write
clZoneIdFb	String	Zone ID feedback
modeZone	Analog	Deprecated, change mode mask in zone
modeRoom	Analog	Deprecated, change mode mask in room
modeFb	Analog	Deprecated, current mode mask
lampRoomState	Multistate	Switch on/off light in room
lampRoomValue	Analog	Set dimlevel in room
lampZoneState	Multistate	Switch on/off light in zone
lampZoneValue	Analog	Set dimlevel in zone
sceneZoneSave	Analog	Save scene in zone
sceneRoomSave	Analog	Save scene in room
sceneZoneRecall	Analog	Recall scene in zone
sceneRoomRecall	Analog	Recall scene in room
lampStateFb	Multistate	Light Feedback (on/off)
lampValueFb	Analog	Light Feedback (%)
occFb	Multistate	Occupancy feedback
luxLevelFb	Analog	Average zone lux level
lampRoomValueIncr	Analog	Increment register for room dimming
lampZoneValueIncr	Analog	Increment register for zone dimming
defaultMode	Analog	Deprecated, light mode on startup
activeModesFb	Analog	Bit mask showing enabled light functions
selectedModeFb	Analog	Number of active light function
autoModeZone	Multistate	Set zone to manual or auto mode
autoModeRoom	Multistate	Set room to manual or auto mode
autoModeFb	Multistate	Current zone mode (manual/auto)
masterFb	Binary	Indicates master controller
overrideFb	Analog	Indicates override value

Table 58: Lighting OPC/BACnet Interface

8.2.1 Zoning

Light modules can be grouped into zones. All light modules within a room having the same zone ID will operate synchronously. A master module will be automatically elected. All modules execute the same control algorithms, but only the master module is allowed to send the control output (light level) to the slave modules. User inputs received by slave modules are directed to the master module.

The *clZoneId* data point assigns the light module to a zone. The default value is A. The current zone ID is reflected by the *clZoneIdFb* data point.

8.2.2 Mode Selection and Feedback

The current light controller mode can be modified by the HMI. The mode is a 16-bit bitmask which enables up to 16 light functions from which the highest-priority function is selected. The default value for the mode is 65535 (hex FFFF) so that all functions are enabled by default.

Note that this interface is usually not necessary, as the modes can setup with static parameters. It is intended to control additional (project specific) light functions based on user input.

The *modeZone* data point sets a new mode for the light zone. The *modeRoom* data points sets a new mode for the room in which the light zone is in.

The current mode is reflected by the *modeFb* feedback data point.

The data point *defaultMode* reads the default Mode, so that one can revert to the default mode by within its value to the *modeZone* or *modeRoom* data point.

The *activeModesFb* data point shows the bit mask of all active modes, these are light modes that currently would like to control the light.

The *selectedModeFb* returns the number of the light function that has been selected from the active modes.

The *masterFb* data point indicates that the controller is the master controller for this lighting zone.

The *overrideFb* data point indicates (0..100%) an active override value or no override (-1).

8.2.3 Manual Dimm Control

The following data points are used to control a light zone or an entire room manually or to revert the room to automatic mode.

The inputs use separate states (0, 1) and values (0-100%), so that lights can be turned off without modifying the last dimm level.

The *lampRoomState* and *lampRoomValue* data points enable manual mode and sets state and value for the entire room. The *lampRoomValueIncr* register increments or decrements the current room light level by the amount written to this data points. It is used for relative dimming buttons. E.g. writing -10 to this register reduces the light level by 10%. Do not use heartbeats on this data point.

The *lampZoneState* and *lampZoneValue* data points enable manual mode and sets state and value for the light zone. The *lampZoneValueIncr* register increments or decements the current zone light level by the amount written to this data points. It is used for relative dimming buttons. E.g. writing -10 to this register reduces the light level by 10%. Do not use heartbeats on this data point.

The *autoModeZone* and *autoModeRoom* data point is used control the CLC in automatic mode:

- MANUAL (1): Manual override is activated
- AUTO (2): Manual override is deactivated
- AUTO_ENABLE (3): Trigger the CLC to turn on (in CLC mode with manual on)
- AUTO_RESET (4): Reset the CLC occupancy hold time (when unoccupied)
- AUTO_LEAVE (5): Disable the CLC and block it for the configured leave time.
- AUTO_DISABLE (6): Disable the CLC (until enabled again).

The current manual/auto mode is reflected in the *autoModeFb* data point.

8.2.4 Scene Control

The scene control data points allow saving and recalling up to 8 scenes.

The *sceneZoneSave* and *sceneRoomSave* data points save the current light level to the given scene number (1-8).

The *sceneZoneRecall* and *sceneRoomRecall* data points restore the saves light level. A scene number of -1 disables the scene controller. An unused scene results in 0% light level.

8.2.5 Lamp Feedback

The current level of the light zone is reflected by the *lampStateFb* and *lampValueFb* data points.

8.2.6 Sensor Feedback

The aggregated occupancy of the light zone is reflected in the *occFb* data point. The average light level can be read from the *luxLevelFb* data point.

8.3 Communication

8.4 Light Functions

8.4.1 Function Summary

The L-ROC library provides the lighting functions shown in Table 59.

Function Number	Function	Mode Value (Bitmask)	Default Priority	Default Enabled
1	Manual Dimming	1	8	Yes
2	Constant Light Control	2	5	Yes
3	Auto On/Off	4	4	No
4	Reserved	8	--	No
5	Scene Control	16	7	No

Table 59: Light functions

A light module can have more than one function enabled. In this case the light functions and the assigned priorities are used to determine the selected light function.

The default setup enables manual dim and and constant light control (CLC). If the zone is occupied and the illumination falls below a threshold, the CLC will become active. If a user turns on the light, the manual mode becomes active. In case of two active modes, the mode with the highest priority is selected. The selected light function then controls the light actuators. If the selected light function is not active anymore, the second highest priority is selected. If no light function is active, the default light level is used (typically 0%).

The CLC and Auto On/Off functions should not be enabled at the same time, as they have similar but not identical functions.

Table 60 shows the light module communication and core parameters.

Parameter Name	Datapoint	Type	Default	Description
Zone ID	clZoneId	String	A	Lighting Zone Id
Default Light Mode	clZoneDefaultMode	Real	65535	Default lighting mode (reset after clZoneDefaultDelay delay)
Default Mode Delay	clZoneDefaultDelay	Real	0 s	Delay from unoccupied state to default mode reset
Zone Location	clZoneLocation	String	""	Location of lighting zone
Enable Default Mode	clZoneDefaultEnable	Binary	TRUE	Enables the default state/value if no controller is active
Default Lamp State	clZoneDefaultState	Binary	FALSE	Default state if no controller is active
Default Lamp Value	clZoneDefaultValue	Binary	0%	Default state if no controller is active
Zone Type	clZoneType	String	""	Lighting Zone Type
Light Level Selection	clLightLevelSel	Enum	AVERAGE (1)	Function for Lux aggregation

Table 60: Lighting communication and base parameters.

Zone ID (clZoneId)

The zone id parameter defines the zone to which this light module belongs.

Default Light Mode (clZoneDefaultMode)
Default Mode Delay (clZoneDefaultDelay)

The default light mode and default mode delay parameters are used for special applications when the HMI is used to change the enabled modes.

For normal applications leave the default light mode at 65535 (all light functions enabled) and the default mode delay at 0 seconds.

If the mode delay parameter is non-zero, the default mode will be restored after the mode delay time after the zone got unoccupied. Thus a user could select a special light function, but the light module reverts to the default mode.

Zone Location (clZoneLocation)

The zone location parameter is an informational parameter for documention the light module location. It has no influence on control functions.

Enable Default Mode (clZoneDefaultEnable)
Default Lamp State (clZoneDefaultState)
Default Lamp Value (clZoneDefaultValue)

The enable default mode parameter (default on) determines the light module behavior when no light function is active. If enabled, the default state and values will be used. If it is disabled, the last light setting will remain until a light function becomes active again.

Normally these parameters can be left at their default values.

Zone Type (clZoneType)

The zone type parameter is an informational parameter for classifying zones with the same parameterization, e.g. office rooms or aisles. It has no influence on control functions. It can be used in the LWEB-900 parameter views to sort light modules with similar parameterization for easier mass configuration.

Light Level Selection (clLightLevelSel)

This parameter selects one of the following aggregation functions for Lux sensors in the Zone:

- AVERAGE (1): The zone uses the average Lux level.
- MINIMUM (2): The zone uses the minimum Lux level. Therefore all segments will be lighted properly, but some segments might be too bright.
- MAXIMUM (3): The zone uses the maximum Lux level. Segments will not exceed the setpoint. Some segments might be too dark.

8.4.2 Manual Dimm Function

The manual dimm function allows manually controlling the light zone. It is typically used with a room operation panel with one or two buttons for turning on/off lights or dimming them manually. Table 61 lists the parameters of the manual dimming mode.

Parameter Name	Datapoint	Type	Default	Description
Manual Priority	clPriorityMan	Real	8	Manual Light Control Priority
Manual On To Auto Delay	clManOnAutoDelay	Real	3600 s	Delay between not occupied and return to auto mode (Light on)
Manual Off To Auto Delay	clManOffAutoDelay	Real	3600 s	Delay between not occupied and return to auto mode (Light off)
Manual Mode	clManMode	Enum	ENABLED (2)	Selects Manual operation (disabled, enabled)

Table 61: Manual dimming parameters

Manual Priority (clPriorityMan)

The manual priority parameter selects the priority of the manual light function. Usually it should be higher than for automatic modes, as manual mode would have no effect else.

Manual On To Auto Delay (clManOnAutoDelay)

When lights are manually switched on and the zone becomes unoccupied for the manual on to auto delay time, it will deactivate the manual light function. The timer starts when the zone becomes unoccupied and stops if the room becomes occupied again within the expiry time. This function ensures that manually switched on lights revert to their default automatic mode in reasonable time (energy savings).

For allowing keeping lights on infinitely, this parameter can be set to 0s.

Manual Off To Auto Delay (clManOffAutoDelay)

When lights are manually switched off and the zone becomes unoccupied for the manual off to auto delay time, it will deactivate the manual light function. The timer starts when the room becomes unoccupied and stops if the room becomes occupied again within the expiry

time. This function ensures that intentionally switched off lights revert to their default automatic mode in reasonable time (comfort improvement).

For bedrooms and similar rooms, this parameter should be set to 0s to disable the feature.

Manual Mode (clManMode)

This data point selects the operation mode of the manual dimming function. It can be disabled (no manual control possible) or enabled.

8.4.3 Constant Light Function

The constant light controller (CLC) uses occupancy and illumination sensors to ensure a certain (minimum) light level. It is used for comfort functions (minimum light level, graded light bands) and for energy saving (lighting only when needed).

The CLC is a closed-loop controller so it continuously compares light level with the selected setpoint and adapts the lamp levels accordingly.

Table 62 shows the parameters of the constant light controller.

Parameter Name	Datapoint	Type	Default	Description
CLC Priority	clPriorityCLC	Real	5	Constant Light Controller Priority
CLC Mode	clCLCMode	Enum	NORMAL	Selects alternative operation modes for CLC controller
CLC Lux Setpoint	clCLCLuxSetpoint	Real	500 Lux	Setpoint for the contant light control algorithm
CLC Artifical Light at 100%	clCLCArtificialLight	Real	700 Lux	Intensity of the artificial light
CLC Relative On Hysteresis	clCLCHysteresisOn	Real	5%	Percentual hysteresis of lux setpoint to turn on lights
CLC Relative Off Hysteresis	clCLCHysteresisOff	Real	5%	Percentual hysteresis of lux setpoint to turn off lights
CLC Deadband	clCLCDeadband	Real	10%	Percentual deadband from setpoint
CLC Lux On Delay	clCLCDelayOn	Real	0 s	Delay to enable controller when brightness falls below hysteresis
CLC Lux Off Delay	clCLCDelayOff	Real	15s	Delay to disable controller when brightness rises above hysteresis
CLC Occupancy Off Delay	clCLCDelaySensorOff	Real	300 s	Time between sensor occupancy off and warning Level
CLC Warmup Time	clCLCDelayWarmup	Real	30 s	Time between constant warmup level and CLC control start
CLC Window Band Offset	clCLCWindowOffset	Real	30%	Lamp value at which window band starts dimming.
CLC Window Band Limit	clCLCWindowLimit	Real	70%	Lamp value at which window band is same as door band
CLC Warn Time	clCLCWarnTime	Real	15 s	Timespan with warning Lamp value before lamp is switched off
CLC Warn Percent	clCLCWarnPercent	Real	50%	Percent of the last LightLevel which is used to indicate warn level
CLC Leave Time	clCLCLeaveTime	Real	120s	Time to leave zone

CLC Control Mode	clCLCControlMode	Enum	CONTROL	Algorithm for light control
CLC Fixed Level	clCLCFixedLevel	Real	100%	Light level for FIXED algorithm

Table 62: Constant Light Controller Parameters

CLC Priority (clPriorityCLC)

The CLC priority defines the priority of its light function. Typically (and per default) it has lower priority than the manual function to allow users to override the automatic control.

CLC Mode (clCLCMode)

The CLC mode selects between different operation modes:

- OFF (0): The CLC is never activated
- NORMAL (1): The CLC is activated by occupancy and low illumination
- NO_OCC (2): The CLC is activated by low illumination only.
- MANUAL (3): The CLC does only start when receiving a AUTO_ENABLE command in the autoModeZone/autoModeRoom data point. It turns off when the zone becomes unoccupied.
- MANUAL_LUX (4): The CLC does only start when receiving a AUTO_ENABLE command in the autoModeZone/autoModeRoom data point. It turns off when the zone becomes unoccupied or the room is bright enough.

CLC Lux Setpoint (clCLCLuxSetpoint)

The lux setpoint parameter selects the desired lux level, e.g. 500 Lux for office rooms.

CLC Artificial Light at 100% (clCLCArtificialLight)

The artificial light level at 100% parameter tells the CLC which brightness to expect from 100% lamp setting. It is used to calculate the initial lamp value before the control algorithm starts.

CLC Relative On Hysteresis (clCLCHysteresisOn)
CLC Relative Off Hysteresis (clCLCHysteresisOff)
CLC Lux On Delay (clCLCDelayOn)
CLC Lux Off Delay (clCLCDelayOff)

The lux hysteresis and delay parameters define the activation/deactivation behavior of the CLC in respect to the light level.

If the lux level rises above the off hysteresis limit, the lux delay off timer starts. If the lux level remains above the off hysteresis limit for that time, the CLC will be deactivated.

If the lux level falls below the on hysteresis limit, the lux delay on timer starts. If the lux level remains below the on hysteresis limit for that time, the CLC will be activated.

CLC Deadband (clCLCDeadband)

The deadband is percentual range around the Lux setpoint in which the CLC will not make any light level changes. It is used to avoid small but sometimes visible changes to the lamp level when not necessary.

CLC Occupancy Off Delay (clCLCDelaySensorOff)

The occupancy off delay parameter defines how long the CLC is active after the zone has become unoccupied. It is used to avoid switching on/off lights too often. A longer time increases comfort (as light level is more often at comfort level). A shorter time increases energy savings but also can put stress on some actuators like fluorescent tubes.

CLC Warmup Time (clCLCDelayWarmup)

The actuators are kept at a fixed light level (calculated by the *clCLCArtificialLight* parameter) before the normal control algorithm is executed. This is to avoid interferences between the control algorithm and the natural warmup behavior of the luminaire.

CLC Window Band Offset (clCLCWindowOffset)
CLC Window Band Limit (clCLCWindowLimit)

These parameters define the behavior of the window light band. When the CLC is active the window band is gradually faded when the CLC output increases. Figure 29 shows the behavior of the window and door light bands.

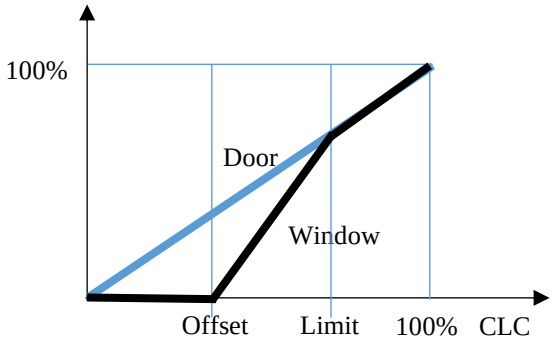


Figure 29: Window band parameters

CLC Warn Time (clCLCWarnTime)
CLC Warn Percent (clCLCWarnPercent)

These parameters define the behavior when the CLC is about to turn off. The CLC then dims the lamp to a lower level to indicate the user that it is about to turn off. The *clCLCWarnTime* parameter defines the time of the warning phase and the *clCLCWarnPercent* defines the percentage of the additional dimming. For example if the CLC has dimmed to 66%, and the *clCLCWarnPercent* parameter is 50%, then the warn dim level will be 33%.

CLC Leave Time (clCLCLeaveTime)

When triggering leave mode, the CLC is turned off and is blocked for this duration. Occupancy is ignored during leave time. At the end of the leave time, the occupancy is evaluated. If the zone is still occupied, the CLC is turned on again, else is remains disabled.

CLC Control Mode (clCLCControlMode)

This parameter allows to select between different control algorithms:

- CONTROL (1): Normal closed-loop algorithm which tries to reach the setpoint.
- FIXED (2): Outputs a constant light level as defined in *clCLCFixedLevel*.

CLC Fixed Level (clCLCFixedLevel)

This parameter defines the light level, when the controller is in FIXED mode.

8.4.4 Auto On/Off Function

The auto on/off light function turns on light based on occupancy, manual input and illumination level. It is typically used for stair cases or aisles where constant light control is not required.

Table 63 shows the parameters of the auto on/off lighting function.

Parameter Name	Datapoint	Type	Default	Description
Auto OnOff Priority	clPriorityAuto	Real	4	Auto ON/OFF function priority
Auto OnOff Mode	clAutoMode	Enum	OFF	Auto on/off modes
Auto OnOff Full Level	clAutoFullLevel	Real	100 %	Light level during clAutoFullTime
Auto OnOff Full Time	clAutoFullTime	Real	120 s	Time for which lights are set to clAutoFullLevel
Auto OnOff Dimm Level	clAutoDimmLevel	Real	50%	Light level during clAutoDimmTime
Auto OnOff Dimm Time	clAutoDimmTime	Real	30 s	Time for which lights are set to clAutoDimmLevel
Auto OnOff Lux Off Hysteresis	clAutoLuxOff	Real	1200 Lux	Upper hysteresis level to deactivate Auto ON/OFF logic
Auto OnOff Lux On Hysteresis	clAutoLuxOn	Real	300 Lux	Lower hysteresis level to activate Auto ON/OFF logic

Table 63: Auto On/Off Lighting Parameters

Auto OnOff Priority (clPriorityAuto)

This parameter defines the priority of the auto on/off lighting function.

Auto OnOff Mode (clAutoMode)

The auto on/off functions supports the following modes:

- OFF (0): The auto on/off mode is disabled.
- OCC (1): The light is turned on when the zone becomes occupied and is automatically turned off after the configured delays when the zone becomes unoccupied again.
- OCC+LUX (2): This mode works like OCC, but the light is only turned on, if the light level is below the configured lux off level.
- LUX (3): This mode turns on light, if the light level falls below to lower hysteresis (lux on) limit and turns off light if the light level raises above the higher hysteresis (lux off) limit.

- MANON_OCCOFF (4): In this mode, light needs to be turned on manually and will be turned off when the zone is unoccupied for the configured delay time. Do not use this mode in parallel with the manual dimming light function, as their timeouts can interfere and result in unexpected light behavior.

- MANON_TIMEDOFF (5): In this mode, light needs to be turned on manually and will be turned off after the configured delay time. Do not use this mode in parallel with the manual dimming light function, as their timeouts can interfere and result in unexpected light behavior.

Auto OnOff Full Level (clAutoFullLevel)
Auto OnOff Full Time (clAutoFullTime)

When light is turned on it will be set to clAutoFullLevel percent for clAutoFullTime seconds.

Auto OnOff Dimm Level (clAutoDimmLevel)
Auto OnOff Dimm Time (clAutoDimmTime)

When light is about to be turned off it is set to clAutoDimmLevel for additional clAutoDimmTime seconds.

Auto OnOff Lux Off Hsysteresis (clAutoLuxOff)
Auto OnOff Lux On Hysteresis (clAutoLuxOn)

These parameters provide the hysteresis for the illumination-aware auto on/off modes.

Light is turned on if the light level falls below to lower hysteresis (lux on) limit. It is turned off if the light level raises above the higher hysteresis (lux off) limit.

8.4.5 Scene Function

The scene lighting function allows saving and recalling scenes over the HMI interface. The scene function supports up to 8 light values.

Table 64 shows the parameters of the scene function.

Parameter Name	Datapoint	Type	Default	Description
Scene Priority	clPriorityScene	Real	7	Priority of Scene Controller

Table 64:Lighting Scene Parameters

Scene Priority (clPriorityScene)

This parameter defines the priority of the scene controller.

Scenes are saved persistently in a string data point using the following format (where <real> is a string-encoded real number:

```
<real>;<real>;<real>;<real>;<real>;<real>;<real>;<real>;
```

8.5 Actuators

8.5.1 Favorite Actuator (clLampFav)

The favorite actuator is used to control lights on arbitrary technologies. The favorite data points are to be connected to the data points of the selected lighting technology.

Favorite	Type	Description
lampDoorState	Binary	Lamp State IO for door light band
lampDoorValue	Real	Lamp Value IO for door light band
lampWindowState	Binary	Lamp State IO for window light band
lampWindowValue	Real	Lamp Value IO for window light band

Table 65: Lighting actuator favorite data points

The door favorite data points are used for the light band near the door or the aisle of an office room, while the window favorite data points are used for the light band facing the window side. If only one light band is required, use the door band.

The value part is used to dim the lights, while the state part switches the light on and off. IF state is off, then also value is set to 0 so that usually only the value favorite data point is required.

8.5.2 DALI Actuator (clDaliLamp2)

The actuator creates 2 DALI ballasts which can be commissioned using the device web interface. The door ballast should be used for a light band near the door or aisle while the window ballast should be used for the ballast facing the window. If only one light band is required, use the door ballast.

8.5.3 OPC Actuator (clLampDali)

This actuator is used to transfer LROC light output to an LOYTEC L-DALI device, but it can also be used for other OPC XML/DA devices.

It creates an OPC client which can be dynamically commissioned to a L-DALI device with a compatible OPC server interface.

8.5.4 Routed DALI Actuator (coRoutedDaliLamp2)

This actuator works like the normal DALI actuator described in section 8.5.2 but has additional parameters for using it as a routed actuator.

Parameter Name	Datapoint	Type	Default	Description
RIO Segment ID	clRioSegId	Int	0	Segment ID
RIO Controller Index	clRioCtrlIdx	Int	0	Controller index
RIO Feedback Enable	clRioFbEnable	Binary	active	Feedback enable

Table 66: sbActuatorRoutedFav Parameters

RIO Segment Id (clRioSegId)

This number defines the segment ID of the connected segment. If this value is 0, the segId input of the block is used, if it is non-zero, it overrides the value of the segId input. The segment ID must be used on the device, else the values are ignored.

RIO Controller Index (clRioCtrlIdx)

This value selects between multiple controllers of the segment. For example, if there are 3 Light controllers, their ctrlIdx input have to match this parameter. A value of 0 means that the actuator is disconnected.

RIO Feedback Enable (clRioFbEnable)

This parameter determines whether the actuator should send a feedback signal. There should be exactly one actuator configured to send a feedback signal.

8.6 Sensors

8.6.1 Light Switch

The light switch can be connected to a light module and provides favorites to link the light controller to physical switches or to data points from other technologies.

Table 67 shows the favorite data points of the light switch module.

Favorite	Type	Description
buttonUp	Binary	Binary IO for dimm up button
buttonDown	Binary	Binary IO for dimm down button
switchState	Binary	On/off state of dimmer control
switchValue	Real	Value of dimmer control (0..100%)

Table 67: Light Switch Favorites

The buttonUp and buttonDown data points are connected to a binary object, like a digital input or an KNX binary object. The meaning and timing of these two data points is determined by the parameters described in Table 68

The switchState and switchValue are an interface used for a dimmer-like input device having a switch and a dim value. These two data points provide a direct input and have no further processing. They directly control the manual dimming function.

Parameter Name	Datapoint clSwitch…	Type	Default	Description
Switch Function Up Short	clSwitchFunctionUpShort	Enum	ON_OFF (3)	Function when up button is pressed momentarily
Switch Function Up Long	clSwitchFunctionUpLong	Enum	UP_DOWN (6)	Function when up button is hold
Switch Function Down Short	clSwitchFunctionDownShort	Enum	ON_OFF (3)	Function when down button is pressed momentarily
Switch Function Down Long	clSwitchFunctionDownLong	Enum	UP_DOWN (6)	Function when down button is hold
Switch Function Both Short	clSwitchFunctionBothShort	Enum	ON_OFF (3)	Function when both buttons are pressed momentarily
Switch Function Both Long	clSwitchFunctionBothLong	Enum	SAVE (7)	Function when both buttons are hold
Switch Dimm Time	clSwitchDimmTime	Real	5 s	Seconds to dimm between 0 and 100%
Switch Short Press Time	clSwitchShortTime	Real	1 s	Maximum time for short press
Switch Location	clSwitchLocation	String	“”	Location of light switch

Table 68:Light Switch Parameters

Switch Function Up Short (clSwitchFunctionUpShort)
Switch Function Up Long (clSwitchFunctionUpLong)
Switch Function Down Short (clSwitchFunctionDownShort)
Switch Function Down Long (clSwitchFunctionDownLong)
Switch Function Both Short (clSwitchFunctionBothShort)
Switch Function Both Long (clSwitchFunctionBothLong)

These parameters define the actions triggered by a short or long press:

- NO_FUNCTION (0): No function.
- ON (1): Turn on light to last dim level. If last dim level was 0%, select 100% instead.
- OFF (2): Turn off light
- ON_OFF (3): Toggle between last dim level and off
- UP (4): Dim up. If light was turned off, turn it on.
- DOWN (5): Dim down.

- UP_DOWN (6): Dim up. If light level reaches 100%, start dimming down. If light level reaches 100% dim up again.
- SAVE (7): Save current light value (deprecated)
- AUTO (8): Activate automatic lighting
- ACT_UP (9): Actuator based dimming (ON and UP)
- ACT_DOWN (10): Actuator based dimming (DOWN and OFF)
- ACT_DOWN1 (11): Actuator based dimming (DOWN to 1%, remain ON)
- ACT_UP/DOWN (12): Actuator based dimming (UP, DOWN alternating)

There are two common setups:

- One button: Use ON_OFF for short presses and UP_DOWN for long presses. This resembles the classical switch.
- Two buttons: Uses two dedicated buttons, one for turning lights on and making them brighted and one for turning lights off and making them darker.
    - Up botton: Use ON for short press and UP for long press.
    - Down button: Use OFF for short press and DOWN for long press.

Switch Dimm Time (clSwitchDimmTime)

The switch dimm time defines the time it takes to dim from 0 to 100% (and vice verse).

Switch Short Press Time (clSwitchShortTime)

The switch short press time defines the maximum time for detecting a short press. Presses longer than this time will be reported as long press.

Switch Location (clSwitchLocation)

This parameter is an informational string to document the switch location. It has no impact on the controller.

8.6.2 Light Switch Core (clHmiButtonCore)

This block sends a light command to a target zone.

The *SegId* input needs to be provided with an existing segment Id. Also area, floor and building Ids are appropriate when the corresponding coupler is on the same device.

The *ZoneId* input needs to be set to the target zone ID. If can be an asterisk (*) to address all zones in the target.

The *DstAddr* input needs to be set to a valid address. Examples would be:

- {TZ} This zone
- {TR} This room
- {TB}.{F}F1.{A}A2.{R}* All rooms in this building, Floor F1, Area A2

The *SecondsLong* input defines the minimum time to trigger a long event.

The *SecondsDouble* input defines the timeout between to short events to trigger a double event.

The *ShortFunction*, *DoubleFunction* and *LongFunction* inputs define which functions are triggered for the corresponding event:

- NOP (1): No operation
- ON (2): Turn on (100%)
- OFF (3): Turn off
- ON_OFF (4): Toggle on/off
- DIMM_UP (5): Turn on and dinm up
- DIMM_DOWN0 (6): Dimm down and turn off
- DIMM_DOWN1 (7): Dimm down to minumum level
- AUTO (8): Return to CLC
- AUTO_ENABLE (9): Manually enable CLC
- AUTO_DISABLE (10): Manually disable CLC
- AUTO_LEAVE(11): Trigger CLC leave function

The *RelativeStep* is reserved for future use.

The *Button* input with its event is used to connect a binary button-like input.

9 Sunblind Functions

9.1 Sunblind Overview

The L-ROC sunblind controller is used to operate sunblinds, raffstores, screens and other shades to provide glare protection, thermal protection or view protection.

The following shading functions are provided by the standard sunblind controller:

- Manual operation (position, rotation)
- Automatic shading based on sun position and outdoor illumination
- Scene control
- Thermal heat/cool support
- Day/night position
- Window contact operation

The following global functions are available:

- Sunblind protection (wind, rain, ice, fire, security alarms)
- Multiple wind groups for complex fasades
- Global weather station
- Central operation (per area, per floor, per building)

The sunblinds support directly controlled sunblinds via digital outputs as well as SMI motors. Other technologies like KNX or LON can also be used.

9.2 OPC/BACNET Interface

Table 69 shows the data points available on the OPC and BACnet interface.

Datapoint	Type	Description
sbZoneId	String	Sunblind zone ID write
sbZoneIdFb	String	Sunblind zone ID feedback
mode	Real	Deprecated, Sunblind mode write
modeFb	Real	Deprecated, Sunblind mode feedback
positionRoom	Real	Set SB position in room
rotationRoom	Real	Set SB rotation in room
functionRoom	Multistate	Set SB function in room
sceneRoomRecall	Real	Recall room scene command
sceneRoomSave	Real	Save room scene command
positionZone	Real	Set SB position in zone
rotationZone	Real	Set SB rotation in zone
functionZone	Multistate	Set SB function in zone
sceneZoneRecall	Real	Recall zone scene command
sceneZoneSave	Real	Save zone scene command
alWindFb	Binary	Wind alarm
alRainFb	Binary	Rain alarm
alIceFb	Binary	Ice alarm
alFireFb	Binary	Fire alarm
alSecurityFb	Binary	Security alarm
functionFb	Multistate	Sunblind function feedback
positionFb	Real	Sunblind position feedback
rotationFb	Real	Sunblind rotation feedback
globalLuxLevelFb	Real	Outdoor lux level
defaultMode	Real	Deprecated, sunblind default mode
location	String	Sunblind location
activeModesFb	Real	Active sunblind modes
selectedModeFb	Real	Highest-priority sunblind mode
autoModeFb	Multistate	Current auto mode (1 for manual, 2 for auto)
autoModeZone	Multistate	Set manual (1) or automatic mode (2)
autoModeRoom	Multistate	Set manual (1) or automatic mode (2)
overrideFb	Binary	Indicates active override
windowsOpenFb	Analog	Number of open windows in zone
masterFb	Binary	Indicated master controller

Table 69: Sunblind OPC/BACnet Interface

Generally the sunblind commands are built from three data points:

1. The function data point selects the movement command:
    a. NULL (-1): No imput. Command will be ignored
    b. SET_OFF (0): Ignored for sunblinds.
    c. SET_ON (1): Ignored for sunblinds.
    d. SET_DOWN (2): Relative downwards movement. The position and rotation data points are interpreted relatively to the current position If

position is not within 0…100%, the position will not be changed. If rotation is not with 0…180°, the rotation will not be changed.

e. SET_UP (3): Relative downwards movement. The position and rotation data points are interpreted relatively to the current position If position is not within 0..100%, the position will not be changed. If rotation is not with 0..180°, the rotation will not be changed.

f. SET_STOP (4): Stop current movement

g. SET_STATE (5): Drive to the position and rotation given by the position and rotation data points. If the *position* value is -1.0, the command uses the current sunblind position and changes only the rotation part. If the *rotation* value is -1000.0, the command uses the current sunblind rotation and changes only the position part.

2. The position data point selects a relative or absolute position dependent on the value of the function data point. Position values are valid from 0 to 100%.

3. The rotation data point selects a relative or absolute rotation dependent on the value of the function data point. Rotation values are valid from -90° to 90° for absolute movements and from 0° to 180° for relative movements.

9.2.1 Zoning

Sunblind modules can be grouped into zones. All sunblind modules within a room having the same zone ID will operate synchronously. A master module will be automatically elected. All modules execute the same control algorithms, but only the master module is allowed to send sunblind drive commands to the slave modules. User inputs received by slave modules are directed to the master module.

The *sbZoneId* data point assigns the sunblind module to a zone. The default value is A. The current zone ID is reflected by the *sbZoneIdFb* data point.

Usually sunblinds with different orientation should not be put into the same zone, as the shading algorithm calculated by the master will only work the sunblinds facing the same direction.

The *location* data points is used to display the location of the sunblind module.

The *overrideFb* indicates that the zone has been overridden.

The *masterFb* indicates that this is the master controller for the zone.

9.2.2 Mode Selection and Feedback

The currently enabled sunblind controller functions can be modified by the HMI. The mode is a 16-bit bitmask which enables up to 16 light functions from which the highest-priority function is selected. The default value for the mode is 65535 (hex FFFF) so that all functions are enabled by default.

Note that this interface is usually not necessary, as the modes can setup with static parameters. It is intended to control additional (project specific) sunblind functions based on user input.

The *mode* data point sets a new mode for the sunblind zone. The *modeRoom* data points sets a new mode for the room in which the sunblind zone is in.

The current mode is reflected by the *modeFb* feedback data point.

The data point *defaultMode* reads the default Mode, so that one can revert to the default mode by within its value to the *mode* data point.

The *activeModesFb* data point shows the bit mask of all active modes, these are sunblind modes that currently would like to control the sunblinds.

The *selectedModeFb* returns the number of the sunblind function that has been selected from the active modes.

9.2.3 Manual Sunblind Control

The following data points are used to control sunblinds manually.

The *functionRoom*, *positionRoom* and *rotationRoom* data points are used to control all sunblind modules in the room. If the *functionRoom* data point is used, manual mode will be automatically entered.

The *functionZone*, *positionZone* and *rotationZone* data points are used to control all sunblind modules in the sunblind zone. If the *functionZone* data point is used, manual mode will be automatically entered.

The *autoModeZone* and *autoModeRoom* data points allow switching the zone or room to automatic mode or manual mode. The *autoModeFb* returns the current state of the sunblind module:

- MANUAL (1): Manual mode is enabled
- AUTO (2): Manual mode is disabled

9.2.4 Scene Control

The scene control data points allow saving and recalling up to 8 scenes.

The sceneZoneSave and sceneRoomSave data points save the current sunblind state to the given scene number (1-8).

The *sceneZoneRecall* and *sceneRoomRecall* data points restore a sunblind state within the zone or room. A scene number of "-1" disables the scene controller. An unused scene results in 0% position.

9.2.5 Alarm Feedback

The alarm feedbacks indicate that the sunblind has moved to the configured protection position and cannot be controlled manually or by an automatic mode.

- *alWindFb*: Wind alarm
- *alRainFb*: Rain alarm
- *alIceFb*: Ice alarm
- *alFireFb*: Fire alarm
- *alSecurityFb*: Security alarm

9.2.6 Sunblind Feedback

The *functionFb*, *positionFb* and *rotationFb* data points indicate the current position. The *functionFb* reads NULL (-1) when no command is executed and SET_STATE (5) when the sunblind is moving, as all relative commands are internally translated to absolute move commands.

9.2.7 Sensor Feedback

The *globalLuxLevelFb* displays the outdoor light level which is used to determine whether shading is required.

The *windowsOpenFb* indicates the number of open windows in the zone

9.3 Sunblind Functions

9.3.1 Function Summary

The L-ROC library provides the sunblind functions shown in Table 70.

Function Number	Function	Mode Value (Bitmask)	Default Priority	Default Enabled
1	Manual	1	7	yes
2	Automatic Shading	2	4	yes
3	Day/Night	4	3	no
4	Thermal	8	0	no
5	Scene Control	16	5	no
6	Window Contact	32	9	no

Table 70: Sunblind functions

A sunblind module can have more than one function enabled. In this case, the sunblind functions and the assigned priorities are used to determine the selected sunblind function.

The default setup enabled manual sunblind control and automatic shading. If the zone is occupied, the sun lights the facede and the outdoor illumination is high enough, the sunblinds are driven to the shading position. The sunblind rotation is automatically calculated from the earth position and façade parameters. If the user overrides the sunblind, it remains in manual mode as long as the zone is occupied plus a configurable timeout.

If no automatic shading and no manual control is requested, the sunblind drives to a default position.

As sunblinds are often mounted outside, they need to be protected from weather conditions like wind or frost. Each sunblind module receives weather data from the building manager and can drive the sunblinds into protection positions.

Table 71 shows the sunblind module communication and core parameters.

Parameter Name	Datapoint	Type	Default	Description
Zone ID	sbZoneId	String	A	Sunblind Zone Id
Default Sunblind Mode	sbZoneDefaultMode	Real	65535	Default sunblind mode (reset after sbZoneDefaultDelay delay)
Default Mode Delay	sbZoneDefaultDelay	Real	0 s	Delay from unoccupied state to default mode reset
Zone Location	sbZoneLocation	String	""	Location of sunblind zone
Enable Default Mode	sbZoneDefaultEnable	Binary	TRUE	Enables the default state/value if no controller is active
Default Position	sbZoneDefaultPosition	Real	0%	Default position if no controller is active
Default Rotation	sbZoneDefaultRotation	Real	0°	Default rotation if no controller is active
Zone Type	sbZoneType	String	""	Sunblind Zone Type
Façade Group	sbZoneFacadeGroup	String	""	Façade group
Façade Direction	sbZoneFacadeDir	Ral	0	0..: global value 1..16: façade directions

Table 71: Sunblind communication and base parameters.

Zone ID (sbZoneId)

The zone id parameter defines the zone to which this sunblind module belongs.

Default Sunblind Mode (sbZoneDefaultMode)
Default Mode Delay (sbZoneDefaultDelay)

The default sunblind mode and default mode delay parameters are used for special applications when the HMI is used to change the enabled modes.

For normal applications leave the default sunblind mode at 65535 (all sunblind functions enabled) and the default mode delay at 0 seconds.

If the mode delay parameter is non-zero, the default mode will be restored after the mode delay time after the zone got unoccupied. So a user could select a special sunblind function, but the sunblind module reverts to the default mode.

Zone Location (sbZoneLocation)

The zone location parameter is an informational parameter for documention the sunblind module location. It has no influence on control functions.

Enable Default Mode (sbZoneDefaultEnable)
Default Position (sbZoneDefaultPosition)
Default Rotation (sbZoneDefaultRotation)

The enable default mode parameter (default on) determines the sunblind module behavior when no sunblind function is active. If enabled, the default position and rotation will be used. If it is disabled, the last sunblind setting will remain until a sunblind function becomes active again.

Normally these parameters can be left at their default values.

Zone Type (sbZoneType)

The zone type parameter is an informational parameter for classifying zones with the same parameterization, e.g. office rooms or aisles. It has no influence on control functions. It can be used in the LWEB-900 parameter views to sort sunblind modules with similar parameterization for easier mass configuration.

Façade Group (sbZoneFacadeGroup)
Façade Direction (sbZoneFacadeDir)

The façade parameters define to which weather façade the module listens. The façade group can by empty for the default façade group. The direction can have the value of 0 to use global weather data (from the single-value weather blocks) or 1..16 to select the one of the 16 supported façade directions to receive weather data.

This affects the wind speed, solar irradiation value and outdoor lux level.

9.3.2 Sunblind Alarms

The sunblind module supports weather alarms. The alarm logic is hard-wired and overrides all sunblind functions. The sunblind alarms are using the following priorities (from highest to lowest):

1. Fire Alarm
2. Wind Alarm
3. Ice Alarm
4. Security Alarm
5. Rain Alarm

Table 72 shows the sunblind alarm parameters.

Parameter Name	Datapoint	Type	Default
Ice Alarm	sbAlarmIceMode	Enum	UP
Fire Alarm	sbAlarmFireMode	Enum	UP
Rain Alarm	sbAlarmRainMode	Enum	DISABLED
Security Alarm	sbAlarmSecurityMode	Enum	DISABLED
Wind Alarm	sbAlarmWindMode	Enum	UP
Wind Alarm Off Delay	sbAlarmWindDelayOff	Real	60 s
Wind Alarm On Delay	sbAlarmWindDelayOn	Real	0 s
Wind Alarm Off Limit	sbAlarmWindLimitOff	Real	10 m/s
Wind Alarm On Limit	sbAlarmWindLimitOn	Real	15 m/s
Alarm Special Position	sbAlarmSpecialPosition	Real	0%
Alarm Special Rotation	sbAlarmSpecialRotation	Real	0°
Ice Alarm On Delay	sbAlarmIceDelayOn	Real	0 s
Ice Alarm Off Delay	sbAlarmIceDelayOff	Real	0 s
Fire Alarm On Delay	sbAlarmFireDelayOn	Real	0 s
Fire Alarm Off Delay	sbAlarmFireDelayOff	Real	0 s
Rain Alarm On Delay	sbAlarmRainDelayOn	Real	0 s
Rain Alarm Off Delay	sbAlarmRainDelayOff	Real	0 s
Security Alarm On Delay	sbAlarmSecurityDelayOn	Real	0 s
Security Alarm Off Delay	sbAlarmSecurityDelayOff	Real	0 s
Alarm Watchdog Start	sbAlarmWatchdogStart	Enum	FIRST_RCV

Table 72: Sunblind alarm parameters.

Alarm Modes
Alarm Special Position
Alarm Special Rotation

The alarm modes define the reaction of the sunblind to an alarm condition:

- ALARM_DISABLED (1): No alarm function
- ALARM_UP (2): Sunblind moves up (0%, 0°)
- ALARM_DOWN (3): Sunblind moves down (100%, 0°)
- ALARM_SHUT (4): Sunblind moves down and shuts (100%, 90°)
- ALARM_SPECIAL (5): Sunblind moves to the position given by the sbAlarmSpecialPosition and sbAlarmSpecialRotation parameters.

Alarm Off Delays
Alarm On Delays

When an alarm condition occurs, the alarm is delayed for the alarm on delay. If the alarm is revoked during the time, the sunblind alarm is not activated. If the alarm is active for more than the on-delay, the sunblind enters the alarm state

When an alarm condition is revoked, the sunblind remains in the alarm state for the off-delay time. If the alarm reoccurs during the time, the sunblind remains in alarm state.

Wind Limit Off
Wind Limit On

These values define at which wind speeds the wind alarm is activated (on) or deactivated (off). It needs to be set according to the sunblind manufacturers specifications.

Alarm Watchdog Start

The alarm functions rely on regular updates from the weather station. Therefore communcication can be check with a watchdog function. This parameter determines how the watchdog is used:

- FIRST_RCV (1): The watchdog is started when the first weather data is received. This mode is suitable for the setup phase where the building manager might not be online all time.
- ALWAYS (2): The watchdog is always started. Setting the parameter to this value also starts the watchdog.
- NEVER (3): The watchdog is never started. Setting the parameter to this value also stops the watchdog.

9.3.3 Manual Function

The manual sunblind function allows setting position and/or rotation of a sunblind using a room operation panel or LWEB project. Table 73 shows the parameters of the manual sunblind function.

Parameter Name	Datapoint	Type	Default	Description
Manual Priority	sbPriorityMan	Real	7	Manual Sunblind Control Priority
Manual To Auto Delay	sbManAutoDelay	Real	0 s	Delay between not occupied and return to auto mode

Table 73: Manual sunblind parameters

Manual Priority (sbPriorityMan)

The manual priority selects the priority of the manual sunblind function. Usually it should be higher than for the automatic modes, as manual modes would have no effect else.

Manual To Auto Delay (sbManAutoDelay)

If the sunblind is in manual mode and the zone is unoccupied for for than the manual to auto delay time, it reverts to automatic modes. The timer starts when the zone becomes unoccupied. If the zone becomes occupied during the interval, it remains in manual mode.

A value of 0s disables the manual to auto transition.

9.3.4 Shading Function

The automatic shading function controls the sunblind to protect the inhabitants from glare. It uses the following values to calculate the correct sunblind position:

- Sun position (automatically calculated)
- Sunblind orientation
- Occupancy
- Outdoor illumination

Table 74 lists the parameters of the shading module

Parameter Name	Datapoint	Type	Default	Description
Shading Priority	sbPriorityShading	Real	4	Sunblind shading controller priority
Shading Use Global Glare	sbShadingGlareUseGlobal	Bool	TRUE	Defines to use the global glare signal to activate the automatic
Shading Glare Lux Limit	sbShadingGlareLux	Real	--	Min global lux value to start shading
Shading Glare Position	sbShadingGlarePos	Real	100%	Position of the sunblind during the automatic mode
Shading Glare On Delay	sbShadingGlareDelayOn	Real	20s	Delay before start shading in seconds.
Shading Glare Off Delay	sbShadingGlareDelayOff	Real	20s	Delay before stopping shading in seconds.
Shading Glare Standby Delay	sbShadingGlareDelayStandby	Real	120s	Time where sunblind remains in standby (closed but rotated to 0)
Shading Min Rotation Change	sbShadingMinRotChange	Real	5°	Rotations less than this value will be surpressed
Shading Occupancy Sensor Off Delay	sbShadingOccSensorDelayOff	Real	--	Delay after occupancy off until shading is deactivated
Shading Slat Spacing	sbShadingSlatSpacing	Real	50mm	Space between the slats in mm
Shading Slat Width	sbShadingSlatWidth	Real	50mm	Slat width in mm
Shading Azimuth Start	sbShadingSunAzimuthStart	Real	80°	Sun azimuth for enabling shading algorithm
Shading Azimuth Center	sbShadingSunAzimuthCenter	Real	0°	Orientation of Slats (0...North, 90...East, 180...South, 270...W
Shading Azimuth End	sbShadingSunAzimuthEnd	Real	70°	Sun azimuth for disabling shading algorithm
Shading Mode	sbShadingMode	Enum	NORMAL	Selects alternate sunblind modes
Shading Rotation Offset	sbShadingRotOffset	Real	5°	Additional rotation offset for tolerance compensation
Shading Elevation Start	sbShadingSunElevationStart	Real	0°	Sun elevation at which shading starts

Table 74: Shading parameters

Shading Priority (sbPriorityShading)

This parameter defines the shading function priority.

Shading Use Global Glare (sbShadingGlareUseGlobal)

If this parameter is true, the shading algorithm uses a global glare signal from the weather station instead of the outdoor brightness.

Shading Glare Lux Limit (sbShadingGlareLux)

If the sbShadingGlareUseGlobal parameter is false, this parameter is compared with the current outdoor brightness to determine if shading is necessary.

Shading Glare Position (sbShadingGlarePos)

This parameter defines the position of the sunblind when automatic shading is active. It can be set to values less than 100% to allow more sunlight during shading.

Shading Glare On Delay (sbShadingGlareDelayOn)
Shading Glare Off Delay (sbShadingGlareDelayOff)
Shading Glare Standby Delay (sbShadingGlareDelayStandby)

These parameters define the timing behavior when the glare condition changes. If the glare condition starts, the activation is delayed for *sbShadingGlareDelayOn* seconds. This is used to prevent that the sunblinds drive if the sun temporarily shines through clouds.

The glare condition is reset after *sbShadingGlareDelayOff* seconds.

If the glare condition ends, the sunblind remains in 0°-rotation for *sbShadingGlareDelayStandby* seconds. This allows to quickly return to the shading rotation when the glare condition starts again and prevents that the sunblinds have to drive to 0% position immediately.

Shading Min Rotation Change (sbShadingMinRotChange)

This parameter prevents that minimal rotation changes are sent to the actuators.

Shading Occupancy Sensor Off Delay (sbShadingOccSensorDelayOff)

This parameter controls how long the zone needs to be unoccupied to deactivate the shading function. If the zone becomes occupied again during the interval, it remains occupied. If the automatic shading function deactivates, the next lower priority function, e.g. thermal support will be activated.

Shading Slat Spacing (sbShadingSlatSpacing)
Shading Slat Width (sbShadingSlatWidth)

These two parameters define the geometry of the slats. The spacing parameter is the distance between two slats. Ths width parameter is the length of the slat. For the shading algorithm only the quotient between spacing and width is necessary, so these parameters can be left at default for most sunblind types.

Shading Azimuth Start (sbShadingSunAzimuthStart)
Shading Azimuth Center (sbShadingSunAzimuthCenter)

Shading Azimuth End (sbShadingSunAzimuthEnd)
Shading Elevation Start (sbShadingSunElevationStart)

These parameters define the orientation and shading limits of the sunblind.

The azimuth center parameter should be set to the direction the sunblind is facing (north=0°, east=90°, south=180°, west=270°)

The azimuth start and end parameters determine the sun position for which shading shall be active. For example if there sunblind faces south and another building blocks vision to east, the start parameter can be increased to avoid shading while the sun is blocked by the other building.

The elevation start parameter defines the minimum sun elevation for which shading should start. For example if another building covers the sky, it can be raised to avoid shading while the sun is covered by the other building.

Shading Mode (sbShadingMode)

The shading mode can have on of the following values:

- NORMAL (1): Shading uses the occupancy sensor
- NO_OCC (2): There is no occupancy sensor, the zone is always occupied
- HVAC_OCC (3): The zone uses the occupancy of an overlapping HVAC zone. This allows to use the HVAC scheduler for activating shading
- DISABLED (4): The shading function is disabled.

Shading Rotation Offset (sbShadingRotOffset)

This parameter adds a constant rotation to the calculated rotation during shading. It is used to compensate tolerances in algorithm, actuator precision and sunblind geometry.

9.3.5 Day/Night Function

The day/night function drives the sunblind to defined positions during day and/or night. This is used to create a uniform façade during day or night.

Table 75 shows the parameters of the Day/Night function.

Parameter Name	Datapoint	Type	Default	Description
DayNight Mode	sbDayNightMode	Enum	OFF	Switch to enable day, night or day+night mode
DayNight Priority	sbPriorityDayNight	Real	3	Day/Night control priority
DayNight Day Delay	sbDayNightDayDelay	Real	--	Minutes between sunrise and driving to day position
DayNight Day Position	sbDayNightDayPosition	Real	--	Sunblind position in day mode
DayNight Day Rotation	sbDayNightDayRotation	Real	--	Sunblind rotation in day mode
DayNight Night Delay	sbDayNightNightDelay	Real	--	Minutes between sunset and driving to night position
DayNight Night Position	sbDayNightNightPosition	Real	--	Sunblind position in night mode
DayNight Night Rotation	sbDayNightNightRotation	Real	--	Sunblind rotation in night mode

Table 75: Day/night shading parameters

DayNight Mode (sbDayNightMode)

This parameter selects the operating modes of the day/night module:

- OFF (1): Module is deactivated
- DAY (2): The module is activated during day.
- NIGHT (3): The module is activated during night.
- DAYNIGHT (4): The module is always active.

DayNight Priority (Day/Night control priority)

This parameter defines the priority of the day/night module. Usually it is lower than automatic shading or manual control.

DayNight Day Delay (sbDayNightDayDelay)
DayNight Night Delay (sbDayNightNightDelay)

These parameters define the offsets to sunrise and sunset in minutes. For example to activate night mode 30 minutes after sunset and day mode 30 minutes before sunrise, set sbDayNightDayDelay to -30 and sbDayNightNightDelay to +30.

DayNight Day Position (sbDayNightDayPosition)
DayNight Day Rotation (sbDayNightDayRotation)
DayNight Night Position (sbDayNightNightPosition)
DayNight Night Rotation (sbDayNightNightPosition)

These parameters define position and rotation during day and night mode.

9.3.6 Thermal Function

The thermal function uses the sunblind to support HVAC when zones are unoccupied.

The thermal function uses the following values for operation

- Sunshine (via Lux value or Irradiation sensor)
- Outdoor Temperature
- Inddor Temperature
- Occupancy
- Occupied setpoints from HVAC module

Table 76 shows the parameters of the thermal function.

Parameter Name	Datapoint	Type	Default	Description
Thermal HVAC Mode Selection	sbThermalHvacMode	Enum	OFF	Source of heat/cool demand
Thermal Sensor Selection	sbThermalSensorMode	Enum	SUN_ OUTTEMP	Which sensors to use for determining heat/cool availability
Thermal Occupancy Mode	sbThermalOccMode	Enum	NORMAL	Selects occupancy source for thermal controller
Thermal Priority	sbPriorityThermal	Real	0	Sunblind thermal controller priority
Thermal Open Position	sbThermalOpenPosition	Real	0	Sunblind open position
Thermal Open Rotation	sbThermalOpenRotation	Real	0	Sunblind open rotation
Thermal Close Position	sbThermalClosePosition	Real	0	Sunblind close position
Thermal Close Rotation	sbThermalCloseRotation	Real	0	sbThermalCloseRotation
Thermal Temp Hysteresis	sbThermalTempHyst	Real	0.3°C	Temperature hysteresis
Thermal Sunshine On Delay	sbThermalSunOnDelay	Real	900s	Time between sunshine and terminal load mode
Thermal Sunshine Off Delay	sbThermalSunOffDelay	Real	900s	Time between sunshine off and thermal mode disable
Thermal Occupancy Off Delay	sbThermalOccOffDelay	Real	900s	Delay after occupancy mode off and terminal load mode
Thermal Sunshine Lux Limit	sbThermalSunshineLux	Real	2000	Minimum Lux to enable terminal load mode
Thermal Sun Hysteresis	sbThermalSunHyst	Real	20%	Hysteresis (%) for Lux or Solar Irradiation
Thermal Sun Solar Irradiation Limit	sbThermalSunshineSolar	Real	200 W/m²	Minium solar irradiation for thermal mode

Table 76: Sunblind thermal parameters

Thermal HVAC Mode Selection (sbThermalHvacMode)

This parameter selects the thermal function:

- OFF (1): Thermal function is disabled.
- MAN_HEAT (2): The thermal function always tries to heat.

- MAN_COOL (3): The thermal function always tries to cool.
- AUTO_TEMP_DIFF (4): The thermal function uses space temperature and occupied heat/cool setpoints to determine heat/cool demand. This mode is recommended as it also uses the thermal mode to heat/cool the room in unoccupied state.
- AUTO_HVAC_MODE (5): The thermal function uses the effective HVAC mode to determine whether to heat or cool.
- AUTO_TERM_LOAD (6): The thermal function uses the output of ther HVAC controller to determine whether to heat or cool. This means if the HVAC controller does neither heat nor cool, thermal support is disabled.[4]

Thermal Sensor Selection (sbThermalSensorMode)

This parameter defines which sensors are used:

- SUN_OUTTEMP (1): Solar irradiation/Lux level and outdoor temperature are used
- SUN_ONLY (2): Only solar irradiation/lux level is used.
- OUTTEMP_ONLY (3): Only outdoor temperature is used.

Solar irradiation has higher priority than outdoor temperature. This makes a difference if the sunblinds can be used to isolate the building (dependent on sunblind type).

Table 77 shows the selected sunblind position dependent on parameters and outdoork conditions.

Mode	Indoor Temp Ti vs Outdoor Temp To	Sun	SUN+ OUTTEMP	SUN+ ONLY	OUTTEMP ONLY
Heat	Ti > To	No	Close	Close	Close
Heat	Ti > To	Yes	Open	Open	Close
Heat	Ti < To	No	Open	Close	Open
Heat	Ti < To	Yes	Open	Open	Open
Cool	Ti > To	No	Open	Close[5]	Open
Cool	Ti > To	Yes	Close	Close	Open
Cool	Ti < To	No	Close	Close	Close
Cool	Ti < To	Yes	Close	Close	Close

[4] This mode is recommended because the HVAC controller tries to conserve expensive energy while using solar power is almost free. If the HVAC controller does not heat a room because it is unoccupied, it still is preferable to use solar irradiation to heat above the unoccupied setpoint.

[5] In cooling mode when using SUN_ONLY mode it is assumed that the outdoor temperature is above the indoor temperature. Therefore the sunblinds are closed independent on the sun condition.

Table 77: Thermal conditions

Thermal Occupancy Mode (sbThermalOccMode)

This parameter defines how occupancy is determined:

- NORMAL (1): The occupancy sensor is used
- NO_OCC (2): The occupancy is not used and occupancy is assumed.
- HVAC_OCC (3): The occupancy output of the HVAC controller is used. This allows using the HVAC scheduler for scheduling the thermal function.

Thermal Priority (sbPriorityThermal)

This parameter defines the priority of the thermal function. It is usually set below all other functions.

Thermal Open Position (sbThermalOpenPosition)
Thermal Open Rotation (sbThermalOpenRotation)
Thermal Close Position (sbThermalClosePosition)
Thermal Close Rotation (sbThermalCloseRotation)

These parameters define the positions and rotations when the thermal module opens or closes the sunblind. The close rotation can be set to less than 90° to let in some sun light even during thermal protection.

Thermal Temp Hysteresis (sbThermalTempHyst)

This parameter defines the hysteresis for temperature comparison to avoid unnecessary sunblind moves.

Thermal Sunshine On Delay (sbThermalSunOnDelay)
Thermal Sunshine Off Delay (sbThermalSunOffDelay)

These parameters define how low irraditation needs to be present or absent to activate or deactivate the thermal function. If irradiation is high enough for more than sbThermalSunOnDelay seconds, the thermal function is activated. If irradiation is low enough for more than sbThermalSunOffDelay seconds, the thermal function is deactivated.

Thermal Occupancy Off Delay (sbThermalOccOffDelay)

This delay defines how long a zone must be unoccupied to activate the thermal function.

Thermal Sunshine Lux Limit (sbThermalSunshineLux)
Thermal Sun Solar Irradiation Limit (sbThermalSunshineSolar)
Thermal Sun Hysteresis (sbThermalSunHyst)

These parameters define the mimimum lux level or solar irradiation level for activating the thermal function. If both sensors are available, each of them activates the function. The hysteresis parameter allows make activation/deactivation more unlikely to avoid uncessary sunblind moves.

9.3.7 Scene Control Function

The sunblind scene function allows saving and recalling scenes over the HMI interface. The scene function supports up to 8 sunblind positions.

Table 78 shows the parameters of the scene function.

Parameter Name	Datapoint	Type	Default	Description
Scene Priority	sbPriorityScene	Real	5	Sunblind scene controller priority

Table 78:Sunblind Scene Parameters

Scene Priority (sbPriorityScene)

This parameter defines the priority of the scene controller.

Scenes are saved persistently in a string data point using the following format (where <real> is a string encoded real number:

```
<real>|<real>;<real>|<real>;<real>|<real>;<real>|<real>;<real>|<real>;<real>
|<real>;<real>|<real>;<real>|<real>
```

The first real is the sunblind position, while the second real is the sunblind rotation.

9.3.8 Window Contact Function

The window contact function allows settings the sunblind to a certain position when the window contact of the zone is opened.

For example, it can be used to open the sunblinds of a door when it is opened.

Table 79 shows the parameters of the window contact function.

Parameter Name	Datapoint	Type	Default	Description
Window Contact Priority	sbPriorityWinCont	Real	9	Window contact priority
Window Contact Mode	sbWinContMode	Enum	OFF (1)	Window contact mode
Window Contact Position	sbWinContPosition	Real	0 %	Sunblind position on window contact state
Window Contact Rotation	sbWinContRotation	Real	0 °	Sunblind rotation on window contact state

Table 79:Sunblind Window Contact Parameters

Window Contact Priority (sbPriorityWinCont)

This parameters defines the priority of the window contact function.

Window Contact Mode (sbWinContMode)

This parameter defines when the window contact function is active:

- OFF(1): Window contact function is disabled
- POSOPEN (2): Window contact function is active when the contact is open.
- POSCLOSE (2): Window contact function is active when the contact is closed.

Window Contact Position (sbWinContPosition)
Window Contact Rotation (sbWinContRotation)

There parameters define the sunblind position when the window contact function is active.

9.3.9 Year Shade Progression

The year shade progression function is parameterized in the sunblind controller and the actuators that belong to the sunblind controller.

Parameter Name	Datapoint	Type	Default	Description
YSP Mode	sbYspMode	Enum	OFF (1)	Yearshade mode
YSP Shade Factor	sbYspShadeFactor	Analog	0.01	Factor for Lux value in shading calculation

Table 80:Year Shade Progression Controller Parameters

Table 80 show the parameters of the year shade progression function.

YSP Mode (sbYspMode)

The YSP Mode determines how shading is calculated:

- OFF(1): The YSP function is deactivated
- ACTUATOR (2): Each actuator calculates shading on its own and opens the sunblinds if it is in shading mode.
- ZONE (3): If all actuators of the zones are shaded, the zone lux value will be modified so that the normal shading function assumes that shading is not necessary.
- The *sbYspShadeFactor* data point defines the factor for multiplying the lux value in this case. In this mode, every actuator calculates its shading on its own, but the decision is made per zone.

YSP Shade Factor (sbYspShadeFactor)

This factor is used to modify the lux value in case that all actuators are shaded in ZONE mode. For example, if all actuators are shaded and the outdoor brightness is 50000 Lux, the brightness for this zone will be lowered to 500 Lux, so that the shading module can determine that shading is not necessary.

The other part of the YSP configuration is done in the actuators, as shown in Table 81.

Parameter Name	Datapoint	Type	Default	Description
Window Name	sbActuatorWindowName	String	“”	3D model window name

Table 81:Year Shade Progression Actuator Parameters

Window Name (sbActuatorWindowName)

The window name is found in all library shading actuators. If set, it determines which 3D face in the 3D model will be used for shading calculation. It needs to be set to an existing name within the DXF file.

9.4 Actuators

9.4.1 Favorite Actuator (sbActuatorFav)

The favorite actuator is used to control sunblinds on arbitrary technologies. The favorite data points are to be connected to the data points of the selected lighting technology

Favorite	Type	Description
sbUp	Real	Sunblind up command in milliseconds
sbDown	Real	Sunblind down command in milliseconds

Table 82: Sunblind favorite actuator data points

Table 82 lists the favorite data points of the sunblind actuator. If the sunblind shall drive up or down the corresponding data point is set to the required drive time in milliseconds. Value 0 means to stop movement while a negative number means to switch on the motor direction permanently. Normally these favorites are connected to a LIOB DO which needs to be put into “Duration” mode.

Parameter Name	Datapoint	Type	Default	Description
Actuator Init Mode	sbActuatorInit	Enum	OPEN (2)	Selects actuator behavior at start
Actuator Open Time	sbActuatorTimeOpen	Real	60 s	Sunblind open time from 100 to 0%
Actuator Close Time	sbActuatorTimeClose	Real	60 s	Sunblind close time from 0% to 100%
Actuator Rotation Time	sbActuatorTimeRotation	Real	1 s	Sunblind rotation time from minimum to maximum rotation
Actuator Min Drive Time	sbActuatorTimeMinDrive	Real	0.05 s	Minimum drive time
Actuator Rotation Max	sbActuatorRotationMax	Real	90 °	Maximum rotation (-90 full inwards, 0 center, 90 full outwards)
Actuator Rotation Min	sbActuatorRotationMin	Real	0 °	Minimum rotation (-90 full inwards, 0 center, 90 full outwards)
Actuator Indication Factor	sbActuatorIndFactor	Real	5	Position feedback every sbActuatorIndFactor -th time.
Actuator Overdrive	sbActuatorOverdrive	Real	10 %	Overdrive in % when opening/closing to 0% or 100% position
Actuator Location	sbActuatorLocation	String	“”	Location of sunblind
Actuator Open Time Offset	sbActuatorOffsetTimeOpen	Real	0 ms	Constant time offset added to open movements
Actuator Close Time Offset	sbActuatorOffsetTimeClose	Real	0 ms	Constant time offset added to close movements
Actuator Alarm Delay	sbActuatorDelayAlarm	Real	0 s	Actuator delay for alarm commands
Actuator Auto Delay	sbActuatorDelayAuto	Real	0 s	Actuator delay for automatic mode commands

Table 83: Sunblind favorite actuator parameters

Table 83 shows the parameters of the sunblind actuator.

Actuator Init Mode (sbActuatorInit)

This mode determines the startup behavior of the sunblind actuator:

- PERSISTENT (1): The actuator remembers the last position and assumes that the sunblind is at the saved position. As this can be wrong (power loss, circuit breaker), it is only recommended for demonstration systems.

- INIT (2): The actuator references to the up position on startup. Thus the controller is sure that the actuator is at a defines position.

Actuator Open Time (sbActuatorTimeOpen)
Actuator Close Time (sbActuatorTimeClose)

These parameters define the time to move from 0% to 100%. They can be different for both positions and need to be determined once per sunblind type.

Actuator Rotation Time (sbActuatorTimeRotation)

This parameter defines the time to rotate from the minium rotation to the maximum rotation (or the other direction). This value needs to be determined once per sunblind type.

Actuator Min Drive Time (sbActuatorTimeMinDrive)

The min drive time suppresses movement shorted than this value in seconds. It prevents that the sunblind tries to move but fails to do so by inertia or friction.

Actuator Rotation Max (sbActuatorRotationMax)
Actuator Rotation Min (sbActuatorRotationMin)

These parameters define the rotation limits of the sunblind. Usually sunblinds move from 0°…90° or -90° to 90°. A positive value means turning the slats outwards down. It can be set to lower values to prevent positions where the slats could freeze together (e.g. 0°…80°)

Actuator Indication Factor (sbActuatorIndFactor)

The sunblind actuator permanently calculates the position of the sunblind during movements. This factor defines how often the calculated position is updated in the feedback data points. Only the nth calculated position is reported, so high values result in a low update rate.

Actuator Overdrive (sbActuatorOverdrive)

This percentage parameter defines the amount of overdrive when the sunblind drives to 0% or 100% position. This allows to make sure that the sunblind is at its calculated position as relative movements, e.g. automatic shading, can result in slight errors.

Actuator Location (sbActuatorLocation)

This is a string data point for documentation purposes. It is not used by the application.

Actuator Open Time Offset (sbActuatorOffsetTimeOpen)
Actuator Close Time Offset (sbActuatorOffsetTimeClose)

These values are added to each calculated move command to compensate inertia of motor or mechanics. If the sunblinds constantly stop before the calculated position/rotation, try adding some extra time here for compensation.

Actuator Alarm Delay (sbActuatorDelayAlarm)
Actuator Auto Delay (sbActuatorDelayAuto)

These values delay a sunblind move command to prevent that all motors are switched on/off at the same time.

The alarm delay is used when the sunblind is driven by an alarm condition. The auto delay is used when the sunblind is driven by an automatic sunblind function. Manual movements are never delayed as they typically switch on only a small number of motors.

For example, one could add 0.5 seconds of cumulative delay for each floor to implement a stagger delay when the sunblinds drive up due to a wind alarm.

9.4.2 SMI Actuator

The SMI actuator creates an SMI device which can be commissioned on the LROC Web UI. Please read the LOYTEC Device User Manual, chapter 10.3 on how to commission and calibrate SMI devices.

Parameter Name	Datapoint	Type	Default	Description
Actuator Alarm Delay	sbActuatorDelayAlarm	Real	0s	Actuator delay for alarm commands
Actuator Auto Delay	sbActuatorDelayAuto	Real	0s	Actuator delay for automatic mode commands

Table 84: SMI Actuator Parameters

Table 84 shows the parameters of the SMI actuator:

Actuator Alarm Delay (sbActuatorDelayAlarm)
Actuator Auto Delay (sbActuatorDelayAuto)

These values delay a sunblind move command to prevent that all motors are switched on/off at the same time.

The alarm delay is used when the sunblind is driven by an alarm condition. The auto delay is used when the sunblind is driven by an automatic sunblind function. Manual movements are never delayed as they typically switch on only a small number of motors.

For example, one could add 0.5 seconds of cumulative delay for each floor to implement a stagger delay when the sunblinds drive up due to a wind alarm.

9.4.3 Routed Favorite Actuator

The sbActuatorRoutedFav actuator is similar to the actuator described in section 9.4.1, but has additional parameters for using it as a routed actuator.

Parameter Name	Datapoint	Type	Default	Description
RIO Segment ID	sbRioSegId	Int	0	Segment ID
RIO Controller Index	sbRioCtrlIdx	Int	0	Controller index
RIO Feedback Enable	sbRioFbEnable	Binary	active	Feedback enable

Table 85: sbActuatorRoutedFav Parameters

RIO Segment Id (sbRioSegId)

This number defines the segment ID of the connected segment. If this value is 0, the segId input of the block is used, if it is non-zero, it overrides the value of the segId input. The segment ID must be used on the device, else the values are ignored.

RIO Controller Index (sbRioCtrlIdx)

This value selects between multiple controllers of the segment. For example, if there are 3 sunblind controllers, their ctrlIdx input have to match this parameter. A value of 0 means that the actuator is disconnected.

RIO Feedback Enable (sbRioFbEnable)

This parameter determines whether the actuator should send a feedback signal. There should be exactly one actuator configured to send a feedback signal.

9.4.4 Routed SMI Actuator

The sbActuatorRoutedFav actuator is similar to the actuator described in section 9.4.2, but has additional parameters for using it as a routed actuator.

Parameter Name	Datapoint	Type	Default	Description
RIO Segment ID	sbRioSegId	Int	0	Segment ID
RIO Controller Index	sbRioCtrlIdx	Int	0	Controller index
RIO Feedback Enable	sbRioFbEnable	Binary	active	Feedback enable

Table 86: sbActuatorRoutedFav Parameters

RIO Segment Id (sbRioSegId)

This number defines the segment ID of the connected segment. If this value is 0, the segId input of the block is used, if it is non-zero, it overrides the value of the segId input. The segment ID must be used on the device, else the values are ignored.

RIO Controller Index (sbRioCtrlIdx)

This value selects between multiple controllers of the segment. For example, if there are 3 sunblind controllers, their ctrlIdx input have to match this parameter. A value of 0 means that the actuator is disconnected.

RIO Feedback Enable (sbRioFbEnable)

This parameter determines whether the actuator should send a feedback signal. There should be exactly one actuator configured to send a feedback signal.

9.5 Sensors

9.5.1 Sunblind Switch

The sunblind switch can be connected to a sunblind module and provides favorites to link the sunblind controller to physical switches or to data points from other technologies.

Table 67 shows the favorite data points of the light switch module.

Favorite	Type	Description
btnUp	Binary	Binary IO for up button
btnDown	Binary	Binary IO for down button

Table 87: Sunblind Switch Favorites

The btnUp and btnDown data points are connected to a binary object, like a digital input or a KNX binary object. The meaning and timing of these two data points is determined by the parameters described in Table 68

Parameter Name	Datapoint clSwitch…	Type	Default	Description
Switch Function Up Short	sbSwitchFunctionUpShort	Enum	ON_OFF (3)	Function when up button is pressed momentarily
Switch Function Up Long	sbSwitchFunctionUpLong	Enum	UP_DOWN (6)	Function when up button is hold
Switch Function Down Short	sbSwitchFunctionDownShort	Enum	ON_OFF (3)	Function when down button is pressed momentarily
Switch Function Down Long	sbSwitchFunctionDownLong	Enum	UP_DOWN (6)	Function when down button is hold
Switch Function Both Short	sbSwitchFunctionBothShort	Enum	ON_OFF (3)	Function when both buttons are pressed momentarily
Switch Function Both Long	sbSwitchFunctionBothLong	Enum	SAVE (7)	Function when both buttons are hold
Switch Rotation Delta	sbSwitchRotationDelta	Real	10°	Sunblind rotation step
Switch Short Press Time	sbSwitchShortTime	Real	1 s	Maximum time for short press
Switch Location	sbSwitchLocation	String	“”	Location of sunblind switch
Switch Close Position	sbSwitchClosePosition	Real	100%	Position for close command
Switch Close Rotation	sbSwitchCloseRotation	Real	0°	Roation for close command

Table 88:Light Switch Parameters

Switch Function Up Short (sbSwitchFunctionUpShort)
Switch Function Up Long (sbSwitchFunctionUpLong)
Switch Function Down Short (sbSwitchFunctionDownShort)
Switch Function Down Long (sbSwitchFunctionDownLong)
Switch Function Both Short (sbSwitchFunctionBothShort)
Switch Function Both Long (sbSwitchFunctionBothLong)

These parameters define the actions triggered by a short or long press:

- STEP_UP (1): Rotate up by *sbSwitchRotationDelta* degrees.
- UP (2): Move to open position.
- STEP_DOWN (3): Rotate down by *sbSwitchRotationDelta* degrees.

- DOWN (4): Move to close position as defined by *sbSwitchClosePosition* and *sbSwitchCloseRotation*.
- UP (4): Dim up. If light was turned off, turn it on.
- STOP (5): Stop movement.
- AUTO (6): Enable automatic shading.

Switch Short Press Time (sbSwitchShortTime)

The switch short press time defines the maximum time for detecting a short press. Presses longer than this time will be reported as long press.

Switch Location (sbSwitchLocation)

This parameter is an informational string to document the switch location. It has no impact on the controller.

Switch Close Position (sbSwitchClosePosition)
Switch Close Rotation (sbSwitchCloseRotation)

These parameters define the position when the sunblind is manually closed. Typical settings are 100%/90° to completely close the blinds, or 100%/0° to avoid darkening the room too much automatically.

9.5.2 Sunblind Switch Core (sbHmiButtonCore)

This block sends a sunblind command to a target zone.

The *SegId* input needs to be provided with an existing segment Id. Also area, floor and building Ids are appropriate when the corresponding coupler is on the same device.

The *ZoneId* input needs to be set to the target zone ID. If can be an asterisk (*) to address all zones in the target.

The *DstAddr* input needs to be set to a valid address. Examples would be:

- {TZ} This zone
- {TR} This room
- {TB}.{F}F1.{A}A2.{R}* All rooms in this building, Floor F1, Area A2

The *SecondsLong* input defines the minimum time to trigger a long event.

The *SecondsDouble* input defines the timeout between to short events to trigger a double event.

The *ShortFunction*, *DoubleFunction* and *LongFunction* inputs define which functions are triggered for the corresponding event:

- NOP (1): No operation
- STEP_UP (2): Rotate up one step
- UP(3): Drive to upper position
- STEP_DOWN (4): Rotate down one step
- DOWN (5): Drive to lower position
- STOP (6): Stop drive
- AUTO (7): Return to automatic shading

The *RelativeStep* input defines the rotation steps in degrees for the STEP_UP and STEP_DOWN commands.

The *Button* input with its event is used to connect a binary button-like input.

9.6 Year Shade Progression

9.6.1 Introduction

The year shade progression is an additional feature of the sunblind module. Based on a three-dimensional (3D) model it is possible to determine if a sunray would directly shine onto a window surface.

The base for the year shade progression is the DXF (Drawing Exchange Fileformat) standard. This file-format is intended to be used to exchange drawings between different CAD-software tools. In order to make use of the year shade progression it is necessary to create a DXF-file containing information about the own building and the surroundings. Latter are considered all objects (trees, buildings, ect.) which have a high possibility of casting shadows onto the own building.

The following sections will therefore explain the required tools and the workflow for creating three dimensional models. In continuation the parametrization of windows will be explained. This includes a description of the available datapoints and the location. Finaly, the validation-process of the created file is described.

9.6.2 Tools

The year shade progression requires a DXF-file containing the three dimensional model. It is therefore important to ensure a successful DXF-File export. Former Google SketchUp, now SketchUp Make 2017, has been successfully used in order to generate and export successfully DXF Files of a scene. SketchUp itself requires the installation of the following third party plugin: https://www.guitar-list.com/download-software/convert-sketchup-skp-files-dxf-or-stl (Last Access: 06.02.2020). This plugin enables the export of the entire model as a tetrahedral mesh into a DXF-File.

In order to install the third-party plugin it is necessary to lauch SketchUp. Open the Menu *Window* and navigate to *Extension-Manager*. This should open a new dialog offering a button labeld *install Extensions*. Once this button is pressed, a file-dialog should be brought up. Navigate to the location of the downloaded plugin and click *Open*. This should install automatically the export plugin into the SketchUp Make 2017 environment.

In order to verify and validate the generated DXF-File it is possible to make use of freeware-software such as *eDrawings*. It provides a Visualization in order to determine surfaces, which have not been exported successfully. This tool (Viewer) is available for different operating systems. There are no special configurations and it is recommended to follow the onscreen instructions during the installation process. Link: http://www.edrawingsviewer.com (Last Access: 06.02.2020)

9.6.3 Creating Scenes

The specifications for the year shade progression can be split into three main categories. As a general specification can be considered, that all labels in the entire scene must not contain any spaces. The first category of specifications refers to windows and focuses on the naming and placement inside the scene. The naming is specified by a regular expression and must meet the following structure:

*window_.**
e.g.: window_1floor_seg01

At the moment of placing a window inside the scene it is necessary to consider, that the window should be placed in front of the buildings wall as described in Figure 30.

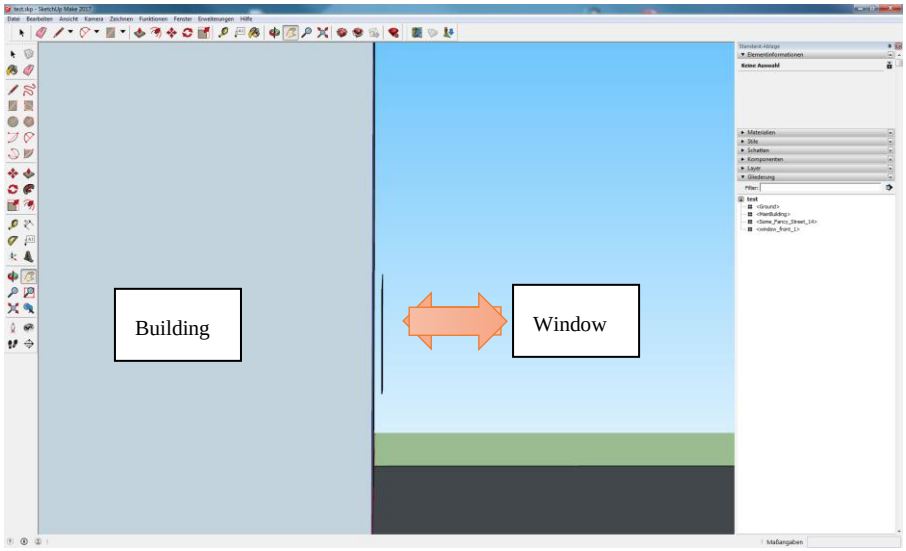


Figure 30: Placement of windows in a 3D model for the year shade progression

This placement ensures that not all four corners of the window collide with triangles, which belong to the building. Placing windows inside of the building would cause the building itself to cast a shadow onto the window. The distance between the buildings wall and the window depends on the overall scene. However, it is recommended to use a distance between five to ten centimiters.

The second category of specifications refers to the building itself. Surfaces that belong to the building in which the year shade progression modules operate, must contain *Building* in their label or at least match the regular expression: .**Building.**

e.g.: MainBuilding, SideBuilding, BuildingTower, MainBuildingTower, etc.

It is recommended, to continuously check if created surfaces of the building are correctly exported by the export-plugins. Missing triangles can lead to erroneous values, as the sun is capable of targeting the surfaces from undesired angles with the sunrays.

The third category of specifications refers to all other objects in the three-dimensional scene. Labels of these objects are freely chooseable and should not contain the keywords *Building* or *window*, as these are reserved for windows and the main building itself.

In continuation the creation of a simple three-dimensional model of a building and its surroundings is presented. The following Figure 31 gives an overview of the scene.

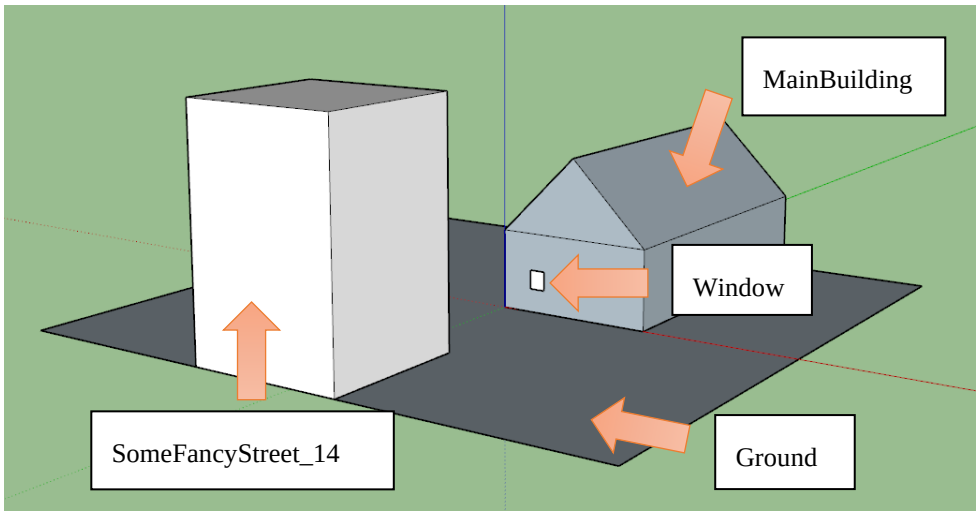


Figure 31: Example of a model for the year shade progression

The presented example is fully created with SketchUp Make 2017 and the already mentioned SKP to DXF/STL export plugin. After opening SketchUp Make 2017 an empty workspace is presented, containing a dummy-object labeld with <*Chris*>, as shown in Figure 32.

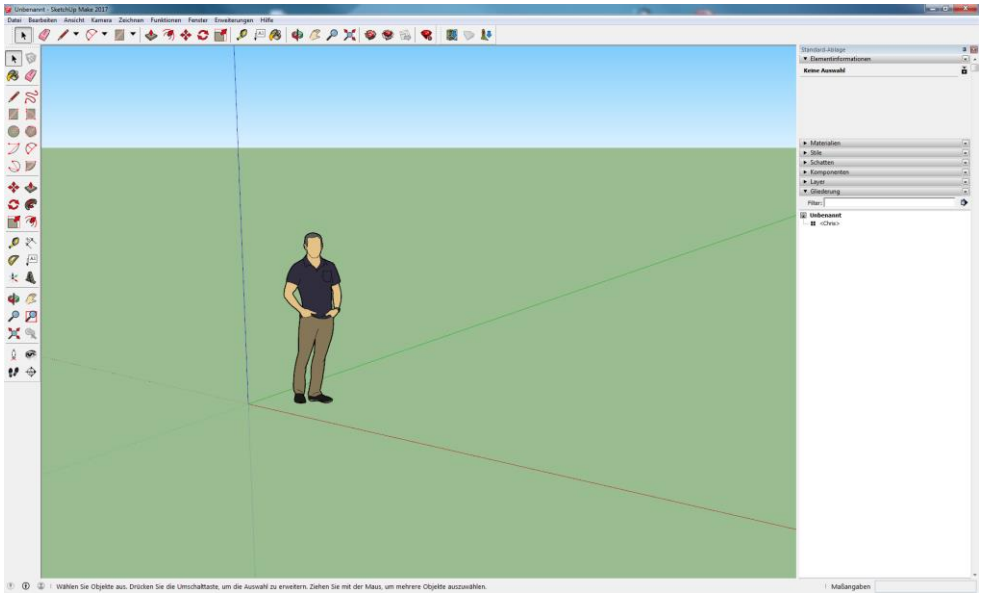


Figure 32: Initial SketchUp Make 2017 workspace

This dummy-object can be removed by marking <*Chris*> with the mouse and pressing delete. The now obtained workspace is ready for creating a dxf-model of the desired scene. However, three lines are still visible. The green (y-axis), blue (z-axis) and red (x-axis) lines describe the geometrical axis of the scene (these cannot be removed). These axes play an important role when orienting the entire scene towards north.

It is highly recommended to begin with the building itself, in which the year shade progression is operated in. As a first step, it is suggested to begin with the outline of the buildings-ground plane. The ground-plane of the entire scene is the surface described by the vectors of the x-axis and the y-axis, as described in Figure 33.

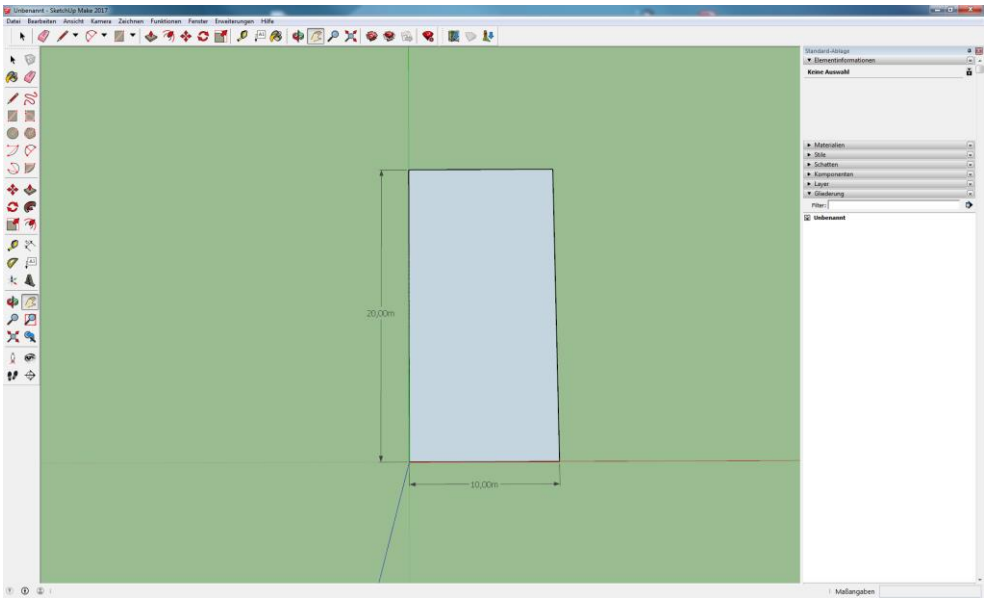


Figure 33: Ground plane for the building

The ground plane of the building is drawn with the help of the rectangle-tool, which can be found in the toolbar.

The next step, shown in Figure 34, involves the creation of four walls, which describe the outline of the building.

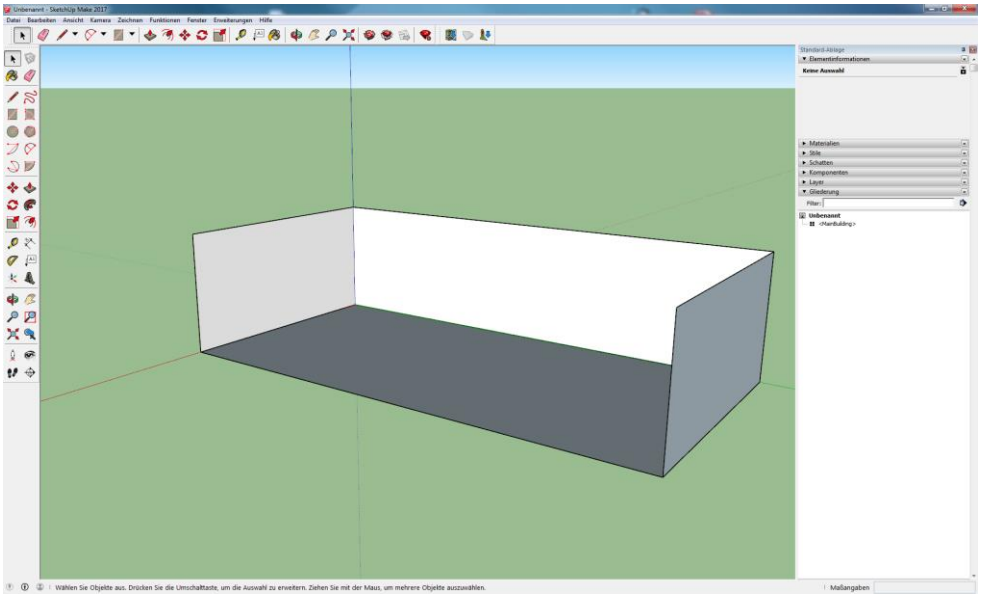


Figure 34: Creation of walls for the building

Once the last wall is closed, SketchUp automatically also closes the top surface. However, the top surface is not needed. By eliminating this surface (single-click and delete) the resulting file-size of the DXF-file is reduced.

In continuation, the roof of the building is designed. For ease of this tutorial, a simple standard roof has been drawn with the help of the pencil-tool. This step is visualized in Figure 35.

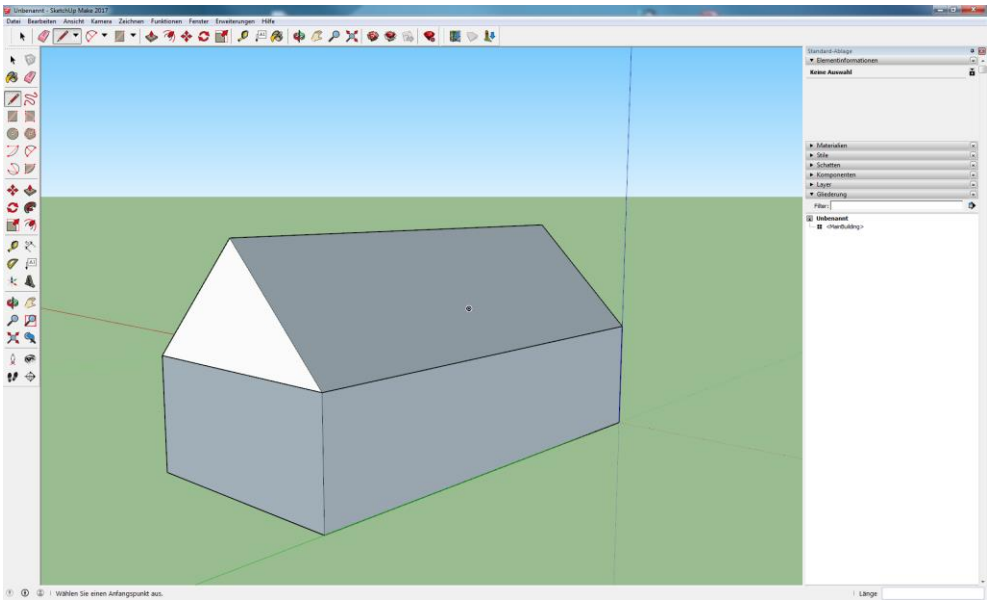


Figure 35: Roof-design for the year shade progression model

Once the entire building is drawn, it is important to mention, that most of the CAD-software tools provide the ability to define outwards and inwards surfaces of a drawn plane. SketchUp visualizes inwards faces of a surface in grey (Figure 36), while outwards faces are colord in white (Figure 37). In order to change the orientation of a face, it is necessary to mark the surface with a single-click. An additional right-click brings up a context menu and *Reverse Face* can be selected (Figure 38). This automatically swaps inwards and outwards faces of the selected plane.

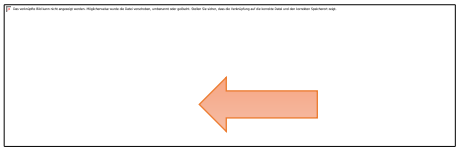


Figure 36: Year shade progression model with reveresed face

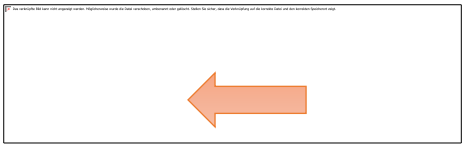


Figure 37: Year shade progression model with correctly alligned face

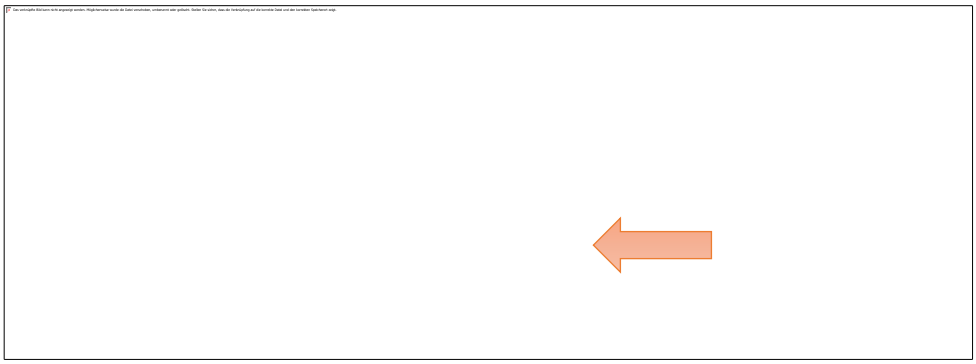


Figure 38: Context menu to reverse faces

Note:	*Faces with a wrong orientation cause no issues for calculations conducted by year shade progression algorithms. Reversed faces manifest in the visualization, incorporated in the WebUI. The result is a camera facing the inside of the building, instead of being oriented towards the surroundings.*

The next step already involves assigning the drawn building the unique identifier. This is achieved by marking the entire surface through tripple-clicking on of the the surfaces of the building. An additional right-click brings up a context-menu. Select *Create Component*. This process is essential, as it is responsible for creating the correct group-code in the DXF-file. In the dialog, shown in Figure 39, insert the label for the building.

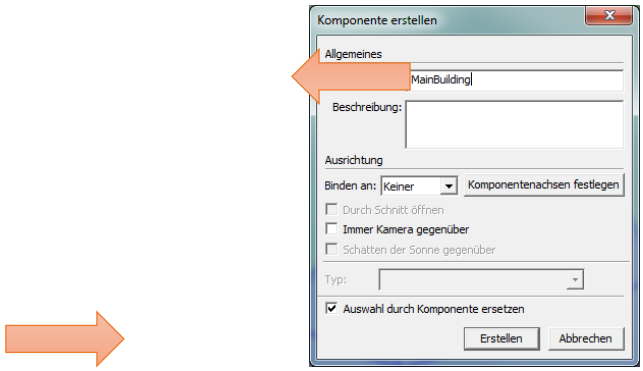


Figure 39: Dialog - Create Component

Once the button *Create* is pressed, a new object shoul be displayed in the workspace outline. In continuation the window for this building is drawn and included into the workspace. The windows are one of the essential components in the entire DXF-scene. They represent the "targets" for sunrays. All windows should be placed outside the wall. An appropriate distance between window-rectangle and the buildings wall can be defined with approximately 5 to 10 centimeters. For this example, a distance of 5 centimeters has been used. Once the window is placed in the model, it is necessary to create a component. This is achieved in the same way as with the building. A tripple-click on the window and an additional right-click onto the marked surface brings up the context-menu and *Create Component* can be selected. As specified, windows shall contain the prefix *window_* in the definition field. After pressing the *Create* button, the new component is displayed in the outline of the workspace. This window is now named *window_front_1*. However, any combination matching the regular expression *window. An asterisk* * is allowed. In order to assign window names to an actuator during the parametrization, it is possible to include floor, area, segment and sunblind-module information into the window-tag. E.g.: window_fl1_a2_seg02_sb1 (window on floor 1, area2, segment 02 for sunblind 1)

Doors which are equipped with sunblinds, such as those used on a terrace, are also considered windows. Therefore, naming of these must comply with the specifications for regular windows.

The surroundings are considered all objects, which are not part of the building or the category windows. Surroundings for this tutorial are modeled by simple cubes. Again, it is recommended to start with the ground plane and in order to reduce time, the extrusion-tool can be used to generate a three dimensional obect. Once the surrounding is drawn, it is again necessary to create a component, as described in the previous steps. The naming of this component must meet the specifications for surrounding objects.

Finally, the ground plane of the entire scene has to be drawn. It is recommended to specify the dimensions three to four times bigger than the farthest object inside the scene. This avoids windows being hit by negative elevation values.

Once the entire model is drawn, it is necessary to align the entire scene towards north. This is achieved by marking all components inside the workspace and using the rotation-tool. This tool requires 3 substeps. The first click with this tool specifies the center of rotation. The second click specifies the first axis from which to rotate. The final click defines the last rotation-point. Note, that the rotation-tool is colored in the color of the axis (x, y, z) around which will be rotated.

In order to export the entire scene, all components need to be selected. Open the File Menu and select *Export to DXF or STL*. Several dialogs will now pop up. The first will determine the Units in which the coordinates and components dimensions shall be exported, please select *Millimiters*. The second dialog specifies the DXF export options. Here it is necessary to select *triangular mesh*. Finally a file-dialog will ask for the location where to store the DXF-file.

Information about surrounding buildings (width, length, height) are, depending on the country, difficult to aquire. Geo-locating the entire scene in SketchUp Make 2017 is a built in feature and can reduce the need for aligning the entire model towards north or estimating the dimensions of surrounding objects and buildings. In order to add the geolocation to the model, navigate to the menu *File*, select *Geo-Location* and *Add Location*. The dialog *Add Location* asks the user to introduce the location and to select the region, as shown in Figure 40.

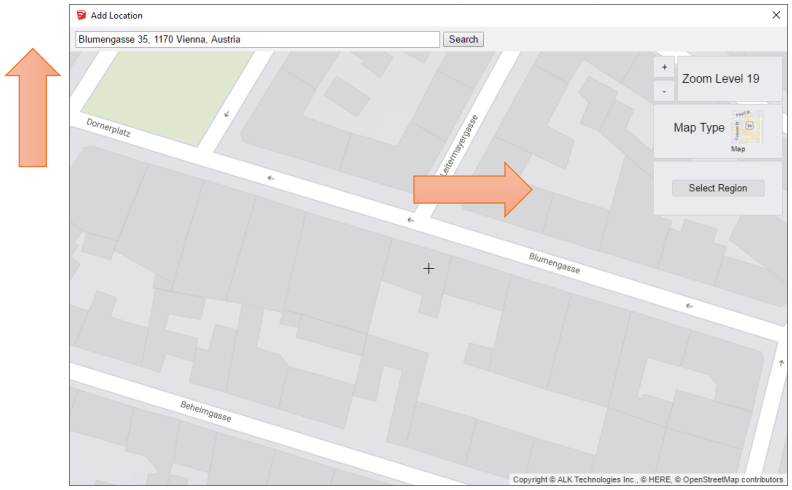


Figure 40: Geolocation of a model

The selected region can then be imported by pressing the button *Import*. The axis x, y and z are automatically aligned with the provided location, as shown in Figure 41.

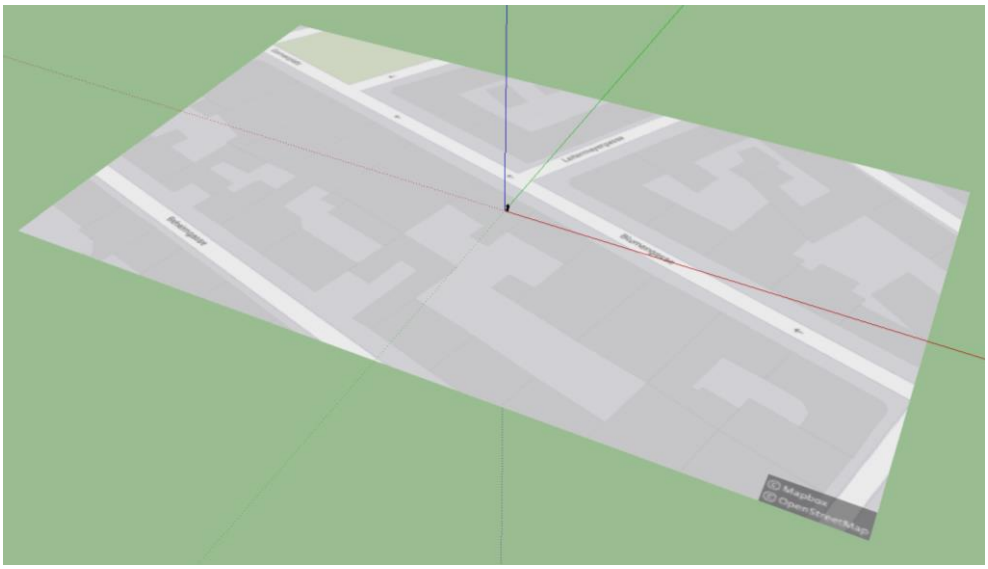


Figure 41: Import of geolocation for the year shade progression model

The imported data originates from open-streetmap and is pasted as an image into the xy-plane. The imported locations builds the base for modeling ground-planes of surrounding buildings and ensures preciser positioning of buildings and other objects.

9.6.4 Parameterization

The year shade progression has been designed as an additional feature to the sunblind module. The results of the implemented algorithm can take direct effect on the position of each individual actuator (ACTUATOR-MODE) or contribute to the position and rotation of an entire sunblind-zone (ZONE-MODE). This parameter is located in the sunblinds *CommLogic* in the subsection *Yearshade* called *sbYspMode*. The second parameter (*sbLuxSMultiplicator*) in this subsection is directly linked to the ZONE-MODE. This multiplicator is in charge of changing the received lux-value for the entire sunblind zone.

The third and last parameter *sbActuatorWindowName* is located in each actuator submodule. This string specifies the so-called window-tag and must correspond to a string specified in the DXF-file which describes a window. This string is case-sensitive.

The composite included in the actuator (YspDll) provides several outputs:

- outSun: This output describes the percentage of a window-surface exposed to the sun
- outDbgMsg: If an error occurs, than this output provides a human readable version of the output outErr
- outErr: This output provides an error code

The following Table 89 gives an overview of all available error codes.

Error Code	outDbgMsg	Description
0	No Error	
-1	Unknown Error	An unknown error has occurred
-2	Memmory Error	The year shade progression module was not able to allocate enough memory for the required operation
-3	File Error	This error is printed when errors concerning the DXF-file have occurred. This involves “File not found” and “File could not be opened”
-4	Parsing error in line: x	Error reported by the parser, when parsing a new DXF-file. (x = number of line)
-5	Value Error	A function received an invalid value
-6	Failure	
-7	Tag error in line: x	Error reported by the parser, when parsing a new DXF-file. (x = number of line)
-8	No registered DLL-FB-YSP	Error reported by the firmware, if the request-queue is empty
-9	Azimuth Error	Error reported by the firmware, as soon as a wrong azimuth value is passed. Az =[0,360]
-10	Elevation Error	Error reported by the firmware, as soon as a wrong elevation value is passed. El = [-90, 90]
-11	Instance Number Error	Error reported by the firmware if inInstanceNumber is out of bounds. inInstanceNumber = [0, 1]
-12	Window not found	Error reported by the firmware, as soon as a wrong window-tag is provided. The window-tag must be present in the DXF-File
-13	Currently parsing	The year shade progression instances are currently parsing the content of the DXF-file

Table 89: Error codes of the year shade progression

The DXF-file can be uploaded through the web-user interface. The upload-form is located in the Menu *Config* and the sub-menu *Yearshade Progression*. This page in the webui also

provides the possibility to delete and download the DXF-file located on the controller. In addition information about the file is provided to the user.

The visualization of the DXF-file content is located in the web user interface sub-menu *Yearshade Progression* of the menu *Statistics*. This page provides the user with information about detected windows, surroundings (season-tag) and a visualization of the scene. The list of windows includes two additional entries at the top of the list. A blank entry, which provides the user with an overview of the three dimensional model. The second entry called Bird Eye View provides the user with a top-view of the entire scene. This form of visualization is especially interesting for validating if the entire model has been aligned correctly towards north.

The subsection *Simulation* enables the user to simulate different dates and times in order to determine if any surrounding objects would cast shadows based on the specified sun position. This feature needs to be activated by checking the checkbox *Active*. Once checked, the webui provides input-fields for the desired date and time. Clicking the button *Simulate*, the values for azimuth and elevation are calculated and the sun is repositioned inside the visualization.

9.6.5 Validation

In order to determine if a generated DXF-file is suitable for the year shade progression, it is necessary to take a closer look at the content of this file. The DXF-file itself is a textbased file, which can be opened with Editors such as Notepad++.

The export algorithm of the DXF/STL export plugin generates the greatest possible triangles. This means for a rectangle, that it is divided into two trinagles as shown in Figure 42.

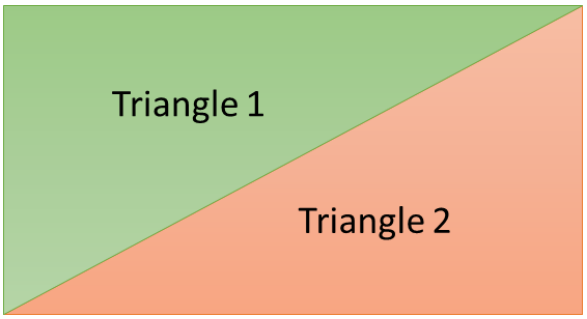


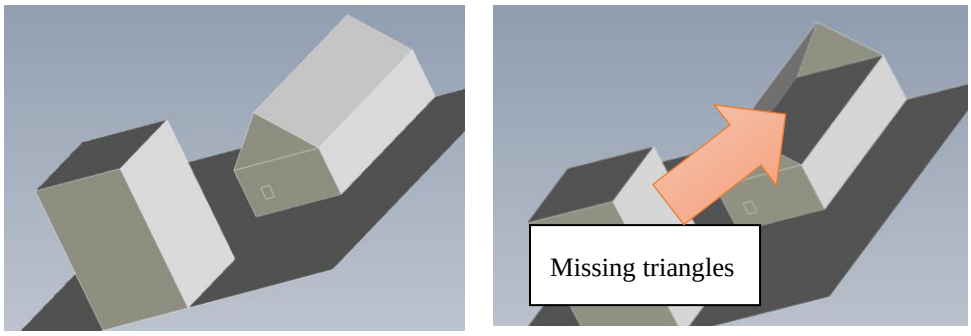
Figure 42: DXF/STL export of rectangles

This division into triangles would result in Table 90 for the previously mentioned example.

Component	Count
Ground	2
MainBuilding	18
Some_Fancy_Street_14	12
window_front_1	2

Table 90: DXF/STL export results

For bigger scenes, however, this is a very time intensive process. Therefore, eDrawings can be used to check if triangles have been correctly exported. The following Figure 43 shows the difference between correctly and unsuccessful exported triangles.



(Successful export) (Unsuccessful export)

Figure 43: Difference between successful and incorect DXF/STL export

In addition, the visualization provided in the webui, can be used to verify, if the entire scene has been aligned correctly towards north. This is achieved by loading the Page *Yearshade Progression* in the *Statistics* menu and selecting *Bird's-eye View* among the available window-tags. Figure 44 shows the corresponding select-element in order to switch between the model-view, the birds-eye view and the window-view. The model-view (first and blank entry in the select-list of the window-tag) itself just provides a standard overview of the scene. The window-view on the other hand, offers the user the possibility to position the camera at the same location as the selected window.

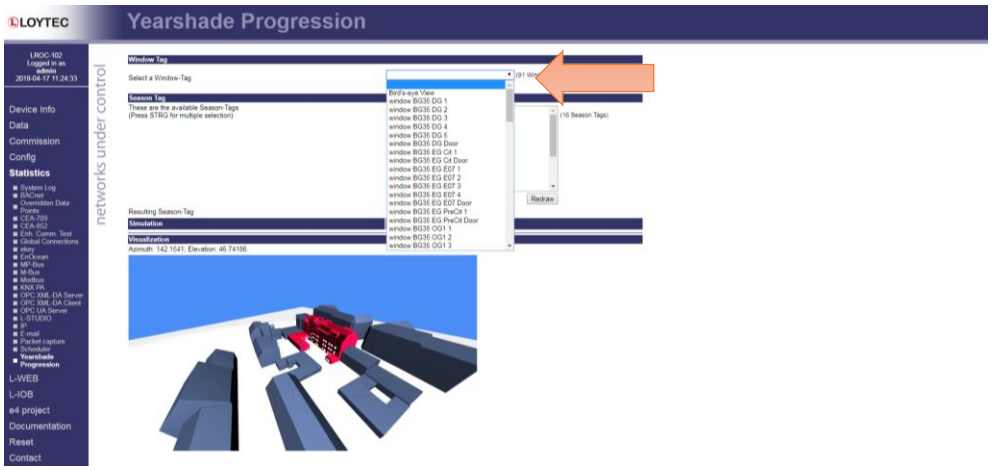


Figure 44: Switching views in the year shade progression visualization

If a window is selected (window-view) and the camera faces towards the building, than, the faces of the window need to be reversed. In addition, once a window is selected, the corresponding window-tag is printed below the select-element.

The current sun-position is displayed above the visualization with azimuth and elevation values. The sun itself is visualized as an orange sphere inside the entire scene. The window-view offers a field of view of approximately 140 degrees in azimuth and 90 degrees in elevation. This limited field of view could be a reason for not displaying the sun in window-view.

10 Window Function

10.1 Window Overview

The window module allows to control a motorized window to achieve the following functions:

- Night purge: This function allows to automatically use cold night are to purge excess heat from the building
- Protection: The window can be controlled by the following environmental conditions:
  - Rain: Limit max position
  - Fire: Allow to control window position by override
  - Wind: Limit max position
  - Dewpoint: Close window
  - Outdoor temperature: Limit max position
- Central functions: Windows or window groups can be controlled from the area, floor or building level.
- Manual control: The user can override the automatic functions within the current weather limits.

10.2 OPC/BACnet Interface

Table 91 shows the data points available on the OPC and BACnet interface.

Datapoint	Type	Description
wndZoneId	String	Window Zone ID
wndManCmd	Multistate	Manual Command
wndManArg	Real	Manual Command Argument
wndPosFb	Real	Window position
wndEffMaxPosFb	Real	Maximum position feedback
wndWindAlarmFb	Binary	Wind alarm active
wndRainAlarmFb	Binary	Rain alarm active
wndOATAlarmFb	Binary	Outdoor Air Temperature alarm active
wndDewAlarmFb	Binary	Dewpoint alarm active
wndFireAlarmFb	Binary	Fire alarm active
wndCommAlarmFb	Binary	Communication alarm active
wndIsMaster	Binary	Module is zone master

Table 91: Window OPC/BACnet Interface

10.2.1 Zoning

Window modules can be grouped into zones. All window modules within a room having the same zone ID will operate synchronously. A master module will be automatically elected. All modules execute the same control algorithms, but only the master module is allowed to send window drive commands to the slave modules. User inputs received by slave modules are directed to the master module.

The *wndZoneId* data point assigns the sunblind module to a zone. The default value is W1.

Note: *The window module uses the same communication module as the HVAC module. Please do not use the same zone IDs for window and HVAC zones.*

The *wndIsMaster* data point indicates that this module is the master module for its zone.

10.2.2 Manual Window Control

The following data points are used to control windows manually.

The *wndManCmd* and *wndManArg* data points are used to control all window modules in the room.

The *wndManCmd* data points supports the following functions:

- NOP(1): No operation
- SET_ABS(2): Set window to absolute position given by *wndManArg*.
- SET_REL(3): Change manual position by *wndManArg*.
- STOP(4): Stop window and enter manual mode.
- AUTO(5): Disable manual override and enable automatic mode again.

The manual control data points also indicate the last manual command entered in the same or another zone member.

10.2.3 Alarm Feedback

The alarm feedbacks indicate that the sunblind has moved to the configured protection position and cannot be controlled manually or by an automatic mode.

- *wndWindAlarmFb:* This alarm is asserted when the wind speed limits the maximum open position.
- *wndRainAlarmFb:* This alarm is asserted when a rain alarm limits the maximum open position
- *wndOATAlarmFb:* When the outdoor air temperature exceeds the high temperature or falls below the low temperature, this alarm is asserted and the window position is limited.
- *wndDewAlarmFb:* This alarm is asserted when the indoor temperature is below the outdoor dewpoint temperature.
- *wndFireAlarmFb:* This alarm indicates a general or fire override.
- *wndCommAlarmFb:* This alarm indicates that the communication to the weather station failed and the window module is fixed to the safe position (closed).

10.2.4 Window Feedback

The *wndPosFb* data point indicates the current window position.

The wndEffMaxPos data point displays the currently maximum allowed window position due to weather alarms.

10.3 Window Functions

10.3.1 Function Summary

The window module allows to control a motorized window for night purge applications.

The primary function is to open the windows during night when the environment conditions are suitable for cooling the building in an energy-efficient way.

The night purge function opens the window based on indoor and outdoor temperature automatically. Additional conditions are outside dewpoint temperature, wind, rain and, fire override and occupancy.

Users can manually open the windows based on the limits posed by the environment conditions.

Window zones can be centrally overridden and have their own wind communication, as a wind alarm not necessarily closes the window, but might limit the max open position depending on wind speed.

10.3.2 Parameters

Table 60 shows the light module communication and core parameters.

Parameter Name	Datapoint	Type	Default	Description
Zone ID	wndZoneId	String		Window Zone Id
Façade Group	wndFacadeGroup	String		Façade group
Façade Direction	wndFacadeDir	Analog	0	Façade direction (1..16)
Night Purge Enable	wndNightPurgeEnable	Binary	TRUE	Enable Night Purge Function
Night Purge Setpoint	wndNightPurgeSetpoint	Analog	20 °C	Night purge setpoint
Night Purge Start	wndNightPurgeStart	Analog	0h	Start hour for night purge
Night Purge End	wndNightPurgeEnd	Analog	6h	End hour for night purge
Night Purge Limit Low	wndNightPurgeLimitLow	Analog	10 °C	Night purge outdoor temp limit
Night Purge Setpoint Hyst On	wndSetpointHystOn	Analog	0.5 K	Setpoint hysteresis
Night Purge Setpoint Hyst Off	wndSetpointHystOff	Analog	0 K	Setpoint hysteresis
Night Purge Outdoor Hyst On	wndOutdoorHystOn	Analog	3K	Outdoor hysteresis
Night Purge Outdoor Hyst Off	wndOutdoorHystOff	Analog	1K	Outdoor hysteresis
Rain Pos Limit	wndRainLimit	Analog	0%	Limit during rain condition
Outdoor Temp Pos Limit	wndOATPosLimit	Analog	0%	Limit during cold or hot outdoor condition
Outdoor Temp Limit High	wndOATLimitHigh	Analog	25°C	High temperature alarm
Outdoor Temp Limit Low	wndOATLimitLow	Analog	10°C	Low temperature alarm
Occupancy Hold Time	wndOccHoldTime	Analog	900s	Occupancy hold time

Table 92: Window module parameters.

Zone ID (wndZoneId)

The zone id parameter defines the zone to which this window module belongs.

Façade Group (wndFacadeGroup)
Façade Direction (wndFacadeDir)

The façade parameters define to which wind limit the module listens. The façade group can by empty for the default façade group. The direction can have the value of 0 for disable wind function or 1..16 to select the one of the 16 supported façade directions to receive wind limit data.

Night Purge Enable (wndNightPurgeEnable)

The enable parameter is the main switch for the night purge function.

Night Purge Setpoint (wndNightPurgeSetpoint)

The night purge function is activated when the room temperature is avobe the night purge setpoint. See also the hysteresis parameters below

Night Purge Start (wndNightPurgeStart)
Night Purge End (wndNightPurgeEnd)

The night purge function is active between the start and end hour given by these data points. The start hour can be larger than the end hour to enable night purge over midnight.

Night Purge Limit Low (wndNightPurgeLimitLow)

This parameter locks the night purge function below this temperature. This function is implemented with a fixed 0.5K hysteresis.

Night Purge Setpoint Hyst On (wndSetpointHystOn)
Night Purge Setpoint Hyst Off (wndSetpointHystOff)
Night Purge Outdoor Hyst On (wndOutdoorHystOn)
Night Purge Outdoor Hyst Off (wndOutdoorHystOff)

The night purge function is activated when

1. The indoor temperature is greater than the setpoint + wndSetpointHystOn
2. The indoor temperature is greater than the outdoor temperature + wndOutdoorHystOn

The night purge function is deactivated when

1. The indoor temperature is less than the setpoint + wndSetpointHystOff
2. The indoor temperature is less than the outdoor temperature + wndOutdoorHystOff

Rain Pos Limit (wndRainLimit)

This parameter selects the maximum allowed open position during rain conditions.

Outdoor Temp Pos Limit (wndOATPosLimit)

This parameter selects the maximum allowed close position during low/high outdoor temperature conditions.

Outdoor Temp Limit High (wndOATLimitHigh)
Outdoor Temp Limit Low (wndOATLimitLow)

These parameters select the low and high temperature at which the outdoor temperature alarm is asserted.

Occupancy Hold Time (wndOccHoldTime)

This parameter defines the occupancy hold time.

10.3.3 Functional Description

The window module has the following modes using the following priorities in decreasing order:

1. Override (Fire or other alarm)
2. Dew Point alarm (forced close)
3. Manual control
4. Automatic control (Night purge)

The window module does not employ hard-coded alarms, but allows to limit the maximum window position.

The following conditions are processed, starting with highest priority.

1. Wind: The wind speed is processed in the area, floor or building coupler and translated into a maximum open position. The wndWindLimitFav block is used to implement this function. Any window can be opened up to this limit.
2. Fire (Override): The fire alarm overrides all manual or automatic functions, except the wind alarm. If the fire control needs to open windows more than the current wind limit, it has to change the centrally distributed wind limit.
3. Dewpoint: If the outdoor dewpoint temperature is above the minimum zone temperature, the windows will be closed
4. Rain limit: On rain condition, the maximum open position is limited.
5. Outdoor temperature function: For very hot or cold days, the maximum position can be limited.

The minimum of the alarm limits are applied to every other function, including wind alarm, manual mode and night purge program.

The night purge application has the following behavior.

The night purge application starts when all following conditions are true:

1. Night purge is enabled
2. Zone is unoccupied
3. Time is between start and end hour
4. Indoor temp > setpoint + wndSetpointHystOn
5. Indoor temp > outdoorTemp + wndOutdoorHystOn
6. Outdoor temp > wndNightPurgeLimitLow + 0.5K

The night purge application stops when any of the following conditions becomes true:

1. Night purge is disabled
2. Zone is occupied
3. Time is not between start and end hour
4. Indoor temp < setpoint + wndSetpointHystOff
5. Indoor temp < outdoorTemp + wndOutdoorHystOff
6. Outdoor temp < wndNightPurgeLimitLow - 0.5K

Manual operation can override night purge. The manual operation is always limited by the maximum allowed window position. The user can also stop the motor to enter manual mode, but this is only possible to stop a manual drive command.

10.4 Window Actutators

10.4.1 Favorite Actuator (sbWindowFav)

The favorite actuator is used to control windows on conventional motors.

Favorite	Type	Description
ActClose	Analog	Window close command in milliseconds
ActOpen	Analog	Window open command in milliseconds
WindowClosed	Binary	Window close contact
WindowOpen	Binary	Window open contact

Table 93: Window favorite actuator data points

Table 93 lists the favorite data points of the window actuator. If the window shall close or open, the corresponding data point is set to the required drive time in milliseconds. Value 0 means to stop movement while a negative number means to switch on the motor direction permanently. Normally these favorites are connected to a LIOB DO which needs to be put into “Duration” mode.

Parameter Name	Datapoint	Type	Default	Description
Override Segment Id	SegId	Analog	0	Selects an alternate segment
Segment ID Feedback	SegIdFb	Analog		Shows the effective segment Id.
Override Ctrl Index	CtrlIdx	Analog	0	Defines the windows controller index
Actuator connected	IsConnected	Binary		Shows whether this actuator is connected to a controller
Actuator Close Time	CloseTime	Analog	30s	Window close time
Actuator Open Time	OpenTime	Analog	30s	Window open time
Actuator Contact Mode	ContactMode	Multi-state	NONE (1)	Window contact mode
Actuator Contact Close Pos	ContactClosePos	Analog	5%	Position of the close contact
Actuator Contact Open Pos	ContactOpenPos	Analog	95%	Position of the open contact
Actuator Alarm Delay	AlarmDelay	Analog	2s	Delay for alarm-initiated movements.
Actuator Auto Delay	AutoDelay	Analog	5s	Delay for automatic function movements
Actuator Command Fb	CommandFb	Analog	-	Requested position
Actuator Position Fb	PositionFb	Analog	-	Current position
Window Closed	WindowClosed	Binary	-	Close window contact state
Window Open	WindowOpen	Binary	-	Open window contact state
Actuator Contact Fault	ContactFault	Multistate	-	Contact fault alarm

Table 94: Window favorite actuator data points.

Table 94shows the data points of the window actuator.

Override Segment Id (SegId)
Segment ID Feedback (SegIdFb)
Override Ctrl Index (CtrlIdx)
Actuator connected (IsConnected)

The actuator can be either connected to the controller directly, or be configured to connect to an window controller on the save device.

The *SegId* and *CtrlIdx* parameters select the window controller, when they are not 0. The *SegIdFb* data point display the effective segment. The *IsConnected* data point indicates that the actuator communicated with its controller.

Actuator Close Time (CloseTime)
Actuator Open Time (OpenTime)

These parameters define the time for opening or closing a window. The controller automatically drives longer when moving to either end position to ensure that the window is closed or opened entirely.

Actuator Alarm Delay (AlarmDelay)
Actuator Auto Delay (AutoDelay)

The delay parameters define how long the actuator will delay a command due to an alarm condition and due to an automatic function. This can be used to stagger the motor movements for avoiding current peaks.

Actuator Command Fb (CommandFb)
Actuator Position Fb (PositionFb)

The *CommandFb* data point displays the last commanded position of the actuator while the *PositionFb* data point displays the current (calculated) position.

Actuator Contact Mode (ContactMode)
Window Closed (WindowClosed)
Window Open (WindowOpen)
Actuator Contact Fault (ContactFault)

The actuator provides two inputs to monitor the window state.

The *ContactMode* parameter selects which contacts are available:

- NONE (1)
- CLOSE (2)
- OPEN (3)
- CLOSE_OPEN(4)

The *WindowClosed* and *WindowOpen* data points indicate whether the contacts are open or closed.

NOTE: *The contacts are not used as end position sensors for motor deactivation. The connected motor has to turn of automatically at end positions..*

If the open and close contacts are not activated for the movement duration, an error is reported in the ContactFault data point:

- OK (1)
- NOT_CLOSING (2): The close contact was not activated although the window should be closed.
- UNEXPECTED_CLOSED (3): The close contact is activated although the window is open
- NOT_OPENING (4): The open contact was not activated although the window should be opened.
- UNEXPECTED_OPEN (5): The open contact is activated although the window should be closed
- BOTH_CONTACTS (6): Both contacts are activated which indicated electrical or mechanical contact failure.

The motor position and the feedback contacts are used to act as a window contact sensor for the LROC system. It uses the window position and available physical contacts to signal open windows to the other control functions.

Actuator Contact Close Pos (ContactClosePos)
Actuator Contact Open Pos (ContactOpenPos)

The *ContactClosePos* and *ContactOpenPos* parameters define where the contacts are placed on the percentile movement span. This is used for the contact check logic.

11 General Functions

11.1 Routed AO Actuator

The coRoutedAO block allows to dynamically send the output of a HVAC or Light controller to an analog data point. Note that routed sunblind actuators have special functions so that they are described in section 9.4.

Parameter Name	Datapoint	Type	Default	Description
RIO Segment ID	rioSegId	Int	0	Segment ID
RIO Controller Index	rioCtrlIdx	Int	0	Controller index
RIO AO Type	rioAOType	Enum	UNASSIGNED (1)	Output type
RIO Name	rioName	String	""	Description

Table 95: coRoutedAO Parameters

Table 95 shows the parameters of the coRoutedAO block.

RIO Segment Id (rioSegId)

This number defines the segment ID of the connected segment. If this value is 0, the segId input of the block is used, if it is non-zero, it overrides the value of the segId input. The segment ID must be used on the device, else the values are ignored.

RIO Controller Index (rioCtrlIdx)

This value selects between multiple controllers of the segment. For example, if there are 3 Light controllers, their ctrlIdx input have to match this parameter. A value of 0 means that the actuator is disconnected.

RIO AO Type (rioAOType)

This selects the control value to be send to the analog output.

- UNASSIGNED (1): Value is ignored
- HEAT (2): Heat sequence output (%)
- COOL (3): Cool sequence output (%)
- 6WAY_2_6_10 (4): 6way-valve with 2V, 6V and 10V (to be implemented)
- FLOW (5): Flow sequence output (%)
- LAMP_DOOR (6): Door lamp value (%)
- LAMP_WINDOW (7): Window lamp value (%)
- HEATCOOL (8): Heat/Cool for 2-pipe systems (%)

RIO Name (rioName)

This parameter is a string for display purposes.

11.2 Implementing Control Buttons

The LROC library allows to implement control functions, like light and sunblind switches with any communication technology and also for remote scenarios, for example:

- A standard two-button light switch that controls the local zone
- A central two-button sunblind switch that controls an entire room
- A central floor switch for reverting all light or sunblind zones to auto mode.

These functions can be implemeted by a CAT providing the binary datapoints for the switches (e.g. EnOcean, Local IO or KNX) and one or more blocks described in this section.

An example is given by the *LdaliBm2HmiTemplate* block which can be copied to the project and adapted as needed.

The logic blocks for the implementation are described here:

- Light: Section 8.6.2
- Sunblind: Section 9.5.2

12 Firmware Update

The L-ROC firmware supports remote upgrade over the network and the USB ports.

The following firmware images are available for the different device models:

- LROC-102/400/401/402: linx2_a5d3_6_2_0_20170614_1323.dl

12.1 Firmware Update via LWEB-900

The preferred option to upgrade an L-ROC system is to upload the firmware into tht LWEB-900 server and to assign the firmware to all LROC devices, as described in [5].

12.2 Firmware Update via the Configurator

The firwmare image can be updated using the Configurator. For this purpose, it is recommended to have the device connected to the Ethernet and to have a valid IP configuration. The Configurator must be installed as described in the LINX Configurator User Manual [2].

To Update the Firmware using the Configurator

1. Start the Configurator from the Windows Start menu: Start → Programs → LOYTEC LINX Configurator → LOYTEC LINX Configurator.

2. Select the menu: **Connection → Connect via FTP**. This opens the FTP connection dialog as shown in Figure 45.

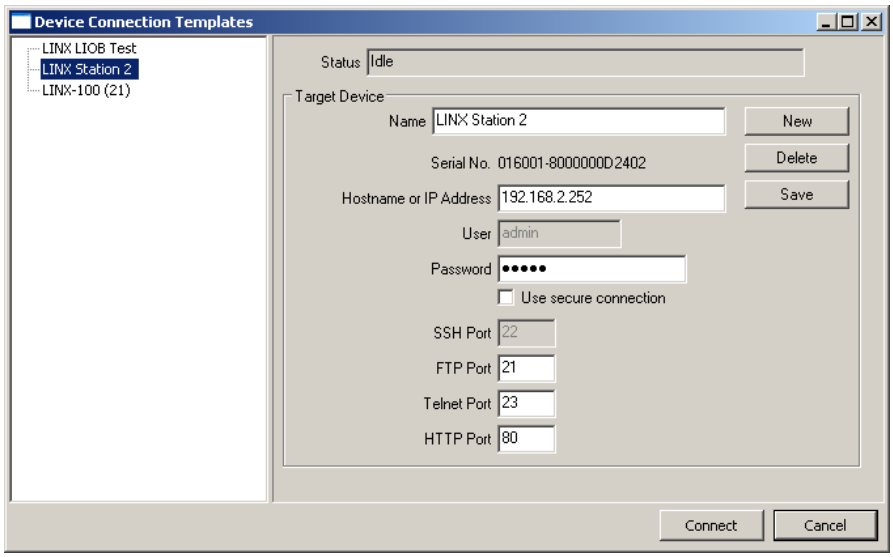


Figure 45: FTP connection dialog.

3. In the connection dialog, enter the IP address of the device as well as the admin user's password. The default password is 'loytec4u' (older firmware versions used 'admin'). This can be changed via the Web interface and reset via the console UI.

4. If the device uses other port settings than the standard settings or the device is operated behind a NAT router, adapt the FTP and Telnet ports accordingly.

5. Click on **Connect**.

6. Optionally, check for updates by selecting the menu **Help → Check for updates …**. This function checks for new firmware and Configurator versions.

7. Select the menu: **Firmware → Update …**

8. This opens the **Firmware Update** dialog as shown in Figure 46. Click on the button ... and select the firmware image.

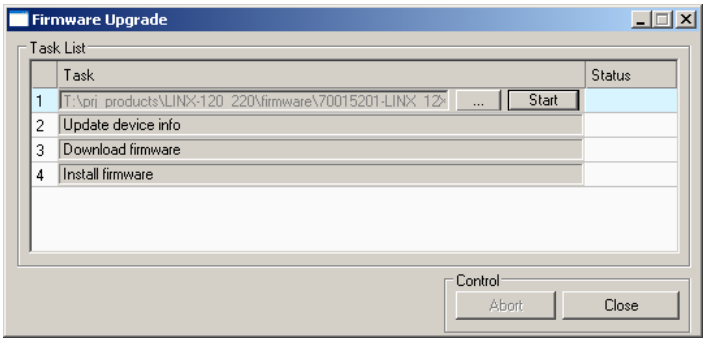


Figure 46: Firmware Update dialog of the Configurator.

9. Click on **Start** and observe the download progress.

10. When the download is complete, a dialog appears. Click **OK**.

11. In the Firmware Update dialog, click **Close**.

12. The device's firmware has now been successfully upgraded.

12.3 Firmware Update via the Web Interface

The device's firmware can also be upgraded using the Web interface. This option can be found in the **Config** menu under the **Firmware** item. For more details refer to the LOYTEC Device User Manual [1].

13 Troubleshooting

13.1 Technical Support

LOYTEC offers free telephone and e-mail support for the L-INX product series. If none of the above descriptions solves your specific problem please contact us at the following address:

LOYTEC electronics GmbH
Blumengasse 35
A-1170 Vienna
Austria / Europe

e-mail : ***support@loytec.com***
Web : ***http://www.loytec.com***
tel : ***+43/1/4020805-100***
fax : ***+43/1/4020805-99***

or

LOYTEC Americas Inc.
N27 W23957 Paul Road
Suite 103
Pewaukee, WI 53072
USA

e-mail: ***support@loytec-americas.com***
Web: ***http://www.loytec-americas.com***
tel: ***+1 (512) 402 5319***
fax: ***+1 (262) 408 5238***

or

LOYTEC Asia Corporation Ltd.
16F.-3, No. 155, Zhongyang Rd
Xindian District
New Taipei City 23150
Taiwan

e-mail: ***support-asia@loytec.com***
tel: ***+886 (2) 8913 7838***
fax: ***+886 (2) 8913 7830***

14 Specifications

14.1 Physical Specifications

14.1.1 LROC-10X

Operating Temperature (ambient)	0°C to +40°C
Storage Temperature	–10°C to +60°C
Humidity (non condensing) operating and storage	10 to 90 % RH
Power Consumption	24 VDC or 24 VAC ±10 %, typ. 2.5 W
In rush current	up to 950 mA @ 24 VAC
Power Supply 24V	24 VDC ±10 % (do not connect if SMI or DALI is used)
Enclosure	Installation enclosure 159 mm wide, DIN 43 880
Installation	DIN rail mounting (EN 50 022) or wall mounting
Environmental Protection	IP 40 (enclosure), IP 20 (screw terminals)

Table 96: General LROC-100/101/102 Specifications

14.1.2 LROC-40x

Operating Temperature (ambient)	0°C to +40°C		
Storage Temperature	-10°C to +85°C		
Humidity (non condensing) operating and storage	10 to 90 % RH		
Power Supply Mains	85-240 VAC, 50-60 Hz, 15 W max.		
Power Supply 24V	24 VDC ±10 % (do not connect if SMI or DALI is used) Mains supply and 24V supply must not be used simultaneously.		
Dimensions [mm]	340 x 144 x 70 (L x W x H)		
Installation	mountable directly via two oblong holes (ø 7 mm, distance 315 mm) or system distribution box LBOX-ROCx		
Environmental Protection	IP 40 (enclosure), IP 20 (screw terminals)		
Types	LROC-400	LROC-401	LROC-402
Interface	2x Ethernet (100Base-T), 2x USB-A, 1x LSTAT, 1x MP-Bus, 1x KNX		

	1x SMI (inc. SMI Power Relay), 1x DALI (inc. DALI PSU 110 mA typ[6]./250 mA max.), 1x RS-485, 1x EXT, 1x EnOcean (Europe 868 MHz)		-
Universal Input (UI)	10	-	10
Digital Input (DI)	2	-	2
Analog Output (AO)	8	-	8
Digital Output L1 (DO)	12 (Relay)	-	12 (Relay)
Digital Output L2 (DO)	12 (Relay)	-	12 (Relay)
Digital Output L3 (DO)	8 (Triac)	-	8 (Triac)
Digital Output Specification	Relay: 10 A, 277 VAC, 30 VDC Triac: 0.5 A, 24-240 VAC, 50-60 Hz		

Table 97: General LROC-400/401/402 Specifications

Types		LROC-400	LROC-401	LROC-402
Serial Numbers		025401- 025402- 025403- 025404-	033101- 033102- 033103- 033104-	033201- 033202- 033203- 033204-
Power Consumption idle. [W] • Ethernet connected • Backlight ON	P_{idle}	3.4		
Power Consumption per Relay switched ON [W]	P_{Relay}	0.15	-	0.15
DALI (160 mA typ.) [W]	P_{DALI}	2.9	2.9	-
LSTAT Supply Power [W]	P_{LSTAT}	$P_{LSTAT} = 15\ W - P_{idle} - (N_{Relay} \times P_{Relay}) - P_{DALI}$		
LSTAT Supply Power [W] (worst case)	$P_{LAST,WC}$	5.1	8.7	8.0

ⓘ The overall power consumption must not exceed 15 W!

Table 98: LROC-400/401/402 Power Consumption (serial number prefix ending in 01 to 04)

[6] LROC-40x models manufactured after calendar W15/2018 can supply up to 160mA on the DALI bus. Former models provide up to 110mA DALI supply current. All variants have 250mA short current, so that they cannot be used in parallel to other DALI power supplies.

Types		LROC-400	LROC-401	LROC-402
Power Consumption idle. [W] • Ethernet connected • Backlight ON	P_{idle}	3.4		
Power Consumption per Relay switched ON [W]	P_{Relay}	0.20	-	0.20
DALI (160 mA typ.) [W]	P_{DALI}	2.9	2.9	-
LSTAT Supply Power [W]	P_{LSTAT}	$P_{LSTAT} = 15\ W - P_{idle} - (N_{Relay} \times P_{Relay}) - P_{DALI}$		
LSTAT Supply Power [W] (worst case)	$P_{LAST,WC}$	3.9	8.7	6.8

ⓘ The overall power consumption must not exceed 15 W!

Table 99: LROC-400/401/402 Power Consumption (serial number prefix ending in 05 and later)

14.2 I/O Specification

14.2.1 UI - Universal Input

UIs are universal inputs for four different input types. They have an input voltage range of 0V to 10V, and can withstand up to 30V. The UIs correspond to class 1 with a relative accuracy of +/-1% (of measured value) between 1V and 10V, and an absolute accuracy of +/-10mV between 0V and 1V. The ADC resolution is 16 bits. Galvanically isolated sensors resp. switches must be connected. Universal inputs can be configured as:

- **Binary Input (Digital Input):** input impedance > 20kΩ, sampling period 10ms.
    - In voltage mode, the threshold values are < 0.8V for low level and > 2V for high level.
    - In resistance mode, the threshold values are < 1.9kΩ for low level and > 6.7kΩ for high level.

    Between the threshold values, the resulting level of the UI is not defined.
- **Voltage Metering 0-10V**: input impedance > 20kΩ, sampling period < 1s.
- **Current Loop 4-20mA**: input impedance 249Ω, sampling period < 1s. An internal shunt of 249Ω is available for some universal inputs. Otherwise, an external resistor of 249Ω must be used as a shunt.
- **Resistance Measurement**: input impedance 10kΩ, sampling period < 1s. Resistors in the range of 1kΩ to 100kΩ can be measured.

The average sampling period *p* of analog inputs depends on the number of active (non-disabled) universal inputs *n* that are configured in analog mode. The formula for *p* is:

$$p = n * 125ms$$

This means if e.g. only two UIs are configured as analog inputs, a new sample is taken every 250ms (on average) for each of the two inputs. The UIs configured as digital inputs are unaffected (sampling period always 10ms) by this formula.

14.2.2 DI - Digital Input, Counter Input (S0-Pulse)

DIs are fast binary inputs, which can also be used as counter inputs (S0). They follow the S0 specification for electric meters and have a sampling period of 10ms. They change state at a load of 195Ω between the DI terminal and GND. Galvanically isolated sensors resp. switches must be connected.

14.2.3 AO - Analog Output

AOs are analog outputs with a signal range of 0V to 10V (up to 12V), a resolution of 10 bits, and a maximum output current of 10mA (short circuit proof). The accuracy over the whole range is +/-100mV.

14.2.4 DO - Digital Output

The following digital outputs are available:

- Relay 6A Output: Switching capacity 6A, 250VAC resp. 30VDC. Max in-rush current 6A, max. 600W (resistive) @ 250VAC.
- Relay 10A Output: Switching capacity 10A, 250VAC resp. 30VDC. Max in-rush current 10A, max. 1600W (resistive) @ 250VAC.
- Relay 16A Output: Switching capacity 16A, 250VAC resp. 30VDC. Max in-rush current 80A, max. 2000W (resistive) @ 250VAC.
- TRIAC Output: Switching capacity 0.5A, 24 to 230VAC. No external relays must be connected.

When switching higher loads than specified an interface relay must be used. When connecting an external interface relay to a L-IOB relay, a quenching circuit like a varistor or RC element must be used.

14.3 Resource Limits

14.3.1 L-ROC Models

Table 100 specifies the resource limits of the different L-ROC models in firmware 6.4.4. Note that the resource limits depend on firmware version.

Model / Limits	LROC-102	LROC-400	LROC-401	LROC-402	LDALI-PLC4	
Total number of data points	30,000	30,000	30,000	30,000	30,000	
OPC Tags	10,000	10,000	10,000	10,000	10,000	
User Registers	10000	1,0000	1,0000	1,0000	1,0000	
NVs (static, dynamic)	1,000	1,000	1,000	1,000	1,000	
External NVs	2,000	2,000	2,000	2,000	2,000	
Alias NVs (ECS and legacy mode)	2,000	2,000	2,000	2,000	2,000	
Address table entries/legacy	1000/ 15	1000/ 15	1,000/ 15	1000/ 15	1,000/ 15	
LONMARK Calendar objects	1 (25 calendar patterns)					
LONMARK Scheduler objects	100 (max. AST configuration size 384KB, 64 data points per scheduler)					
LONMARK Alarm Servers	1	1	1	1	1	
BACnet objects (analog, binary, multi-state)	2000	2000	2000	2000	2000	
BACnet client mappings	5000	5000	5000	5000	5000	
BACnet scheduler objects	100	100	100	100	100	
BACnet calendar objects	25	25	25	25	25	
BACnet notification classes	32	32	32	32	32	
BDT max recommended	100	100	100	100	100	
KNX Communication Objects (per interface)	1000	1000	1000	1000	1000	
Trend Logs	512	512	512	512	512	
Total trended data points	256	256	256	256	1000	
Total aggregated size	6MB	60MB	6MB	60MB	60MB	
E-mail templates	100	100	100	100	100	
Math objects	100	100	100	100	100	
Alarm Logs	10	10	10	10	10	
Modbus data points	4,000	4,000	4,000	4,000	-	
M-Bus data points	1,000	1,000	1,000	1,000	-	
EnOcean devices	100	64	64	64	-	
EnOcean data points	1000	1000	1000	1000	-	
SMI devices (per channel)	16	16	16	-	-	
Connections (local)	1,000	1,000	1,000	1,000	2,000	
Connections (global)	250	250	250	250	250	
L-WEB Clients (concurrent)	32	32	32	32	32	
L-IOB Modules	24	2 (IP)	2 (IP)	2 (IP)	24 (IP)	
DALI Channels	-	1	1	-	4	
DALI Ballasts / Channel	-	64	64	-	64	
DALI Sensors / Channel	-	16	16	-	16	
DALI Buttons / Channel	-	64	64	-	64	
MP-Bus Channels (builtin)	-	1	1	1	-	
MP-Bus Devices	16	16	16	16	-	

Table 100: Resource limits of different L-ROC models

15 References

[1] LOYTEC Device User Manual, LOYTEC electronics GmbH, Document № 88086503, May 2017.

[2] LINX Configurator User Manual, LOYTEC electronics GmbH, Document № 88086703, May 2017.

[3] NIC User Manual 4.2, LOYTEC electronics GmbH, Document № 88067217, April 2013.

[4] LWEB-802/803 User Manual 2.4, LOYTEC electronics GmbH, Document № 88074217, December 2016.

[5] LWEB-900 User Manual 2.2, LOYTEC electronics GmbH, Document № 88081504, June 2017.

[6] L-VIS User Manual 6.1, LOYTEC electronics GmbH, Document № 88068521, January 2017.

[7] LIOB-10x/x5x User Manual 5.0, LOYTEC electronics GmbH, Document № 88078611, May 2017.

16 Revision History

Date	Version	Author	Description
2017-08-08	6.2	TR	Initial version
2018-06-18	2.1.4	TR	Changed version to follow library version Changed DALI powersupply specification to 160mA for production dates after KW15/2018 Clarified that 24V AC is not allowed on LROC-40x Added Routed IO documentation
2018-09-20	2.1.5	TR	Added Year Shade Progression Information Added new data points in HVAC, Lighting and Sunblinds Described improved HVAC optimum start function Minor corrections
2019-01-16	2.1.6	TR	Added LDALI-MS2 support Added sbShadingGlareDelayOff parameter Added new CLC control modes Added new CLC enable/disable leave commands
2019-04-01	2.1.8	TR	Updated power calculation table for LROC-400/401/402 for serial prefix numbers ending in 05 and later. Updated information on BACnet weather data points. Light switches support DALI dimming.
2019-07-26	2.1.10	TR	Added HEATCODE mode for coRoutedAO. Added sections for light and sunblind buttons Added external flow override documentation
2019-11-15	2.1.12	TR	Added Window module and actuator Added flow aggregation documentation Added multi façade weather functions Added new CLC commands
2020-02-14	2.1.14	DB	Year shade progression documentation
2020-01-21	2.1.14	TR	Added multi-façade sunblind parameters
2020-03-02	2.1.16	TR	What is new 2.1.16