Electronic Intelligent Controls, S.L. 08830 Sant Boi de Llobregat Tel.: +34 93 652 55 21 www.e-controls.es
C/Múrcia, 35 F Barcelona, Spain Fax: +34 93 652 55 22 info@e-controls.es

e-Multisensor® LON TP/FT-10

MOTION SENSOR, LIGHT AND TEMPERATURE SENSOR
FOR LONWORKS® CONTROL NETWORKS
Ref: MS.623000-000

Functional Profile

Version 0.1.0

This document describes the network variables and the configuration parameters of the device that defines its LonWorks® network interface. The application is defined by logical objects (functional profiles) according to the LONMARK™ Interoperability Directives.

Resource Files version 1.1

Device Functional Profiles

Quantity	Code	Functional Profile Name	Version
1	0000	Node Object	1.0
1	1010	Light Sensor	1.0
1	1060	Occupancy Sensor	1.0
1	1040	Temperature Sensor	1.0
1	3050	Constant Light Controller	1.0
1	3071	Occupancy Controller	1.0

e-controls
electronic intelligent controls, s.l.

Functional Profile:

Node Object

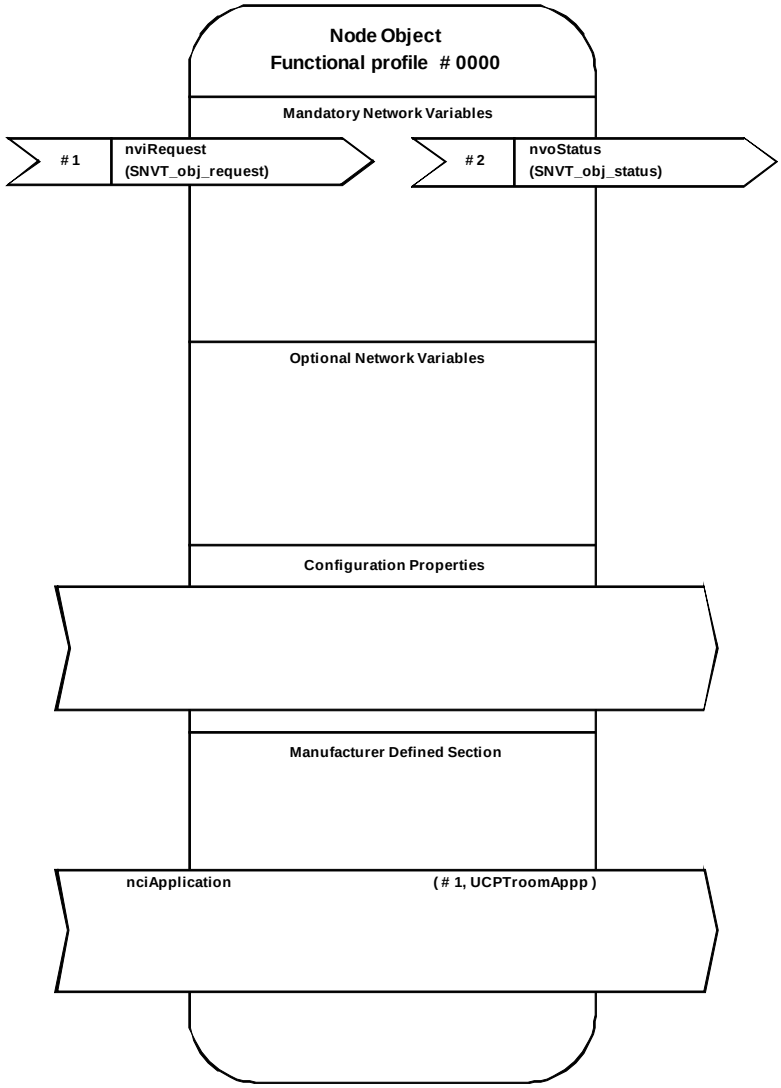
08504 v1.0

Content

1. Description

The *Node Object* is a functional profile used for the network tools to test and manage the functional profiles of the device.

2. Network interface



3. Network variables

3.1. nviRequest

```
network input SNVT_obj_request nviRequest;
```

This input network variable provides the mechanism to request an operation or a mode for a functional block within a device

Type

SNVT_obj_request

```
typedef struct
{
      unsigned long object_id;
      object_request_t object_request;
} SNVT_obj_request;
```

Valid range

object_id

0…6	Index to the desired profile
Other	Invald

object_request

```
typedef enum object_request_t {
      /*  0 */    RQ_NORMAL,
      …
      /*  2 */    RQ_UPDATE_STATUS,
      …
      /*  5 */    RQ_REPORT_MASK,
      …
      /* -1 */    RQ_NUL = -1
} object_request_t;
```

RQ_NORMAL	If the specified functional block was in the disabled or overridden state, this request cancels that state, and returns the functional block to normal operation.
RQ_UPDATE_STATUS	Requests the status of the specified functional block to be sent to the **nvoStatus** output network variable. The state of the functional block is unchanged.
RQ_REPORT_MASK	Requests a *status mask* reporting the status bits that are supported by the specified functional block to be sent to the **nvoStatus** output network variable.

Default value

{ 0, 0 } Request the normal state to the `NodeObject`.

3.2. nvoStatus

```
network output SNVT_obj_status nvoStatus;
```

This output network variable reports the status for any functional block on a device. It is also used to report the status of the entire device and all functional blocks on the device.

Tipo

SNVT_obj_status

```
typedef struct
{
      unsigned long object_id;
      unsigned invalid_id           :1;         // bit0
      unsigned invalid_request      :1;         // bit1
      ...
      unsigned report_mask          :1;         // bit11
      ...
} SNVT_obj_status;
```

Valid range

Any value within the defined limits of ***SNVT_obj_status***.

Default value

{ 0, [0…0] }

When Transmitted

- After reset.
- A request is received by the **nviRequest** input network variable.

Default Service Type

Unspecified, but acknowledged service is recommended. It is also a recommendation that the network variable be polled.

e-controls
electronic intelligent controls, s.l.

Functional Profile:

Light Sensor

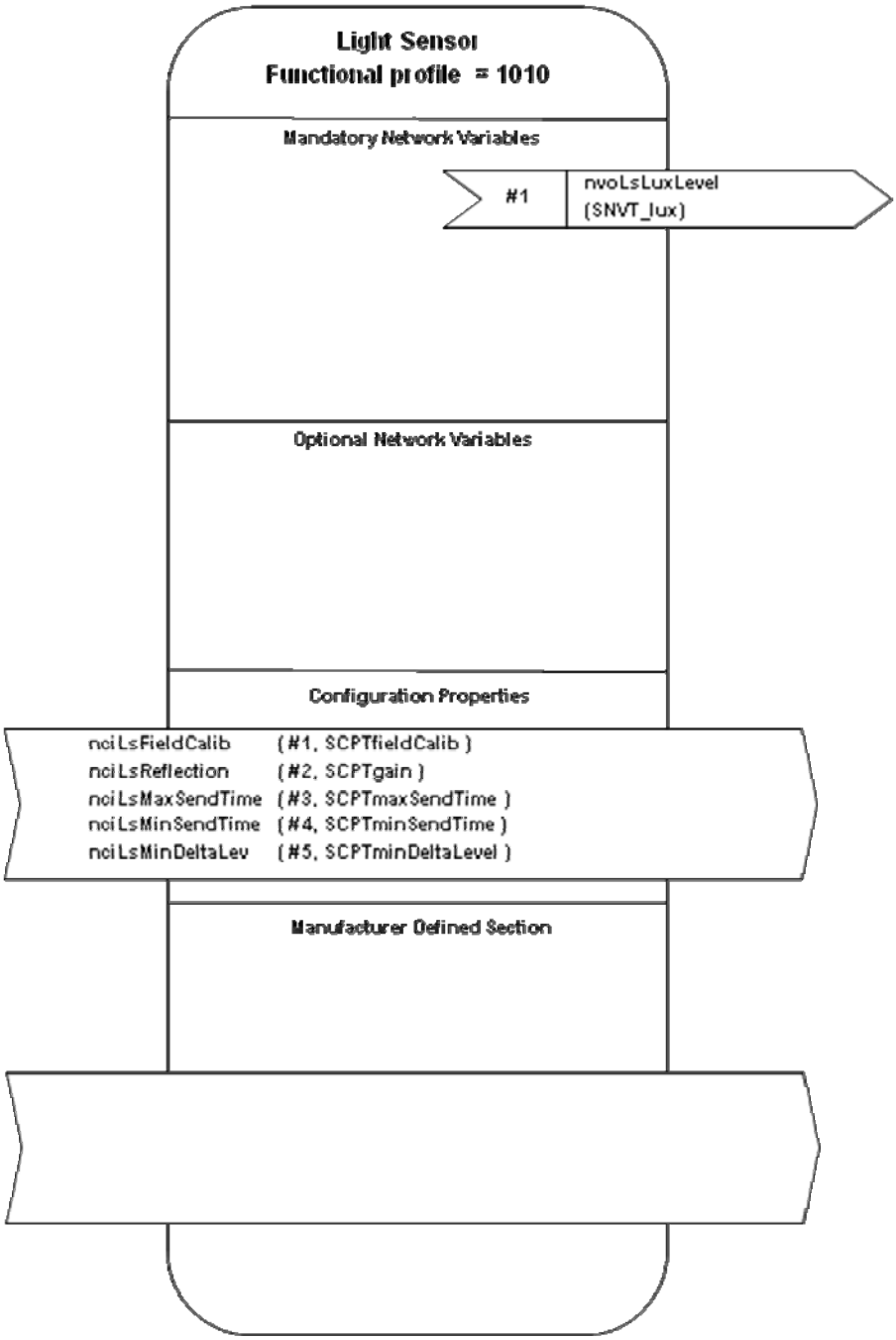
10501 r1.0

Contents

1. Overview

The *Light Sensor* object is used in a device with a sensor that measures the ambient light level.

2. Network interface



3. Network variables

3.1. nvoLsLuxLevel

```
network output SNVT_lux nvoLsLuxLevel;
```

This variable provides the ambient light level of the area where the sensor is installed.

Type

SNVT_lux

```
typedef unsigned long SNVT_lux;
```

Valid range

0 .. 1000 (0 .. 1000 lux, 1 lux resolution)

Default value

{ 65535 } Invalid value

4. Configuration properties

4.1. nciLsFieldCalibr

```
network input cp SCPTfieldCalib nciLsFieldCalibr;
```

This configuration property is used by the light sensor to adjust the gain factor to the environment conditions. To adjust the gain factor the ambient lux measured with an external luxmeter must be assigned to this property.

Type

SCPTfieldCalib, derived from SNVT_lux.

Valid range

1 .. 1000	(1 .. 1000 lux, 1 lux resolution)

Default value

{ 0 }	Defines that the gain factor is not adjusted to the environment. Factory default value.

4.2. nciLsReflection

```
network input cp SCPTgain nciLsReflection;
```

This configuration property defines the reflection level of the surface below the sensor and is used to adjust the gain factor for the measured illumination level.

Adjusting is needed because the amount of the light reflected back to the sensor element from the alight surface differs.

The gain factor is adjusted from the reflection percentage of the surface below the sensor. For example, if the reflection percentage is of 20%, the `multiplier` field must be assigned to 100 and the `divisor` the value 20.

It can also be adjusted the gain factor without known the reflection percentage of the surface, with the help of a luxmeter. In this case the field `divisor` must be filled with the lux value measured by the sensor of the device and provided through `nvoLsLuxLevel` and the field `multiplier` with the measured value of the luxmeter.

NOTE: This property is not effective if the configuration property `nciLsFieldCalibr` is different from 0, this means, if the device has been adjusted with the `nciLsFieldCalibr` property.

Type

SCPTgain, derived from SNVT_muldiv.

```
typedef struct
{
     unsigned long multiplier;
     unsigned long divisor;
} SNVT_switch;
```

Valid range

`multiplier (external lux meter reading)`

1 .. 65535 (1 .. 1000 lux, 1 lux resolution)

`divisor (internal sensor reading)`

1 .. 65535 (1 .. 1000 lux, 1 lux resolution)

Default value

{ 1, 1 } Gain 1

4.3. nciLsMaxSendTime

```
network input cp SCPTmaxSendTime nciLsMaxSendTime;
```

This configuration network variable is used to control the maximum period of time that expires before the object automatically transmits he current value of the `nvoLsLuxLevel` output network variable. This provides a heartbeat output that can be used by destination objects to ensure that the object is still healthy. The heartbeat output may be disabled by setting a value to zero.

Type

SCPTmaxSendTime, derived from SNVT_time_sec.

Valid range

0 - 6553,4s

Default value

The default value is 1 minute.

4.4. nciLsMinSendTime

```
network input cp SCPTminSendTime nciLsMinSendTime;
```

This configuration network variable is used to control the minimum period between output network variable transmissions (maximum transmission rate). It provides a way to tailor the output network variable transmission rate to available bandwidth.

Type

SCPTminSendTime, derived from SNVT_time_sec.

Valid range

0 - 6553,4s

Default value

The default value is 1 second.

4.5. nciLsMinDeltaLev

```
network input cp SCPTminDeltaLevel nciLsMinDeltaLev;
```

This configuration property is used to determine the amount by which the value obtained by the data acquisition application must change before `nvoLsLuxLevel` is transmitted.

Type

SCPTminDeltaLevel, derived from SNVT_lev_cont.

Valid range

The valid range is 0.0% - 100.0% in 0.5% steps

Default value

The default value is 2%.

e-controls
electronic intelligent controls, s.l.

Functional Profile:

Occupancy Sensor

10501 r1.0

Contents

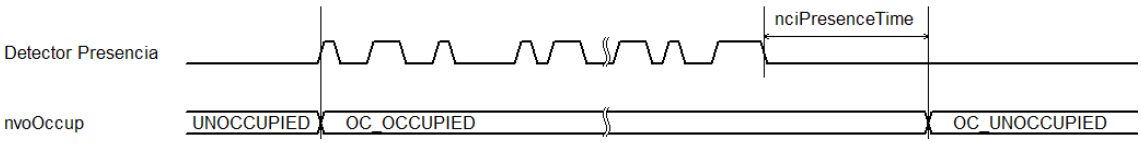
1. Overview

The *Occupancy Sensor* object is used in the device with a motion sensor that detects occupancy in a room or an area. The change to *Occupied* state is dome by a change on the sensor, while the change to *Unoccupied* is produced when the time defined in the configuration variable `nciOsDebounce` expires.

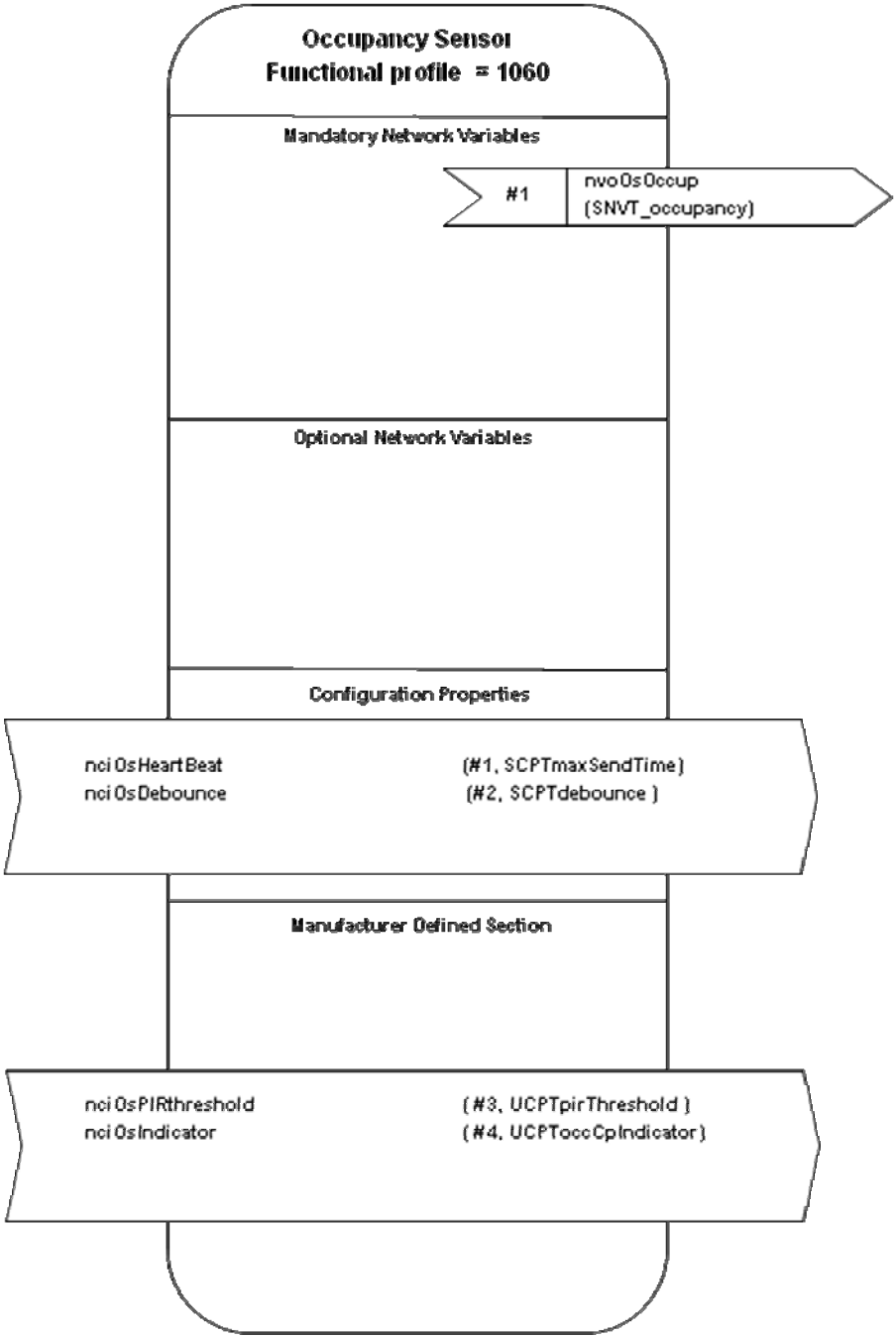
The object's output variable is usually connected to a controller, which takes an action depending on the occupancy state, like turning on a light or changing the temperature setpoint.

1.1. Operation

A rising edge from the hardware sensor forces the output to go to the occupied state. A falling edge triggers a timer. When the timer expires, the output goes to the unoccupied state. This timer is retriggered with each rising edge.



2. Network interface



3. Network variables

3.1. nvoOsOccup

```
network output SNVT_occupancy nvoOsOccup;
```

This output network variable provides the occupancy state of the hardware sensor output.

Type

SNVT_occupancy

```
typedef occup_t SNVT_occupancy;
```

Valid range

```
typedef enum occup_t
{
     /*  0 */  OC_OCCUPIED,
     /*  1 */  OC_UNOCCUPIED,
     …
     /* -1 */  OC_NUL = -1
} occup_t;
```

Default value

```
{-1}            OC_NUL
```

4. Configuration properties

4.1. nciOsDebounce

```
network input cp SCPTdebounce nciOsDebounce;
```

This configuration property defines the hold time value before setting the unoccupied state at the network output variable when no occupancy is detected by the hardware sensor.

Type

SCPTdebounce, derived from SNVT_time_sec.

Valid range

0 - 6553.4 seconds (with 0,1 second resolution)

Default value

{ 3 } 300 miliseconds

4.2. nciOsHeartBeat

```
network input cp SCPTmaxSendTime nciOsHeartBeat
```

This configuration network variable defines the repeat period between to value update sent on the bus. The aim 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 SVNT.

Type

SCPTmaxSendTime, derived from SNVT_time_sec.

Valid range

0 – 6553,4 by steps of 0.1 s

Default value

{ 1200 } 2 minutes

4.3. nciOsPIRthreshold

```
network input cp UCPTpirThreshold nciOsPIRthreshold
```

This property is used to configure the occupancy sensor sensitivity. It fixes the sensitivity level to detect motion in the zone.

Type

UCPTpirThreshold, derived from enumerate sensitivity_level_t.

Valid range

```
typedef enum sensitivity_level_t {
     LEVEL_1_MIN = 0,
     LEVEL_2     = 1,
     LEVEL_3     = 2,
     LEVEL_4     = 3,
     LEVEL_5     = 4,
     LEVEL_6     = 5,
     LEVEL_7     = 6,
     LEVEL_8      = 7,
     LEVEL_9      = 8,
     LEVEL_10     = 9,
     LEVEL_11     = 10,
     LEVEL_12     = 11,
     LEVEL_13     = 12,
     LEVEL_14     = 13,
     LEVEL_15     = 14,
     LEVEL_16_MAX = 15,
     LEVEL_NUL   = -1}
 sensitivity_level_t;
```

LEVEL_1_MIN: minimum sensitivity
LEVEL_16_MAX: maximum sensitivity
LEVEL_NUL: the sensitivity level is disabled and the sensor does not signal the OCCUPIED value

Default value

{LEVEL_12}

4.4. nciOsIndicator

```
network input cp UCPToccCpIndicator nciOsIndicator
```

This configuration property defines if the Led indicator of the motion sensor must blink or not when motion is detected.

Type

UCPToccCpIndicator, derived from UNVT_occCpIndicator (boolean_t) .

Valid range

0: Occupancy indicator enabled
1: Occupancy indicator disabled

Default value

{ 1 } Occupancy indicator enabled

e-controls
electronic intelligent controls, s.l.

Functional Profile:

Temperature Sensor

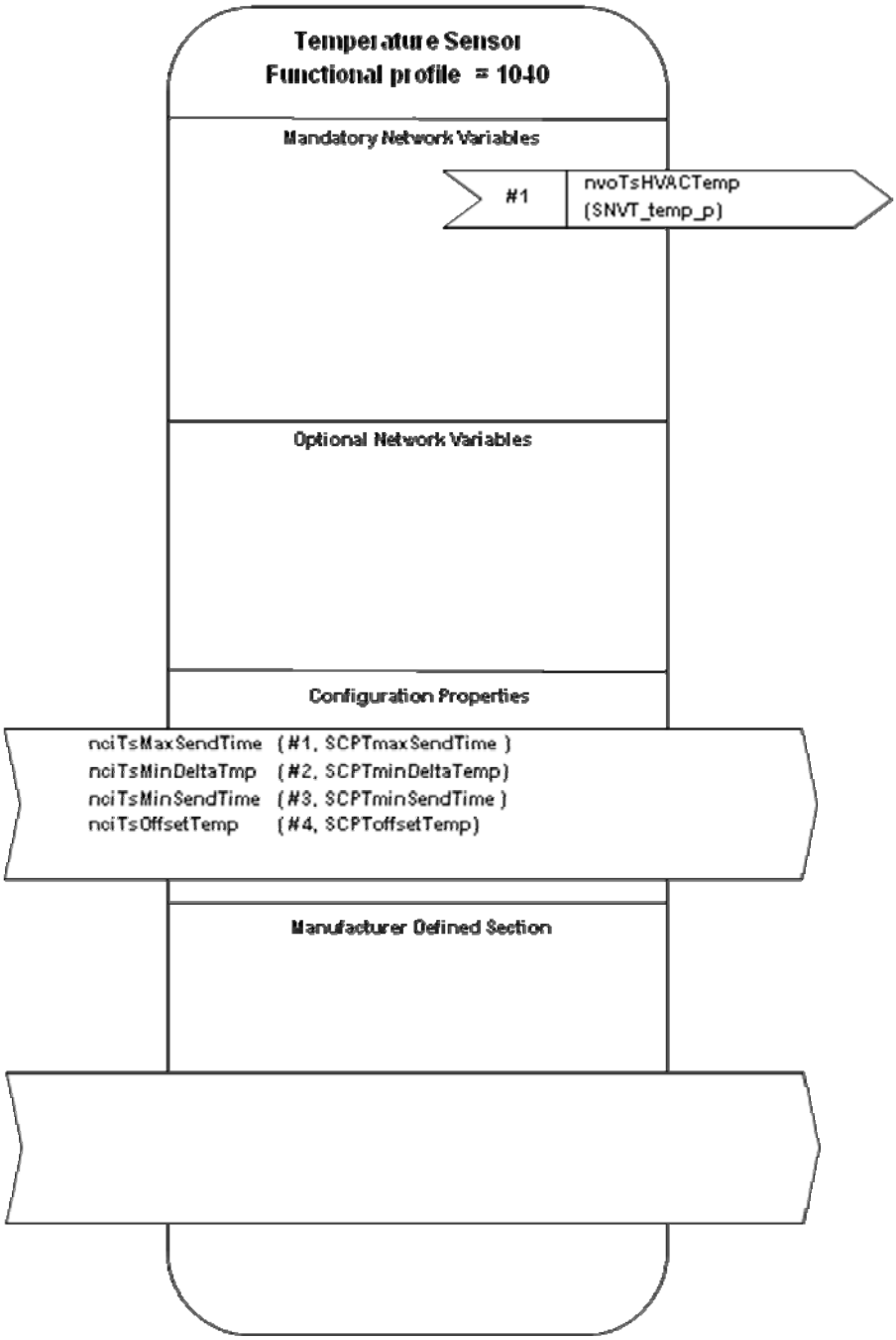
10501 r1.0

Contents

1. Overview

This document describes the profile of an HVAC temperature sensor object. The *Temperature Sensor* object is used in a device with a sensor that measures the temperature.

2. Network interface



3. Network variables

3.1. nvoTsHVACTemp

```
network output SNVT_temp_p nvoTsHVACTemp
```

This output network variable provides the temperature of the area where the sensor is placed.

Type

SNVT_temp_p

```
typedef signed long SNVT_temp_p;
```

Valid range

```
-273,17 .. 327,66 (0,01 resolution degrees Celsius)
```

Invalid value

32767 (0x7FFF)

Default value

N/A

4. Configuration properties

4.1. nciTsMaxSendTime

`network input SCPTmaxSendTime cp nciTsmaxSendTime`

This configuration property is used to indicate the maximum period of time that expires before the sensor object automatically updates the network variable `nvoTsHVACTemp`. This provides a mechanism to know that the device is alive. It can be disabled defining its value to 0.

Type

SCPTmaxSendTime, derived from SNVT_time_sec.

Valid range

0 .. 6553,4 seconds (0,1 seconds resolution)

Default value

{ 300 } 5 minutes.

Notes

If this configuration property is filled with a value of 0 seconds, then the sensor object never automatically updates its output variables.

4.2. nciTsMinDeltaTmp

`network input SCPTminDeltaTemp cp nciTsMinDeltaTmp`

This configuration property indicates the minimum temperature change required to update the output network variable `nvoTsHVACTemp`. This property type is used to limit the amount of values sent per minimum unit time.

Type

SCPTminDeltaTemp, derived from SNVT_temp_p

Valid range

-273,17 .. 327,66 (0,01 resolution degrees Celsius)

Default value

{1 } 1 degree celsius

4.3. nciTsMinSendTime

`network input SCPTminSendTime cp nciTsMinSendTime`

This configuration property indicates the minimum period between output network variables transitions.

Type

SCPTminSendTime, derived from SNVT_time_sec

Valid range

0 .. 6553,4 seconds (0,1 seconds resolution)

Default value

{ 5 } 5 seconds

Notes

If this configuration property is filled with a value of 0 seconds, then the sensor object does not have a minimum period between output network variables.

4.4. nciTsOffsetTemp

`network input SCPToffsetTemp cp nciTsOffsetTemp`

This configuration property is used to adjust the device temperature with a offset provided to the `nvoTsHVACTemp` value.

Type

SCPToffsetTemp, derived from SNVT_temp_p.

Valid range

-273,17 .. 327,66 (0,01 resolution degrees Celsius)

Default value

{0 } 0 degrees Celsius

e-controls
electronic intelligent controls, s.l.

Functional Profile:

Constant Light Controller

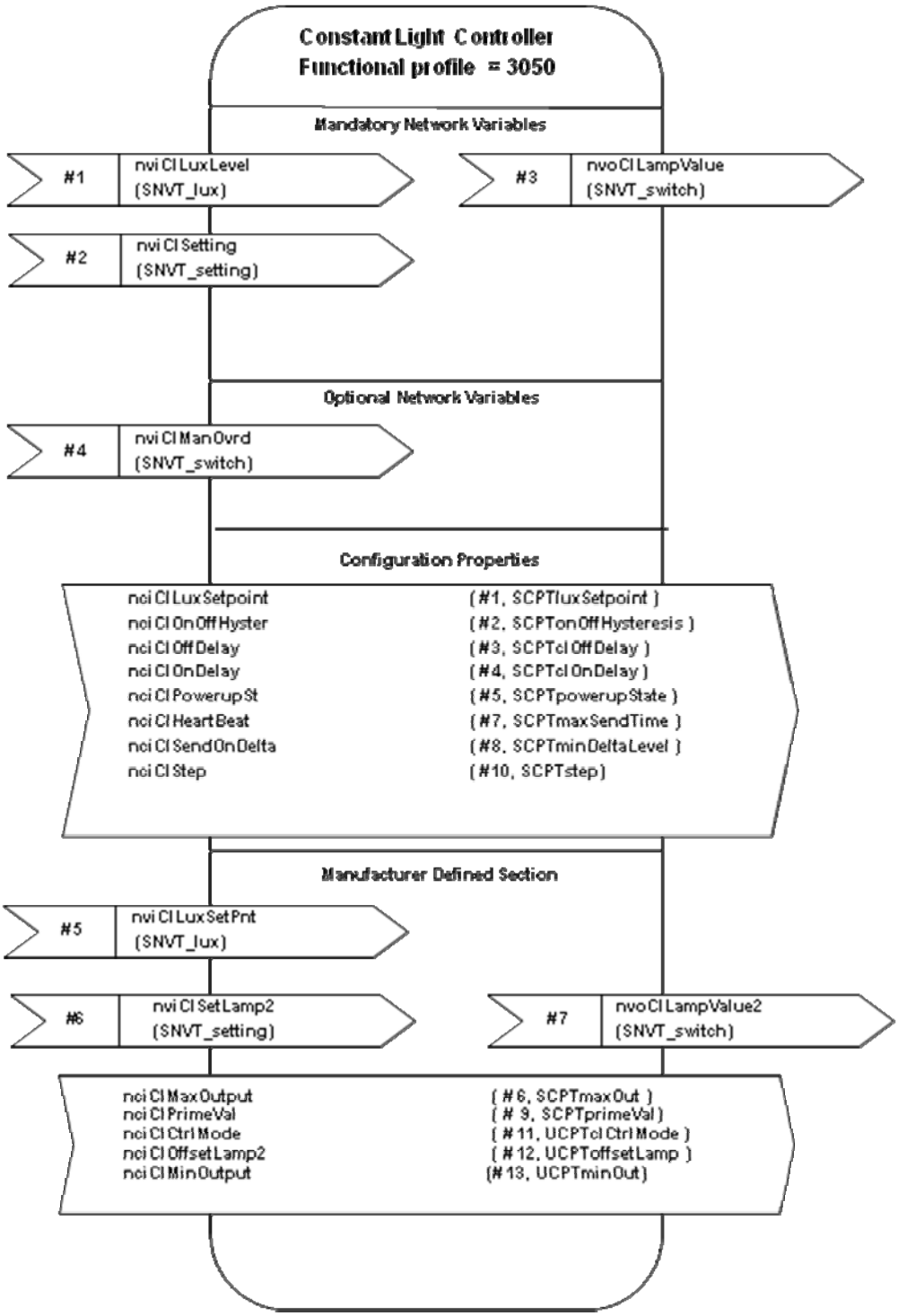
10501 r1.0

Contents

1. Overview

The *Constant Light Controller* object is responsible for providing a constant light level in a room through an automatic adjustment mechanism of the brightness of a lamp or group of luminaries. The lighting level is automatically adjusted depending on the ambient light in the room and the contribution of external light in it.

2. Network interface


Constant Light Controller
Functional profile = 3050
Mandatory Network Variables
#1
nviClLuxLevel
(SNVT_lux)
#3
nvoClLampValue
(SNVT_switch)
#2
nviClSetting
(SNVT_setting)
Optional Network Variables
#4
nviClManOvrd
(SNVT_switch)
Configuration Properties
nciClLuxSetpoint (#1, SCPTluxSetpoint)
nciClOnOffHyster (#2, SCPTonOffHysteresis)
nciClOffDelay (#3, SCPTclOffDelay)
nciClOnDelay (#4, SCPTclOnDelay)
nciClPowerupSt (#5, SCPTpowerupState)
nciClHeartBeat (#7, SCPTmaxSendTime)
nciClSendOnDelta (#8, SCPTminDeltaLevel)
nciClStep (#10, SCPTstep)
Manufacturer Defined Section
#5
nviClLuxSetPnt
(SNVT_lux)
#6
nviClSetLamp2
(SNVT_setting)
#7
nvoClLampValue2
(SNVT_switch)
nciClMaxOutput (#6, SCPTmaxOut)
nciClPrimeVal (#9, SCPTprimeVal)
nciClCtrlMode (#11, UCPTclCtrlMode)
nciClOffsetLamp2 (#12, UCPToffsetLamp)
nciClMinOutput (#13, UCPTminOut)

3. Network variables

3.1. nviClLuxLevel

```
network input SNVT_lux nviClLuxLevel;
```

This input network variable provides the ambient light level of the zone.

Type

SNVT_lux

```
typedef unsigned long SNVT_lux;
```

Valid range

0 .. 1000 (0 .. 1000 lux, 1 lux resolution)

Default value

{ 0 } 0 lux

3.2. nviClLuxSetPnt

```
network input SNVT_lux nviClLuxSetPnt;
```

This input network variable is used to define a new setpoint value on the CLC. The variable is initialized by the `nciClLuxSetpoint` .

When the value of `nviClLuxSetpoint` is modified, the value of `nciClLuxSetpoint` is overwritten permanently.

When the device is configured as an ON/OFF controller (see 4.12 `nciClCtrlMode`) this network variable operates as the ON/OFF threshold value.

Type

SNVT_lux

```
typedef unsigned long SNVT_lux;
```

Valid range

0 .. 1000 (0 .. 1000 lux, 1 lux resolution)

Default value

The values is taken from `nciClLuxSetpoint`

3.3. nviClSetting

```
network input SNVT_setting nviClSetting;
```

This input network variable selects the operating mode and adjusts the setpoint of the constant light controller. Modes are `SET_ON`, `SET_OFF`, `SET_DOWN`, `SET_UP` and `SET_STOP`.

- The `SET_ON` mode turns on the constant light controller (CLC) which then starts to control the lamp value output `nvoClLampValue` so that the illumination level provided by the `nviClLuxLevel` equals to the setpoint value `nviClLuxSetPnt`.
- The `SET_OFF` mode turns off the controller and the lamp value output `nvoClLampValue` to `{OFF,0}`.
- The setpoint of the controller can temporarily be stepped upwards and downwards by the modes `SET_UP` and `SET_DOWN` respectively. The changes made to the setpoint value are not stored permanently into the memory. Next time when the `SET_ON` mode is selected the original setpoint value is restored to `nviClLuxSetPnt`.

Type

SNVT_setting

```
typedef struct
{
     setting_t function;
     unsigned short setting;
     signed long rotation;
} SNVT_setting;
```

Valid range

`function`

Value	Name	Description
-1	SET_NUL	Invalid value, keep on current operation mode.

0	SET_OFF	Turn off controller and output.
1	SET_ON	Turn on controller and output.
2	SET_DOWN	Step down light gradually until reception of SET_STOP or controller reaches minimal allowable setpoint.
3	SET_UP	Step up light setpoint gradually until reception of SET_STOP or controller reaches maximum allowable setpoint.
4	SET_STOP	Stop the CLC algorithm assigning the setpoint to the read lux level (the input variable nviClLuxLevel).

`setting`

Not used.

`rotation`

Not used.

Default value

```
{SET_NUL, 0, 0}     -
```

3.4. nviClSetLamp2

```
network input SNVT_setting nviClSetLamp2;
```

This input network variable sets if the second output, `nvoClLampValue2` is turned on or turned off.

Type

SNVT_setting

```
typedef struct
{
     setting_t function;
     unsigned short setting;
     signed long rotation;
} SNVT_setting;
```

Valid range

`function`

Value	Name	Description
-1	SET_NUL	Invalid value, keep on current operation mode.
0	SET_OFF	Turn off controller and output.
1	SET_ON	Turn on controller and output.

`setting`

Not used

`rotation`

Not used.

Default value

`{SET_NUL, 0, 0}` -

3.5. nviClManOvrd

```
network input SNVT_switch nviClManOvrd;
```

This input network variable provides the possibility to control manually the lamp output `nvoClLampValue`. When it receives a new value the constant light controller is turned off, and the input value is directly passed to the lamp value output `nvoClLampValue`.

If the state of the received data has value 255 (undefined), the constant light controller is again turned on and de output variable `nvoClLampValue` will take the corresponding value.

Type

SNVT_switch

```
typedef struct
{
     unsigned value;
     signed state;
} SNVT_switch;
```

Valid range

`value`

0 .. 200	(0% .. 100%, 0,5% resolution)

`state`

0	Off
1	On (Note: If `value` = 0, output is off)
-1	Undefined: controller turns on.

Default value

{ 0, -1 }	Controller is enabled.

3.6. nvoClLampValue

```
network output SNVT_switch nvoClLampValue;
```

This variable provides the state for the lamp actuator (ON or OFF) and the percentage level of intensity.

Type

SNVT_switch

```
typedef struct
{
     unsigned value;
     signed state;
} SNVT_switch;
```

Valid range

`value`

0 .. 200	(0% .. 100%, 0,5% resolution)

`state`

0	Off
1	On (Note: If `value` = 0, output is off)
-1	Invalid value.

Default value

{ 0, 0 }	Output is Off

3.7. nvoClLampValue2

```
network output SNVT_switch nvoClLampValue2;
```

This variable provides a second value and state for a secondary lamp actuator. The value equals the value of the main lamp actuator, `nvoClLampValue`, plus the offset set by the property variable `nciClOffsetLamp2`.

Type

SNVT_switch

```
typedef struct
{
     unsigned value;
     signed state;
} SNVT_switch;
```

Valid range

`value`

0 .. 200	(0% .. 100%, 0,5% resolution)

`state`

0	Off
1	On (Note: If `value` = 0, output is off)
-1	Invalid value.

Default value

{ 0, 0 }	Output is Off

4. Configuration properties

4.1. nciClLuxSetpoint

```
network input SCPTluxSetpoint cp nciClLuxSetpoint;
```

This configuration property is used to initialize the `nviClLuxSetPnt` value after reset. The setpoint value can also be changed temporarily from the control input by the `nviClSetting`. This property is overwritten by changes made to `nviClLuxSetPnt`.

When the device is configured as an ON/OFF controller (see 4.12 `nciClCtrlMode`) this property operates as a ON/OFF threshold value.

Type

SCPTluxSetpoint, derived from SNVT_lux.

```
typedef unsigned long SNVT_lux;
```

Valid range

0 .. 1000	(0 .. 1000 lux, 1 lux resolution)

Default value

{ 500 }	500 lux

4.2. nciClOnOffHyster

```
network input SCPTonOffHysteresis cp nciClOnOffHyster
```

This configuration parameter defines the setpoint level to switch on and off the Lighting through the network variable `nvoClLampValue`. The controller operates overt the output using this network variable and the `nciClOffDelay` and `nciClOnDelay` variables (read 4.3 and 4.4).

This network variable is not used when the device is configured to operate in ON/OFF mode.

Tipo

SCPTonOffHysteresis, derived from SNVT_level_cont.

```
typedef unsigned short SNVT_level_cont;
```

Valid range

0 .. 200	(0% .. 100%, 0,5% resolution)

Default value

{ 0 }	0%, automatic switching off and on is disabled when operates in CLC mode.

4.3. nciClOffDelay

```
network input SCPTclOffDelay cp nciClOffDelay;
```

This configuration property defines the time during which the condition setpoint + hysteresis (defined by `nciClOnOffHyster`) must be always valid regarding the illumination level of the zone, to switch off the lighting.

In CLC mode this property works when the light level is at minimum value.

Tipo

SCPTclOffDelay, derived from SNVT_time_sec.

Valid range

0 – 6553,4 s (0,1 seconds resolution)

Default value

{ 3000 } 5 minutes.

4.4. nciClOnDelay

```
network input SCPTclOnDelay cp nciClOnDelay;
```

This configuration property defines the time during which the condition setpoint + hysteresis (defined by `nciClOnOffHyster`) must be always valid regarding the illumination level of the zone, to switch off the lighting.

This configuration property defines the time during which the condition setpoint – hysteresis (defined by `nciClOnOffHyster`) must be always valid regarding the illumination level of the zone, to switch on the lighting.

Tipo

SCPTclOffDelay, derived from SNVT_time_sec.

Valid range

0 – 6553,4 s (0,1 seconds resolution)

Default value

{ 50 } 5 seconds.

4.5. nciClPowerupSt

```
network input SCPTpowerupState cp nciClPowerupSt;
```

This configuration property sets the state (mode) of a controller object after power-up or reset. The state can either be ON or OFF.

Type

SNVT_setting

```
typedef struct
{
     setting_t function;
     unsigned short setting;
     signed long rotation;
} SNVT_setting;
```

Valid range

`function`

Value	Name	Description
0	SET_OFF	Turn off controller and output.
1	SET_ON	Turn on controller and output.
other	-	Turn off controller and output.

`setting`

0 Not used.

`rotation`

0 Not used.

Default value

`{SET_OFF, 0, 0}` Controller turned off.

4.6. nciClMaxOutput

```
network input SCPTmaxOut cp nciClMaxOutput;
```

This configuration property determines the maximum value of lighting in constant light mode for the output variable `nvoClLampValue`.

When operating through the variable `nviClManOvrd` the output value of `nvoClLampValue` is not limited.

Type

`SCPTmaxOut`***, derived from SNVT_lev_cont.***

Valid range

0,0 % – 100,0 % (0,5% steps resolution)

Default value

{ 200 } 100 %

4.7. nciClMinOutput

```
network input UCPTminOut cp nciClMinOutput;
```

This configuration property determines the minimum value of lighting in constant light mode for the output variable `nvoClLampValue`.

When operating through the variable `nviClManOvrd` the output value of `nvoClLampValue` is not limited.

Type

`UCPTminOut`***, derived from SNVT_lev_cont.***

Valid range

0,0 % – 100,0 % (0,5% steps resolution)

Default value

{ 20 } 10 %

4.8. nciClHeartBeat

```
network input SCPTmaxSendTime cp nciClHeartBeat;
```

This configuration property is used to control the maximum period of time before the object automatically re-transmits the current lamp value `nvoClLampValue`. This provides a heartbeat output that can be used by destination objects to ensure that the object is still healthy. The output may be disabled by setting a value to zero.

Type

SCPTmaxSendTime, derived from SNVT_time_sec.

Valid range

0 – 6553,4 (0.1 seconds resolution)

Default value

{ 3000 } 5 minutes

4.9. nciClSendOnDelta

```
network input SCPTminDeltaLevel cp nciClSendOnDelta;
```

This configuration property is used to determine the amount by which the lamp value output must change before `nvoClLampValue` is transmitted. The lamp value output is always updated when the state (ON/OFF) is changed.

Type

SCPTminDeltaLevel, derived from SNVT_lev_cont.

Valid range

0,0 % – 100,0 % (0,5% steps resolution)

Default value

{ 1 } 0,5 %

4.10. nciClPrimeVal

```
network input SCPTprimeVal cp nciClPrimeVal;
```

This configuration property defines the switch-on value of the output network variable `nvoClLampValue` after a reset or at anytime the CLC is switched on.

Type

SCPTprimeVal, derived from SNVT_switch.

```
typedef struct
{
        unsigned value;
        signed state;
} SNVT_switch;
```

Valid range

Value

0 ..200 (0%... 100%, resolution 0,5%)

State
0 OFF
1 ON (if value = 0 is OFF)
-1 Invalid

Default value

{1, 100 } ON, 50%.

4.11. nciClStep

```
network input SCPTstep cp nciClStep;
```

This configuration property is used in the CLC to define the increment value applied to the output network variable `nvoClLampValue` until the setpoint level configured is reached.

Type

SCPTstep, derived from SNVT_lev_cont.

Valid range

0,0 % – 100,0 % (0,5% steps resolution)

Default value

{ 4 } 2 %

4.12. nciClCtrlMode

```
network input UCPTclCtrlMode cp nciClCtrlMode;
```

This configuration property determines the controller operating mode. It can operate as a Constant Light Controller (CLC) or in switching mode as an ON/OFF controller.

Type

`UCPTclCtrlMode`***, derived from UNVT_cl_Ctrl_Mode.***

```
typedef enum
{
        CL_ONOFF = 0,
        CL_CONSTANT = 1
} clc_mode_t;
```

Valid range

0: Switching ON/OFF mode
1: Constant Light Controller mode

Default value

{ 1 } Constant Light Controller mode

4.13. nciClOffsetLamp2

```
network input UCPToffsetLamp cp nciClOffsetLamp2;
```

This configuration property is used by the `nvoClLampValue2` variable. The `nvoClLampValue2` adopts the value of the `nvoClLampValue` variable plus the offset set by this configuration property.

Type

UCPToffsetLamp, derived from SNVT_lev_cont.

Valid range

0,0 % – 100,0 % (0,5% steps resolution)

Default value

{ 40 } 20 %

e-controls
electronic intelligent controls, s.l.

Functional Profile:

Occupancy Controller

10501 r1.0

Contents

1. Overview

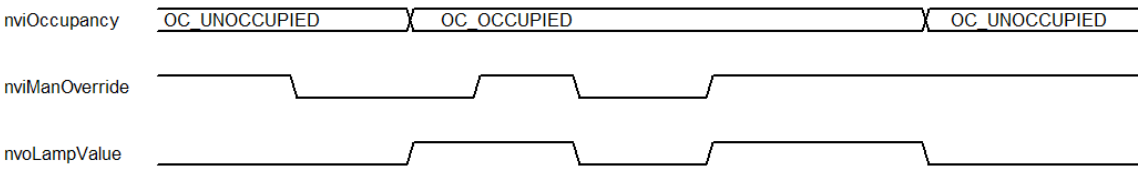
The *Occupancy Controller* object is used to control an actuator, typically a lighting system, depending on the occupancy status of the room. When the room is occupied output is activated, whereas if the room becomes unocupied the output is disabled.

The controller has an input to manually control the output when the room is occupied.

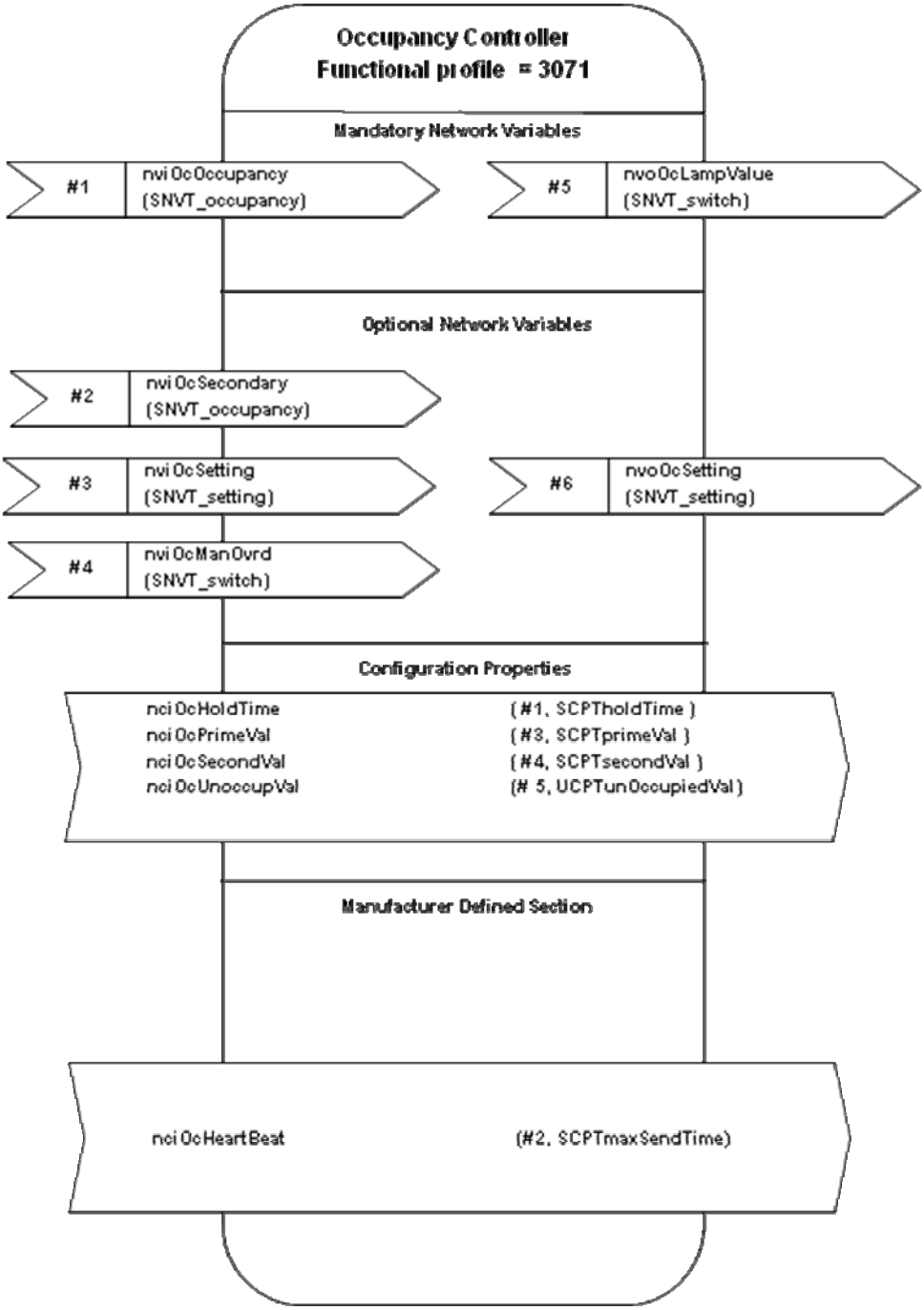
1.1. Operation

The value of the output variable `nvoOcLampValue` is determined by the value of `nviOCOccupancy`:

- when it goes from Unoccupied to Occupied, the output is automatically activated.
- when it goes from Occupied to Unoccupied, output is automatically disabled when `nciOcHoldTime` expires.
- while in Occupied, you can change the output value by the variable `nviOcManOverride`.



2. Network interface



3. Network variables

3.1. nviOcOccupancy

```
network input SNVT_occupancy nviOcOccupancy;
```

This input network variable provides the occupancy status of the room to the controller.

Type

SNVT_occupancy

```
typedef occup_t SNVT_occupancy;
```

Valid range

```
typedef enum occup_t
{
     /*  0 */  OC_OCCUPIED,
     /*  1 */  OC_UNOCCUPIED,
     …
     /* -1 */  OC_NUL = -1
} occup_t;
```

Default value

```
{-1}            OC_NUL
```

Notes

For all other enumerations of SNVT_occupancy, no action is taken by the occupancy controller.

3.2. nviOcSetting

```
network input SNVT_setting nviOcSetting;
```

This input network variable is used to switch ON and OFF the controller. The ON mode turns on the controller (automatic mode) which then controls the lamp value output `nvoOcLampValue`. The OFF mode turns off the controller and the `nvoOcLampValue` output.

Type

SNVT_setting

```
typedef struct
{
     setting_t function;
     unsigned short setting;
     signed long rotation;
} SNVT_setting;
```

Valid range

`function`

Value	Name	Description
0	SET_OFF	Turn off controller and output.
1	SET_ON	Turn on controller.

The other enumerations are not used.

`setting`

Not used.

`rotation`

Not used.

Default value

`{SET_ON, 0, 0}` Controller turned on.

3.3. nviOcManOverride

```
network input SNVT_switch nviOcManOverride;
```

Any change in this input network variable is forwarded to the variable `nvoOcLampValue` output.

When the variable `nviOcOccupancy` is set at `OC_OCCUPIED`, any change on `nviOcManOverride` is forwarded to the `nvoOcLampValue` output.

When the variable `nviOcOccupancy` changes to `OC_UNOCCUPIED`, the controller sets the output to OFF and changes in `nviOcManOverride` are not processed.

Type

SNVT_switch

```
typedef struct
{
     unsigned value;
     signed state;
} SNVT_switch;
```

Valid range

`value`

0 .. 200	(0% .. 100%, 0,5% resolution)

`state`

0	Off
1	On (Note: If `value` = 0, output is off)
-1	Invalid value.

Default value

{ 0, -1 }	Invalid

3.4. nviOcSecondary

```
network input SNVT_occupancy nviOcSecondary;
```

This variable is used to define a second light level in a zone. When the value changes to `OC_OCCUPIED`, the value defined in `nciOcSecondVal` is loaded into `nvoOcLampValue`, if the value in `nviOcOccupancy` is `Oc_UNOCCUPIED`.

Type

SNVT_occupancy

```
typedef occup_t SNVT_occupancy;
```

Valid range

```
typedef enum occup_t
{
     /*  0 */  OC_OCCUPIED,
     /*  1 */  OC_UNOCCUPIED,
     …
     /* -1 */  OC_NUL = -1
} occup_t;
```

Default value

```
{-1}            OC_NUL
```

Notes

For all other enumerations of `SNVT_occupancy`, no action is taken by the occupancy controller.

3.5. nvoOcLampValue

```
network output SNVT_switch nvoOcLampValue;
```

This output network variable provides the state (ON or OFF) and the percentage level of intensity. The switching off time can be delayed with the configuration property `nciOcHoldTime`.

Type

SNVT_switch

```
typedef struct
{
     unsigned value;
     signed state;
} SNVT_switch;
```

Valid range

`value`

0 .. 200	(0% .. 100%, 0,5% resolution)

`state`

0	Off
1	On (Note: If `value` = 0, output is off)
-1	Invalid value.

Default value

{ 0, 0 }	Lamp Off

3.6. nvoOcSetting

```
network input SNVT_setting nvoOcSetting;
```

This output network variable defines the actual operating mode of the controller, following this table:

nviOcSetting	nviOcOccupancy	nvoOcSetting
SET_ON	OC_OCCUPIED	SET_ON
	OC_UNOCCUPIED	SET_OFF
SET_OFF	any	SET_OFF

The SET_OFF value can be delayed with the configuration parameter nciOcHoldTime.

Type

SNVT_setting

```
typedef struct
{
     setting_t function;
     unsigned short setting;
     signed long rotation;
} SNVT_setting;
```

Valid range

`function`

Value	Name	Description
0	SET_OFF	Turns Off the controller and network variable output
1	SET_ON	Turns On controller

The other enumeration values are not used.

`setting`

Not used.

`rotation`

Not used.

Default value

`{ SET_OFF, 0, 0 }`	Output is off

4. Configuration properties

4.1. nciOcHoldTime

```
network input SCPTholdTime cp nciOcHoldTime;
```

This configuration property is used to set the hold time value before switching off the lamp when the area is unoccupied. If the neighboring area is occupied the lamp is not switched off, but switched to the value specified by `nciOcSecondVal`.

When the hold time is elapsed, if the controller has been overridden before, it is automatically set back to the normal operating mode.

This configuration property is used to define the time between the controller changes to `OC_UNOCCUPIED` until the `nvoOcLampValue` changes automatically to Off. The value of this parameter also affects to the output variable `nvoOcSetting.`

Type

SCPTholdTime, derived from SNVT_time_sec.

Valid range

0 .. 6553,4 seconds (0,1 seconds resolution)

Default value

{ 600 } 60 seconds.

Notes

If this configuration property is defined with a value of 0 seconds, then the lamp output `nvoOcLampValue` and `nvoOcSetting` are turned off immediately when the controller changes to `OC_UNOCCUPIED`.

4.2. nciOcHeartBeat

```
network input SCPTmaxSendTime cp nciOcHeartBeat;
```

This configuration property defines the perod time between an automatic update of the network variables `nvoOcLampValue` and `nvoOcSetting`. This heartbit ensures that the controller is alive. It can be disabled setting the value to 0.

Type

SCPTmaxSendTime, derived from SNVT_time_sec.

Valid range

0 – 6553,4 by steps of 0.1 s

Default value

{ 1200 } 2 minutes

4.3. nciOcPrimeVal

```
network input SCPTprimeVal cp nciOcPrimeVal;
```

The value of this configuration property is used to undate `nvoOcLampValue` when the controller turns to Occupied state through `nviOcOccupancy`.

Type

SCPTprimeVal, derived from SNVT_switch.

```
typedef struct
{
        unsigned value;
        signed state;
} SNVT_switch;
```

Valid range

Value

0 ..200 (0%... 100%, resolution 0,5%)

State
0 OFF
1 ON (if value = 0 is OFF)
-1 Invalid

Default value

{1, 200 } ON, 100%.

4.4. nciOcSecondVal

`network input SCPTsecondVal cp nciOcSecondVal;`

The value of this configuration property is used to update `nvoOcLampValue` when the controller turns to Occupied state through the `nviOcSecondary` and the variable `nviOcOccupancy` is disabled.

Type

SCPTsecondVal, derived from SNVT_switch.

typedef struct
{
unsigned value;
signed state;
} SNVT_switch;

Valid range

Value

0 ..200 (0%... 100%, resolution 0,5%)

State
0 OFF
1 ON (if value = 0 is OFF)
-1 Invalid

Default value

{0, 0 } OFF, 0%.

4.5. nciOcUnocVal

`network input UCPTunOccupiedValcp nciOcUnocVal;`

This configuration property is used to update the variable `nvoOcLampValue` when the controller changes to Unoccupied state.

Type

UCPTunOccupiedVal, derived from SNVT_switch.

```
typedef struct
{
        unsigned value;
        signed state;
} SNVT_switch;
```

Valid range

Value

0 ..200 (0%... 100%, resolution 0,5%)

State
2 OFF
3 ON (if value = 0 is OFF)
-1 Invalid

Default value

{0, 0 } OFF, 0%.