# PZM4146 Software Users Guide
# Version 1
# Draft 12 Feb. 2004

# 1 Software functions.

The following section is describing the software function profiles implemented in the Lonbox Comfort and Light Controller model PZM4146.

The PZM4146 is designed to fulfil the requirement for operating and controlling lighting, heating, cooling, ventilation and sunblind or windows in modern Intelligent Building Installations, using the LonWorks network standard ANSI/EIA/CEN 709.1 or as a stand-alone unit.

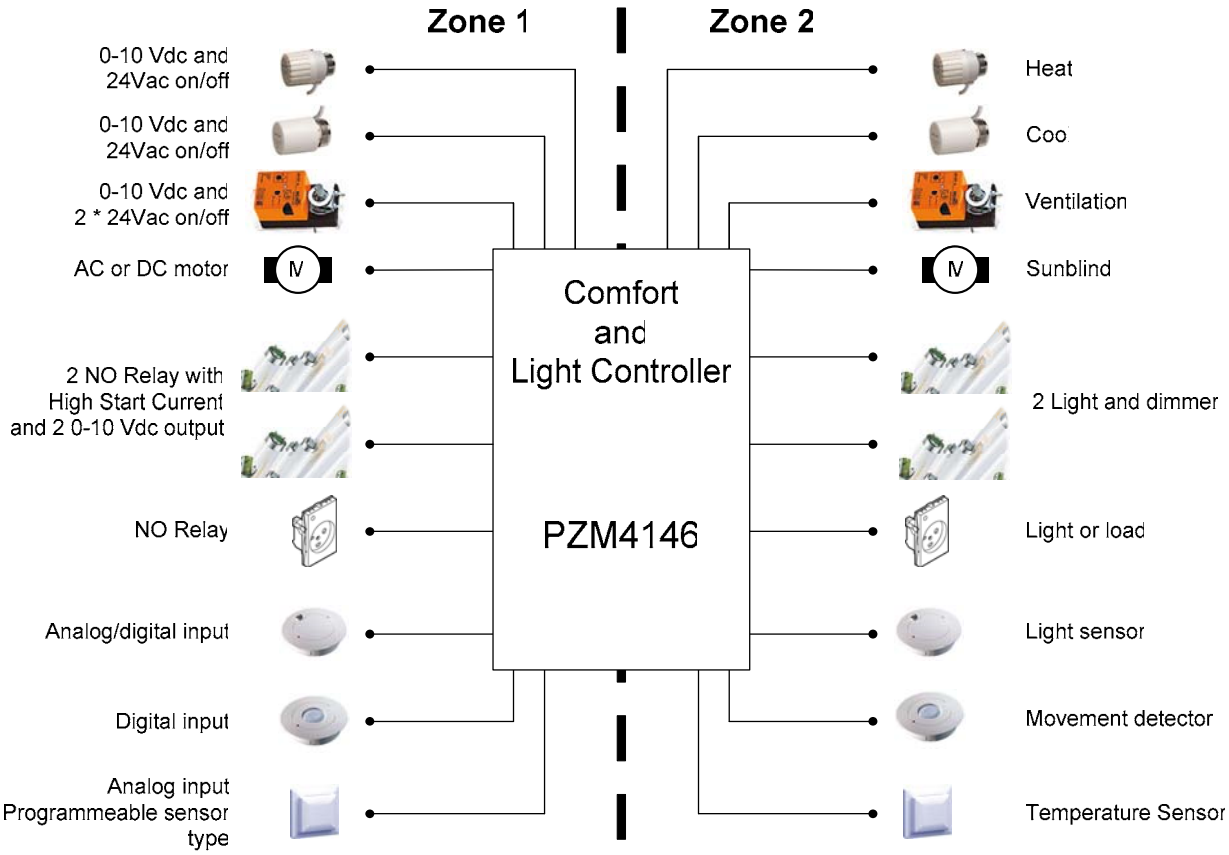



**Figure 1 The main application for PZM4146**

## *1.1 General purpose function profiles:*

The following is describing the general purpose software functions implemented in the Lonbox Comfort and Light Controller model PZM4146.

### 1.1.1 Scene Controller (3251), 6 objects

The scene controller object is used to control scenes. Input from an event triggering source e.g. a scene panel or Building Management System (BMS) different scene numbers will be triggered. Scene values are stored in the controller memory. Scene setup is done by sending values or storing current values under a given scene number. Current values are obtained internally. Stored values are reported using scene configuration feedback output.

A scene panel output is normally connected to the scene controller input and each lamp or groups of lamps have their own Scene controller.

| Controller # | Name in LonMaker list |
|---|---|
| 1 | SceneControl_1 |
| 2 | SceneControl_2 |
| 3 | SceneControl_3 |
| 4 | SceneControl_4 |
| 5 | SceneControl_5 |
| 6 | SceneControl_6 |

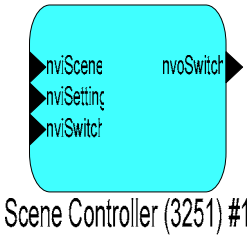



**Figure 2 The scene controller object**

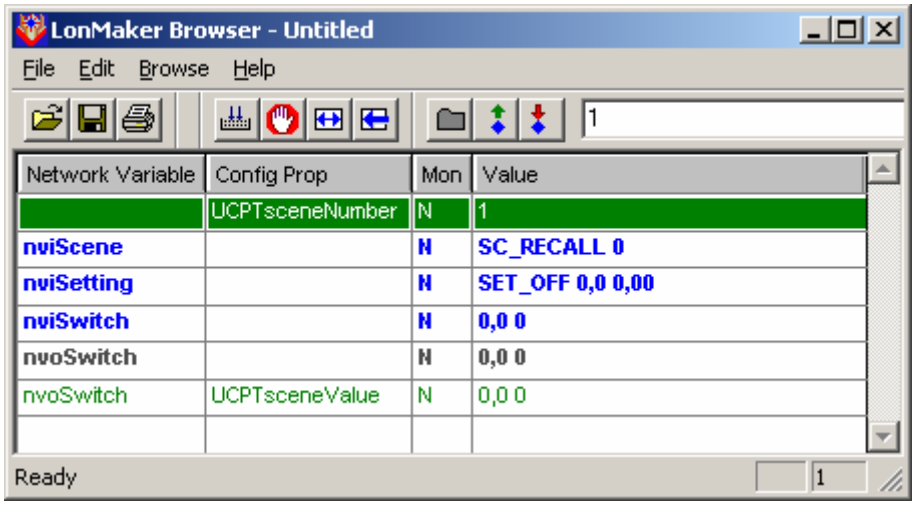



**Figure 3. Browsing the scene controller object**

### 1.1.1.1 Network Variable Interface

| Variable name | Default Value | Type | Description |
|---|---|---|---|
| nviScene | SC_RECALL 0 | SNVT_scene (115) | Scene trigger input |
| nviSetting | SET_OFF 0,0 0,00 | SNVT_setting (117) | Setting control function |
| nviSwitch | 0,0 0 | SNVT_switch (95) | Direct control input |
| | | | |
| nvoSwitch | 0,0 0 | SNVT_switch (95) | Control output |

#### *1.1.1.1.1 nviScene*

This input network variable triggers a scene or loads the scene preset memory with current values. If the recalled scene number is not found in the preset memory, the controller takes no action.

##### 1.1.1.1.1.1 Valid Range

The valid enumeration is RECALL with associated scene number (1 to 255). Scene number 0 is not used.

##### 1.1.1.1.1.2 Configuration Considerations

This network variable is subject to the receive heartbeat time, nciRcvHrtBt.

#### *1.1.1.1.2 nviSetting*

This input network variable selects the operating mode of the controller. The operating modes are: ON or OFF

The ON mode turns on the controller which then starts to control the lamp value output triggered by the incoming scene trigger nviScene. The OFF mode turns off the controller and the lamp value output.

#### *1.1.1.1.3 nviSwitch*

This optional input network variable provides a direct control method for the output (nvoSwitch). This input overrides other inputs or ongoing fades. It can be used for scene adjusting when “learn current” function is used.

The valid range of the state is as defined for nviSwitch where 0 means OFF and 1 means ON. The 8-bit intensity value contains a value from 0 to 200, representing minimum to maximum (0%-100% in steps of 0.5%) intensity. A state value of 0xFF indicates the switch value is undefined

#### *1.1.1.1.4 nvoSwitch*

This output network variable provides the switch output for an actuator triggered by the scene.

##### 1.1.1.1.4.1 Valid Range

The valid range of the state is as defined for nviSwitch where 0 means OFF and 1 means ON. The 8-bit intensity value contains a value from 0 to 200, representing minimum to maximum (0%-100% in steps of 0.5%) intensity. A state value of 0xFF indicates the switch value is undefined

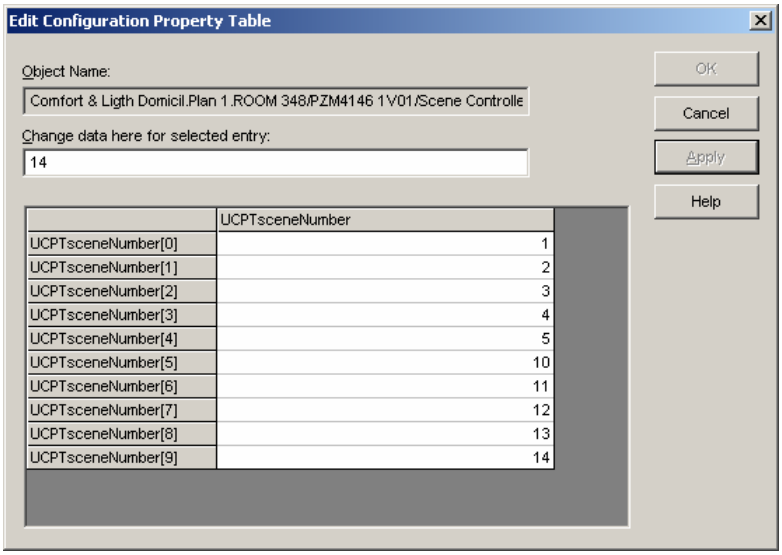
### 1.1.1.2 Configuration Properties

| Configuration | Default Value | Description |
|---|---|---|
| UCPTsceneNumber ( ) | 1 | |
| UCPTsceneValue ( ) | 0,00 | |

#### *1.1.1.2.1 UCPTsceneNumber*

The UCPTsceneNumber is a configuration table property. The table is used to configure which input scene number, there will use a specific index number in the scene controller. Related to the fixed index number from UCPTsceneNumber(0) to UCPTsceneNumber(9) you can configure which scene that shall trigger this index number.

I the below windows list, you can see that input scene 1 uses index (0) and that scene 12 uses index (7).

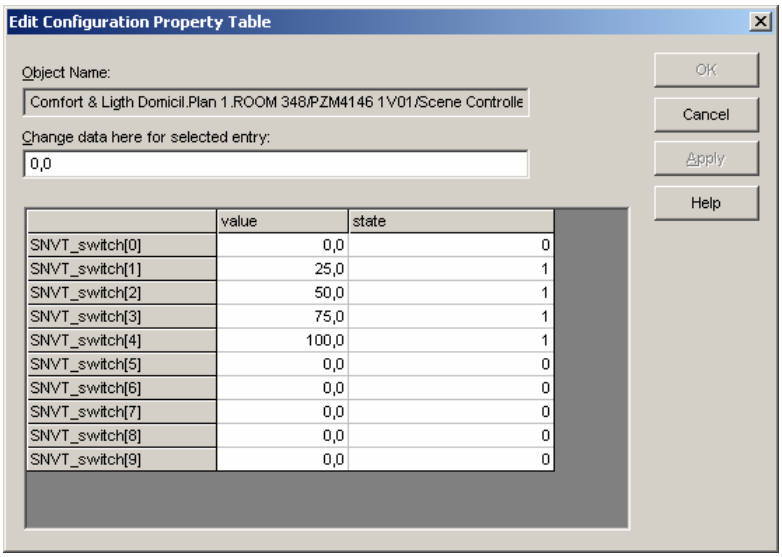



Find the relation between the scene number index and the scene value index in the next section describing the configuration property UCPTsceneValue.

### *1.1.1.2.2 UCPTsceneValue*

The UCPTsceneValue is a configuration table property. The table is used to configure a SNVTswitch value related to a specific switch index number. Related to the fixed index number from SNVT_switch(0) to SNVT_switch(9) you can configure the SNVTswitch value and state for each index number.

I the below windows list, you can see that index SNVT_switch(1) is configured to 25% and state 1 and that index SNVT_switch(3) is configured to 75% and state 1.



The combination using the two index's gives a very high flexibility when configuring the scene controller. In the below list is showed the fixed relation between the two index's, UCPTsceneNumber(X) and SNVT_switch(X). As you can see in the below table, will input scene number 3 relate to index UCPTsceneNumber(2). This relation will trigger that, the controller will use the value from index SNVT_switch(2) and send 50% and state 1 out on the nvoSwitch output variable.

| **UCPTsceneNumber** | | **UCPTsceneValue** | | |
|---|---|---|---|---|
| **Index** | **Scene Number** | **Index** | **Switch %** | **State** |
| [0] | 1 | [0] | 0,0 | 0 |
| [1] | 2 | [1] | 25,0 | 1 |
| **[2]** | **3** | **[2]** | **50,0** | **1** |
| [3] | 4 | [3] | 75,0 | 1 |
| [4] | 5 | [4] | 100,0 | 1 |
| [5] | 10 | [5] | 0 | 0 |
| [6] | 11 | [6] | 0 | 0 |
| [7] | 12 | [7] | 0 | 0 |
| [8] | 13 | [8] | 0 | 0 |
| [9] | 14 | [9] | 0 | 0 |

### 1.1.2 Occupancy Sensor (1060), 2 objects

The following describes the function for the occupancy sensor object. This object is used with a hardware sensor whose output is either in an occupied or unoccupied state. The occupancy sensor object is used to detect occupancy in a room or an area and keep the occupied state until no occupancy can be detected. The output from the occupancy sensor object is typically connected to a controller. The occupancy controller takes care of the proper action and calculates application delay or hold times as appropriate.

| PIR # | Connector # | Name in LonMaker list |
|---|---|---|
| 1 | 37.1 | OccupSensor_1 |
| 2 | 37.2 | OccupSensor_2 |

**Figure 1 The Occupancy hardware/software relation**

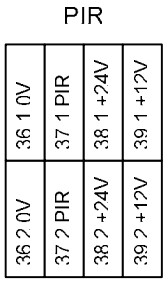



**Figure 2 The occupancy hardware input**

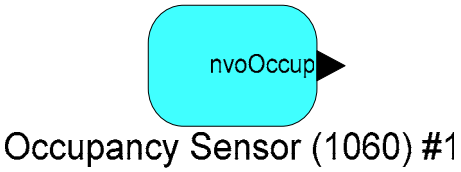



**Figure 4 The occupancy sensor object**

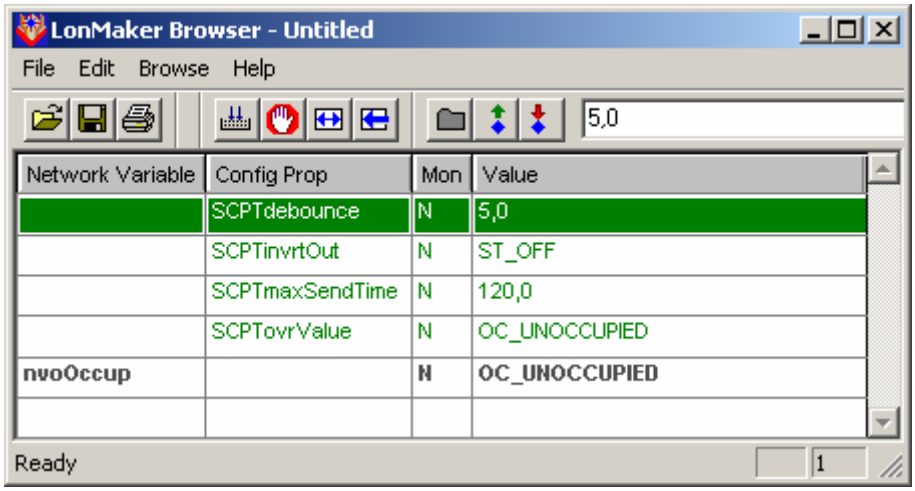



**Figure 5 Browsing the occupancy sensor object**

### 1.1.2.1 Network Variable Interface

| Variable name | Default Value | Type | Description |
|---|---|---|---|
| nvoOccup | OC_UNOCCUPIED | SVNT_occupancy (109) | Sensor output |

#### *1.1.2.1.1 nvoOccup*

This output network variable provides the qualified state of the hardware sensor output connected to the Comfort and Ligth Controller. The sensor input can be inverted with the configuration property SCPTinvrtOut.

The variable is transmitted when the occupancy state changes from unoccupied to occupied or when the occupancy state has changed from occupied to unoccupied at the frequency of the heartbeat. The maximum update rate is fixed by the heartbeat frequency and the minimum update rate is fixed by the debouncing time. The default service type is acknowledged.

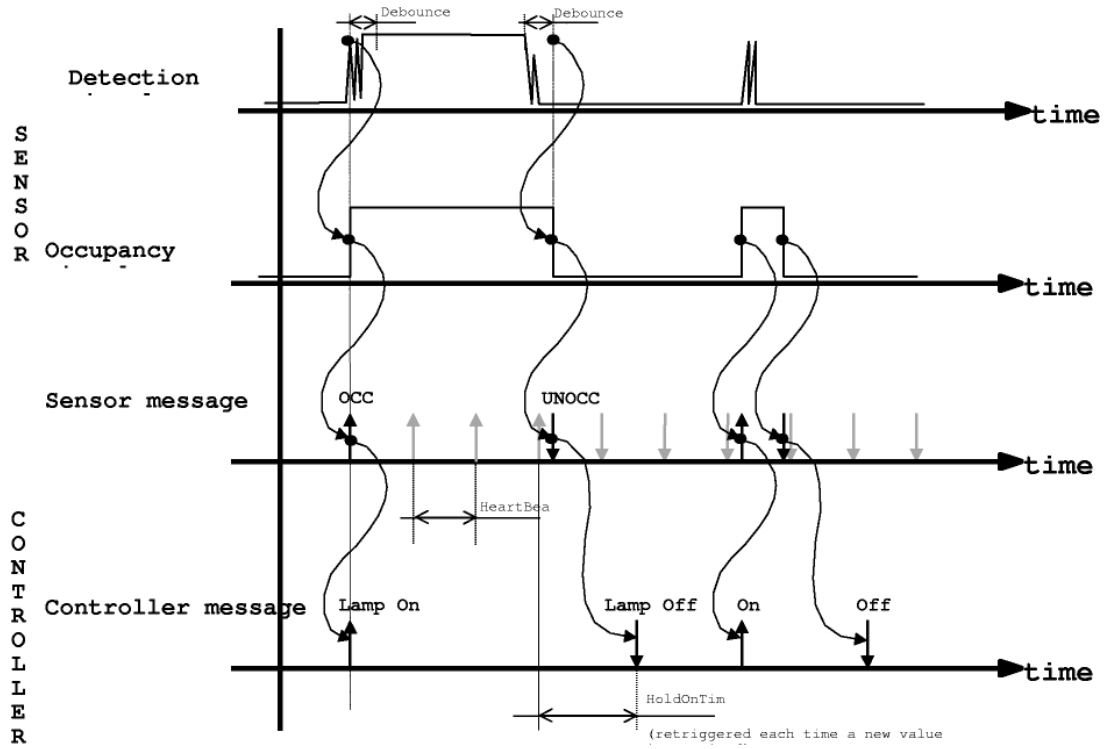
### 1.1.2.2 Configuration Properties

| Configuration | Default Value | Description |
|---|---|---|
| SCPTdebounce (139) | 5,0 | Debounce filter time |
| SCPTinvrtOut (16) | ST_OFF | Invert the output |
| SCPTmaxSendTime (49) | 120,0 | Heartbeat time |
| SCPTovrValue (33) | OC UNOCCUPIED | State when in override mode |

#### *1.1.2.2.1 SCPTdebounce*

This mandatory configuration network variable defines the debounce time to generate the detection envelop and the OCCUPIED and NON OCCUPIED messages. Valid range is 0.0 - 6553.4 by steps of 0.1s.

The following timing diagram defines the different timers used in the profile, it illustrate the debounce filter function.




### *1.1.2.2.2 SCPTinvrtOut*

The SCPTinvrtOut configuration property is used to invert the the input signa, so sensor with normal close also can be used.

The property is either:

| **SCPTinvrtOut** | |
|---|---|
| ST_OFF | Not inverted input |
| ST_ON | Inverted input |

### *1.1.2.2.3 SCPTmaxSendTime*

SCPTmaxSendTime is a configuration property. This configuration defines the repeat period between to value update sent on the bus also called a heartbeat. The aims of the heartbeat is to be sure that the sensor is alive and to permit a controller to have multiple sensors on the same input

**Remark** : Even if the heartbeat is optional, the heartbeat functionality
shall be implemented in the Occupancy Sensor with a 2 minutes repeat period, (default value).

#### *1.1.2.2.4 SCPTovrValue*

This override value sets the value the actuator should adopt when the object is overridden and behavior is OV_SPECIFIED.

### 1.1.3 Occupancy Controller (3071), 6 objects

The following section is describing the occupancy controller. Typically the occupancy controller input is connected to the occupancy sensor and the output to a lamp. A switch can be used to turn the occupancy controller into ON and OFF mode. An additional switch can override the controller and directly control the lamp. The occupancy controller can also be driven by an occupancy sensor from a neighboring area, secondary input. The secondary function gives the users a secure feeling, due to the light around. The occupancy controller can also operate another controller, eg. a constant light controller.

| Controller# | Name in LonMaker list |
|---|---|
| 1 | OccupancyCtrl_1 |
| 2 | OccupancyCtrl_2 |
| 3 | OccupancyCtrl_3 |
| 4 | OccupancyCtrl_4 |
| 5 | OccupancyCtrl_5 |
| 6 | OccupancyCtrl_6 |

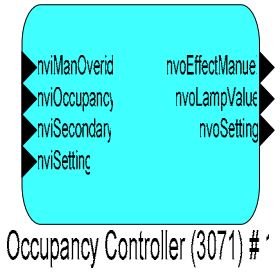



**Figure 6 The Occupancy controller object**

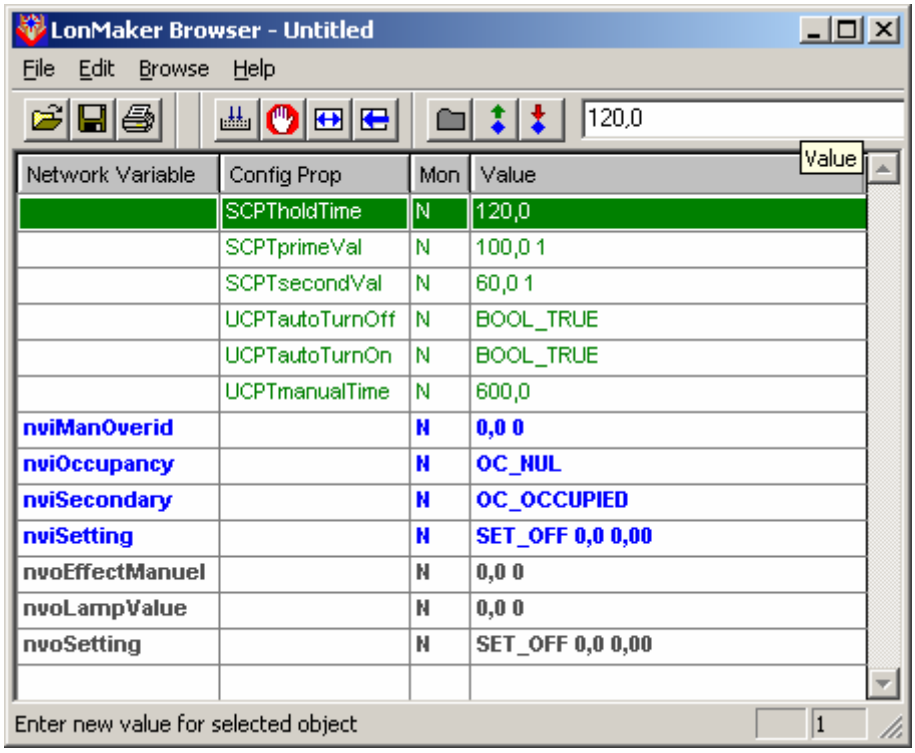



**Figure 7 Browsing the occupancy controller object**

### 1.1.3.1 Network Variable Interface

| Variable name | Default Value | Type | Description |
|---|---|---|---|
| nviManOverride | 0,0 0 | SNVT_switch (95) | Manual Operation |
| nviOccupancy | OC_NUL | SVNT_occupancy (109) | Occupancy input |
| nviSecondary | OC_UNOCCUPIED | SVNT_occupancy (109) | Neighbourhood Occupancy input |
| nviSetting | SET_OFF 0,0 0,00 | SNVT_setting (117) | Setting control input |
| | | | |
| nvoEffectManuel | 0,0 0 | SNVT_switch (95) | Manual indicator output |
| nvoLampValue | 0,0 0 | SNVT_switch (95) | Controller output |
| nvoSetting | SET_OFF 0,0 0,00 | SNVT_setting (117) | Setting control output |

#### *1.1.3.1.1 nviManOverride*

Manual override input, enables the local and manual control of the lamp value output.

#### *1.1.3.1.2 nviOccupancy*

Occupancy status input value, provides the occupancy status for the area.

#### *1.1.3.1.3 nviSecondary*

Secondary occupancy status input value. It provides the occupancy status of a neighbourhood area, in order to provide low-level lighting around an occupied area for a feeling of security.

#### *1.1.3.1.4 nviSetting*

Setting input (Auto/Off). This mode can be either be ON(AUTO) or OFF.

#### *1.1.3.1.5 nvoEffectManuel*

This output variable reflect the status of manual override.

#### *1.1.3.1.6 nvoLampValue*

Lamp value output. State for the lamp actuator ON or OFF, and the percentage level of intensity.

#### *1.1.3.1.7 nvoSetting*

Setting output. Select the operating mode for another controller, e.g., constant light controllers.

### 1.1.3.2 Configuration Properties

| Configuration | Default Value | Description |
|---|---|---|
| SCPTholdTime (91) | 600 | Occupancy delay timer |
| SCPTprimeVal (155) | 100,0 1 | Output value when occupied |
| SCPTsecondVal (156) | 60,0 1 | Output for Value neighboring |
| | | |
| UCPTautoTurnOff | BOOL TRUE | Turn off selection, true or false |
| UCPTautoTurnOn | BOOL TRUE | Turn on selection, true or false |
| UCPTmanualTime | 600 | Delay timer when manual control |

#### *1.1.3.2.1 SCPTholdTime*

Hold time for occupied state after there is no occupancy detected. This timer is used for both the main occupancy input and for the secondary input.

#### *1.1.3.2.2 SCPTprimeVal*

The default output value when an area is occupied.

#### *1.1.3.2.3 SCPTsecondVal*

The default output value when the neighboring area is occupied, when the secondary occupancy sensor input variable is active.

#### *1.1.3.2.4 UCPTautoTurnOff*

This configuration property is used to select if the controller automatically switch off the output when it time out the timer.

| UCPTautoTurnOff | Description |
|---|---|
| BOOL_TRUE | Automatic off activated |
| BOOL_FALSE | No automatic off activated |

#### *1.1.3.2.5 UCPTautoTurnOn*

This configuration property is used to select if the controller automatically switch on the output when occupancy is detected.

| UCPTautoTurnOn | Description |
|---|---|
| BOOL_TRUE | Automatic on activated |
| BOOL_FALSE | No automatic on activated |

#### *1.1.3.2.6 UCPTmanualTime*

This configuration property is used to set the timer value used during a manual override input on the nviManOverride. The manual timer is also serviced by the occupancy input and will be set every time an occupancy signal is received. Hereafter the timer will countdown and when it times out manual override will be deactivated.

### 1.1.4 Partition Wall Controller (3252), 2 objects

The Partition Wall Controller is used in spaces that can be divided into smaller sections. The object passes data to the area next to it, if the space is open (no partitioning). If the space is divided, then light-switch data is not passed to the other side of the separating wall.

The LonMark PartitionWall Controller is based only on Scenes. The Prolon Control Systems version expand the controller to both Scene, Occupancy and Switch.

| Controller# | Name in LonMaker list |
|---|---|
| 1 | PartWallCtrl_1 |
| 2 | PartWallCtrl_2 |

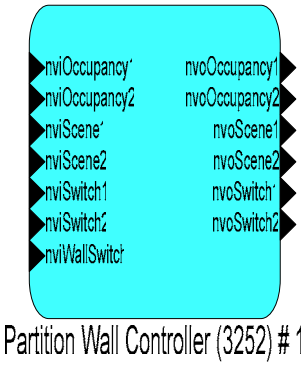



**Figure 8 The Partition wall controller object**

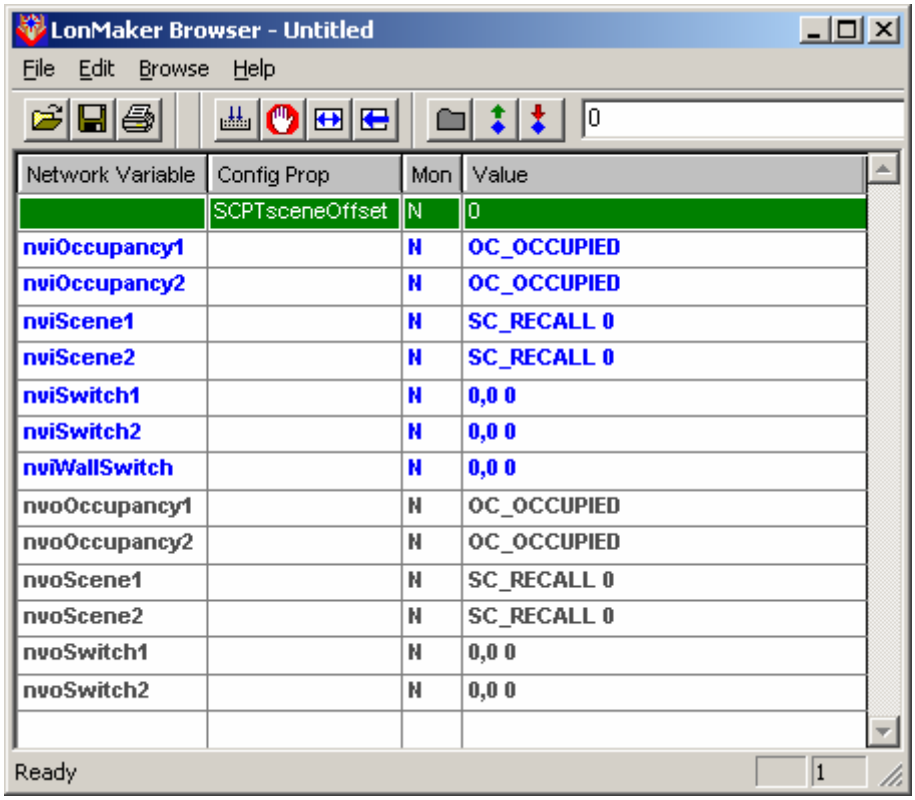



**Figure 9 Browsing the partition wall controller object**

### 1.1.4.1 Network Variable Interface

| Variable name | Default Value | Type | Description |
|---|---|---|---|
| nviWallSwitch | 0,0 0 | SNVT_switch (95) | Partition wall switch input |
| nviScene1 | SCENE_RECALL 0 | SNVT_scene (115) | Primary input |
| nviScene2 | SCENE_RECALL 0 | SNVT_scene (115) | Secondary input |
| nviOccupancy1 | OC_OCCUPIED | SVNT_occupancy (109) | Primary input |
| nviOccupancy2 | OC_OCCUPIED | SVNT_occupancy (109) | Secondary input |
| nviSwitch1 | 0,0 0 | SNVT_switch (95) | Primary input |
| nviSwitch2 | 0,0 0 | SNVT_switch (95) | Secondary input |
| | | | |
| nvoScene1 | SCENE_RECALL 0 | SNVT_scene (115) | Primary output |
| nvoScene2 | SCENE_RECALL 0 | SNVT_scene (115) | Secondary output |
| nvoOccupancy1 | OC_OCCUPIED | SVNT_occupancy (109) | Primary output |
| nvoOccupancy2 | OC_OCCUPIED | SVNT_occupancy (109) | Secondary output |
| nvoSwitch1 | 0,0 0 | SNVT_switch (95) | Primary output |
| nvoSwitch2 | 0,0 0 | SNVT_switch (95) | Secondary output |

#### *1.1.4.1.1 nviWallSwitch*

Partition wall switch input. The valid range of the input is, partition open state=ON and partition closed state=OFF. Other enumerations of the state are discarded; value field has no effect.

#### *1.1.4.1.2 nviScene1*

Primary side scene input. Read in data from the scene panels and other sensors on the primary side of the partition wall.

#### *1.1.4.1.3 nviScene2*

Secondary side scene input. Read in data from the scene panels and other sensors, of the secondary side of the partition wall.

#### *1.1.4.1.4 nviOccupancy1*

Primary side occupancy input. Read in data from the occupancy sensors on the primary side of the partition wall.

#### *1.1.4.1.5 nviOccupancy2*

Secondary side occupancy input. Read in data from the occupancy sensors on the secondary side of the partition wall.

#### *1.1.4.1.6 nviSwitch1*

Primary side Switch input. Read in data from the switch panels and other sensors on the primary side of the partition wall.

#### *1.1.4.1.7 nviSwitch2*

Secondary side switch input. Read in data from the switch panels and other sensors, of the secondary side of the partition wall.

#### *1.1.4.1.8 nvoScene1*

Primary side scene output. Provide the scene output to the scene controllers on the primary side of the partition wall.

#### *1.1.4.1.9 nvoScene2*

Secondary side scene output. Provide the scene output to the scene controllers on the secondary side of the partition wall.

#### *1.1.4.1.10 nvoOccupancy1*

Primary side occupancy output. Provide the occupancy output to the occupancy controllers on the primary side of the partition wall.

#### *1.1.4.1.11 nvoOccupancy2*

Secondary side occupancy output. Provide the occupancy output to the occupancy controllers on the secondary side of the partition wall.

#### *1.1.4.1.12 nvoSwitch1*

Primary side switch output. Provide the switch output on the primary side of the partition wall.

#### *1.1.4.1.13 nvoSwitch2*

Secondary side switch output. Provide the switch output on the secondary side of the partition wall.

### 1.1.4.2 Configuration Properties

| Configuration | Default Value | Description |
|---|---|---|
| SCPTsceneOffset () | 0 | Scene offset from primary to secondary side |

#### *1.1.4.2.1 SCPTsceneOffset*

The scene offset configuration property, is used to set the offset for scene number, when data is forwarded from primary to secondary side of the partition wall controller.

### 1.1.5 Digital Functional Block, 4 objects

The digital function object is used to make logical functions as AND, OR, NAND and NOR gates. In a simple configuration it can be used to invert a signal, both input and output can be inverted. The object can also be used as a latch gate and a relay function. The gate function have two input and one output.

The logical function object help the system integrator with quick and easy changes and new functions without requiring a new firmware for the products in the network.

| Digital function# | Name in LonMaker list |
|---|---|
| 1 | DigitalFunc_1 |
| 2 | DigitalFunc_2 |
| 3 | DigitalFunc_3 |
| 4 | DigitalFunc_4 |

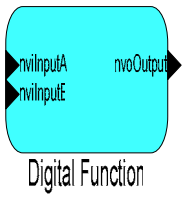



**Figure 10 The digital function object**

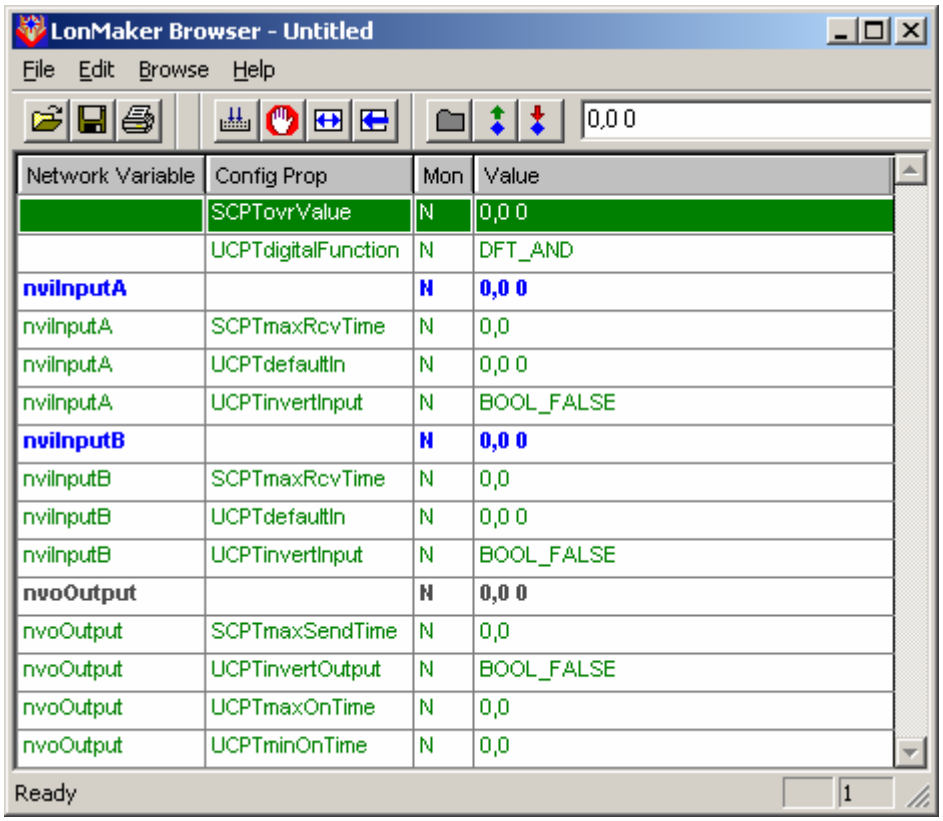



**Figure 11 Browsing the digital function object**

### 1.1.5.1 Network Variable Interface

| Variable name | Default Value | Type | Description |
|---|---|---|---|
| nviInputA | 0,0 0 | SNVT_switch (95) | Gate input A |
| nviInputB | 0,0 0 | SNVT_switch (95) | Gate input B |
| | | | |
| nvoOutput | 0,0 0 | SNVT_switch (95) | Gate output |

#### *1.1.5.1.1 nviInputA*

This input variable is the A input to the gate function. The input switch variable is used as a digital indicator. Logic 0 = 0,0 % state 0, any other combinations are logic 1.

#### *1.1.5.1.2 nviInputB*

This variable is the B input to the gate function. The input switch variable is used as a digital indicator. Logic 0 = 0,0 % state 0, any other combinations are logic 1

#### *1.1.5.1.3 nvoOutput*

This variable is the output from the gate function,  it's a SNVT_switch type variable.
Logic 0 = 0,0% state 0.
Logic 1 = 100% state 1.

### 1.1.5.2 Configuration Properties

| Configuration | Default Value | Description |
|---|---|---|
| SCPTovrValue (33) | 0,0 0 | State when in override mode |
| SCPTmaxRcvTime (48), nviInputA | 0,0 | Maximum receive time |
| SCPTmaxRcvTime (48), nviInputB | 0,0 | Maximum receive time |
| SCPTmaxSendTime (49) | 0,0 | Maximum send time |
| | | |
| UCPTdigitalFunction (3:86) | DFT_AND | Set the function metode |
| UCPTdefaultIn, nviInputA | 0,0 0 | Set a fixed input value |
| UCPTdefaultIn, nviInputB | 0,0 0 | Set a fixed input value |
| UCPTinvertInput, nviInputA | BOOL_FALSE | Invert the input A |
| UCPTinvertInput, nviInputB | BOOL_FALSE | Invert the input B |
| UCPTinvertOutput | BOOL_FALSE | Invert the output |
| UCPTmaxOnTime | 0,0 | Maximum output ON time |
| UCPTminOnTime | 0,0 | Minimum output ON time |

#### *1.1.5.2.1 SCPTovrValue*

This configuration type set the value for the output variable, when the object is in override mode. The variable type is a SNVT_switch (95).  The value is used on the nvoOutput variable from the object.

#### *1.1.5.2.2 SCPTmaxRcvTime, nviInputA*

The maximum period of time that may expire with no updates on the associated input network variables before the object goes into heartbeat failure mode. A zero value disables.

#### *1.1.5.2.3 SCPTmaxRcvTime, nviInputB*

The maximum period of time that may expire with no updates on the associated input network variables before the object goes into heartbeat failure mode. A zero value disables.

#### *1.1.5.2.4 SCPTmaxSend Time*

The maximum period of time between consecutive transmissions of the current value.

### *1.1.5.2.5 UCPTdigitalFunction*

This property is used to select the required digital function from the list below.

| UCPTdigitalFunction | Index | description |
|---|---|---|
| DFT_AND | 0 | Make a logical AND function, set output ON if both inputs are ON. |
| DFT_OR | 1 | Make a logical OR function, set the output ON if any of the two inputs are ON. |
| DFT_LATCH | 2 | Latch input 1 to the output and clear the latch when input 2 goes ON. |
| DFT_RELAY | 3 | Set output to same value as input 1, if input 2 is ON. If input 2 is OFF, output is also off. |

#### 1.1.5.2.5.1 DFT_AND

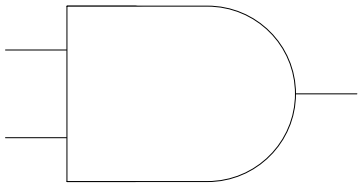


**Figure 12 AND Gate**

| AND Gate | | |
|---|---|---|
| **Input A** | **Input B** | **Output** |
| 0 | 0 | 0 |
| 0 | 1 | 0 |
| 1 | 0 | 0 |
| 1 | 1 | 1 |

#### 1.1.5.2.5.2 DFT_AND inverted output

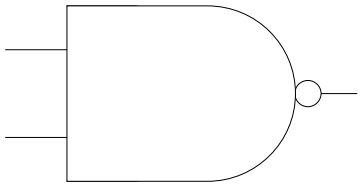


**Figure 13 NAND Gate**

| NAND Gate | | |
|---|---|---|
| **Input A** | **Input B** | **Output** |
| 0 | 0 | 1 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 0 |

#### 1.1.5.2.5.3 DFT_OR

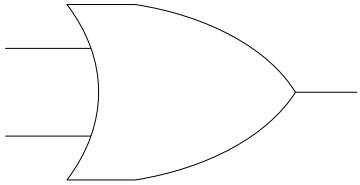


**Figure 14 OR gate**

| OR Gate | | |
|---|---|---|
| **Input A** | **Input B** | **Output** |
| 0 | 0 | 0 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 1 |

#### 1.1.5.2.5.4 DFT_OR inverted output

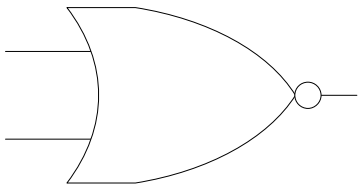


**Figure 15 NOR Gate**

| NOR Gate | | |
|---|---|---|
| **Input A** | **Input B** | **Output** |
| 0 | 0 | 1 |
| 0 | 1 | 0 |
| 1 | 0 | 0 |
| 1 | 1 | 0 |

#### 1.1.5.2.5.5 DFT_LATCH

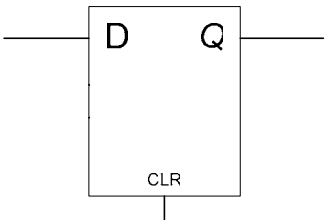



**Figure 16 Latch**

| LATCH | | |
|---|---|---|
| **D** | **CLR** | **Q** |
| Shift to 1 | 0 | 1 |
| Shift to 0 | 0 | Q |
| X | 1 | 0 |

#### 1.1.5.2.5.6 DFT_LATCH inverted output

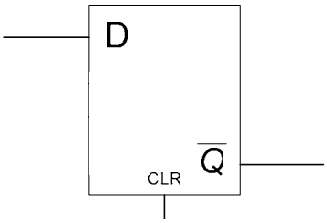



**Figure 17 Latch inverted out**

| LATCH | | |
|---|---|---|
| **D** | **CLR** | **Q** |
| Shift to 1 | 0 | 0 |
| Shift to 0 | 0 | Q |
| X | 1 | 1 |

#### 1.1.5.2.5.7 DFT_RELAY

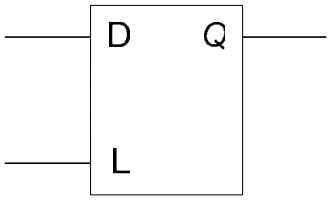



**Figure 18 Relay gate**

| Relay gate | | |
|---|---|---|
| **D** | **L** | **Q** |
| 1 | 1 | 1 |
| 0 | 1 | 0 |
| X | 0 | 0 |

#### 1.1.5.2.5.8 DFT_RELAY inverted output

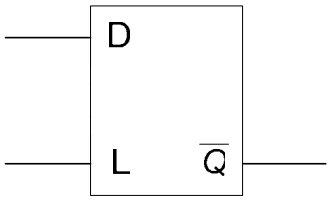



**Figure 19 Relay inverted output**

| Relay gate | | |
|---|---|---|
| **D** | **L** | **Q** |
| 1 | 1 | 0 |
| 0 | 1 | 1 |
| X | 0 | 1 |

#### *1.1.5.2.6 UCPTdefaultIn, nviInputA*

This configuration property is used to select a fixed value on the input A. This can be used as a comparator function with a fixed input to the gate and only one active input to the gate. The variable type is a SNVT_switch (95).

#### *1.1.5.2.7 UCPTdefaultIn, nviInputB*

This configuration property is used to select a fixed value on the input B. This can be used as a comparator function with a fixed input to the gate and only one active input to the gate. The variable type is a SNVT_switch (95)

#### *1.1.5.2.8 UCPTinvertInput, nviInputA*

Configuration property used to select if the input A is inverted or not.

| **UCPTinvertOutput** | **Description** |
|---|---|
| BOOL_TRUE | Input inverted |
| BOOL_FALSE | Input not inverted |

#### *1.1.5.2.9 UCPTinvertInput, nviInputB*

Configuration property used to select if the input B is inverted or not.

| **UCPTinvertOutput** | **Description** |
|---|---|
| BOOL_TRUE | Input inverted |
| BOOL_FALSE | Input not inverted |

#### *1.1.5.2.10 UCPTinvertOutput*

Configuration property used to select if the output is inverted or not.

| **UCPTinvertOutput** | **Description** |
|---|---|
| BOOL_TRUE | Output inverted |
| BOOL_FALSE | Output not inverted |

#### *1.1.5.2.11 UCPTmaxOnTime*

The configuration property is used to limit the output in time. It will limit the active output time. The time is specified in seconds. If this function not used the time shall be set to 0,0 seconds, equal to no limitation. If the value is set to 10 seconds you will get an active output in 10 seconds even in the input is active for 20 seconds. It can be used as a ONE SHOOT function.

#### *1.1.5.2.12 UCPTminOnTime*

The configuration property is used to set a minimum time for the output. The output will as a minimum always have an active output equal to the minimum on time, even if the input to the gate is shorter than the minimum on time. The time is specified in seconds. If this function not used the

time shall be set to 0,0 seconds, equal to no minimum active time. This function can enlarge a short active ON signal and make sure to get an active output in the defined period.

### 1.1.6 Analog Functional Block, 4 objects

The analog function object is used to make analog functions as minimum out, maximum out, add inputs, subtract inputs, multiply inputs, divided inputs, take average of input and fin the difference between the inputs.

The analog function object help the system integrator with quick and easy changes and new functions without requiring a new firmware for the products in the network.

| Analog function# | Name in LonMaker list |
|---|---|
| 1 | AnalogFunc_1 |
| 2 | AnalogFunc_2 |
| 3 | AnalogFunc_3 |
| 4 | AnalogFunc_4 |

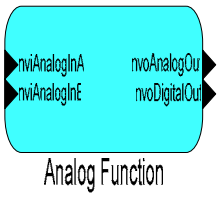



**Figure 20 The analog function object**

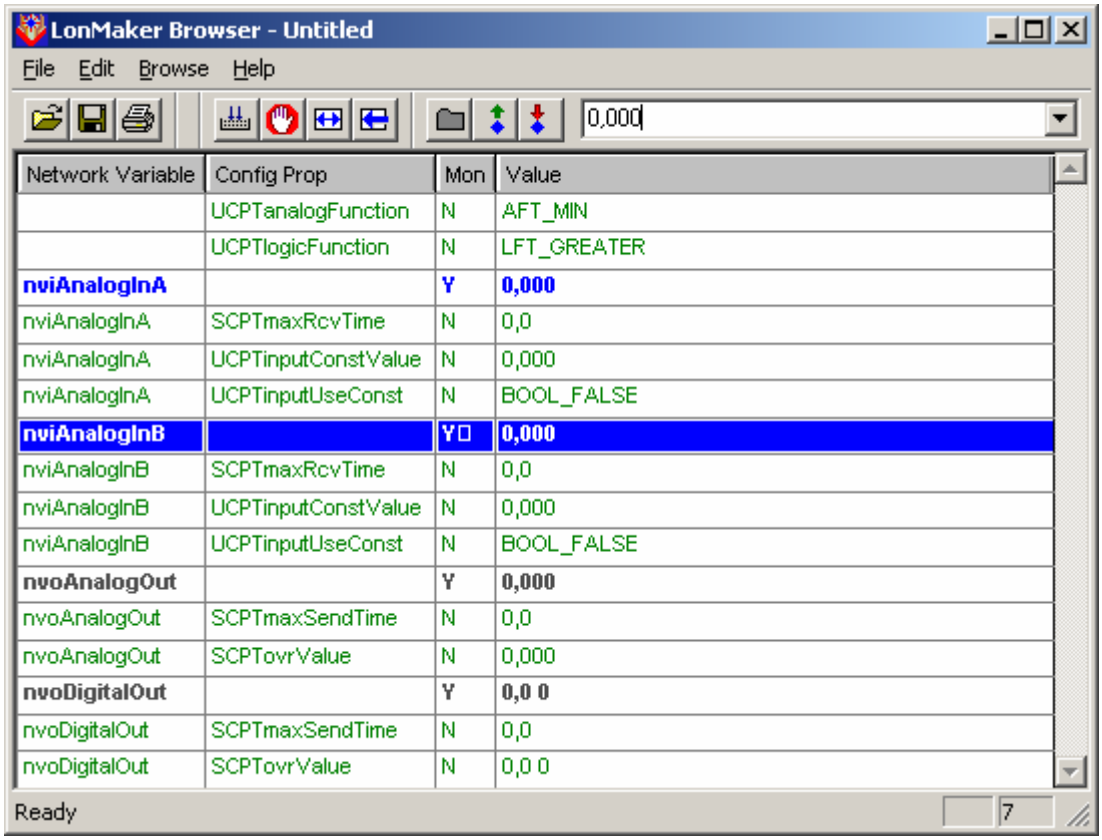



**Figure 21 Browsing the analog function object**

### 1.1.6.1 Network Variable Interface

| Variable name | Default Value | Type | Description |
|---|---|---|---|
| nviAnalogInA | 0,0 | SNVT_lev_percent (81) | Analog input A |
| nviAnalogInB | 0,0 | SNVT_lev_percent (81) | Analog input B |
| nvoAnalogOut | 0,0 | SNVT_lev_percent (81) | Analog output |
| nvoDigitalOut | 0,0 0 | SNVT_switch (95) | Digital output |

#### *1.1.6.1.1 nviAnalogInA*

This variable is the A input to the analog function.

#### *1.1.6.1.2 nviAnalogInB*

This variable is the B input to the analog function.

#### *1.1.6.1.3 nvoAnalogOut*

This variable is the analog output from the analog function. The output is processed by the analog function and how it is processed is depending on the function selection with the configuration property UCPTanalogFunction. The function could be AFT_MIN and this will send the minimum value out on the output variable.

#### *1.1.6.1.4 nvoDigitalOut*

This variable is the digital output from the analog function. The output is processed by the analog function and how it is processed is depending on the function selection with the configuration property UCPTlogicFunction. The result is always a digital output related to the analog inputs. If the UCPTlogicFunction is LFT_GREATER, you will get an ON output if the A is greator than B. The ON output is 100,0 % state 1, the OFF output is 0,0% state 0.

### 1.1.6.2 Configuration Properties

| Configuration | Default Value | Description |
|---|---|---|
| SCPTovrValue, nvoAnalogOut | 0,0 | Value in override mode |
| SCPTovrValue, nvoDigitalOut | 0,0 0 | Value in override mode |
| SCPTmaxRcvTime, nviAnalogInA | 0,0 | Maximum receive time |
| SCPTmaxRcvTime, nviAnalogInB | 0,0 | Maximum receive time |
| SCPTmaxSend Time, nvoAnalogOut | 0,0 | Maximum send time |
| SCPTmaxSend Time, nvoDigitalOut | 0,0 | Maximum send time |
| | | |
| UCPTanalogFunction (3:85) | AFT_MIN | Set analog function type |
| UCPTlogicFunction (3:87) | LFT_GREATER | Set logic function type |
| UCPTinputConstValue, nviAnalogA | 0,0 | Constant value on A |
| UCPTinputConstValue, nviAnalogB | 0,0 | Constant value on B |
| UCPTinputUseConst, nviAnalogA | BOOL_FALSE | Select if constant used A |
| UCPTinputUseConst, nviAnalogB | BOOL_FALSE | Select if constant used B |

#### *1.1.6.2.1 SCPTovrValue, nvoAnalogOut*

This configuration property set the value used in override mode. The value is a SNVT_lev_percent (81). This value will be send out from the object in override mode on nvoAnalogOut.

#### *1.1.6.2.2 SCPTovrValue, nvoDigitalOut*

This configuration property set the value used in override mode. The value is a SNVT_switch (95). This value will be send from the object in override mode on nvoDigitalOut.

#### *1.1.6.2.3 SCPTmaxRcvTime, nviAnalogInA*

The maximum period of time that may expire with no updates on the associated input network variables before the object goes into heartbeat failure mode. A zero value disables.

#### *1.1.6.2.4 SCPTmaxRcvTime, nviAnalogInB*

The maximum period of time that may expire with no updates on the associated input network variables before the object goes into heartbeat failure mode. A zero value disables.

#### *1.1.6.2.5 SCPTmaxSend Time, nvoAnalogOut*

The maximum period of time between consecutive transmissions of the current value.

#### *1.1.6.2.6 SCPTmaxSend Time, nvoDigitalOut*

The maximum period of time between consecutive transmissions of the current value.

#### *1.1.6.2.7 UCPTanalogFunction*

This configuration property is used to select the function in the analog function object. Depending on the setting you can either calculate, get the minimum, maximum or average value or find the difference between the 2 input, A and B.

| UCPTanalogFunction | Index # | Description |
|---|---|---|
| AFT_MIN | 0 | Set the output to the smallest of the two input values. |
| AFT_MAX | 1 | Set the output to the largest of the two input values. |
| AFT_ADD | 2 | Add the two input values. |
| AFT_SUB | 3 | Subtract input 2 from input 1. |
| AFT_MUL | 4 | Multiply the two input values. |
| AFT_DIV | 5 | Divide input value 1 by input value 2. |
| AFT_AVG | 6 | Take the average of the two inputs |
| AFT_DIF | 7 | Take the difference between the two inputs. |

#### *1.1.6.2.8 UCPTlogicFunction*

This configuration property select the logically function between input A and B and reflect it as a digital state, if the statement is true or not.

| UCPTlogicFunction | Index # | Description |
|---|---|---|
| LFT_GREATER | 0 | A greater than B |
| LFT_GREATER_EQUAL | 1 | A greater or equal to B |
| LFT_SMALLER | 2 | A smaller than B |
| LFT_SMALLER_EQUAL | 3 | A smaller or equal to B |
| LFT_EQUAL | 4 | A equal to B |
| LFT_DEVIATES | 5 | A deviates from B |

#### *1.1.6.2.9 UCPTinputConstValue, nviAnalogA*

With this property a constant value is configured, it can be used on input A. The UCPTinputUseConst select if it is used or not.

#### *1.1.6.2.10 UCPTinputConstValue, nviAnalogB*

With this property a constant value is configured, it can be used on input B. The UCPTinputUseConst select if it is used or not.

#### *1.1.6.2.11 UCPTinputUseConst, nviAnalogA*

This property select if the constant value UCPTinputConstValue is used on input A instead of a input from the network.

| UCPTinputUseConst | Description |
|---|---|
| BOOL_TRUE | Constant value used |
| BOOL_FALSE | Constant value not used |

#### *1.1.6.2.12 UCPTinputUseConst, nviAnalogB*

This property select if the constant value UCPTinputConstValue is used on input B instead of a input from the network.

| **UCPTinputUseConst** | **Description** |
|---|---|
| BOOL_TRUE | Constant value used |
| BOOL_FALSE | Constant value not used |

## *1.2 Lighting profiles:*

The following is describing the lightning software functions implemented in the Comfort and light controller Lonbox PZM4146.

### 1.2.1 Light Sensor Input (1010), 2 objects

The following describes the light sensor object. The object is used to measure ambient light levels. The light sensor object is used together with controller objects, such as the constant light controller. Typically the light sensor output is connected to the constant light controller input.

The light object service the hardware input used for light sensors and convert the hardware signals to a digital SNVT resource on the network. The hardware connected is detected as a 0 til 10 voltage input signal.

| LUX # | Connector # | Name in LonMaker list |
|---|---|---|
| 1 | 41.1 | LightSensor_1 |
| 2 | 41.2 | LightSensor_2 |

**Figure 3 The light sensor hardware/software relation**

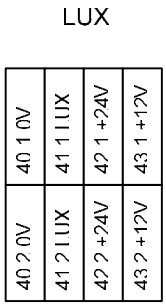



**Figure 4 The light sensor hardware input**

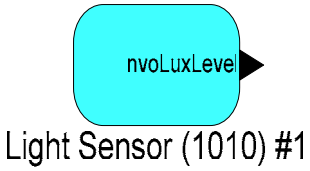



**Figure 22 The light sensor object**

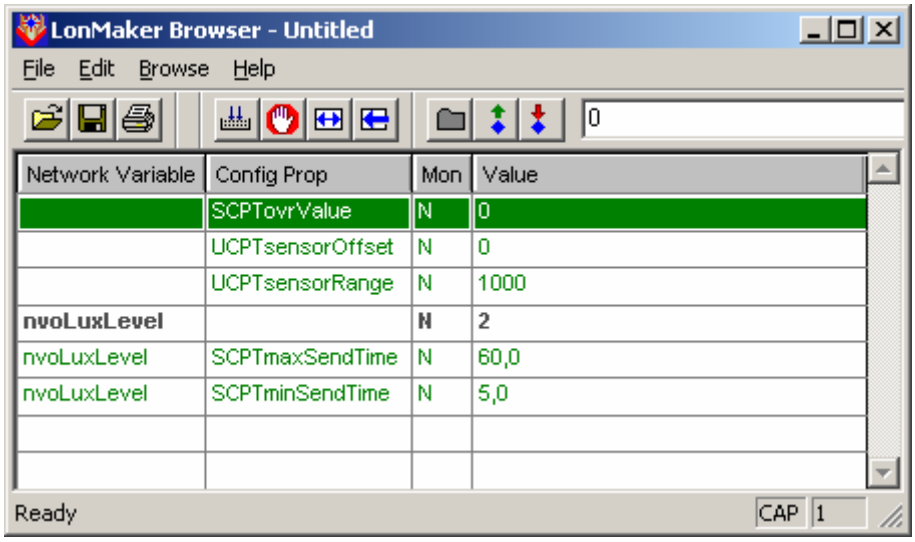



**Figure 23 Browsing the light sensor object**

### 1.2.1.1 Network Variable Interface

| Variable name | Default Value | Type | Description |
|---|---|---|---|
| nvoLuxLevel | 0 | SNVT_lux (79) | Output light level |

#### *1.2.1.1.1 nvoLuxLevel*

This variable is the output from the light sensor object. The hardware monitors the 0 to 10 volt input voltage and send out the variable manipulated by the configuration properties UCPTsensorOffset and UCPTsensorRange

### 1.2.1.2 Configuration Properties

| Configuration | Default Value | Description |
|---|---|---|
| SCPTovrValue (33) | 0 | Lux level when overwrite |
| SCPTmaxSendTime (49) | 60,0 | Maximum receive time |
| SCPTminSendTime (52) | 1,0 | Minimum send time |
| | | |
| UCPTsensorRange | 1/1 | Lux level output range |
| UCPTsensorOffset | 0 | Lux level offset from zero |

#### *1.2.1.2.1 SCPTovrValue*

This configuration property set the value for the output variable, when the object is in override mode. The variable type is a SNVT_lux (79). The value is used on the nvoLuxLevel variable from the object.

#### *1.2.1.2.2 SCPTmaxSendTime*

This configuration property set the value for the send heartbeat. Maximum period of time that expires before the object automatically transmits the present value of the lux level output NV.

#### *1.2.1.2.3 SCPTminSendTime*

This configuration property set the value for the minimum send time. Minimum period between output NV transmissions (maximum transmission rate).

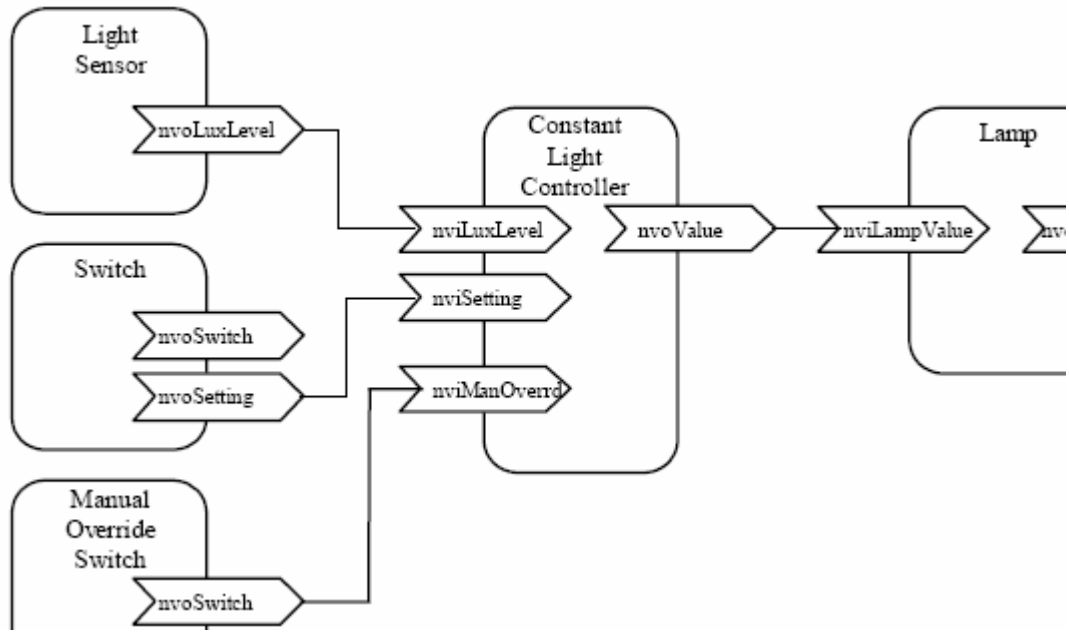
#### *1.2.1.2.4 UCPTsensorRange*

This configuration property is used to calibrate the external hardware input. It will calibrate the sensor range output. If the value is set to 1000, 10 volt on the input will be represented as 1000 lux and the highest value send out is 1000 lux.

#### *1.2.1.2.5 UCPTsensorOffset*

This is used to calibrate the external hardware input. It will calibrate the sensor ra is used to calibrate the external hardware input. It will calibrate the sensor offset failure from the value 0. If the value is set to 20, 0 volt on the input will be represented as 20 lux on the network and the lowest value send out is 20 lux.

### 1.2.1.3 Light sensor object typically use



### 1.2.2 Lamp Actuator (3040), 6 objects

The following describes the lamp actuator object. The lamp actuator object is used to control the illumination level of a hardware lamp output. The lamp actuator is used with switch and controller devices such as the constant light controller and scene controller. A switch object output nvoSwitch is connected to the input nviLampValue of the lamp actuator. Controller objects can be used between switch type sensors and lamp actuators. In cases of multiple sensors the feedback connection can be used to synchronize a group of switches.

| Lamp # | Zone # | Connector # | Name in LonMaker list |
|---|---|---|---|
| 1 | 1 | 1.1 | Lamp_1_1 |
| 2 | 1 | 4.1 | Lamp_2_1 |
| 3 | 1 | 7.1 | Lamp_3_1 |
| 1 | 2 | 1.2 | Lamp_1_2 |
| 2 | 2 | 4.2 | Lamp_2_2 |
| 3 | 2 | 7.2 | Lamp_3_2 |

**Figure 5 The lamp actuators software hardware relation**

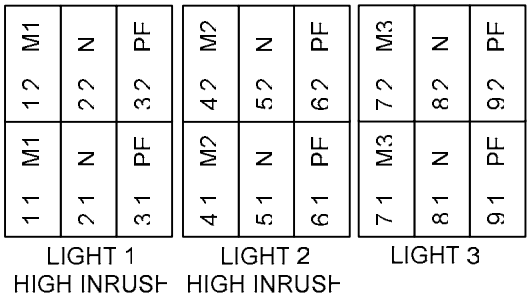



**Figure 6 The lamp actuator hardware outputs**

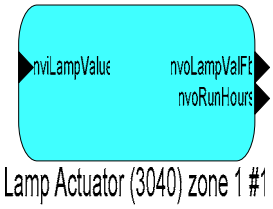



**Figure 24 The lamp actuator object**

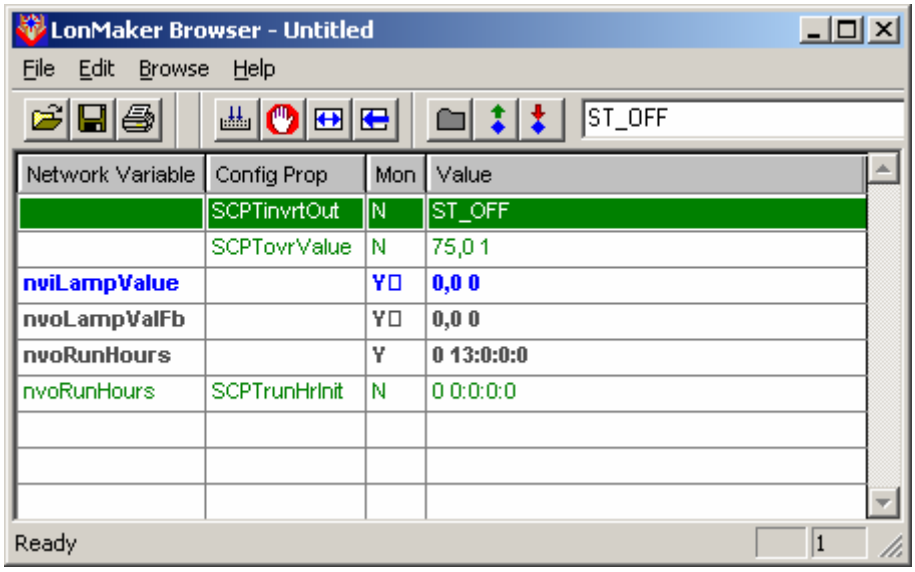



**Figure 25 Browsing the lamp actuator object**

### 1.2.2.1 Network Variable Interface

| Variable name | Default Value | Type | Description |
|---|---|---|---|
| nviLampValue | 0,0 0 | SNVT_switch (95) | Control input |
| nvoLampValueFb | 0,0 0 | SNVT_switch (95) | Lamp actuator feed back |
| nvoRunHours | 0 0:0:0:0 | SNVT_elapsed_tm (87) | Elapsed ON time |

#### *1.2.2.1.1 nviLampValue*

This variable is the control input. Permits other devices to control the Lamp Actuator level and state.

#### *1.2.2.1.2 nvoLampValueFb*

Lamp feedback output variable. State of the Lamp Actuator (ON or OFF) and the percentage level of intensity.

#### *1.2.2.1.3 nvoRunHours*

This output variable contains the value for the accumulated actuator ON time. The Comfort and Light Controller automatically monitor the lamp actuator's ON time. The value can be set and reset with the configuration property SCPTrunHrInit. The timer is enabled counting time, whenever the lamp actuator is turned ON. Elapsed time have following format:

day hour : minute : second : millisecond.

It is only the day and hour field in the structure that is used. Internally the counter is counting in seconds, but it only updates the hour and date field.

### 1.2.2.2 Configuration Properties

| Configuration | Default Value | Description |
|---|---|---|
| SCPTovrValue (33) | 75,0 1 | Lamp value when overwrite |
| SCPTinvrtOut (16) | ST_OFF | Setting if output is inverted |
| SCPTrunHrInit (135) | 0 0:0:0:0 | Set or reset Running hours |

#### *1.2.2.2.1 SCPTovrValue*

This configuration type set the value for the output variable, when the object is in override mode. The variable type is a SNVT_switch (95). The value is used on the nvoLampValue variable from the object.

#### *1.2.2.2.2 SCPTinvrtOut*

This configuration property set the parameter, if the lamp output is inverted or not. This possibility is used when an external relay is used and the electrically installation shall be fail-safe.

| SCPTinvrtOut | Description |
|---|---|
| ST_OFF | Output not inverted |
| ST_ON | Output inverted |

#### *1.2.2.2.3 SCPTrunHrInit*

This configuration is used to initialize the running hour counter value. The counter is send out on the SNVT variable nvoRunHours.

Elapsed time have the following format: day hour : minute : second :millisecond.

It is only the day and hour field in the structure that is used. Internally the counter is counting in seconds, but it only updates the hour and date field in the structure.

To change the value it's important to change value from what it is and not only resend the value existing in the property all ready.

For example:

The configuration property value is already 0 0:0:0:0 and we want to set the nvoRunHours to 0 0:0:0:0 using the configuration property. To do this you need to change the configuration property different from 0 0:0:0:0, it could be 0 1:0:0:0. Here after we set the value back to 0 0:0:0:0 and now the nvoRunHours is reset to 0 0:0:0:0.

Another example:

We want to set the nvoRunHours to 18 days and 1 hour (18 1:0:0:0). The configuration property value is 0 0:0:0:0, we can directly enter 18 1:0:0:0 to change the nvoRunHours.

### 1.2.2.3 Lamp object typically use

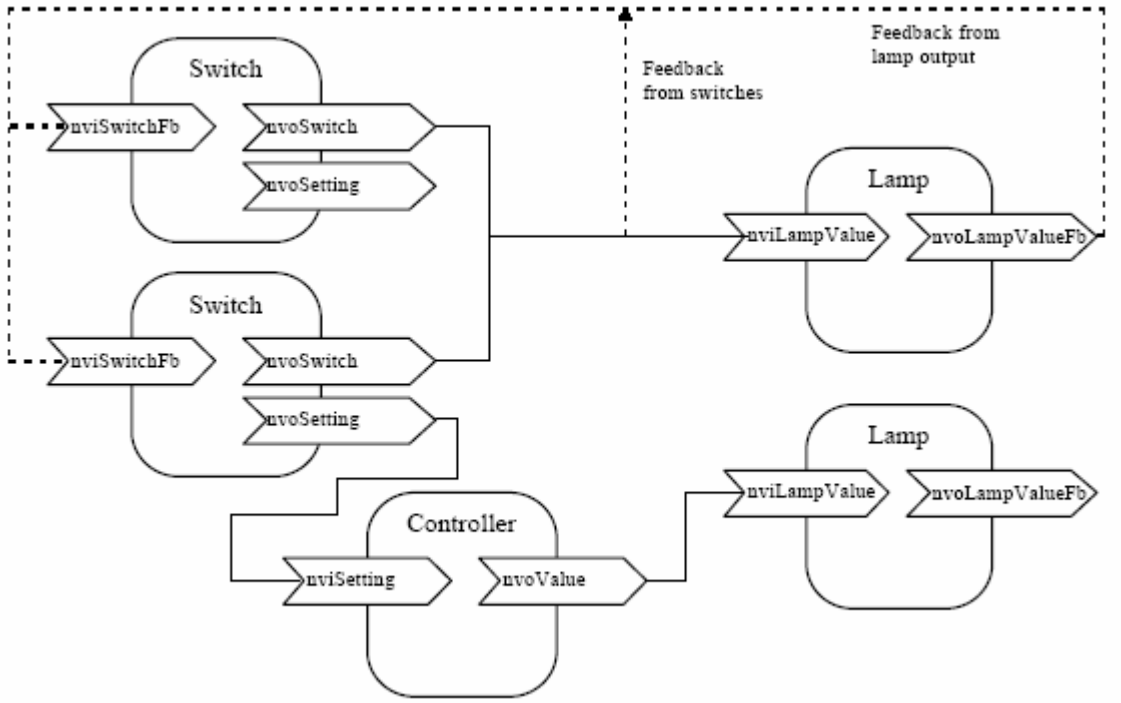



**Figure 26 Typically use of the lamp actuator object**

### 1.2.3 Constant Light Controller (3050), 4 objects

The following section is describing constant light controller object. The controller input is the ambient light level and the output is the state and illumination level to the lamp actuator.

Typically the constant light controller input is connected to a light sensor object and the output to a lamp actuator object. An extern input can be used to turn the constant light controller object into AUTO and OFF mode. Also the illumination level set point can temporarily be adjusted upwards and downwards. An additional switch can override the controller. When manual override input is written to, the constant light controller object is turned into MANUAL mode and the data is directly passed to the lamp.

The controller is designed from the experience Prolon have got, from the many building automation installation we have delivered components for. The controller is calculating the failure between the set point and the measured light level. If the failure is more than a fixed 8% failure dead band, the lamp output will be either be increased or decreased. The correction on the lamp output will change with the speed configured by the value in the configuration parameter step value.

| **Controller #** | **Name in LonMaker list** |
|---|---|
| 1 | ConstantLight_1 |
| 2 | ConstantLight_2 |
| 3 | ConstantLight_3 |
| 4 | ConstantLight_4 |

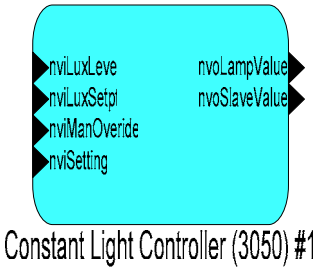



**Figure 27 The constant light controller object**

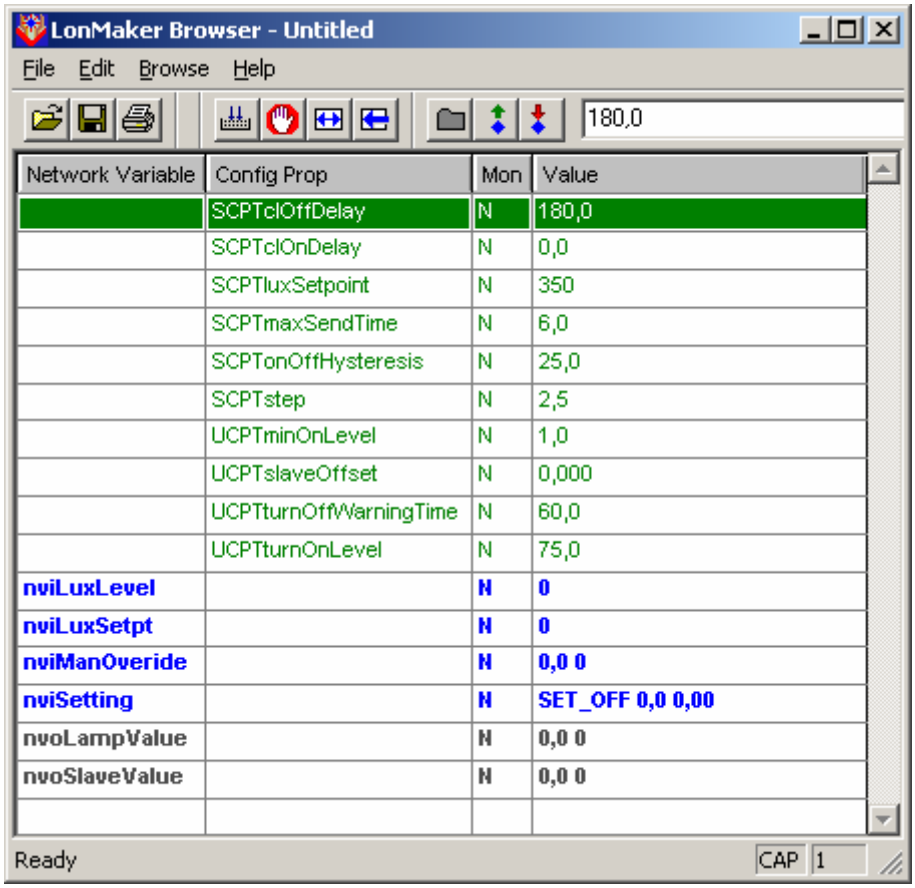



**Figure 28 Browsing the constant light controller object**

### 1.2.3.1 Network Variable Interface

| Variable name | Default Value | Type | Description |
|---|---|---|---|
| nviLuxLevel | 0 | SNVT_lux (79) | Ambient light level input |
| nviSetting | SET_OFF 0,0 0,00 | SNVT_setting (117) | Mode selection, setpoint |
| nviManOverride | | SNVT_switch (95) | Manual override control |
| nviLuxSetpt | 0 | SNVT_lux (79) | Temporary set point input |
| | | | |
| nvoLampValue | 0,0 0 | SNVT_switch (95) | Lamp actuator output |
| nvoSlaveValue | 0,0 0 | SNVT_switch (95) | Slave lamp actuator output |

#### *1.2.3.1.1 nviLuxLevel*

This input variable is sourcing the ambient light level to the controller object. The constant light controller object will calculate an output based on this value and it's failure from the set point defined with the SCPTluxSetpoint property or the temporary set point set by the input variable nviLuxSetpt. The primary output is the variable nvoLampValue and secondary the nvoSlaveValue.

#### *1.2.3.1.2 nviSetting*

This input network variable selects the operating mode and adjusts the setpoint of the constant light controller. Modes are ON, OFF, DOWN or UP.

The ON mode turns on the constant light controller into automatic, which then starts to control the lamp value output, so that the illumination level on the output which is provided by the nviLuxLevel equals to the setpoint value.

The OFF mode turns off the controller and the lamp value output.

The setpoint of the controller can temporarily be stepped upwards and downwards. The changes made to the setpoint value are not stored permanently into the memory. Next time when ON mode is selected the original setpoint from the SCPTluxSetpoint property is restored. The variable nviSetting uses SNVT_setting enumerations ON, OFF, UP and DOWN. With UP and DOWN functions a percentile value (0%-100% in 0.5% steps) is used to define the relative size of an increment/decrement.

#### *1.2.3.1.3 nviManOverride*

This variable is the manual override input and provides enable of manual control for the lamp value output.

#### *1.2.3.1.4 nviLuxSetpt*

This variable is the manual input for a temporary light set point. The set point will be reset to the set point defined by SCPTluxSetpoint property when the controller object is set ON with the setting input or when the Comfort and Light Controller is booted.

#### *1.2.3.1.5 nvoLampValue*

This output network variable provides the state for the lamp actuator (ON or OFF) and the percentage level of intensity. The valid range of the state is as defined for SNVT_switch where 0 means OFF and 1 means ON. The 8-bit intensity value contains a value 0 to 200, representing minimum to maximum (0%-100%) intensity. This variable I the primary output from the controller.

#### *1.2.3.1.6 nvoSlave Value*

This output network variable provides the state for the lamp actuator (ON or OFF) and the percentage level of intensity. The valid range of the state is as defined for SNVT_switch where 0 means OFF and 1 means ON. The 8-bit intensity value contains a value 0 to 200, representing minimum to maximum (0%-100%) intensity. This variable I the secondary output from the controller, normally used as a slave function. The slave output can be offset from the primary master output. Please see the following graphic to understand the offset function. The offset level can be configured med the parameter UCPTslaveOffset configuration property.

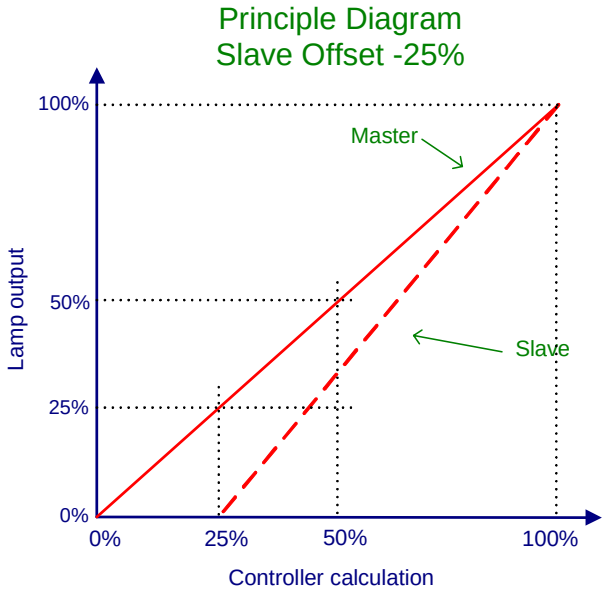



**Figure 29 Understanding the light slave output**

### 1.2.3.2 Configuration Properties

| Configuration | Default Value | Description |
|---|---|---|
| SCPTclOffDelay | 180,0 | Delay time switching Off |
| SCPTclOnDelay | 0,0 | Delay time switching On |
| SCPTluxSetpoint | 350 | Controller light level set point |
| SCPTmaxSendTime (52) | 6,0 | Maximum send time |
| SCPTonOffHysteresis | 25,0 | Hysteresis between On and OFF |
| SCPTstep | 2,5 | Speed for the regulation |
| | | |
| UCPTminOnLevel | 1,0 | The lowest light level |
| UCPTslaveOffset | 0,0 | Slave offset from master |
| UCPTturnOffWarningTime | 60,0 | Warning time before switching OFF |
| UCPTturnOnLevel | 75,0 | On level for automatic regulation |

#### *1.2.3.2.1 SCPTclOffDelay*

This configuration property is used to configure the delay time, before the automatic regulation is switching OFF the output after no light is required any longer.

The light will automatic switch off, when the output from the controller is 0% light level and the required light level has been obtained compared to the light level set point. Hereafter the light will switch OFF delayed by the SCPTclOffDelay time configuration.

#### *1.2.3.2.2 SCPTclOnDelay*

This configuration property is used to configure the delay time, before the automatic regulation is starting after is has been switched off, when light is required. Light is required if the measured light

level is lower than the set point, minus the SCPTonOffHysteresis value. Hereafter the light will turn on delayed be the time, configured in the SCPTclOnDelay property.

#### *1.2.3.2.3 SCPTluxSetpoint*

This property is used to configure the light level set point defined in lux. The value can be in the range 0 to 65636.

#### *1.2.3.2.4 SCPTmaxSendTime*

This configuration property set the value for the send heartbeat. Maximum period of time that expires before the object automatically transmits the present value of the lux level output NV.

#### *1.2.3.2.5 SCPTonOffHysteresis*

#### *1.2.3.2.6 SCPTstep*

This configuration property is used to configure the speed for the regulation loop. The property describe the maximum value in % point, that the output can change in one step. The output will change every second if necessary and the maximum change in % point is defined in the UCPTstep.

#### *1.2.3.2.7 UCPTminOnLevel*

This configuration property is used to configure the minimum level for the light when the light is on. The value is described in %.

#### *1.2.3.2.8 UCPTslaveOffset*

This configuration property is used to configure the slaves offset from the master. The offset can be positive or negative related to the master lamp output. It is not a parallel offset . The value is described in %.

#### *1.2.3.2.9 UCPTturnOffWarningTime*

This configuration property is used to configure the warning time before the light turns off. This function is warning that the light soon will turn off.

#### *1.2.3.2.10 UCPTturnOnLevel*

This configuration property is used to configure the level the controller will start on when switching on in automatic mode.

### 1.2.3.3 Constant light controller object typically use

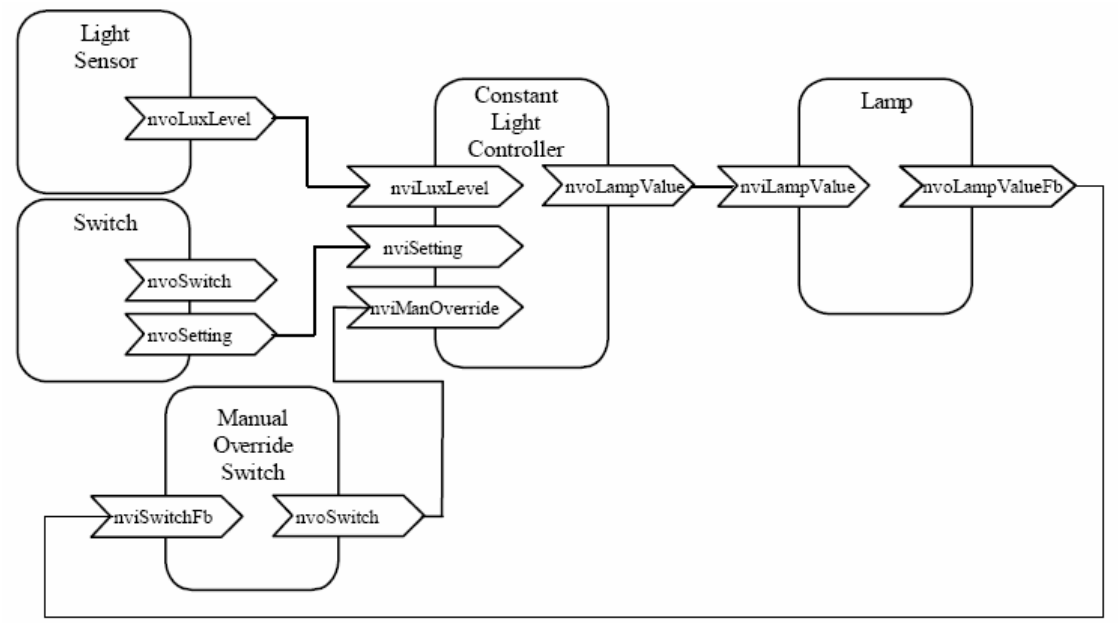



**Figure 30 Constant light controller typically use**

## *1.3 Space Comfort profiles:*

The following list is describing the space comfort software functions implemented in the Comfort and Light Controller Lonbox PZM4146.

### 1.3.1 HVAC Temperature Sensor Input (1040), 2 objects

The temperature sensor object is used to measure on a hardware input from a temperature sensor and make this measurement available on the network as a SNVT variable resource.

The hardware sensor connected to the input, can be one from many types as NTC I various models and PT1000. The temperature measured is often used as input to the comfort space controller object, controlling the space temperature in a zone.

| Temp # | Connector # | Name in LonMaker list |
|---|---|---|
| 1 | 44.1 – 45.1 | TempSensor_1 |
| 2 | 44.2 – 45.2 | TempSensor_2 |

**Table 1 The software and hardware relation**

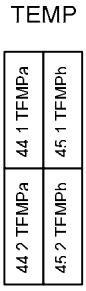



**Figure 7 The temperature hardware input**

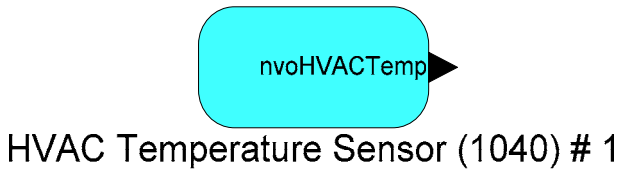



**Figure 31 The temperature sensor object**

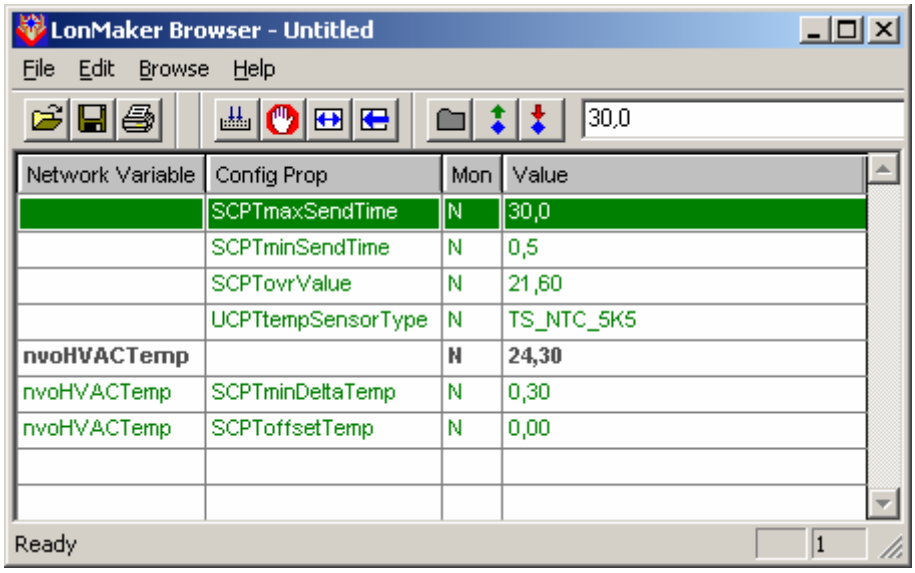



**Figure 32 Browsing the temperature sensor object**

### 1.3.1.1 Network Variable Interface

| Variable name | Default Value | Type | Description |
|---|---|---|---|
| nvoHVACTemp | | | |

#### *1.3.1.1.1 nvoHVACTemp*

This output network variable is used to source a temperature signal to a control object as a space comfort controller.

### 1.3.1.2 Configuration Properties

| Configuration | Default Value | Description |
|---|---|---|
| SCPTmaxSendTime (49) | 30,0 | Heartbeat maximum send time |
| SCPTminSendTime (52) | 0,5 | Heartbeat minimum send time |
| SCPTovrValue (33) | 22,0 | Temperature value when overwrite |
| SCPTminDeltaTemp ( ) | 0,30 | Temp. change before variable send |
| SCPToffsetTemp ( ) | 0,00 | Temp. hardware offset adjustment |
| | | |
| UCPTtempSensorType ( ) | TS_NTC_1K8 | Temperature type selection |

#### *1.3.1.2.1 SCPTmaxSendTime*

This configuration property set the value for the send heartbeat. Maximum period of time that expires before the object automatically transmits the present value of the lux level output NV.

#### *1.3.1.2.2 SCPTminSendTime*

This configuration property set the value for the minimum send time. Minimum period between output NV transmissions (maximum transmission rate).

#### *1.3.1.2.3 SCPTovrValue*

This configuration property set the value for the output variable, when the object is in override mode. The variable type is a SNVT_temp ( ). The value is used on the nvoHVACTemp variable from the object.

#### *1.3.1.2.4 SCPTminDeltaTemp*

This configuration property is used to configure how large a temperature change, that will activate an event sending out an update on the nvoHVACTemp variable. The value is in

#### *1.3.1.2.5 SCPToffsetTemp*

This configuration property type is used the adjust the hardware sensor connected to the comfort and light controller temperature input and compensate the temperature failure on the sensor. The value is subtracted or added from the measured sensor value.

#### *1.3.1.2.6 UCPTtempSensorType*

This configuration property type is used to select the hardware temperature sensor type connected to the comfort and light controller.

| **UCPTtempSensorType** | **Index #** | **Description** |
|---|---|---|
| TS_NTC_1K8 | | NTC sensor 1K8 |
| TS_NTC_5K5 | | NTC sensor 5K5 |
| TS_NTC_5K5_GROUNDED | | NTC 5K5 one grounded wire |
| TS_NTC_1015_0_60 | | NTC 0 to 30 degrees Celsius |
| TS_NTC_1015_30_30 | | NTC -30 to 30 degrees Celsius |
| TS_NTC_1015_0_30 | | NTC 0 to 30 degrees Celsius |
| TS_PTC_1000 | | Standard PT1000 sensor |

### 1.3.2 Space Comfort Controller (8500), 2 objects

The following section is describing the space comfort controller software object. This object is a temperature controller used to automatic controlling heat and or cool in a zone. The object receive a temperature on its input variable and decide if there is need for heat or cool. The decision is taken with reference to a set of set points configured in the object. Hereafter the controller will use a PI function controlling either heat or cool output.

The space comfort controller has both a primary heat and cool output and a secondary heat and cool output. This gives many possibilities when configuring the controller functions. In cooling situations the ventilation can take the first stage for cooling and a real cooler unit can take the second stage if the first stage cannot cool down, to the set point used.

The controller is designed to work with partition walls, combining two separate controllers into one, when a partition wall is open.

The controller also have build-in compensation when no need for heat in the zone, but it‘s very cold outside. Using the outdoor temperature compared to a configured minimum level property, a minimum heat signal is send out the heat output even if there is no need for heat in the zone, giving a better space comfort, specially close to the windows area.

| Controller # | Name in LonMaker list |
|---|---|
| 1 | SpaceComfort_1 |
| 2 | SpaceComfort_2 |


**Figure 33 The space comfort controller**


| Network Variable | Config Prop | Mon | Value |
|---|---|---|---|
| | SCPTholdTime | N | 600,0 |
| | SCPThvacType | N | HVT_GENERIC |
| | SCPTmaxRcvTime | N | 250,0 |
| | SCPTmaxSendTime | N | 120,0 |
| | SCPTsetPnts | N | 23,00,25,00,28,00,21,00,19,00,16,00 |
| | UCPTcoolGain | N | 10,0000 |
| | UCPTcoolITime | N | 600,0 |
| | UCPTcoolValveMotionT | N | 10080 |
| | UCPTenergyHoldMode | N | EHM_NORMAL |
| | UCPTheatGain | N | 10,0000 |
| | UCPTheatITime | N | 600,0 |
| | UCPTheatValveMotionT | N | 10080 |
| | UCPTminCool | N | 0,000;0,000;0,000 |
| | UCPTminHeat | N | (-10,00;30,000),(10,00;0,000) |
| | UCPTminHeatInStandby | N | BOOL_TRUE |
| **nviApplicMode** | | **N** | **HVAC_AUTO** |
| **nviCoolPrSlve** | | **N** | **0,000** |
| **nviCoolSeSlve** | | **N** | **0,000** |
| **nviEnrgyHldOf** | | **N** | **0,0 0** |
| **nviHeatPrSlve** | | **N** | **0,000** |
| **nviHeatSeSlve** | | **N** | **0,000** |
| **nviOccSched** | | **N** | **OC_OCCUPIED,OC_OCCUPIED,0** |
| **nviOccSensor** | | **N** | **OC_OCCUPIED** |
| **nviOutdoorTmp** | | **N** | **327,67** |
| **nviSetpoint** | | **N** | **327,67** |
| **nviSetptOffst** | | **N** | **0,00** |
| **nviSlave** | | **N** | **0,0 0** |
| **nviSpaceTemp** | | **N** | **327,67** |
| **nviValveOvrid** | | **N** | **HVO_OFF,0,000,0** |
| **nvoCoolPrimry** | | **N** | **0,000** |
| nvoCoolPrimry | UCPTcoolPrimPart | N | 100,0 |
| **nvoCoolSecond** | | **N** | **0,000** |
| nvoCoolSecond | UCPTcoolSecPart | N | 0,0 |
| **nvoEffOccup** | | **N** | **OC_OCCUPIED** |
| **nvoEffSetpt** | | **N** | **0,00** |
| **nvoHeatCool** | | **N** | **HVAC_AUTO** |
| **nvoHeatPrimry** | | **N** | **0,000** |
| nvoHeatPrimry | UCPTheatPrimPart | N | 100,0 |
| **nvoHeatSecond** | | **N** | **0,000** |
| nvoHeatSecond | UCPTheatSecPart | N | 0,0 |
| **nvoSetpoint** | | **N** | **0,00** |
| **nvoSpaceTemp** | | **N** | **327,67** |
| **nvoUnitStatus** | | **N** | **HVAC_AUTO,0,000,0,000,0,000,0,000,0,000,0** |

**Figure 34 Browsing the space comfort controller object**

### 1.3.2.1 Network Variable Interface

| Variable name | Default Value | Type | Description |
|---|---|---|---|
| nviApplicMode | HVAC_AUTO | SNVT_hvac_mode (108) | Application mode input. |
| nviCoolPrSlve | 0,0 | SNVT_lev_percent (81) | Cool slave input primary |
| nviCoolSeSlve | 0,0 | SNVT_lev_percent (81) | Cool slave input secondary |
| nviEnrgyHldOf | 0,0 0 | SNVT_switch (95) | Energy hold-off input |
| nviHeatPrSlve | 0,0 | SNVT_lev_percent (81) | Heat slave input primary |
| nviHeatSeSlve | 0,0 | SNVT_lev_percent (81) | Heat slave input secondary |
| nviOccSched | OC_OCCUPIED, OC_OCCUPIED,0 | SNVT_tod_event (128) | Operation mode selection |
| nviOccSensor | OC_OCCUPIED | SNVT_occupancy (109) | Occupancy sensor input |
| nviOutdoorTmp | 327,067 | SNVT_temp_p (105) | Outdoor temperature |
| nviSetpoint | 327,067 | SNVT_temp_p (105) | Absolute set point |
| nviSetptOffst | 0,0 | SNVT_temp_p (105) | Set point offset value |
| nviSlave | 0,0 0 | | |
| nviSpaceTemp | 327,067 | SNVT_temp_p (105) | Temperature input |
| nviValveOvrid | HVO_OFF,0,000,0 | SNVT_hvac_overid (111) | Valve override input |
| | | | |
| nvoCoolPrimary | 0,0 | SNVT_lev_percent (81) | Primary cool out |
| nvoCoolSecond | 0,0 | SNVT_lev_percent (81) | Secondary cool out |
| nvoEffOccup | OC_OCCUPIED | SNVT_occupancy (109) | Effective occupancy out |
| nvoEffSetpt | 0,0 | SNVT_temp_p (105) | Effective setpoint output |
| nvoHeatCool | HVAC_AUTO | SNVT_hvac_mode (108) | Effective heat/cool out |
| nvoHeatPrimary | 0,0 | SNVT_lev_percent (81) | Primary heat out |
| nvoHeatSencond | 0,0 | SNVT_lev_percent (81) | Secondary heat out |
| nvoSetpoint | 0,0 | SNVT_temp_p (105) | Local setpoint output |
| nvoSpaceTemp | 327,067 | SNVT_temp_p (105) | Effective temp. output |
| nvoUnitStatus | HVAC_AUTO, 0,000,0,000, 0,000,0,000, 0,000,0 | SNVT_hvac_status (112) | Unit status output |

#### *1.3.2.1.1 nviApplicMode*

Application mode input. Used to coordinate the SCC with any supervisory controller.

#### *1.3.2.1.2 nviCoolPrSlve*

Slave input used when in slave mode. This value will be send out on the primary cool output, when the object is forced into slave mode, with the variable SNVT_switch on the nviSlave input.

#### *1.3.2.1.3 nviCoolSeSlve*

Slave input used when in slave mode. This value will be send out on the secondary cool output, when the object is forced into slave mode, with the variable SNVT_switch on the nviSlave input.

#### *1.3.2.1.4 nviEnrgyHldOf*

Energy hold-off input. This input is used to stop heating and cooling while allowing the unit to protect the space from temperature extremes.

#### *1.3.2.1.5 nviHeatPrSlve*

Slave input used when in slave mode. This value will be send out on the primary heat output, when the object is forced into slave mode, with the variable SNVT_switch on the nviSlave input.

#### *1.3.2.1.6 nviHeatSeSlve*

Slave input used when in slave mode. This value will be send out on the secondary heat output, when the object is forced into slave mode, with the variable SNVT_switch on the nviSlave input.

#### *1.3.2.1.7 nviOccSched*

Occupancy scheduler input. Command the SCC into different occupancy modes using the SNVT_tod_event structure.

##### 1.3.2.1.7.1 SNVT_tod_event struture:

| **Field 1** | **Field 2** | **Field 3** |
|---|---|---|
| current_state | next_state | time_to_next_state |

Tod event field description

| **Field** | **Description** | **Refference** |
|---|---|---|
| 1 current_state | Occupancy, current | Occup_t table |
| 2 next_state | Occupancy, next | Occup_t table |
| 3 time_to_next_state | Time to next state | 0 to 65535 minutes |

1.3.2.1.7.1.1 Occup_t, Occupancy code names table:

| **Occup_t** | ***Index*** |
|---|---|
| OC_NUL | *-1* |
| OC_OCCUPIED | *0* |
| OC_UNOCCUPIED | *1* |
| OC_BYPASS | *2* |
| OC_STANDBY | 3 |

#### *1.3.2.1.8 nviOccSensor*

Occupancy sensor input. Indicate the presence of occupants in the controlled space/zone. This input start the hold timer and count down from the set value in the hold time property SCPTholdTime. The variable is SNVT_occupancy.

#### *1.3.2.1.9 nviOutdoorTmp*

Outdoor air temperature input. Represents information from an outdoor air temperature sensor. Is used to calculate minimum heat, compensation for chilled windows or other building parts, when the outdoor temperature is very low.

#### *1.3.2.1.10 nviSetpoint*

Absolute temperature setpoint input. Allow the temperature set points for the occupied and standby modes to be changed via the network directly.

#### *1.3.2.1.11 nviSetptOffst*

Setpoint offset input. Shifts the effective occupied and standby temperature setpoints by adding this value to the present setpoints.

#### *1.3.2.1.12 nviSlave*

#### *1.3.2.1.13 nviSpaceTemp*

Space temperature input. Connect an external space temperature sensor to the object.

#### *1.3.2.1.14 nviValveOvrid*

Water valve override input. Command the controller into a manual mode for overriding water valves using the SNVT_hvac_ove structure.

##### 1.3.2.1.14.1 SNVT_hvac_ove struture:

| Field 1 | Field 2 | Field 3 |
|---|---|---|
| state | percent | flow |

HVAC ove field description

| Field | Description | Refference |
|---|---|---|
| 1 state | Occupancy, current | hvac_overd_t table |
| 2 percent | Valve value | % full scale |
| 3 Flow | Valve flow | 0 to 65535 |

hvac_overid_t, HVAC override state table:

The following function is implemented in the object.

| hvac_overid_t | *Index* |
|---|---|
| HVO_OFF,0,000,0 | 0 |

#### *1.3.2.1.15 nvoCoolPrimary*

Primary cool output variable. This variable is a SNVT_lev_percent and is used to control a cool actuator object.

#### *1.3.2.1.16 nvoCoolSecond*

Secondary cool output variable. This variable is a SNVT_lev_percent and is used to control a cool actuator object.

#### *1.3.2.1.17 nvoEffOccup*

Effective occupancy output. Actual occupancy mode of the unit

#### *1.3.2.1.18 nvoEffSetpt*

Effective setpoint output. Monitors the effective temperature setpoint and send it out on the network.

#### *1.3.2.1.19 nvoHeatCool*

Effective heat or cool output. Actual heat or cool mode of the unit.

#### *1.3.2.1.20 nvoHeatPrimary*

Primary heat output variable. This variable is a SNVT_lev_percent and is used to control a heat actuator object.

#### *1.3.2.1.21 nvoHeatSencond*

Secondary heat output variable. This variable is a SNVT_lev_percent and is used to control a heat actuator object.

#### *1.3.2.1.22 nvoSetpoint*

Local setpoint output. Space temperature setpoint value if a setpoint device is hardwired.

#### *1.3.2.1.23 nvoSpaceTemp*

Effective space temperature output. Used to monitor the effective space temperature that the SCC is using for control.

#### *1.3.2.1.24 nvoUnitStatus*

Unit status output. Reports the SCC status. This output variable is a structure giving a total status for the space comfort controller:
*mode, heat_output_primary, heat_output_secondary, cool_output, econ_output, fan_output, in_alarm*

Structure fields

| **# 1** | **# 2** | **# 3** | **# 4** | **# 5** | **# 6** | **# 7** |
|---|---|---|---|---|---|---|
| Mode | heat_output_primary | heat_output_secondary | cool_output | econ_output | fan_output | in_alarm |

Fields descriptions

| Field | Description | Refference |
|---|---|---|
| 1 mode | HVAC mode names | **hvac_t** |
| 2 heat_output_primary | Primary heat | % of full scale |
| 3 heat_output_secondary | Secondary heat | % of full scale |
| 4 cool_output | Cool output | % of full scale |
| 5 econ_output | Economizer output | % of full scale |
| 6 fan_output | Fan output | % of full scale |
| 7 in_alarm | In alarm state | alarm value |

hvac_t description table:

| hvac_t | | Index # |
|---|---|---|
| HVAC_NUL | | -1 |
| HVAC_AUTO | | 0 |
| HVAC_HEAT | | 1 |
| HVAC_MRNG_WRMUP | | 2 |
| HVAC_COOL | | 3 |
| HVAC_NIGHT_PURGE | | 4 |
| HVAC_PRE_COOL | | 5 |
| HVAC_OFF | | 6 |
| HVAC_TEST | | 7 |
| HVAC_EMERG_HEAT | | 8 |
| HVAC_FAN_ONLY | | 9 |
| HVAC_FREE_COOL | | 10 |
| HVAC_ICE | | 11 |
| HVAC_MAX_HEAT | | 12 |
| HVAC_ECONOMY | | 13 |
| HVAC_DEHUMID | | 14 |
| HVAC_CALIBRATE | | 15 |
| HVAC_EMERG_COOL | | 16 |
| HVAC_EMERG_STEAM | | 17 |

### 1.3.2.2 Configuration Properties

| Configuration | Default Value | Description |
|---|---|---|
| SCPTholdTime | 600,0 | Occupancy timer value |
| SCPThvacType (169) | HVT_GENERIC | HVAC unit type identifier |
| SCPTmaxRcvTime | 250,0 | Heartbeat maximum receive time |
| SCPTmaxSendTime | 120,0 | Heartbeat maximum send time |
| SCPTsetPnts | 23,00,25,00,28,00,21,00, 19,00,16,00 | Set points for heat and cool |
| | | |
| UCPTcoolGain | 10,0000 | PI gain cool in procent |
| UCPTcoolTime | 600,0 | PI time cool in seconds |
| UCPTcoolValveMotionT | 10080 | Interval between valve motion |
| UCPTenergyHoldMode | EHM_NORMAL | Energy hold mode selection |
| UCPTheatGain | 10,0000 | PI gain heat in procent |
| UCPTheatITime | 600,0 | PI time heat in seconds |
| UCPTheatValveMotionT | 10080 | Interval between valve motion |
| UCPTminCool | 0,000;0,000;0,000 | Minimum cool level selection |
| UCPTminHeat | (-10,00;30,000),(10,00;0,000) | Minimum heat level selection |
| UCPTminHeatInStandby | BOOL_TRUE | Heat standby function |
| UCPTcoolPrimPart | 100,0 | Not implemented |
| UCPTcoolSecPart | 0,0 | Not implemented |
| UCPTheatPrimPart | 100,0 | Not implemented |
| UCPTheatSecPart | 0,0 | Not implemented |

#### *1.3.2.2.1 SCPTholdTime*

This configuration property is used to set the timer value for the occupancy timer. This timer is used to hold the space/zone occupied after the last occupancy input. The timer is described in seconds.

#### *1.3.2.2.2 SCPThvacType*

HVAC unit type identifier. This value is set by the manufacturer to allow an integrator to know the function of this SCC device. The default value is HVT_GENERIC.

| SCPThvacType | Index # |
|---|---|
| HVT_NUL | -1 |
| HVT_GENERIC | 0 |
| HVT_FAN_COIL | 1 |
| HVT_VAV | 2 |
| HVT_HEAT_PUMP | 3 |
| HVT_ROOFTOP | 4 |
| HVT_UNIT_VENT | 5 |
| HVT_CHILL_CEIL | 6 |
| HVT_RADIATOR | 7 |
| HVT_AHU | 8 |
| HVT_SELF_CONT | 9 |

#### *1.3.2.2.3 SCPTmaxRcvTime*

Maximum receive time. The maximum period of time that may expire with no updates on the associated input network variables before the object goes into heartbeat failure mode. A zero value disables. This configuration property is used for the SNVT input

#### *1.3.2.2.4 SCPTmaxSendTime*

Send heartbeat. Maximum period of time that expires before the specified NV outputs will automatically be updated again.

#### *1.3.2.2.5 SCPTsetPnts*

This configuration property is a structure used to set the setpoints for heat and cool, in occupied, unoccupied and standby mode.

Default values

| Field 1 | Field 2 | Field 3 | Field 4 | Field 5 | Field 6 |
|---|---|---|---|---|---|
| 23,00 | 25,00 | 28,00 | 21,00 | 19,00 | 16,00 |

Setpoints structure:

| Field 1 | Field 2 | Field 3 | Field 4 | Field 5 | Field 6 |
|---|---|---|---|---|---|
| Occupied_cool | standby_cool | unoccupied_cool | occupied_heat | standby_heat | unoccupied_heat |

Field description:

| Field # | Description |
|---|---|
| 1 occupied_cool | *Occupied cooling setpoint. degrees Celsius* |
| 2 standby_cool | *Standby cooling setpoint. degrees Celsius* |
| 3 unoccupied_cool | *Unoccupied cooling setpoint. degrees Celsius* |
| 4 occupied_heat | *Occupied heating setpoint. degrees Celsius* |
| 5 standby_heat | *Standby heating setpoint. degrees Celsius* |
| *6 unoccupied_heat* | *Unoccupied heating setpoint.degrees Celsius* |

The setpoints are used depending on the occupied status as below:

| Effective Occupied | | | |
|---|---|---|---|
| | | Calender | |
| | | Occupied (day) | OC_NUL (night) |
| Sensor | Occupied | Occupied | Occupied |
| | Unoccupied | Standby | Unoccupied |

#### *1.3.2.2.6 UCPTcoolGain*

This configuration property is used to select the gain value in the PI regulator for the cooling part.

#### *1.3.2.2.7 UCPTcooIlTime*

This configuration property is used to select the integral time value in the PI regulator for the cooling part.

#### *1.3.2.2.8 UCPTcoolValveMotionT*

This configuration property is used to select the interval between aa automatic valve motion on the cool valves.

#### *1.3.2.2.9 UCPTenergyHoldMode*

This configuration property is used to select the type of energy hold, that is used when the space comfort controller is in energy hold mode.

#### *1.3.2.2.10 UCPTheatGain*

This configuration property is used to select the gain value in the PI regulator for the heating part.

#### *1.3.2.2.11 UCPTheatITime*

This configuration property is used to select the integral time value in the PI regulator for the heating part.

#### *1.3.2.2.12 UCPTheatValveMotionT*

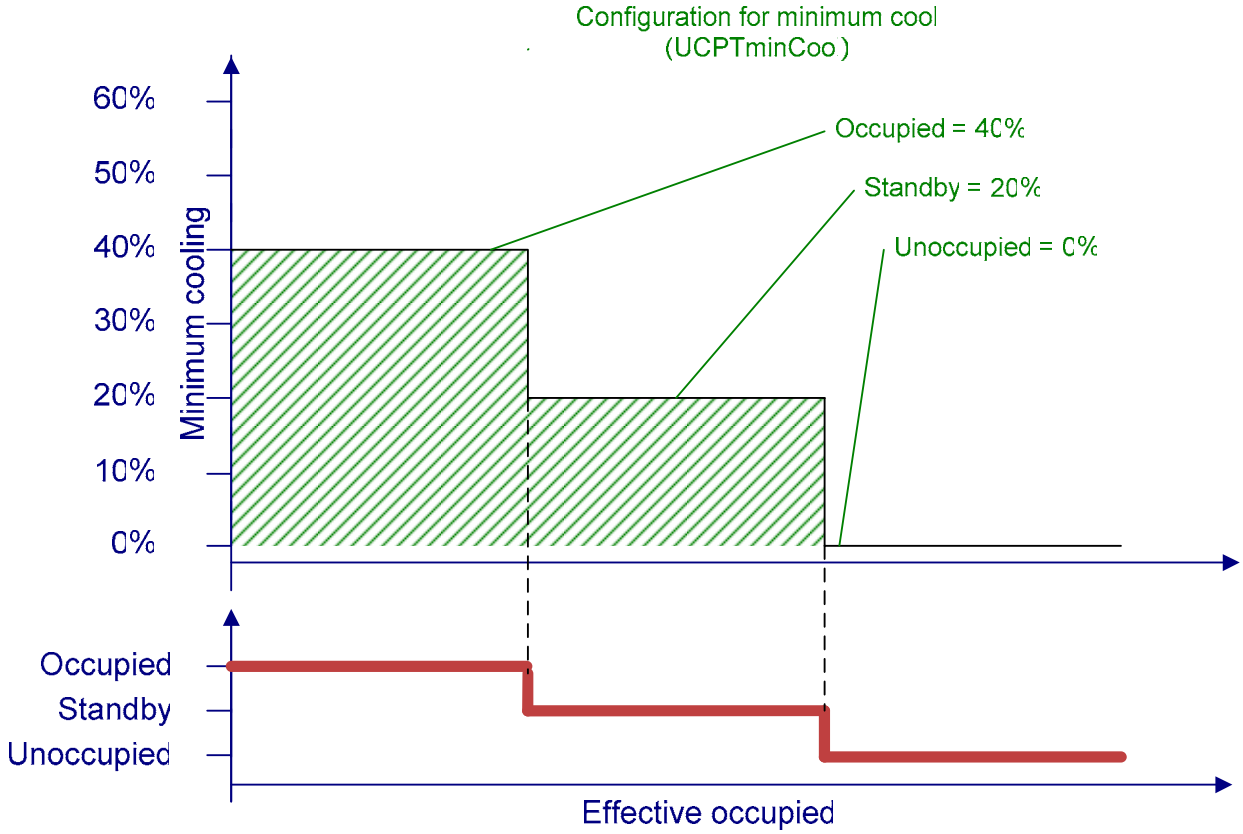
This configuration property is used to select the interval between aa automatic valve motion on the heat valves.

#### *1.3.2.2.13 UCPTminCool*

The configuration property UCPTminCool is used to configure the minimum cool output level on each state for the space comfort controller. The property is a s structure:

| Field 1 | Field 2 | Field 3 |
|---|---|---|
| occupied_cool | standby_cool | unoccupied_cool |

Please see the following figure to understand the function of the property.



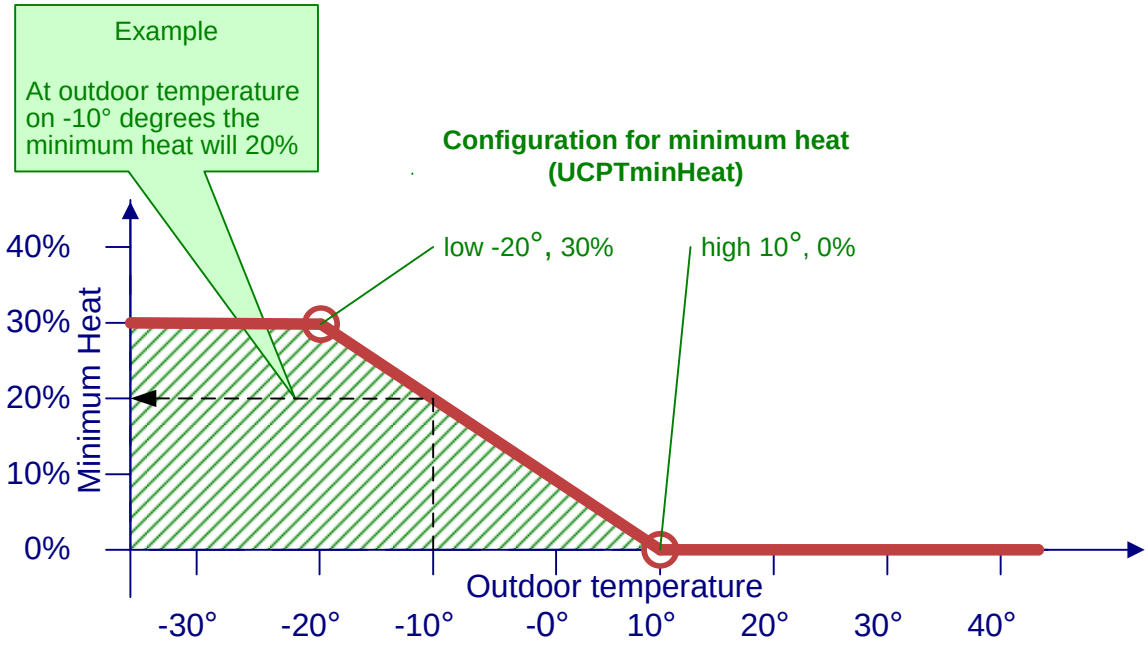

#### *1.3.2.2.14 UCPTminHeat*

The configuration property UCPTminHeat is used to configure the minimum heat output levels depending on the outdoor temperature, for the space comfort controller. The property is a s structure:

The structure:

| Field 1 | Field 2 | Field 3 | Field 4 |
|---|---|---|---|
| | | | |

Please see the following figure to understand the function of the property.




### *1.3.2.2.15 UCPTminHeatInStandby*

This configuration property is used to select if minimum heat is used during standby or not.

| UCPTminHeatInStandby | Description |
|---|---|
| BOOL_TRUE | Minimum heat on during standby |
| BOOL_FALSE | Minimum heat not on during standby |

#### *1.3.2.2.16 UCPTcoolPrimPart*

No function

#### *1.3.2.2.17 UCPTcoolSecPart*

No function

#### *1.3.2.2.18 UCPTheatPrimPart*

No function

#### *1.3.2.2.19 UCPTheatSecPart*

No function

### 1.3.3 Heat Actuator (3:20011), 2 objects

The following describes the heat actuator object. The object is used to control a hardware actuator for heat valves. The object is used to control both a digital, analog and floating actuators. The hardware output is either 24Vac ON/OFF or a 0 to 10Vdc signal with fixed 24Vac supply.

| Heat # | AC # | 0-10 # | Name in LonMaker list |
|---|---|---|---|
| 1 | 66.1 | 64.1 | Heat_1 |
| 2 | 66.2 | 64.2 | Heat_2 |

**Table 2 The heat software hardware relation**

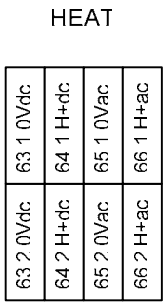



**Figure 8 The heat hardware outputs**

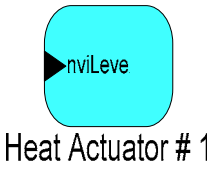



**Figure 35 The heat actuator object**

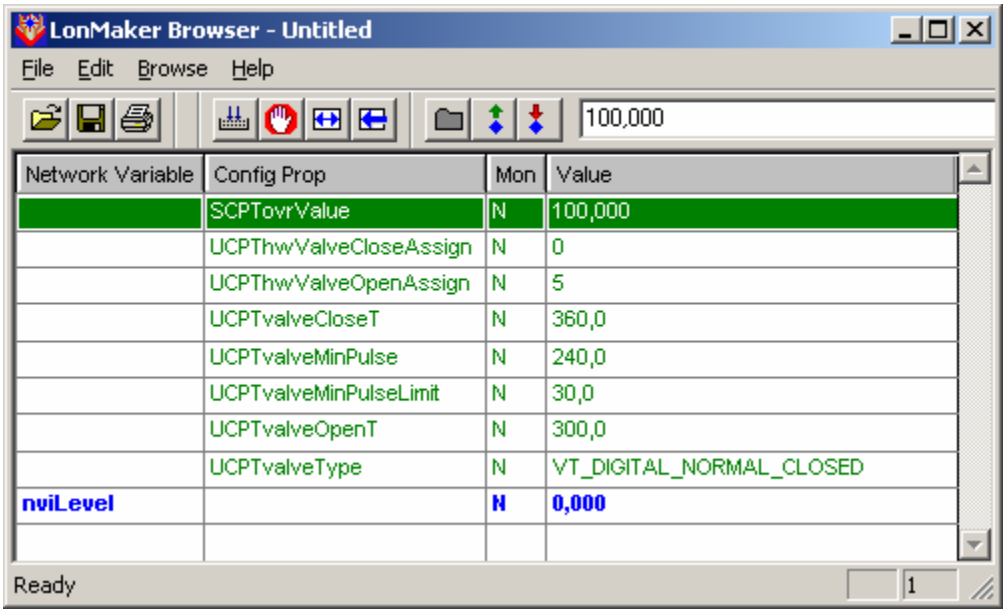



**Figure 36 Bowsing the heat actuator object**

The heat actuator object receive a control level in % and with help from the configuration properties a calculation is made, to decide how to actuate the hardware outputs.

When the object is used as a digital actuator control, the hardware output is turned ON and OFF with an interval related to the control level in %, giving approximately an average flow trough the valve matching the control input % level.

When the control input is under the valve minimum pulse limit level, the algorithm will calculate the pause length between the ON pulses and use a fixed ON length pulse using the valve minimum pulse with a security margin.

When the control input is over the valve minimum pulse limit level, the algorithm will calculate a duty cycle modulation using the valve open, close and minimum time compensating for the slow reaction of the actuator. This will generate an average flow trough the valve, matching the control input % level.

When the object is used as an analog actuator control, the hardware output is modulated to a 0 to 10 Vdc signal related to the control input in %, giving approximately a flow trough the valve matching the control input % level. In this configuration the digital hardware output can be configured as the 24Vac supply for the 0 to 10 Vdc actuator.

### 1.3.3.1 Network Variable Interface

| Variable name | Default Value | Type | Description |
|---|---|---|---|
| nviValueLevel | 0 % | SNVT_lev_percent (82) | Input control value |

#### *1.3.3.1.1 nviValueLevel*

This input network variable is the control level for the hardware actuator. It will control the actuator level in % of the full actuator range from 0 to 100%.

### 1.3.3.2 Configuration Properties

| Configuration | Default Value | Description |
|---|---|---|
| SCPTovrValue (33) | 100 % | SNVT overwrite value |
| UCPThwValveCloseAssign (3:71) | 0 | Select output hardware |
| UCPThwValveOpenAssign (3:70) | 5 or 6 | Select output hardware |
| UCPTvalveCIoseT (3:43) | 360 sec. | Actuator closing time |
| UCPTvalveMinPulse (3:44) | 240 sec. | Minimum reaction time |
| UCPTvalveMinPulseLimit (3:72) | 30 sec. | Level between algorithms |
| UCPTvalveOpenT (3:42) | 300 sec. | Actuator opening time |
| UCPTvalveType (3:73) | VT_DIGITAL NORMAL CLOSED | Actuator type selection |

#### *1.3.3.2.1 SCPTovrValue*

This configuration property set the level on the hardware output, when the object is in override mode. The variable type is a SNVT_lev_percent (82). The value is used on the hardware output.

#### *1.3.3.2.2 UCPThwValveCloseAssign*

UCPThwValveCloseAssign is shorted from, hardware valve close assignment configuration property.

The hwValveCloseAssign configuration is assigning a digital hardware output the heat actuator object. This hardware is used in floating configuration for the close direction.

#### *1.3.3.2.3 UCPThwValveOpenAssign*

UCPThwValveOpenAssign is shorted from, hardware valve open assignment configuration property. The hwValveOpenAssign configuration is assigning a digital hardware output the heat actuator object.

The following list describe the typically relation between the hardware and the software objects. The outputs are not fixed coded in the software to a specific hardware connector and can be assigned to another output due to flexibility and the possibility to use either digital, analog or floating control of the actuators.

| Output # | 24Vac connector # | Used in object |
|---|---|---|
| 1 | 57.1 | Vent 1 Open |
| 2 | 57.2 | Vent 2 Open |
| 3 | 58.1 | Vent 1 Close |
| 4 | 58.2 | Vent 2 Close |
| **5** | **66.1** | **Heat 1** |
| **6** | **66.2** | **Heat 2** |
| 7 | 62.1 | Cool 1 |
| 8 | 62.2 | Cool 2 |

#### *1.3.3.2.4 UCPTvalveCIoseT*

UCPTvalveCloseT is shorted from, valve close time configuration property.

The UCPTvalveCloseT configuration is specifying the speed the actuator can close the valve with. The closing time is used in the algorithm to compensate for the slow actuator speed and make sure that the actuator control the valve best possible. Consult the actuator documentation for getting the correct value.

#### *1.3.3.2.5 UCPTvalveMinPulse*

UCPTvalveMinPulse is shorted from, valve minimum pulse length configuration property.

The UCPTvalveMinPulse configuration is specifying the minimum pulse length possible, getting a water flow in the valve, even if the actuator are not opening 100%. If you can find this value, the algorithm can calculate a better control output. This will also give possibility to use less water, when compensating for cold windows surfaces.

#### *1.3.3.2.6 UCPTvalveMinPulseLimit*

UCPTvalveMinPulseLimit is shorted from, valve minimum pulse limit configuration property. The UCPTvalveMinPulseLimit configuration property is specifying the input level to the object where the software change from one algorithm to another. Under the limit property level, is used a fixed valve ON time and the pause is calculated. Over the property limit level, is used a calculated ON time and a calculated pause time.

#### *1.3.3.2.7 UCPTvalveOpenT*

UCPTvalveOpenT is shorted from, valve open time configuration property.

The UCPTvalveOpenT configuration is specifying the speed the actuator can open the valve with. The opening time is used in the algorithm to compensate for the slow actuator speed and make sure that the actuator control the valve best possible. Consult the actuator documentation for getting the correct value

#### *1.3.3.2.8 UCPTvalveType*

UCPTvalveType is shorted from, valve actuator type configuration property.

The following selections are possible:

| **UCPTvalveType** | **Description** |
|---|---|
| VT_DIGITAL NORMAL CLOSED | 24Vac ON/OFF, normal closed actuator |
| VT_DIGITAL NORMAL OPEN | 24Vac ON/OFF, normal open actuator |
| | |
| VT_ANALOG NORMAL CLOSED | 0 to 10Vdc, normal closed actuator |
| VT_ANALOG NORMAL OPEN | 0 to 10Vdc, normal open actuator |
| | |
| VT_FLOATING | 24Vac Open and 24Vac Close,  drive time |

### 1.3.4 Cool Actuator (3:20011), 2 objects

The following describes the cool actuator object. The object is used to control a hardware actuator for cool valves. The object can be used on both digital, analog and flowting actuators. The hardware output is either 24Vac ON/OFF or a 0 til 10Vdc signal with fixed 24Vac supply.

| Cool # | AC # | 0-10 # | Name in LonMaker list |
|---|---|---|---|
| 1 | 62.1 | 60.1 | Cool_1 |
| 2 | 62.2 | 60.2 | Cool_2 |

**Table 3 The cool software hardware relation**

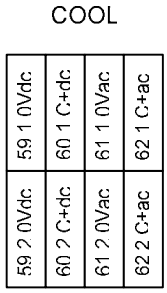



**Figure 9 The cool hardware outputs**

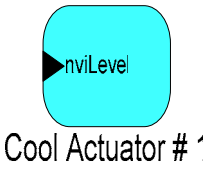



**Figure 37 The cool actuator object**

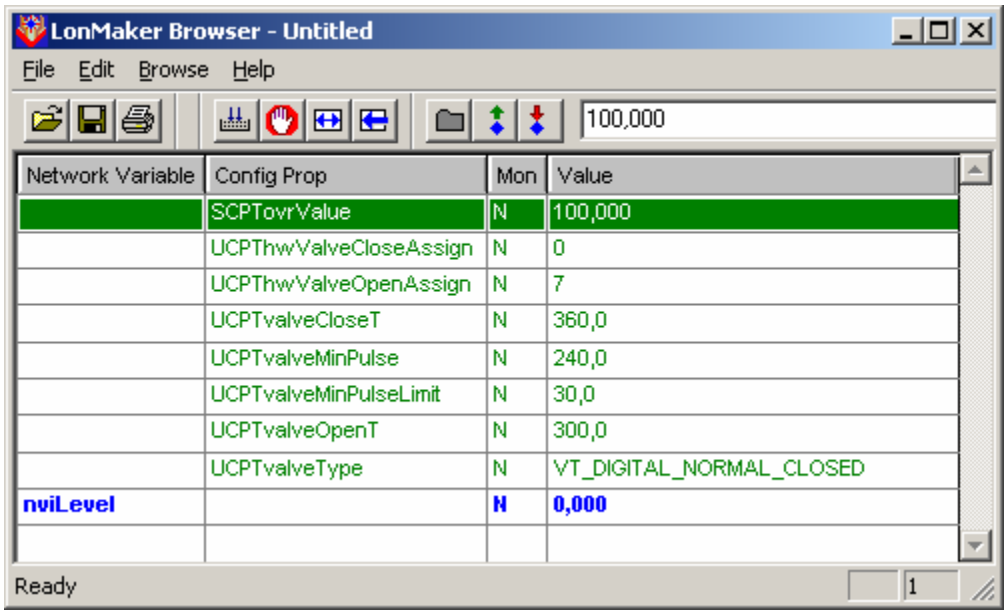



**Figure 38 Browsing the cool actuator object**

The cool actuator object receive a control level in % and with help from the configuration properties a calculation is made, to decide how to actuate the hardware outputs.

When the object is used as a digital actuator control, the hardware output is turned ON and OFF with an interval related to the control level in %, giving approximately an average flow trough the valve matching the control input % level.

When the control input is under the valve minimum pulse limit level, the algorithm will calculate the pause length between the ON pulses and use a fixed ON length pulse using the valve minimum pulse with a security margin.

When the control input is over the valve minimum pulse limit level, the algorithm will calculate a duty cycle modulation using the valve open, close and minimum time compensating for the slow reaction of the actuator. This will generate an average flow trough the valve, matching the control input % level.

When the object is used as an analog actuator control, the hardware output is modulated to a 0 to 10 Vdc signal related to the control input in %, giving approximately a flow trough the valve matching the control input % level. In this configuration the digital hardware output can be configured as the 24Vac supply for the 0 to 10 Vdc actuator.

### 1.3.4.1 Network Variable Interface

| Variable name | Default Value | Type | Description |
|---|---|---|---|
| nviValueLevel | 0 % | SNVT_lev_percent (82) | Input control value |

#### *1.3.4.1.1 nviValueLevel*

This input network variable is the control level for the hardware actuator. It will control the actuator level in % of the full actuator range from 0 to 100%.

### 1.3.4.2 Configuration Properties

| Configuration | Default Value | Description |
|---|---|---|
| SCPTovrValue (33) | 100 % | SNVT overwrite value |
| UCPThwValveCloseAssign (3:71) | 0 | Select output hardware |
| UCPThwValveOpenAssign (3:70) | 7 or 8 | Select output hardware |
| UCPTvalveCIoseT (3:43) | 360 sec. | Actuator closing time |
| UCPTvalveMinPulse (3:44) | 240 sec. | Minimum reaction time |
| UCPTvalveMinPulseLimit (3:72) | 30 sec. | Level between algorithms |
| UCPTvalveOpenT (3:42) | 300 sec. | Actuator opening time |
| UCPTvalveType (3:73) | VT_DIGITAL NORMAL CLOSED | Actuator type selection |

#### *1.3.4.2.1 SCPTovrValue*

This configuration property set the level on the hardware output, when the object is in override mode. The variable type is a SNVT_lev_percent (82). The value is used on the hardware output.

#### *1.3.4.2.2 UCPThwValveCloseAssign*

UCPThwValveCloseAssign is shorted from, hardware valve close assignment configuration property.

The hwValveCloseAssign configuration is assigning a digital hardware output the cool actuator object. This hardware is used in floating configuration for the close direction.

#### *1.3.4.2.3 UCPThwValveOpenAssign*

UCPThwValveOpenAssign is shorted from, hardware valve open assignment configuration property. The hwValveOpenAssign configuration is assigning a digital hardware output the heat actuator object.

The following list describe the typically relation between the hardware and the software objects. The outputs are not fixed coded in the software to a specific hardware connector and can be assigned to another output due to flexibility and the possibility to use either digital, analog or floating control of the actuators.

| Output # | 24Vac connector # | Used in object |
|---|---|---|
| 1 | 57.1 | Vent 1 Open |
| 2 | 57.2 | Vent 2 Open |
| 3 | 58.1 | Vent 1 Close |
| 4 | 58.2 | Vent 2 Close |
| 5 | 66.1 | Heat 1 |
| 6 | 66.2 | Heat 2 |
| **7** | **62.1** | **Cool 1** |
| **8** | **62.2** | **Cool 2** |

#### *1.3.4.2.4 UCPTvalveCIoseT*

UCPTvalveCloseT is shorted from, valve close time configuration property.

The UCPTvalveCloseT configuration is specifying the speed the actuator can close the valve with. The closing time is used in the algorithm to compensate for the slow actuator speed and make sure that the actuator control the valve best possible. Consult the actuator documentation for getting the correct value.

#### *1.3.4.2.5 UCPTvalveMinPulse*

UCPTvalveMinPulse is shorted from, valve minimum pulse length configuration property.

The UCPTvalveMinPulse configuration is specifying the minimum pulse length possible, getting a water flow in the valve, even if the actuator are not opening 100%. If you can find this value, the algorithm can calculate a better control output. This will also give possibility to use less water, when compensating for cold windows surfaces.

#### *1.3.4.2.6 UCPTvalveMinPulseLimit*

UCPTvalveMinPulseLimit is shorted from, valve minimum pulse limit configuration property. The UCPTvalveMinPulseLimit configuration property is specifying the input level to the object where the software change from one algorithm to another. Under the limit property level, is used a fixed valve ON time and the pause is calculated. Over the property limit level, is used a calculated ON time and a calculated pause time.

#### *1.3.4.2.7 UCPTvalveOpenT*

UCPTvalveOpenT is shorted from, valve open time configuration property.

The UCPTvalveOpenT configuration is specifying the speed the actuator can open the valve with. The opening time is used in the algorithm to compensate for the slow actuator speed and make sure that the actuator control the valve best possible. Consult the actuator documentation for getting the correct value

#### *1.3.4.2.8 UCPTvalveType*

UCPTvalveType is shorted from, valve actuator type configuration property.

The following selections are possible:

| UCPTvalveType | Description |
|---|---|
| VT_DIGITAL NORMAL CLOSED | 24Vac ON/OFF, normal closed actuator |
| VT_DIGITAL NORMAL OPEN | 24Vac ON/OFF, normal open actuator |
| | |
| VT_ANALOG NORMAL CLOSED | 0 to 10Vdc, normal closed actuator |
| VT_ANALOG NORMAL OPEN | 0 to 10Vdc, normal open actuator |
| | |
| VT_FLOATING | 24Vac Open and 24Vac Close, drive time |

### 1.3.5 Ventilation Actuator (3:20011), 2 objects

The following describes the ventilation actuator object. The object is used to control a hardware actuator for ventilation actuators. The object can be used on both digital, analog and flowting actuators. The hardware output is either 24Vac ON/OFF or a 0 til 10Vdc signal with fixed 24Vac supply.

| Vent # | AC # | 0-10 volt 0 volt # | 0-10 volt out | Name in LonMaker list |
|---|---|---|---|---|
| 1 | 57.1 open | 54.1 | 55.1 | Vent_1 |
| 1 | 58.1 close | | | |
| 2 | 57.2 open | 54.2 | 55.2 | Vent_2 |
| 2 | 58.2 close | | | |

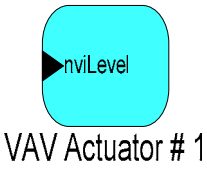



**Figure 39 The ventilation actuator object**

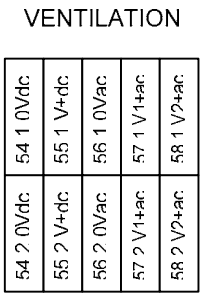



**Figure 10 The Ventilation hardware outputs**

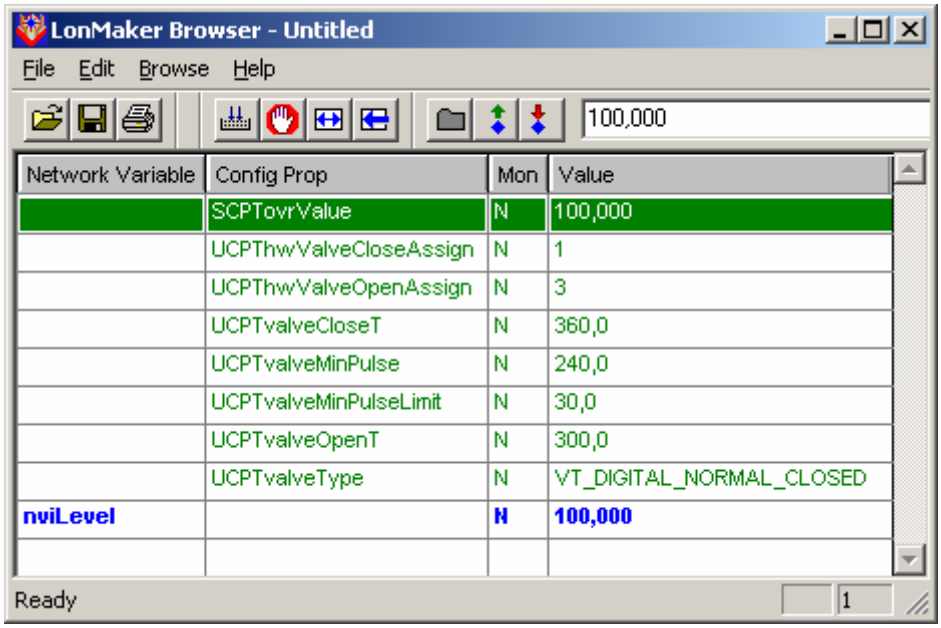



**Figure 40 Browsing the ventilation actuator object**

The ventilation actuator object receive a control level in % and with help from the configuration properties a calculation is made, to decide how to actuate the hardware outputs.

When the object is used as a digital actuator control, the hardware output is turned ON and OFF with an interval related to the control level in %, giving approximately an average flow trough the valve matching the control input % level.

When the control input is under the valve minimum pulse limit level, the algorithm will calculate the pause length between the ON pulses and use a fixed ON length pulse using the valve minimum pulse with a security margin.

When the control input is over the valve minimum pulse limit level, the algorithm will calculate a duty cycle modulation using the valve open, close and minimum time compensating for the slow reaction of the actuator. This will generate an average flow trough the valve, matching the control input % level.

When the object is used as an analog actuator control, the hardware output is modulated to a 0 to 10 Vdc signal related to the control input in %, giving approximately a flow trough the valve matching the control input % level. In this configuration the digital hardware output can be configured as the 24Vac supply for the 0 to 10 Vdc actuator.

### 1.3.5.1 Network Variable Interface

| Variable name | Default Value | Type | Description |
|---|---|---|---|
| nviValueLevel | 0 % | SNVT_lev_percent (82) | Input control value |

#### *1.3.5.1.1 nviValueLevel*

This input network variable is the control level for the hardware actuator. It will control the actuator level in % of the full actuator range from 0 to 100%.

### 1.3.5.2 Configuration Properties

| Configuration | Default Value | Description |
|---|---|---|
| SCPTovrValue (33) | 100 % | SNVT overwrite value |
| UCPThwValveCloseAssign (3:71) | 0 | Select output hardware |
| UCPThwValveOpenAssign (3:70) | 7 or 8 | Select output hardware |
| UCPTvalveCIoseT (3:43) | 360 sec. | Actuator closing time |
| UCPTvalveMinPulse (3:44) | 240 sec. | Minimum reaction time |
| UCPTvalveMinPulseLimit (3:72) | 30 sec. | Level between algorithms |
| UCPTvalveOpenT (3:42) | 300 sec. | Actuator opening time |
| UCPTvalveType (3:73) | VT_DIGITAL NORMAL CLOSED | Actuator type selection |

#### *1.3.5.2.1 SCPTovrValue*

This configuration property set the level on the hardware output, when the object is in override mode. The variable type is a SNVT_lev_percent (82). The value is used on the hardware output.

#### *1.3.5.2.2 UCPThwValveCloseAssign*

UCPThwValveCloseAssign is shorted from, hardware valve close assignment configuration property.

The hwValveCloseAssign configuration is assigning a digital hardware output the ventilation actuator object. This hardware is used in floating configuration for the close direction.

#### *1.3.5.2.3 UCPThwValveOpenAssign*

UCPThwValveOpenAssign is shorted from, hardware valve open assignment configuration property. The hwValveOpenAssign configuration is assigning a digital hardware output the heat actuator object.

The following list describe the typically relation between the hardware and the software objects. The outputs are not fixed coded in the software to a specific hardware connector and can be assigned to another output due to flexibility and the possibility to use either digital, analog or floating control of the actuators.

| Output # | 24Vac connector # | Used in object |
|---|---|---|
| **1** | **57.1** | **Vent 1 Open** |
| **2** | **57.2** | **Vent 2 Open** |
| **3** | **58.1** | **Vent 1 Close** |
| **4** | **58.2** | **Vent 2 Close** |
| 5 | 66.1 | Heat 1 |
| 6 | 66.2 | Heat 2 |
| 7 | 62.1 | Cool 1 |
| 8 | 62.2 | Cool 2 |

#### *1.3.5.2.4 UCPTvalveCIoseT*

UCPTvalveCloseT is shorted from, valve close time configuration property.

The UCPTvalveCloseT configuration is specifying the speed the actuator can close the valve with. The closing time is used in the algorithm to compensate for the slow actuator speed and make sure that the actuator control the valve best possible. Consult the actuator documentation for getting the correct value.

#### *1.3.5.2.5 UCPTvalveMinPulse*

UCPTvalveMinPulse is shorted from, valve minimum pulse length configuration property.

The UCPTvalveMinPulse configuration is specifying the minimum pulse length possible, getting a water flow in the valve, even if the actuator are not opening 100%. If you can find this value, the algorithm can calculate a better control output. This will also give possibility to use less water, when compensating for cold windows surfaces.

#### *1.3.5.2.6 UCPTvalveMinPulseLimit*

UCPTvalveMinPulseLimit is shorted from, valve minimum pulse limit configuration property. The UCPTvalveMinPulseLimit configuration property is specifying the input level to the object where the software change from one algorithm to another. Under the limit property level, is used a fixed valve ON time and the pause is calculated. Over the property limit level, is used a calculated ON time and a calculated pause time.

#### *1.3.5.2.7 UCPTvalveOpenT*

UCPTvalveOpenT is shorted from, valve open time configuration property.

The UCPTvalveOpenT configuration is specifying the speed the actuator can open the valve with. The opening time is used in the algorithm to compensate for the slow actuator speed and make sure that the actuator control the valve best possible. Consult the actuator documentation for getting the correct value

#### *1.3.5.2.8 UCPTvalveType*

UCPTvalveType is shorted from, valve actuator type configuration property.

The following selections are possible:

| **UCPTvalveType** | **Description** |
|---|---|
| VT_DIGITAL NORMAL CLOSED | 24Vac ON/OFF, normal closed actuator |
| VT_DIGITAL NORMAL OPEN | 24Vac ON/OFF, normal open actuator |
| | |
| VT_ANALOG NORMAL CLOSED | 0 to 10Vdc, normal closed actuator |
| VT_ANALOG NORMAL OPEN | 0 to 10Vdc, normal open actuator |
| | |
| VT_FLOATING | 24Vac Open and 24Vac Close,  drive time |

### 1.3.6 Indoor Air Quality Controller (3:20018), 2 objects

The following section is describing the indoor air controller software function object implemented in the Comfort and Light Controller Lonbox PZM4146.

The object is normally used controlling ventilation with reference to a measured CO2 level and a setpoint configured in the controller object.

The ventillation control output can be combined with the control output of a Space Comfort Controller using an Analog Function to select either the control value with the largest level, or the average of the two control values.


**Figure 41 The indoor air quality controller object**

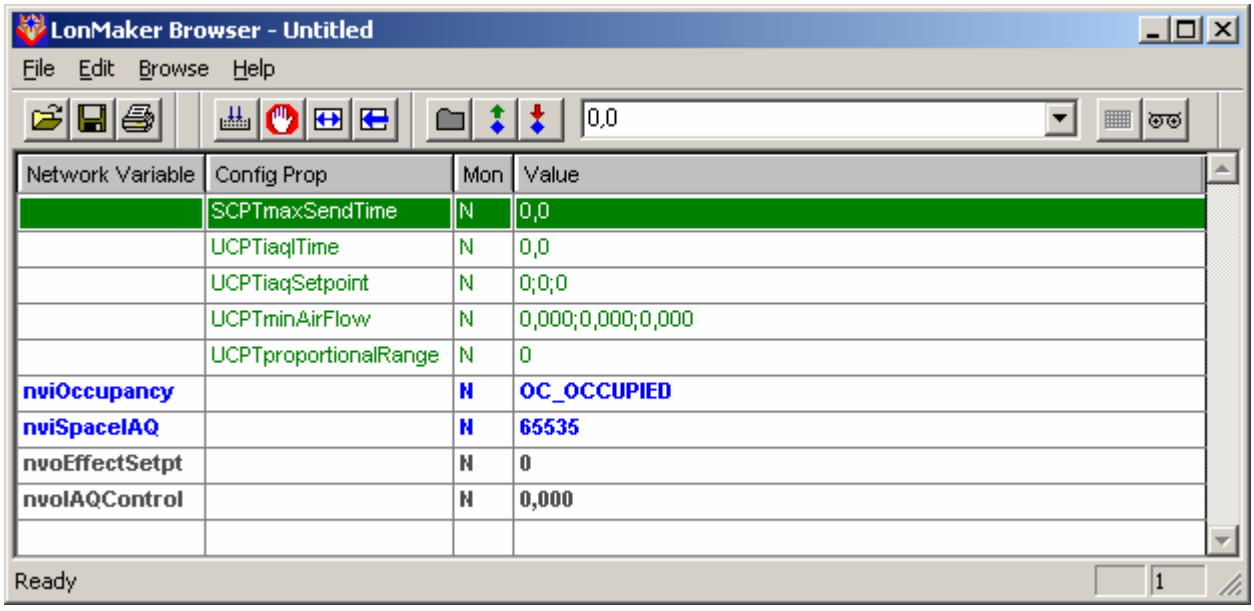



**Figure 42 Browsing the indoor air quality controller object**

#### 1.3.6.1 Network Variable Interface

| Variable name | Default Value | Type | Description |
|---|---|---|---|
| nviOccupancy | OC_UNOCCUPIED | SNVT_occupancy (109 ) | Occupancy input status |
| nviSpaceIAQ | 65353 | SNVT_ppm (29) | Air quality input level |
| nvoEffectSetpt | 0 | SNVT_ppm (29) | Output the current effective setpoint |
| nvoIAQcontrol | 0,000 | SNVT_lev_percent (82) | Control output for actuator |

#### *1.3.6.1.1 nviOccupancy*

Occupancy status input value, provides the occupancy status for the area where the controller is used.

#### *1.3.6.1.2 nviSpaceIAQ*

Air quality input from the air quality sensor, could be a CO2 sensor.

#### *1.3.6.1.3 nvoEffectSetpt*

The nvoEffectSetpt variable will reflect the current effective setpoint.

#### *1.3.6.1.4 nvoIAQcontrol*

The nvoIAQcontrol variable is the output from the controller object and is used to control an actuator object.

### 1.3.6.2 Configuration Properties

| Configuration | Default Value | Description |
|---|---|---|
| SCPTmaxSendTime | | Maximum send time |
| | | |
| UCPTiaqITime | | The regulator integration time (*i*) |
| UCPTiaqSetpoint | | PPM setpoint |
| UCPTminAirFlow | 0,000;0,000;0,000 | Minimum countrol output |
| UCPTproportionalRange | | The regulator proportional range (*p*). |

#### *1.3.6.2.1 SCPTmaxSendTime*

This configuration property set the value for the send heartbeat. Maximum period of time that expires before the object automatically transmits the present value of the nvoIAQcontrol level output.

#### *1.3.6.2.2 UCPTiaqITime*

Integration time for the ait quality PI regulator.

#### *1.3.6.2.3 UCPTiaqSetpoint*

Regulator air quality (ppm) setpoint for each of the occupancy states OCCUPIED, STANDBY and UNOCCUPIED, other states use same setpoint as specified for UNOCCUPIED.

#### *1.3.6.2.4 UCPTminAirFlow*

Minimum countrol output level (air flow) for each of the occupancy states OCCUPIED, STANDBY and UNOCCUPIED, other states use same minimum air flow as specified for UNOCCUPIED.

#### *1.3.6.2.5 UCPTproportionalRange*

Proportional range for the air quality PI regulator.

## *Sunblind profiles:*

The following list is describing the sunblind software functions implemented in the Comfort and light controller Lonbox PZM4146.

### 1.3.7 Sunblind Actuator (3:20000), 2 objects

The sunblind actuator object is controlling the hardware output for the sunblind actuator.

| Actuator # | Ma direction | Mb direction | Name in LonMaker list |
|---|---|---|---|
| 1 | 16.1 | 17.1 | SblndActuator_1 |
| 2 | 16.2 | 17.2 | SblndActuator_2 |

**Table 4 The sunblind software hardware relation**

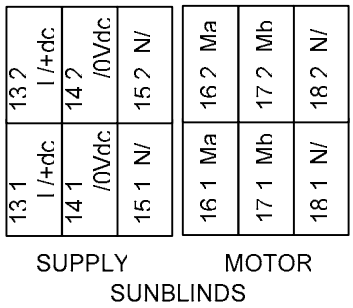



**Figure 11 The sunblibd hardware outputs**

Please consult the hardware manual for more details about AC or DC actuators.

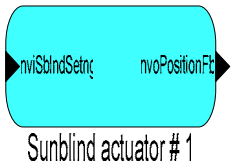



**Figure 43 The sunblind actuator object**

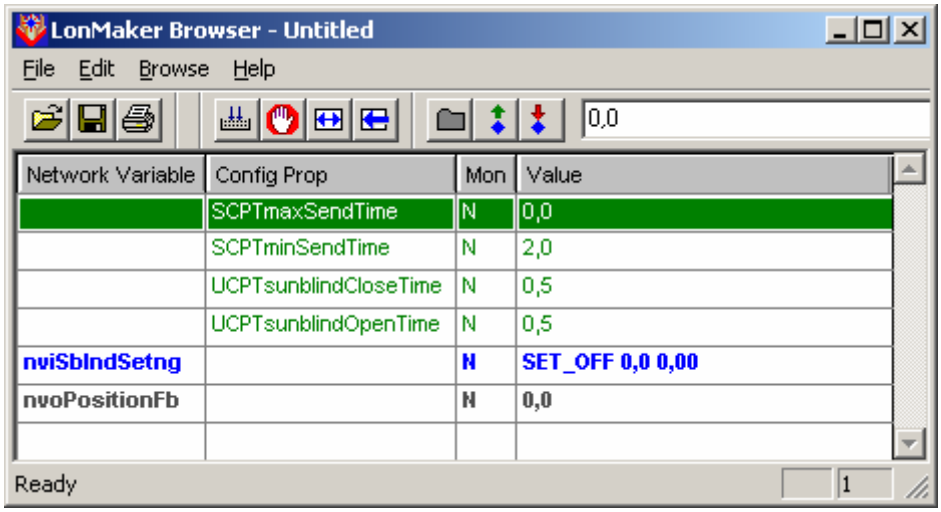



**Figure 44 Browsing the sunblind actuator object**

### 1.3.7.1 Network Variable Interface

| Variable name | Default Value | Type | Description |
|---|---|---|---|
| nviSblndSetng | SET_OFF 0,0 0,00 | SNVT_setting (117) | Setting control |
| nvoPositionFb | 0,0 | SNVT_lev_cont | Output the position |

#### *1.3.7.1.1 nviSblndSetting*

Control input for the sunblind actuator. The variable is a SNVT_setting, this is at structure.

The structure:
(function, setting, rotation)

#### *1.3.7.1.2 nvoPositionFb*

This output variable is used to display the position of the sunblind actuator. The variable is a SNVT_lev_cont, (% of full level).

### 1.3.7.2 Configuration Properties

| Configuration | Default Value | Description |
|---|---|---|
| SCPTmaxSendTime (49) | 0 | Maximum send time |
| SCPTminSendTime (52) | 2 | Minimum send time |
| | | |
| UCPTsunblindCIoseTime | 0,5 | Drive time close |
| UCPTsunblindOpenTime | 0,5 | Drive time open |

#### *1.3.7.2.1 SCPTmaxSendTime*

This configuration property set the value for the send heartbeat. Maximum period of time that expires before the object automatically transmits the present value of the nvoPositionFb level output.

#### *1.3.7.2.2 SCPTminSendTime*

This configuration property set the value for the minimum send time. Minimum period between output NV transmissions (maximum transmission rate).

#### *1.3.7.2.3 UCPTsunblindCIoseTime*

UCPTsunblindCloseTime is shorted from, sunblind close time configuration property.

The UCPTsunblindCloseTime configuration is specifying the speed the actuator can close the actuatorevalve and make sure that the actuator control the valve best possible. Consult the actuator documentation for getting the correct value.

#### *1.3.7.2.4 UCPTsunblindOpenTime*

UCPTsunblindOpenTime is shorted from, sunblind open time configuration property.

The UCPTsunblindOpenTime configuration is specifying the speed the actuator can open the valve with. The opening time is used in the algorithm to compensate for the slow actuator speed and make sure that the actuator control the valve best possible. Consult the actuator documentation for getting the correct value

## 1.4 Converter Profiles

The following list is describing the Converter software functions implemented in the Comfort and light controller Lonbox PZM4146.

### 1.4.1 Switch to setting converter (3:20016) 6 objects

The following section is describing the gateway software functions implemented in the Comfort and light controller Lonbox PZM4146 for interfacing units only sending out switch variables to setting outputs.

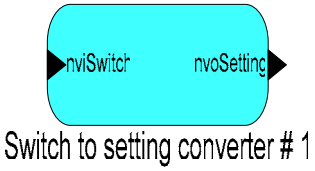



**Figure 45 The switch to setting convertor**

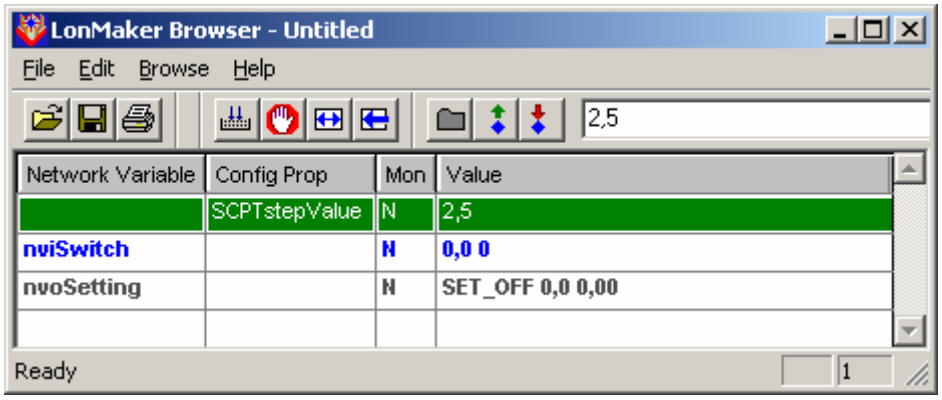



**Figure 46 Browsing the switch to setting object**

#### 1.4.1.1 Network Variable Interface

| Variable name | Default Value | Type | Description |
|---|---|---|---|
| nviSwitch | 0,0 0 | SNVT_switch (95) | Switch input |
| nvoSetting | SET_OFF 0,0 0,00 | SNVT_setting (117) | Setting control output |

##### *1.4.1.1.1 nviSwitch*

This input network variable is used to source the converter object with level and state from the switch variable SNVTswitch.

#### *1.4.1.1.2 nvoSetting*

This variable is the converted output from the switch input.

### 1.4.1.2 Configuration Properties

| Configuration | Default Value | Description |
|---|---|---|
| SCPstepValue | 2,5 | Step size selection |

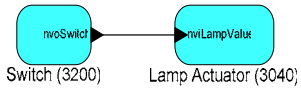
#### *1.4.1.2.1 SCPstepValue*

Step size selection configuration property is used the select how big steps is taken when sending out the seiitng variable.

# 2 Applications and bindings

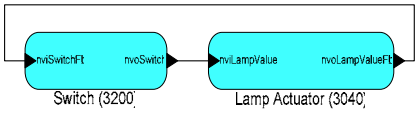
## *2.1 Manual Control, Light Bindings*

### 2.1.1 Simple on and off.



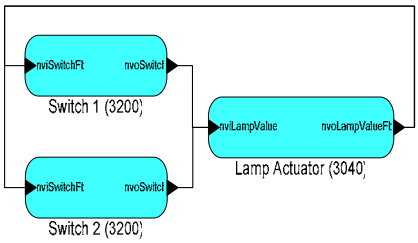

**Figur 47**

### 2.1.2 Simple on and off with feed back.



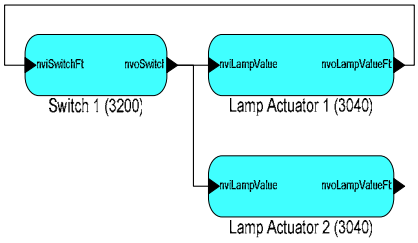

**Figur 48**

### 2.1.3 Multi on and off with feed back.



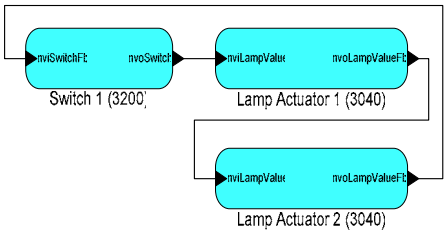

**Figur 49**

### 2.1.4 Simple on and off with multi actuators 1.



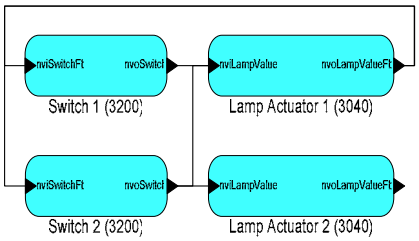

**Figur 50**

### 2.1.5 Simple on and off with multi actuators 2.



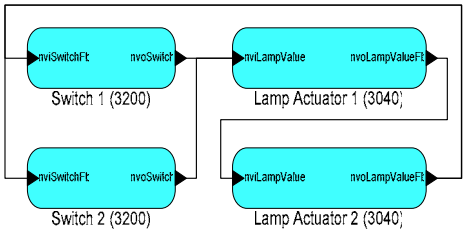

**Figur 51**

### 2.1.6 Multi on and off with multi actuators 1.




**Figur 52**

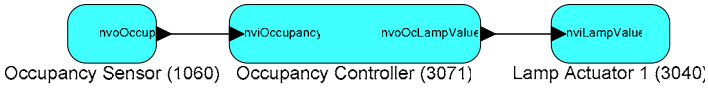
### 2.1.7 Multi on and off with multi actuators 2.




**Figur 53**

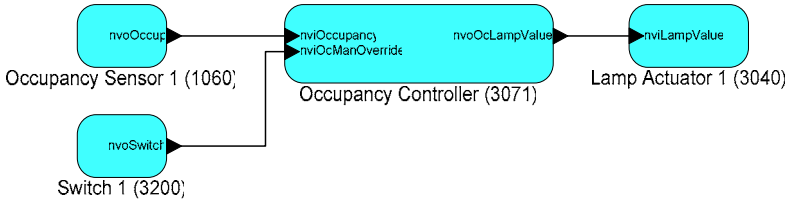
## *2.2 Occupancy automatic, Light Bindings*

### 2.2.1 Occupancy on off.



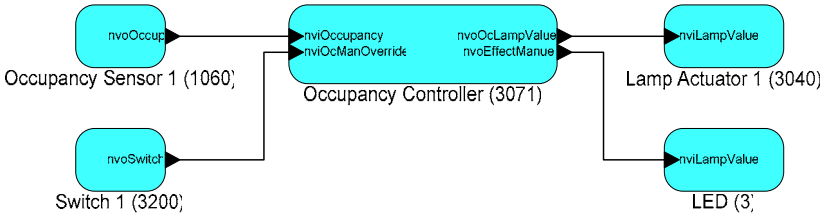

**Figur 54**

### 2.2.2 Occupancy on off, with manual override.


nvoOccup
Occupancy Sensor 1 (1060)
nviOccupancy
nviOcManOverride
nvoOcLampValue
Occupancy Controller (3071)
nviLampValue
Lamp Actuator 1 (3040)
nvoSwitch
Switch 1 (3200)

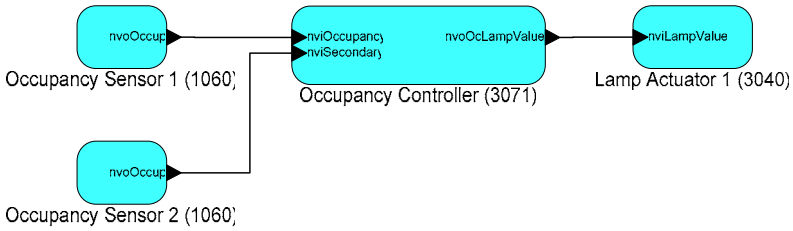
**Figur 55**

### 2.2.3 Occupancy on off, manual override and manual control LED.



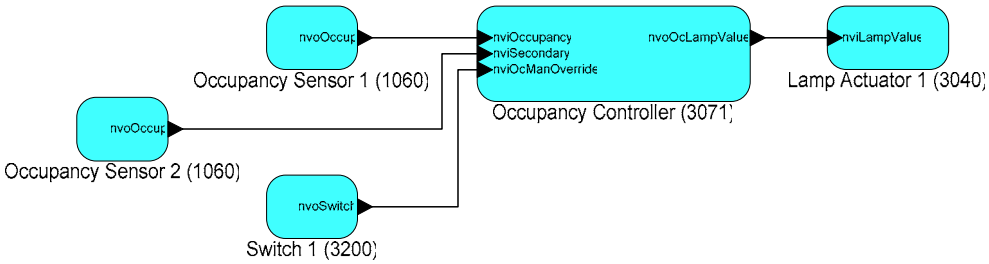

**Figur 56**

### 2.2.4 Occupancy and neighbour occupancy on off.




**Figur 57**

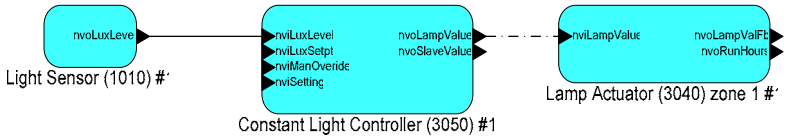
### 2.2.5 Occupancy, neighbour occupancy on off and manual override.




**Figur 58**

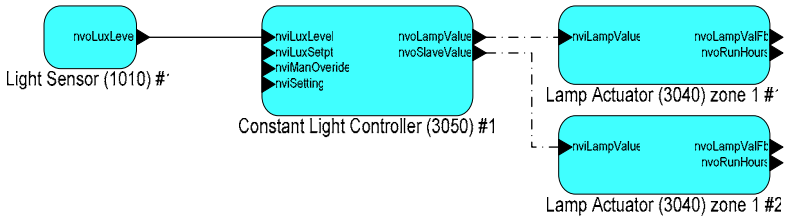
## 2.3 Constant light automatic, Light Bindings

### 2.3.1 Constant light basis automatic



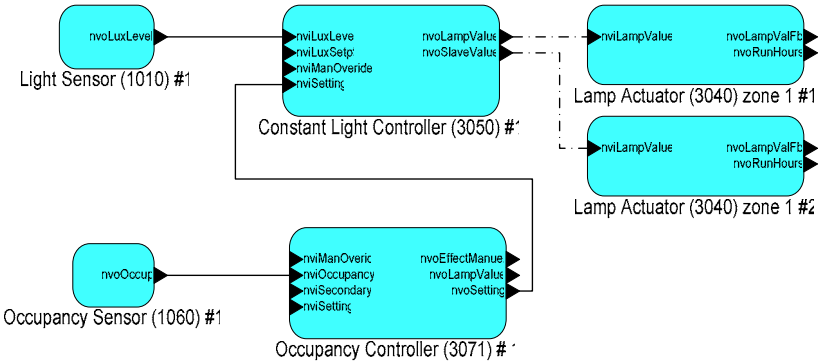

**Figur 59**

### 2.3.2 Constant light with slave output



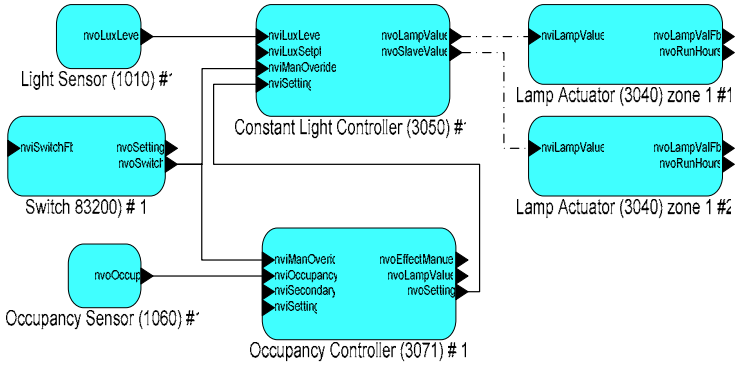

**Figur 60**

### 2.3.3 Constant light and occupancy control




**Figur 61**

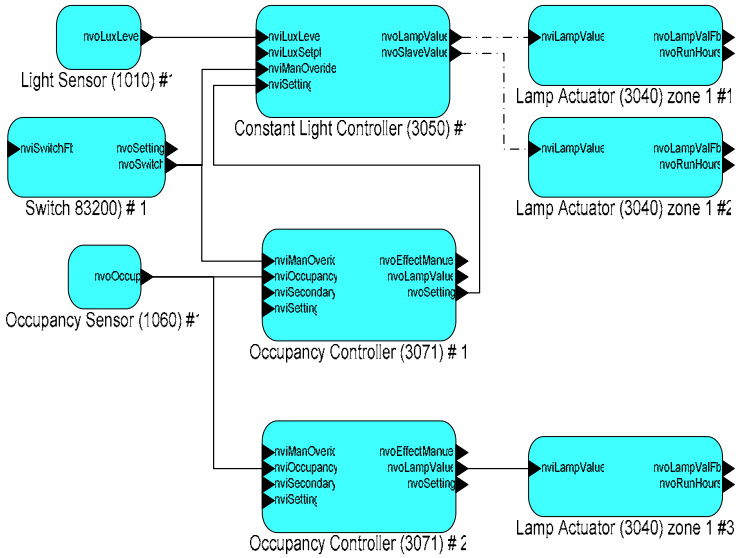
### 2.3.4 Constant light, occupancy and manual control




**Figur 62**

### 2.3.5 Constant light, dual occupancy and a manual control

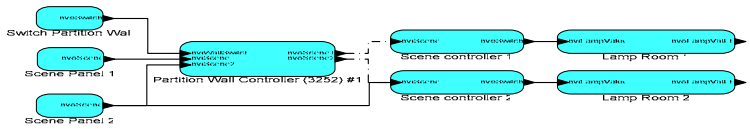
TODO: Describe reason for binding (different hold time for light)




**Figur 63**

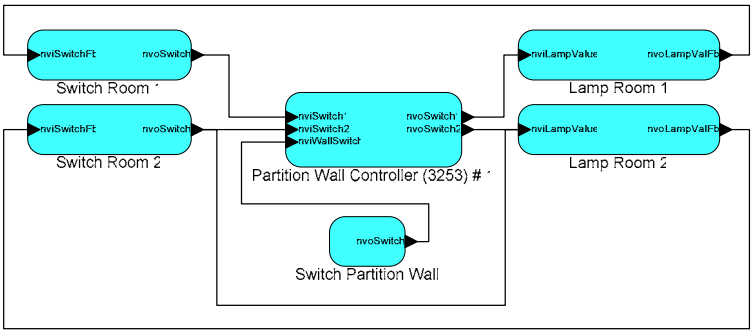
Partition Wall

### 2.3.6 Partition Wall and scene



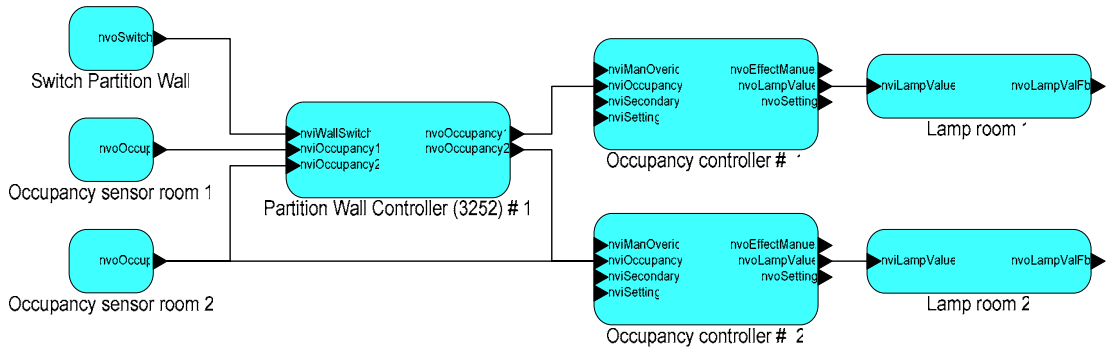

**Figur 64**

### 2.3.7 Partition Wall and switch




**Figur 65**

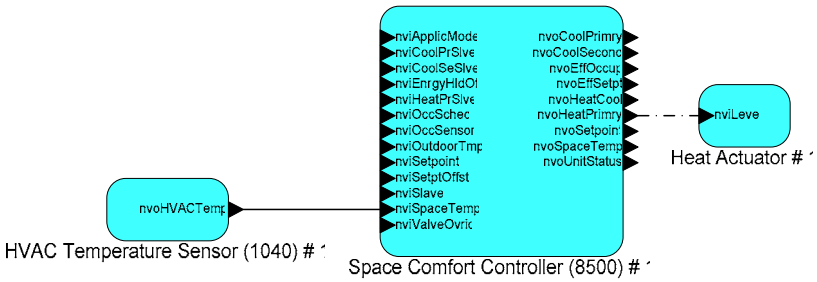
### 2.3.8 Partition wall and occupancy




**Figur 66**

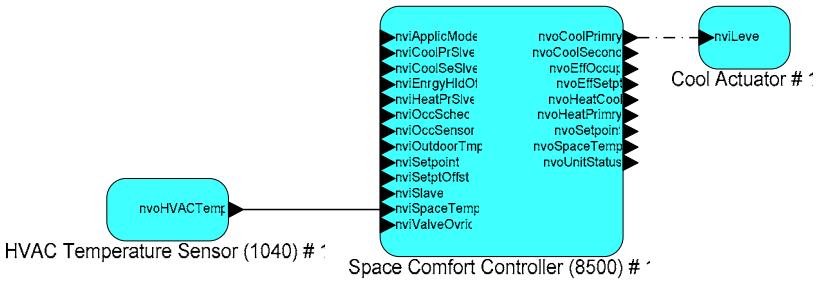
## 2.4 Space Comfort

### 2.4.1 Heating



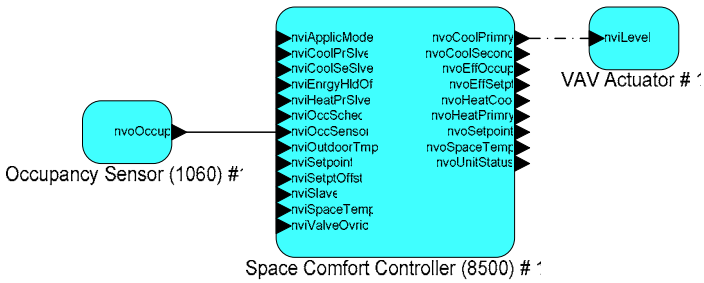

**Figur 67**

### 2.4.2 Cooling



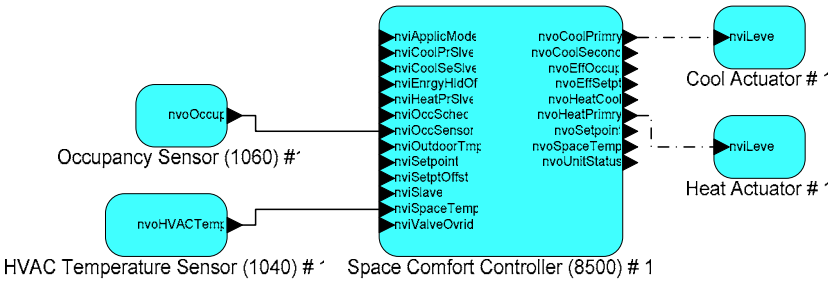

**Figur 68**

### 2.4.3 Ventilation with occupancy control



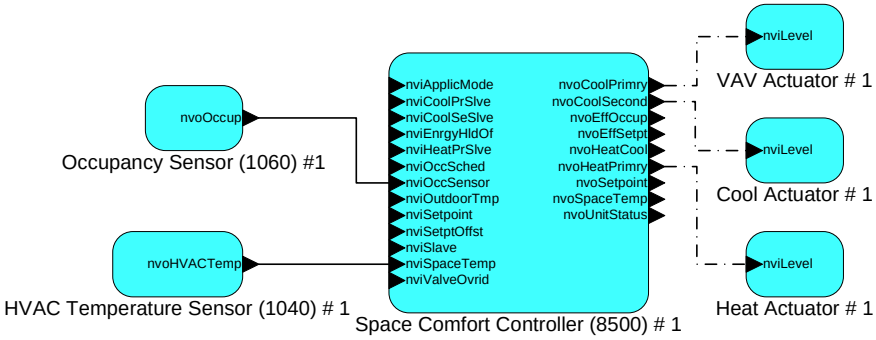

**Figur 69**

### 2.4.4 Heat and cool with occupancy automatic




**Figur 70**

### 2.4.5 Heat, cool and ventilation with occupancy automatic

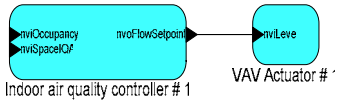



**Figur 71**

## *2.5 Indoor air quality*

### 2.5.1 CO2 controlled ventilation

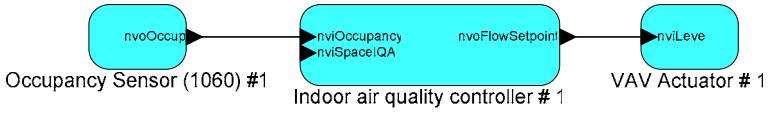
TODO: Add analog sensor




**Figur 72**

### 2.5.2 CO2 controlled ventilation with occupancy automatic

TODO: Add analog sensor




**Figur 73**

### 2.5.3 CO2 controlled ventilation combined with space comfort controller

TODO: insert drawing (bind nvoEffectOcc from space comfort to nviOccupancy)

# 3 Figure list