User Manual

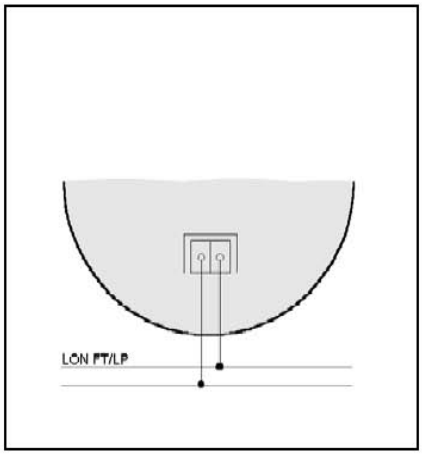
LON Multi-Sensor ILA-22
Art. no.: MTN880551

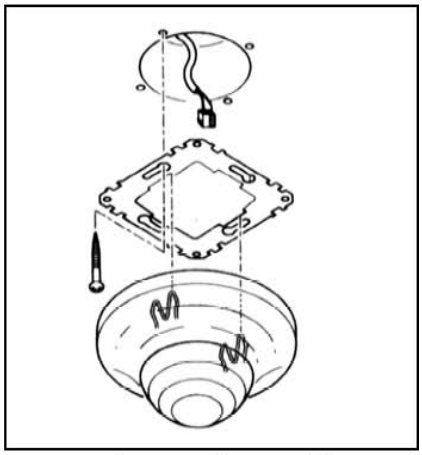
Directory

LON Multi-Sensor ILA-22

Art. no.: MTN880551

1. Description

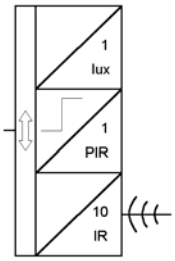




LPT | UPI

- combination of occupancy sensor, light sensor and IR receiver
- IR receiver for control of various room functions (in combination with
- IR Remote Control, art. no MTN570222
- flush-mounting (surface-mounting in combination with Surface Mounting Box, art. no. MTN550619
- circular sensor range with a diameter of approx. 14 m at 2.5 m mounting height
- detection range: 360 degrees
- several detection levels with over all 544 control segments in 136 zones
- brightness sensor for daylight-dependent light control, sensor range: 10 ... 1.000 lux
- dimensions of surface-mounted sensor: 105 x 42.6 mm (D x H)
- software application to translate the detected movements (according to LonMark profile “Occupancy Sensor (1060)” and “Occupancy Controller (3071)”), resp. the detected brightness (LonMark profile “Light Sensor (1010)”) into LON messages for occupancy-dependent light or sunblind control as well as for control of room functions (LonMark profile “Switch (3200)” and “Scene Panel (3250)”) by use of the received IR signals

2. Function



Installed on the ceiling of a room, the Multi-Sensor registers the presence of persons within a certain detection area and simultaneously measures the intensity of the natural light.
Furthermore a built-in infrared receiver is integrated.
The Multi-Sensor is used to control lighting, blinding, heating, ventilation and air-conditioning applications.
The integrated movement detector is based on the "Passive Infrared Method" (PIM).
It detects the difference of thermal radiation (infrared spectrum) of moving objects.
The size of the detection area depends on the room height, e. g. an installation height of 2.5 m results a detection area with a diameter of 14 m on the ground.
It is partitioned into radial levels with over all 544 switching sectors in 136 zones.
The integrated brightness sensor measures permanently the lux level in a range of 10 ... 1.000 lux and propagates the values to the LON network.
It can be used for a constant light control by regulating of the room brightness.
By means of the IR Remote Control up to 10 network channels of the Multi-Sensor can be remotely controlled.

3. Mounting

The Multi-Sensor is for flush-mounted installations in interior rooms.
In combination with the Surface Mounting Box for Multi-Sensors it can also be mounted to ceilings which have not been suspended.
Usually the Multi-Sensor is fixed to a 60 mm flush-mounted box.
The base plate supplied with the Multi-Sensor is fixed with two screws to a 60 mm flush-mounted box or to the surface-mounted Surface Mounting Box for Multi-Sensors.
The Multi-Sensor is connected via a bus terminal clamp and clipped onto the base plate.
There is only the LON link power required to supply the device.
The network is connected via a 2-pole pluggable bus terminal included in delivery which allows up to 4 pairs of wires to be connected.
The terminal is suited for conductor cross-sections of 0.6 .. 0.8 mm².
The Multi-Sensor propagates its Neuron-ID by pressing the service pin.
The service LED indicates the programming state.
For the right operation of the Multi-Sensor an appropriate application program is needed.

4. Remarks

Installation and assembly of electrical devices may take place only by an electrical specialist. When planning and installing an electrical system the relevant standards, guidelines and regulations of the respective country are to be considered. Beyond that the device specifications are to be kept. For project engineering, assembly and line-up detailed expertise of the LON technology is presupposed.
The function of the device is software dependent. Only application programs may be loaded, which are approved for this device.
The system integrator has to carry ensuring that the loaded application program and the configured parameters in it correspond with the outside wiring of the device. This applies in particular if for different use several application programs for a device are available.

5. Technical Data

Power supply	
Power consumption (max.):	<= 285 mW (1 LPU) at DC 42.8 V network voltage
Network interface	
Transceiver type:	LON Link Power Transceiver (LPT-11)
Sensor Head	
Measuring square:	360º
Scope:	Radius of max. 7 m at a mounting height of 2.50 m
Number of zones:	136 with 544 Switch segments
Brightness sensor:	Direct from app. 10 .. 1,000 lux
Controls	
Service pin:	Propagates the Neuron ID
Indicators	
Service LED (red):	lit: network access error
	flashes: module unconfigured
Connections	
Bus:	2-pole plug-in and branch terminal (Type: WAGO 243)
Housing	
Protection class:	IP20 (EN 60 529/IEC 144)
Mounting:	Screw mounting in a flush-mounted wall socket (60mm) or by a separate mounting socket
Dimensions:	Diameter: app. 100 mm Height from ceiling: app. 50 mm
EU Directive:	Low voltage guideline 2006/95/EWG and EMC guideline 2004/108/EEC

6. Application Description

The application "880551MS11C" allows movement-dependent controlling via the LON network using the LON Multi-Sensor ILA-22 and the corresponding actuators. An infrared remote control can be used to activate lightscenes and switching and dimmer commands for blinds and lamps. The following LonMark profiles are implemented in the application: "Light Sensor (1060)" (1x), "Occupancy Sensor (1060)" (2x), "Occupancy Controller (3071)" (2x), "Switch (3200)" (10x) and "Scene Panel" (1x).

Function

General information

The keys on the remote control may be assigned freely to the objects. This is implemented via the parameters `UCPTbuttonMapFirstEvent[i]` and `UCPTbuttonMapSecondEvent[i]`. At the factory, the individual inputs and their sequence are assigned to the individual objects in toggle mode. "FirstEvent" and "SecondEvent" refer to the alternating switching events in single-surface mode.

Brightness sensor

The brightness sensor provides the room brightness option in the LON network. The output `nvoLSLuxLevel` transmits the current room brightness in the format SNVT_lux. The output can be transmitted cyclically (`SCPTmaxSendTime`). The parameters `SCPTminSendTime` and `SCPTminDeltaLevel` are available for limiting the frequency of telegrams. Any deviation of the measured lux value from the lux value at the reference point can be corrected using the parameters `SCPTfieldCalib` (`nciLSfieldCalib`) and `SCPTgain`.

Presence sensor

The application provides the signal of the presence detector in two channels `nvoOSOccupancy[i]`. The output value in "NOT OCCUPIED" mode can be set individually in `UCPTunoccupiedOccCmd[i]`. The current occupancy status of the monitored area is transmitted as soon as it changes or cyclically in the time set in `SCPTmaxSendTime[i]`. If the sensor detects a movement, the time set in `SCPTdebounce[i]` is started and is restarted with each new movement. OC_OCCUPIED is maintained until this individually set time has passed.

Presence controller

If a movement is detected via `nviOCOccupancy[i]` the value defined in `SCPTprimeVal[i]` is transmitted via the output variable `nvoOCLampValue[i]`, e.g. to a "lamp actuator". In addition, the `nvoOCSetting[i]` can be used, if required, to control another controller, e.g. a "constant light controller". Furthermore, the output `nvoOCScene` can be used to control a scene controller.
The variable `nviOCSecondary[i]` also allows controlling to take place depending on the occupancy state of a neighbouring area. If, once the `SCPTholdTime[i]` has passed, no movements are detected any more and a neighbouring area still transmits an occupancy message, the value defined in `SCPTsecondVal[i]` is transmitted via `nvoOCLampValue[i]`.
It can be used e.g. to dim the lighting in a neighbouring area that is not occupied. In addition, a separate scene can be accessed for this condition. A switch-off delay time can be defined in the configuration variable `SCPTholdTime[i]`. If no movement is detected any more in the monitored area, the value {0,0} is transmitted after this time via `nvoOCLampValue[i]` or the value that is set in `SCPTsecondVal[i]`, provided the neighbouring area is also connected. The controller can be activated or deactivated externally via `nviOCSetting[i]`.

The configuration variable UCPTonOffFilter[i] can be used to set whether the light is to be switched on and off in a movement-controlled manner, only switched on or only switched off. It is thus possible to always switch the lighting automatically or to switch it on manually and switch it off automatically if no presence is detected any more, or vice versa.
After switching it off, every telegram is ignored during the interval UCPTignoreTime[i] at the inputs nviOCOccupancy[i] and nviOCSecondary[i] in order to prevent it from being switched back on again by any change in brightness that might be detected by the sensor as a movement.
Within the time interval SCPTmaxSendTime[i], the current values for nvoOCLampValue[i] and nvoOCSetting[i] are transmitted cyclically.
The presence controller can switch depending on the ambient brightness. The ambient brightness is received at nviOCLuxLevel[i]. If the brightness value is above the upper limit set in UCPTluxHystHigh[i], the output nvoOCLampValue[i] is set to {0,0} and the output nvoOCSetting[i] is set to SET_OFF after an adjustable delay time UCPToffDelay[i]. The presence detection is active below the lower limit UCPTluxHystLow[i]. The presence detection works as follows between the lower and upper limits: If the lighting is switched on and if the brightness value is increased, e.g. due to an increasing level of brightness outside, the light remains switched on and the presence detection continues to operate in the normal manner. If no movement is detected any more, the light is switched off after the time configured in SCPTholdTime[i]. If the brightness value is then still above the lower limit, the light is not switched back on again – even if a movement is detected – until the brightness value falls below the set lower brightness limit.

Configuration of the remote control keys

The configuration options for the individual switch objects (LonMark profile #3200) have been kept highly flexible to give the user the option of meeting all requirements of the project.
The parameter UCPTcmdXXX[i] can be used to freely configure the behaviour of the outputs nvoSWswitch[i] and nvoSWsetting[i]. The network outputs can be used to control switch / dimmer actuators and motor control units. Here are a few examples to explain the configuration:

Single-surface switching mode using key 1

In single-surface switching mode, an ON telegram and an OFF telegram are transmitted alternately on pressing a key. A two-way circuit is implemented using the feedback input nviSWswitchFb[0].

UCPTbuttonMapFirstEvent[0] = .bit0 = 1
UCPTbuttonMapSecondEvent[0] = .bit0 = 1
(The first and second time key pressing actions are distinguished and assigned to the first switch object ([0]))
UCPTcmdPushFirstEvent[0] = SET_ON, ?, ?
UCPTcmdPushSecondEvent[0] = SET_OFF, ?, ?
All others UCPTcmdXXX[0] = SET_NO_MESSAGE, ?, ?

Dual-surface switching mode using keys 1+2

In dual-surface switching mode, an ON telegram is transmitted using one key and an OFF telegram is transmitted using another key.

`UCPTbuttonMapFirstEvent[0]` = `.bit0` = 1
`UCPTbuttonMapSecondEvent[0]` = `.bit1` = 1
(The first switching event for the first switch object ([0]) is assigned to the first key and the second switching event for the second switch object ([0]) is assigned to the second key)
`UCPTcmdPushFirstEvent[0]` = SET_ON, ?, ?
`UCPTcmdPushSecondEvent[0]` = SET_OFF, ?, ?
All others `UCPTcmdXXX[0]` = SET_NO_MESSAGE, ?, ?

Single-surface switching/dimmer mode using keys 1+2

In single-surface switching/dimmer mode, an ON and an OFF telegram are transmitted alternately each time a key is released after being pressed briefly. If the key is pressed for a longer time, dimming UP and DOWN takes place alternately. A two-way circuit is implemented using the feedback input `nviSWswitchFb[0]`.

`UCPTbuttonMapFirstEvent[0]` = `.bit0` = 1
`UCPTbuttonMapSecondEvent[0]` = `.bit0` = 1
(The first and second switching events for the first switch object ([0]) are assigned to the first key)
`UCPTcmdReleaseFirstEvent[0]` = SET_ON, ?, ?
`UCPTcmdReleaseSecondEvent[0]` = SET_OFF, ?, ?
`UCPTcmdHoldFirstEvent[0]` = SET_UP, 5%, ?
`UCPTcmdHoldSecondEvent[0]` = SET_DOWN, 5%, ?
All others `UCPTcmdXXX[0]` = SET_NO_MESSAGE, ?, ?

Dual-surface switching/dimmer mode using keys 1+2

In dual-surface switching/dimmer mode, an ON telegram is transmitted each time one key is released after being pressed briefly and an OFF telegram is transmitted each time another key is released after being pressed briefly. Pressing a key for a longer time dims up or down.

`UCPTbuttonMapFirstEvent[0]` = `.bit0` = 1
`UCPTbuttonMapSecondEvent[0]` = `.bit1` = 1
(The first switching event for the first switch object ([0]) is assigned to the first key and the second switching event for the second switch object ([0]) is assigned to the second key)
`UCPTcmdReleaseFirstEvent[0]` = SET_ON, ?, ?
`UCPTcmdReleaseSecondEvent[0]` = SET_OFF, ?, ?
`UCPTcmdHoldFirstEvent[0]` = SET_UP, 5%, ?
`UCPTcmdHoldSecondEvent[0]` = SET_DOWN, 5%, ?
All others `UCPTcmdXXX[0]` = SET_NO_MESSAGE, ?, ?

Dual-surface blinds mode using keys 1+2

In dual-surface blinds mode, an UP telegram is transmitted using one key and a DOWN telegram is transmitted using another key. Releasing the key after pressing it briefly stops the blind.

`UCPTbuttonMapFirstEvent[0]` = `.bit0` = 1
`UCPTbuttonMapSecondEvent[0]` = `.bit1` = 1
(The first switching event for the first switch object ([0]) is assigned to the first key and the second switching event for the second switch object ([0]) is assigned to the second key)
`UCPTcmdPushFirstEvent[0]` = SET_UP, 100%, 360°
`UCPTcmdPushSecondEvent[0]` = SET_DOWN, 100%, 360°
`UCPTcmdReleaseFirstEvent[0]` = SET_STOP, ?, ?
`UCPTcmdReleaseSecondEvent[0]` = SET_STOP, ?, ?
All others `UCPTcmdXXX[0]` = SET_NO_MESSAGE, ?, ?

Scene panel

All digital inputs are automatically also implemented in the scene panel. In the default setting, the corresponding scene number is transmitted when a key is released.
The telegrams at `nvoSPscene` can be set individually for each input for the different input events via `UCPTbuttonRslSceneCmd[i]`, `UCPTbuttonFslSceneCmd[i]` and `UCPTsceneLearnDelay`.

Configuration for push-buttons with learning function

`UCPTbuttonFslSceneCmd[i]` contains the scene to be transmitted on releasing the key (falling edge). The values `UCPTbuttonRslSceneCmd[i]` belonging to the pressing of the key (rising edge) are deactivated with `.function` = SC_NUL. If the key or the input is activated for longer than the time configured in `UCPTsceneLearnDelay`, then `nvoSPscene` = {SC_LEARN, `UCPTbuttonFslSceneCmd[i].scene_number`} (learn command) is transmitted.

Configuration for push-buttons without learning function

`UCPTsceneLearnDelay` is configured to 0. Valid values are only set in one of the parameters `UCPTbuttonFslSceneCmd[i]` or `UCPTbuttonRslSceneCmd[i]`. The corresponding values are deactivated with SC_NUL (`.function`).

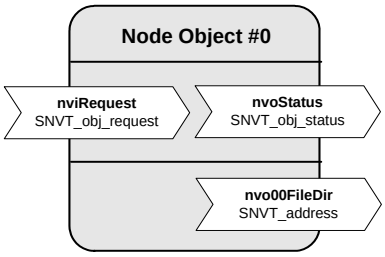
6.1 System requirements

An LNS-compatible LON management tool is required for the configuration of the application!
User-defined configuration property types (UCPTs) are used as parameters in the DirectMemoryAccess. To be able to use the parameters, the device resource files (DRFs) need to be installed **before (!)** creating a device template.

The used LNS must be Version 2.0 or higher.

Network interface / LonMark objects

6.2 Node object (LONMARK® profile #0)



Input variables

`nviRequest`

Type:	SNVT_obj_request
Value range:	Valid object ID in combination with RQ_NORMAL, RQ_UPDATE_STATUS, RQ_REPORT_MASK
Default value:	RQ_NORMAL
Description:	Input that is used to initiate the status messages of the node.

Output variables

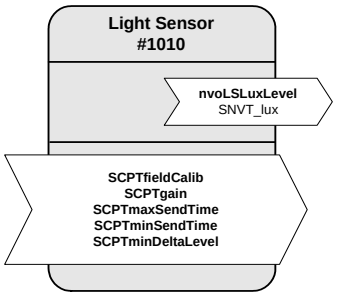
`nvoStatus`

Type:	SNVT_obj_status
Value range:	The status bits supported by the object: nvoStatus.report_mask, nvoStatus.invalid_id, nvoStatus.invalid_request
Default value:	All bits are 0
Description:	Is transmitted when an update is received from `nviRequest`.

`nvo00FileDir`

Type:	SNVT_address
Value range:	16,384 ... 64,767
Default value:	Undefined
Description:	Is required for internal functionality !

6.3 Light sensor (LONMARK® profile #1010)



Output variables

nvoLSLuxLevel

Type:	SNVT_lux
Value range:	0 ... 65,535 lux
Default value:	0
Description:	Outputs the measured brightness in the form of a lux value.

Configuration variables

SCPTfieldCalib (nciLSfieldCalib)

Type:	SNVT_lux
Value range:	0 ... 65,535 lux
Default value:	0
Description:	To compensate deviations between the brightness values at the sensor and at the reference point, a conversion factor SCPTgain needs to be determined. To do this, the lux value measured at the reference point is entered in SCPTfieldCalib (nciLSfieldCalib) , which updates SCPTgain: SCPTgain.multiplier = SCPTfieldCalib (nciLSfieldCalib) and SCPTgain.divisor = active lux sensor value at this time.

SCPTgain

Type:	SNVT_muldiv
Value range:	.multiplier: 0 ... 65,535 .divisor: 0 ... 6.5535
Default value:	.multiplier = 1 .divisor =1
Description:	Memory of the conversion factor of the calibration. A known conversion factor can be entered manually.

SCPTmaxSendTime

Type:	SNVT_time_sec
Value range:	0 ... 6,553.5 s
Default value:	120 s
Description:	The current lux value is transmitted regularly within this interval.

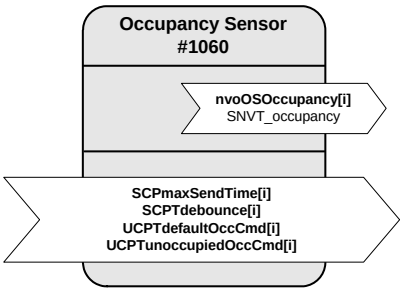
SCPTminSendTime

Type:	SNVT_time_sec
Value range:	0 ... 6,553 s
Default value:	2 s
Description:	Minimum interval between two telegrams.

SCPTminDeltaLevel

Type:	SNVT_lev_cont
Value range:	0 ... 100%
Default value:	2.5%
Description:	The minimum change required for an update of the output variables.

6.4 Occupancy Sensor (LONMARK® profile #1060)



Output variables

nvoOSOccupancy[i]

Type:	SNVT_occupancy
Value range:	OC_OCCUPIED, OC_UNOCCUPIED, OC_BYPASS, OC_STANDBY, OC_NUL
Default value:	see `UCPTdefaultOccCmd[i]`
Description:	Output of the sensor value. If a movement is detected, OC_OCCUPIED is output here. The output value for the "NOT OCCUPIED" state can be defined via the parameter `UCPTunoccupiedOccCmd[i]`.

Configuration parameters

SCPTmaxSendTime[i]

Type:	SNVT_time_sec
Value range:	0 ... 6,553 s
Default value:	120 s
Description:	The maximum period of time between consecutive transmissions of the current value.

SCPTdebounce[i]

Type:	SNVT_time_sec
Value range:	10 ... 6,553 s
Default value:	15 s
Description:	Debounce time that is restarted with each "OCCUPIED" signal. During this time, `nvoOSOccupancy[i]` is maintained as OC_OCCUPIED.

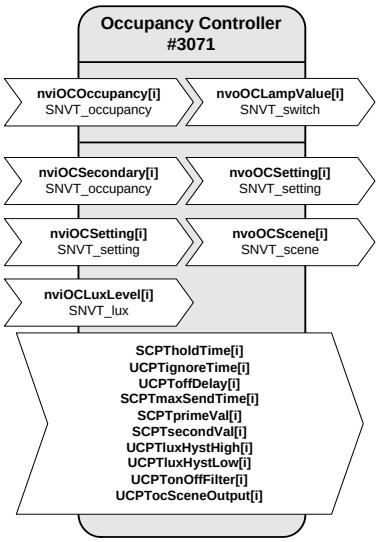
UCPTdefaultOccCmd[i]

Type:	SNVT_occupancy
Value range:	OC_OCCUPIED, OC_UNOCCUPIED, OC_BYPASS, OC_STANDBY, OC_NUL
Default value:	OC_NUL
Description:	The value configured here is issued during the initialisation phase of the sensor head (approx. 2 s).

UCPTunoccupiedOccCmd[i]

Type:	SNVT_occupancy
Value range:	OC_UNOCCUPIED, OC_BYPASS, OC_STANDBY, OC_NUL
Default value:	OC_UNOCCUPIED
Description:	The value set here is transmitted on switching to the "NOT OCCUPIED" status. That means that the presence detector can be adapted to different application cases, e.g. lighting control or heating control. Normally: lighting control = OC_UNOCCUPIED, heating = OC_STANDBY

6.5 Occupancy Controller (LONMARK® profile #3071)



Input variables

nviOCOccupancy[i]

Type:	SNVT_occupancy
Value range:	OC_OCCUPIED, OC_UNOCCUPIED
Default value:	OC_NUL
Description:	Input for the occupied message of the movement sensor

nviOCSecondary[i]

Type:	SNVT_occupancy
Value range:	OC_OCCUPIED, OC_UNOCCUPIED
Default value:	OC_NUL
Description:	Input for the occupied message of a neighbouring area

nviOCSetting[i]

Type:	SNVT_setting
Value range:	.function: SET_OFF, SET_ON, SET_DOWN, SET_UP, SET_STOP
Default value:	.function = SET_ON
Description:	Input for switching the presence controller on and off

nviOCLuxLevel[i]

Type:	SNVT_lux
Value range:	0 ... 65,534 lux
Default value:	0
Description:	Input for the application of a lux value.

Output variables

nvoOCLampValue[i]

Type:	SNVT_switch
Value range:	value: 0 ... 100%; state: 0.1, -1
Default value:	.value = 0 .state = 0
Description:	Control command for the actuator to be switched. If the status of nviOCOccupancy[i] is "OCCUPIED", then the value in SCPTprimeVal[i] is transmitted; if the status of nviOCOccupancy[i] is "NOT OCCUPIED" and the status of nviOCSecondary[i] is "OCCUPIED", the value in SCPTsecondVal[i] is transmitted. If both areas are not occupied, {0,0} is transmitted.

nvoOCSetting[i]

Type:	SNVT_setting
Value range:	.function: SET_ON, SET_OFF
Default value:	.function = SET_OFF
Description:	Control output for a controller, e.g. "Constant Light Controller". The SET_ON command is transmitted once if nviOCOccupancy[i] or nviOCSecondary[i] switch to the "OCCUPIED" state and the value was previously SET_OFF.

nvoOCScene[i]

Type:	SNVT_scene
Value range:	.function: SC_ RECALL .scene_number: 0 ... 255
Default value:	.function = SC_RECALL .scene_number = 0
Description:	Control output for a scene controller; the transmitted scene number is set in the parameter UCPTocSceneOutput[i].

Configuration variables

SCPTholdTime[i]

Type:	SNVT_time_sec
Value range:	1 ... 6,553 s
Default value:	900 s
Description:	If the monitored area is given the status OC_UNOCCUPIED, the switch-off delay nvoOCLampValue[i] set here is either set to {0,0} or, if nviOCSecondary[i] has the status OC_OCCUPIED, to the value set in SCPTsecondVal[i]. The delay time is restarted with each change to OC_UNOCCUPIED.

UCPTignoreTime[i]

Type:	SNVT_time_sec
Value range:	0 ... 6,553 s
Default value:	0
Description:	During this time, each telegram at nviOCOccupancy[i] or nviOCSecondary[i] is ignored after switching off the light, since the change of brightness might incorrectly be reported as a movement by a sensor.

UCPToffDelay[i]

Type:	UNVT_xxxx_time_sec
Value range:	0 ... 65,534 s
Default value:	300 s
Description:	Switch-off delay if the upper brightness limit value is exceeded.

SCPTmaxSendTime[i]

Type:	SNVT_time_sec
Value range:	0 ... 6,553 s
Default value:	0
Description:	The output variables are transmitted cyclically within this interval. The function is deactivated for 0.

SCPTprimeVal[i]

Type:	SNVT_switch
Value range:	value: 0 ... 100%; state: 0,1
Default value:	.value =100% .state = 1
Description:	This value is output via `nvoOCLampValue[i]` if the monitored area switches to the status OC_OCCUPIED.

SCPTsecondVal[i]

Type:	SNVT_switch
Value range:	value: 0 ... 100%; state: 0.1
Default value:	.value = 50% .state = 1
Description:	This value is output via `nvoOCLampValue[i]` if `nviOCSecondary[i]` switches to the status OC_OCCUPIED. Also see the description of the configuration variable `SCPTholdTime[i]`.

UCPTluxHystHigh[i]

Type:	SNVT_lux
Value range:	0 ... 65,534 lux
Default value:	700 lux
Description:	The presence detection can be made dependent on the ambient brightness that is received via `nviOCLuxLevel[i]`. If the brightness value is above the upper limit set here, after an adjustable delay time `UCPToffDelay[i]`, the output `nvoOCLampValue[i]` is reset to {0,0} and the output `nvoOCSetting[i]` is reset to SET_OFF.

UCPTluxHystLow[i]

Type:	SNVT_lux
Value range:	0 ... 65,534 lux
Default value:	500 lux
Description:	If the brightness value received at nviOCLuxLevel[i] is below this value, the presence detection is active. If any movement is detected, the outputs nvoOCLampValue[i] at SCPTprimeVal[i] / SCPTsecondVal[i] and nvoOCSetting[i] are set to SET_ON. If the brightness value is between UCPTluxHystLow[i] and UCPTluxHystHigh[i], no switching process is generated if a movement is detected again. Once the time configured in SCPTholdTime[i] has passed and if no more movements are detected, switching off takes place.

UCPTonOffFilter[i]

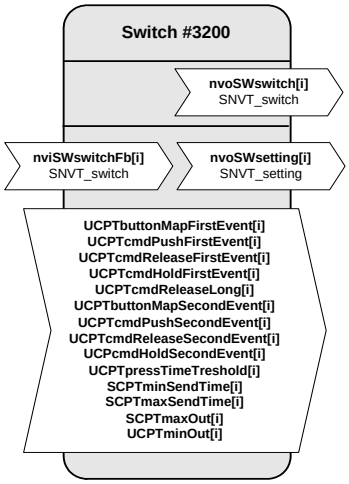
Type:	UNVT_on_off_filter
Value range:	FL_NO FILTER, FL_NO_ON_CMD, FL_NO_OFF_CMD
Default value:	FL_NO FILTER
Description:	Activation/deactivation of a switch-on/switch-off lock: FL_NO FILTER: Switch-on and switch-off telegrams are transmitted. FL_NO_ON_CMD: Switch-on telegrams are not transmitted. FL_NO_OFF_CMD: Switch-off telegrams are not transmitted.

UCPTocSceneOutput[i]

Type:	UNVT_oc_scene
Value range:	.oc_off: 0 ... 255 .oc_secondary: 0 ... 255 .oc_primary: 0 ... 255
Default value:	.oc_off = 1 .oc_secondary = 2 .oc_primary = 3
Description:	Scene number that is issued, depending on the state of the controller at nvoOCScene.

6.6 Switch (LONMARK® profile #3200)

Key 1 (index=0) to key 10 (index=9)



Input variables

`nviSWswitchFb[i]`

Type:	SNVT_switch
Value range:	.value: 0 ... 100% .state: 0, 1 ON: .state = 1 and .value > 0 OFF: .state = 0 or .state = 1 and .value = 0
Default value:	.value = 0 .state = 0
Description:	Feedback value for the implementation of two-way circuits in TOGGLE mode or to be able to apply the current `.value` of the actuator for dimming.

Output variables

`nvoSWswitch[i]`

Type:	SNVT_switch
Value range:	.value: 0 ... 100% .state: 0, 1, -1 ON: .state = 1 and .value > 0 OFF: .state = 0 or .state = 1 and .value = 0
Default value:	.value = 0 .state = 0
Description:	Network variable for controlling switching and dimmer actuators. The functionality is the result of the configuration of the parameters `UCPTbuttonMapXXX` and `UCPTcmdXXX` in the corresponding switch object.

nvoSWsetting[i]

Type:	SNVT_setting
Value range:	.function: SET_OFF, SET_ON, SET_DOWN, SET_UP, SET_STOP, SET_STATE, SET_NUL .setting: 0 ... 100% .rotation: -359.98° ... 360°
Default value:	.function = SET_NUL .setting = 0 .rotation = 0
Description:	Network variable for controlling controller, blind and dimmer actuators. The functionality is the result of the configuration of the parameters UCPTbuttonMapXXX and UCPTcmdXXX in the corresponding switch object.

Configuration variables

The assignment of the keys (1 ... 10) to the objects or switching events is implemented using the parameters UCPTbuttonMapFirstEvent[i] and UCPTbuttonMapSecondEvent[i].

The first time a key is pressed (FirstEvent) usually refers to the switch-on command or a movement command (for blind control). In TOGGLE mode, pressing a key (SecondEvent) usually triggers a switch-off command or a movement command in the opposite direction. In two-key operation however, the first switching event is linked to a certain key (always the same telegram is generated) and the second switching event is linked to another key. See below for a detailed description.

UCPTbuttonMapFirstEvent[i] - Key assignment, first event

Type:	SNVT_state
Value range:	.bit0 ... .bit9 = 0/1
Default value:	.bit[j] = 1; .bit[j<>i] = 0
Description:	Assignment of the keys (1 ... 10) to the software objects. .bit0 ... .bit9 are equivalent to keys 1 ... 10 and the index of the variable is equivalent to the object. The factory settings assume a TOGGLE functionality, i.e.: UCPTbuttonMapFirstEvent[0] with .bit0 = 1, rest = 0 UCPTbuttonMapFirstEvent[1] with .bit1 = 1, rest = 0 UCPTbuttonMapFirstEvent[2] with .bit2 = 1, rest = 0 UCPTbuttonMapFirstEvent[3] with .bit3 = 1, rest = 0 ... Two-key operation requires the following reconfiguration: UCPTbuttonMapFirstEvent[0] with .bit0 = 1, rest = 0 UCPTbuttonMapFirstEvent[1] all = 0 UCPTbuttonMapFirstEvent[2] with .bit2 = 1, rest = 0 UCPTbuttonMapFirstEvent[3] all = 0 ... Special assignments may be configured freely.

`UCPTbuttonMapSecondEvent[i]` - Key assignment, second event

Type:	SNVT_state
Value range:	.bit0 ... .bit9 = 0/1
Default value:	.bit[j] = 1; .bit[j<>i] = 0
Description:	Assignment of the keys (1 ... 10) to the software objects. `.bit0` ... `.bit9` are equivalent to keys 1 ... 10 and the index of the variable is equivalent to the object.

The factory settings assume a TOGGLE functionality, i.e.:

`UCPTbuttonMapSecondEvent[0]` with `.bit0` = 1, rest = 0
`UCPTbuttonMapSecondEvent[1]` with `.bit1` = 1, rest = 0
`UCPTbuttonMapSecondEvent[2]` with `.bit2` = 1, rest = 0
`UCPTbuttonMapSecondEvent[3]` with `.bit3` = 1, rest = 0
...

Two-key operation requires the following reconfiguration:

`UCPTbuttonMapSecondEvent[0]` with `.bit1` = 1, rest = 0
`UCPTbuttonMapSecondEvent[1]` all = 0
`UCPTbuttonMapSecondEvent[2]` with `.bit3` = 1, rest = 0
`UCPTbuttonMapSecondEvent[3]` all = 0
...

Special assignments may be configured freely.

`UCPTcmdXXX[i]` - General

Type:	UNVT_setting
Value range:	.function: SET_OFF, SET_ON, SET_DOWN, SET_UP, SET_STOP, SET_STATE, SET_NO_MESSAGE, SET_NUL .setting: 0 ... 100% .rotation: -359.98° ... 360°
Description:	`UCPTcmdXXX[i].function` is interpreted as follows for `nvoSWswitch[i].state`:

`.function` = SET_OFF	results in `.state` = 0
`.function` = SET_ON, SET_UP, SET_DOWN, SET_STATE	results in `.state` = 1
`.function` = SET_NUL	results in `.state` = -1

If an event is not to generate a telegram, then you must set `.function` = SET_NO_MESSAGE.

`UCPTcmdXXX[i].function` is interpreted as follows for the `nvoSWswitch[i].value`:

`.function` = SET_OFF	results in `.value` = 0
`.function` = SET_NUL, SET_STATE	results in `.value` = `.setting`
`.function` = SET_ON	results in `.value` = `SCPTmaxOut[i]` after a restart or the last value prior to switching off (memory)
`.function` = SET_UP, SET_DOWN	results in `.value` = `.value` +/- `.setting`

`UCPTcmdPushFirstEvent[i]` - First pressing of a key

Type:	UNVT_setting
Value range:	.function: SET_OFF, SET_ON, SET_DOWN, SET_UP, SET_STOP, SET_STATE, SET_NO_MESSAGE, SET_NUL .setting: 0 ... 100% .rotation: -359.98° ... 360°
Default value:	.function = SET_ON .setting = 100% .rotation = 0
Description:	This value is issued at `nvoSWsetting[i]` the first time a key is pressed. See above for the interpretation of `nvoSWswitch[i]` and for general information.

`UCPTcmdReleaseFirstEvent[i]` - First release

Type:	UNVT_setting
Value range:	.function: SET_OFF, SET_ON, SET_DOWN, SET_UP, SET_STOP, SET_STATE, SET_NO_MESSAGE, SET_NUL .setting: 0 ... 100% .rotation: -359.98° ... 360°
Default value:	.function = SET_NO_MESSAGE .setting = 0 .rotation = 0
Description:	This value is issued at `nvoSWsetting[i]` the first time the key is released after the first brief pressing of the key (before the `UCPTpressTimeThreshold[i]` time passes). See above for the interpretation of `nvoSWswitch[i]` and for general information.

`UCPTcmdHoldFirstEvent[i]` - First holding

Type:	UNVT_setting
Value range:	.function: SET_OFF, SET_ON, SET_DOWN, SET_UP, SET_STOP, SET_STATE, SET_NO_MESSAGE, SET_NUL .setting: 0 ... 100% .rotation: -359.98° ... 360°
Default value:	.function = SET_NO_MESSAGE .setting = 0 .rotation = 0
Description:	This value is issued after detecting the longer pressing of the key for the first time (once the `UCPTpressTimeThreshold[i]` time has passed) on `nvoSWsetting[i]`. See above for the interpretation of `nvoSWswitch[i]` and for general information. If `SCPTminSendTime[i]` > 0, the value configured at `nvoSWsetting[i]` is transmitted cyclically. The `.setting` proportion is used as dimming increment at `nvoSWswitch[i]`.

`UCPTcmdReleaseLong[i]` - Release after holding

Type:	UNVT_setting
Value range:	.function: SET_OFF, SET_ON, SET_DOWN, SET_UP, SET_STOP, SET_STATE, SET_NO_MESSAGE, SET_NUL .setting: 0 ... 100% .rotation: -359.98° ... 360°
Default value:	.function = SET_NO_MESSAGE .setting = 0 .rotation = 0
Description:	This value is issued after releasing the key after pressing the key for a long time at `nvoSWsetting[i]`. See above for the interpretation of `nvoSWswitch[i]` and for general information (applies to both events).

`UCPTcmdPushSecondEvent[i]` - Second pressing of a key

Type:	UNVT_setting
Value range:	.function: SET_OFF, SET_ON, SET_DOWN, SET_UP, SET_STOP, SET_STATE, SET_NO_MESSAGE, SET_NUL .setting: 0 ... 100% .rotation: -359.98° ... 360°
Default value:	.function = SET_OFF .setting = 0 .rotation = 0
Description:	This value is issued at `nvoSWsetting[i]` the second time a key is pressed. See above for the interpretation of `nvoSWswitch[i]` and for general information.

`UCPTcmdReleaseSecondEvent[i]` - Second release

Type:	UNVT_setting
Value range:	.function: SET_OFF, SET_ON, SET_DOWN, SET_UP, SET_STOP, SET_STATE, SET_NO_MESSAGE, SET_NUL .setting: 0 ... 100% .rotation: -359.98° ... 360°
Default value:	.function = SET_NO_MESSAGE .setting = 0 .rotation = 0
Description:	This value is issued at `nvoSWsetting[i]` on releasing the key after the second brief pressing of the key (before the `UCPTpressTimeThreshold[i]` time expires). See above for the interpretation of `nvoSWswitch[i]` and for general information.

`UCPTcmdHoldSecondEvent[i]` - Second holding

Type:	UNVT_setting
Value range:	.function: SET_OFF, SET_ON, SET_DOWN, SET_UP, SET_STOP, SET_STATE, SET_NO_MESSAGE, SET_NUL .setting: 0 ... 100% .rotation: -359.98° ... 360°
Default value:	.function = SET_NO_MESSAGE .setting = 0 .rotation = 0
Description:	This value is issued after the detection of the second longer pressing of the key at `nvoSWsetting[i]` (once the `UCPTpressTimeThreshold[i]` time has passed); see above for the interpretation of `nvoSWswitch[i]`. If `SCPTminSendTime[i]` > 0, the value configured cyclically is transmitted at `nvoSWsetting[i]`. The `.setting` proportion is used as dimming increment at `nvoSWswitch[i]`.

`UCPTpressTimeThreshold[i]` - Time threshold for the longer pressing of a key

Type:	SNVT_time_sec
Value range:	0.0 ... 64.0 s
Default value:	1.0 s
Description:	Difference in time between the brief and longer pressing of a key. If this time is 0, `UCPTcmdReleaseLong[i]` is always transmitted after the opening of the contact.

`SCPTminSendTime[i]` - Minimum transmission interval

Type:	SNVT_time_sec
Value range:	0.0 ... 64.0 s
Default value:	0.2 s
Description:	Minimum interval between two telegrams. If this time is > 0, then `UCPTcmdHoldFirstEvent[i]` / `UCPTcmdHoldSecondEvent[i]` is transmitted with this interval once the `UCPTpressTimeThreshold[i]` time has passed.

`SCPTmaxSendTime[i]` - Maximum transmission pause

Type:	SNVT_time_sec
Value range:	0 ... 6.553 s
Default value:	0 (deactivated)
Description:	Maximum time between two transmission cycles. If the time is > 0, the current output values `nvoSWswitch[i]` and `nvoSWsetting[i]` are transmitted in cycles.

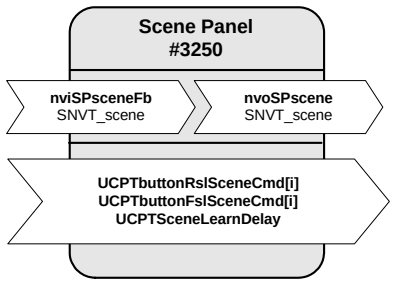
`SCPTmaxOut[i]` - Maximum output value

Type:	SNVT_lev_cont
Value range:	0 ... 100%
Default value:	100%
Description:	Output value for `nvoSWswitch[i].value` on switching on and maximum value for dimming via `nvoSWswitch[i]` and `nvoSWsetting[i]`.

`UCPTminOut[i]` - Minimum output value

Type:	SNVT_lev_cont
Value range:	0 ... 100%
Default value:	5%
Description:	Minimum value for dimming via `nvoSWswitch[i]` and `nvoSWsetting[i]`.

6.7 Scene panel (LONMARK® profile #3250)



Input variables

nviSPsceneFb

Type:	SNVT_scene	
Value range:	.function:	SC_RECALL, SC_LEARN, SC_NUL
	.scene_number:	1 ... 255
Default value:	.function:	SC_NUL
	.scene_number:	255
Description:	No function at present	

Output variables

nvoSPscene

Type:	SNVT_scene	
Value range:	.function:	SC_RECALL, SC_LEARN, SC_NUL
	.scene_number:	1 ... 255
Default value:	.function:	SC_NUL
	.scene_number:	255
Description:	For controlling a scene controller. Pressing the [i+1] key triggers the call `nvoSPscene.function` = SC_RECALL with `nvoSPscene.scene_number` = `UCPTbuttonRslSceneCmd[i]` as scene number. On releasing the key, the scene configured in `UCPTbuttonFslSceneCmd[i]` is accessed. After exceeding the time threshold `UCPTsceneLearnDelay` with the key pressed, `nvoSPscene.function` = SC_LEARN is transmitted with the scene number configured in `UCPTbuttonFslSceneCmd[i]`.	

Configuration variables

`UCPTbuttonRslSceneCmd[i]` - Scene on pressing a key

Type:	SNVT_scene	
Value range:	.function:	SC_RECALL, SC_LEARN, SC_NUL
	.scene_number:	1 ... 255
Default value:	.function:	SC_NUL
	.scene_number:	255
Description:	The [i] key is assigned to a scene number here that is accessed on pressing the key briefly (< `UCPTsceneLearnDelay`).	

`UCPTbuttonFslSceneCmd[i]` - Scene on release

Type:	SNVT_scene	
Value range:	.function:	SC_RECALL, SC_LEARN, SC_NUL
	.scene_number:	1 ... 255
Default value:	.function:	SC_RECALL
	.scene_number:	i+1
Description:	The [i] key is assigned to a scene number here that is accessed on releasing a key. If the key is pressed for a longer time (longer than configured in `UCPTsceneLearnDelay`) this scene number is transmitted along with a learn command.	

`UCPTsceneLearnDelay` - Time threshold scene learning

Type:	SNVT_time_sec
Value range:	0.0 ... 64.0 s
Default value:	10 s
Description:	Difference in time for SC_RECALL or SC_LEARN: After exceeding the time threshold with the key pressed, `nvoSPscene.function` = SC_LEARN is transmitted. The value 0 deactivates the learning function (SC_LEARN).