EN – Software Description

Subject to technical alteration
Version 11.04.07

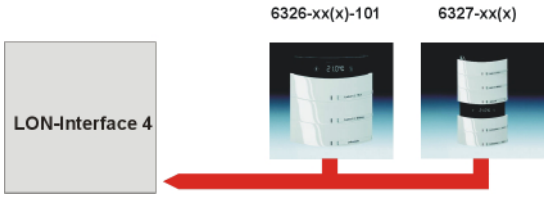
29100...

1 Application

By means of the LON-Interface 4 (bus coupling unit) devices of the Busch-Jaeger EIB programme:

- Triton: 6326-xx and 6327-xx

can be coupled to the building communication system LON of the company Echelon. The LON-Interface is designed as a gateway between the LON bus and the specific Busch-Jaeger device.



The defaults of the LonMark® functions profiles **3200** „Switch“ and **3250** „Scene Panel“ are considered. The function profile (UFTP) **20001** „SpaceComfortController“ is adapted to the LonMark® function profile **8500** „SpaceComfortController“ with regard to the defaults. For extended adjustment possibilities there are user configuration properties (UCPT). The UCPTs used are defined in the Thermokon Device Resource Files from Version 2.1 or higher.

2 Types Available

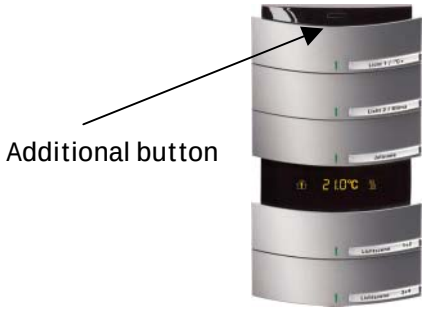
6326-xx Room temperature controller with triple tactile sensor with heating/cooling
6327-xx Room temperature controller with quintuple tactile sensor with heating/cooling
6327-xx with IR

- Universal operating element with room temperature control function, 6327 with IR-reception
- For heating and cooling (PI, PWM or 2-point)
- Button functions (switch/dim/blind/send value/light scene/ventilation function)
- Operating elements: tactile contacts left/right, also for set point and operating mode selection
- Display elements: display of operating mode, temperature, fan stage adjustment via LCD

3 Parameterization of Rockers

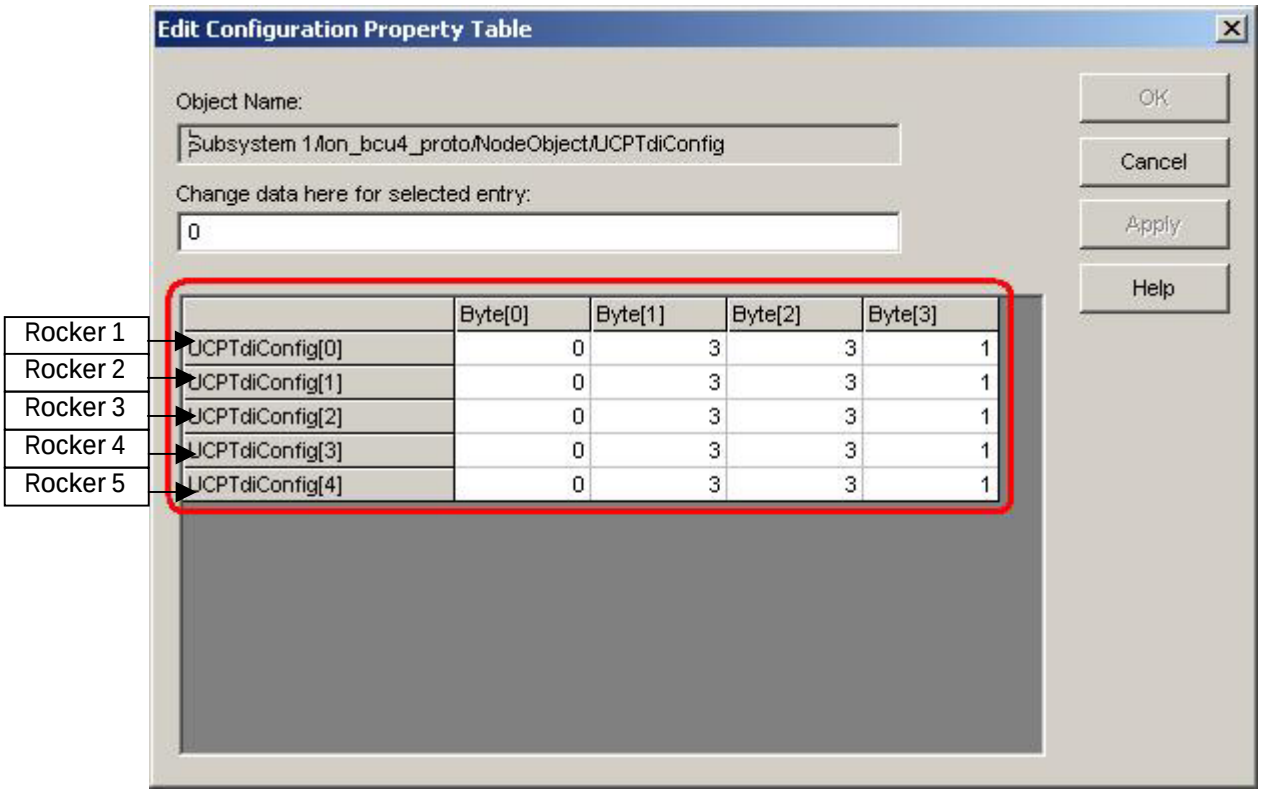
Set Point, Room Occupancy

In order to change the set point respectively the room occupancy, the operating mode must be changed by the additional button. The set point is fix allocated to rocker 1 and the room occupancy to rocker 2.



Button Function

For the parameterization of the individual rockers, the configuration property *UCPTdiConfig[...]* in the Node Object is used. When using the LONMaker, the configuration window shown below can be opened by a double click on the parameter.



Picture 3-1: LONmaker

UCPTdiConfig[0], UCPTdiConfig[1]

The rockers are configured as follows:

- Rocker 1 by UCPTdiConfig[0]
- Rocker 2 by UCPTdiConfig[1]
- Rocker 3 by UCPTdiConfig[2]
- Rocker 4 by UCPTdiConfig[3]
- Rocker 5 by UCPTdiConfig[4]

Byte[0]

The functions of the rocker are configured by the Byte[0]. The following functions are possible:

- 0 - switch
- 1 – dim
- 2 – blinds
- 3 - shutter
- 4 - scene
- 5 - fan stage adjustment

Example: Rocker 1 shall be configured as a dimming function and rocker 2 as a blind function.
UCPTdiConfig**[0]**.*Byte[0] = 1*
UCPTdiConfig**[1]**.*Byte[0] = 2*

Via byte[1] the function of the left rocker button is configured. Via byte[2] the function of the right rocker button is configured. For example, it is possible to select the left button to switch-off (Byte[1] = 6) or switch-on (Byte[1] = 4) the light. (see chapter:0)

Example: The left button shall switch-on the light/ light it up and the right button shall switch-off the light/ darken the light/ dim the light.
UCPTdiConfig**[0]**.*Byte[1] = 3*
UCPTdiConfig**[0]**.*Byte[2] = 7*

Byte[3]

The LED of the rocker is set in Byte[3]. Thus, it can be selected whether the colour shall be changed between red and green with a button actuation (see chapter 4.13).

Example: The LED shall be red with OFF and green with ON.
UCPTdiConfig**[0]**.*Byte[3] = 1*

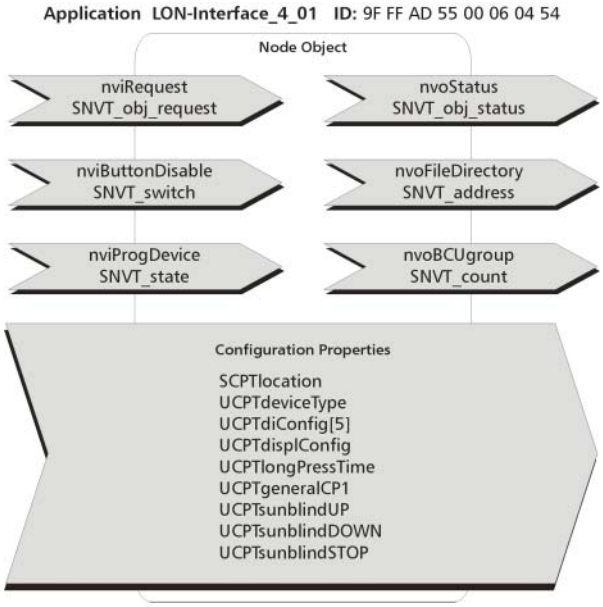
Data Transmission

After having changed the parameters they must be transmitted to the operating element. By setting nviProgDevice.bit0 from 0 to 1 respectively after a voltage reset the new settings are taken over in the control element.

4 Software Description

4.1 Node Object

The Node Objekt supervises and controls the functions of the individual objects in the device. The basic functions required by the LonMark® are supported, whereas general network variables and configuration properties for control and parameterization of the device were added.



Data Transmission:
After a parameterization, *nviProgDevice.bit0* must be set from 0 to 1 , so that the new parameters are taken over. They are also taken over after a voltage reset.

Data Display:
Beside the room temperature the set point can be displayed as well. This configuration is made via *UCPTdisplConfig*.

Button Functionality and Devices Types:
During installation, the used device type (6326-xx, 6327-xx) must be set via the paramenter UCPTdeviceType. It is possible to allocate different functions to the buttons, e.g. light control or scene inquiry.
By *UCPTdiConfig[0],UCPTdiConfig[1],UCPTdiConfig[2], UCPTdiConfig[3]* and *UCPTdiConfig[4]* the configuraiton of the rockers 1, 2, 3, 4 and 5 is made.

4.1.1 Input Variables Node Object:

nviRequest
SNVT Type: SNVT_obj_request, Index 92
Function: Input variable including the functions RQ_NORMAL, RQ_UPDATE_STATUS and RQ_REPORT_MASK.

nviButtonDisable
SNVT Typ: SNVT_switch, Index 95
Funktion: Input variable for diableing the buttons. (Only for Switch, Dim, Blind, Shutter, Scene)

nviProgDevice
SNVT Type: SNVT_state, Index 83
Function: Input variable for parameterization of the control element. After the parameterization *nviProgDevice.bit0* must be set from 0 to 1, so that the new parameters are taken over in the control element.
nviProgDevice.bit0 = 0 ➔ nviProgDevice.bit0 = 1
Acceptance of the new parameters.

4.1.2 Output Variables Node Object:

nvoStatus
SNVT Type: SNVT_obj_status, Index 93
Function: Output variable including the requested status bits „invalid_id“ and „invalid_request“.

nvoFileDirectory
SNVT Type: SNVT_address, Index 114
Function: The output variable makes the address data of the configuration property in the device available to the LON integration tool.

nvoBCUgroup
SNVT Type: SNVT_count, Index 8
Function: Output variable with the LON-Interface type. The LON-Interface type cannot be changed.

4.1.3 Configuration Properties Node Object:

SCPTlocation
SCPT Index: 17, SNVT_str_asc
Function: Additional input option to save information on the location in the device.

UCPTdeviceType
UCPT Index: 42, SNVT_count
Function: The software adaption is made via this configuration property. Valid input values are:

6326-xx	==>	6326
6327-xx	==>	6327

Preset value: 0

UCPTdisplConfig
UCPT Index: 46, SNVT_state
Function: By UCPTdisplConfig the display form can be configured.

Temperature:

UCPTdisplConfig. bit0	bit1		
0	0*	==>	display room temperature
0	1	==>	no temperature display
1	0	==>	display set temperature

Display settings:

Bit2	Bit3		
0	0	==>	display illumination always OFF
0	1*	==>	display illumionation only during operation
1	0	==>	display illumionation off during night operation
1	1	==>	display illumionation always ON

* = Preset values

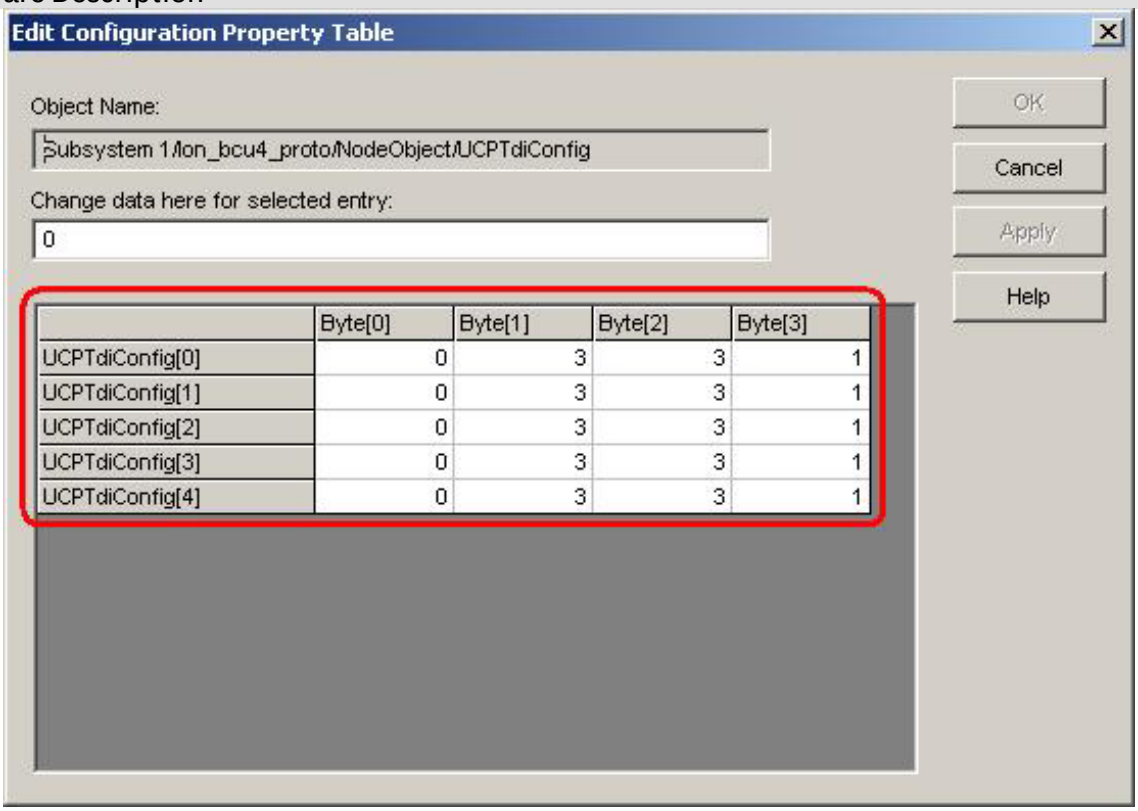
UCPTgeneralCP1
UCPT Index: 7, SNVT_state
Function: Selection of IR-range by a remote control. (IR-function is only with 6327 possible)

bit0		
0	➔	White
1	➔	Blue

UCPTlongPressTime
UCPT Index: 71, typedef struct { SNVT_time_sec dimming; SNVT_time_sec sunblind; SNVT_time_sec scene; SNVT_time_sec universal; }
Function: By means of this configuration property the time for dimming, blinds, scene and universal can be input by a long button actuation.
As for model 6326, the long button actuation for blinds is fixed to 900 ms.
(preset value: 1.0;2.0;2.0;2.0)

UCPTdiConfig[0]...[4]
UCPT Index: 44, typedef struct {unsigned short Byte[4]} UNVT_str_hex4
Function: This configuration property determines the rocker functions and their allocation to the output variables as well as the control logic of the response LED. UCPTdiConfig[0] is fix allocated to rocker 1, UCPTdiConfig[1] to rocker 2, UCPTdiConfig[2] to rocker 3, UCPTdiConfig[3] to rocker 4, UCPTdiConfig[4] to rocker 5 .

UCPTdiConfig[0]	configured	rocker 1
UCPTdiConfig[1]	configured	rocker 2
UCPTdiConfig[2]	configuredr	rocker 3
UCPTdiConfig[3]	configured	rocker 4
UCPTdiConfig[4]	configured	rocker 5
UCPTdiConfig[x] .*Byte[0]*	configured	Function of rocker
UCPTdiConfig[x] .*Byte[1]*	configured	Function of left button
UCPTdiConfig[x] .*Byte[2]*	configured	Function of right button
UCPTdiConfig[x] .*Byte[3]*	configured	Function of LED



Picture 4-1: UCPTdiconfig

UCPTsunblindUP
UCPT Index: 72, SNVT_setting
Function: By means of this configuration parameter you can adjust which SNVT_setting value shall be sent when the blind/shutter is going up. (Preset value: SET_UP 100.0 0.0)

UCPTsunblindDOWN
UCPT Index: 73, SNVT_setting
Function: By means of this configuration parameter you can adjust which SNVT_setting value shall be sent when the blind/shutter is going down. (Preset value: SET_DOWN 100.0 0.0)

UCPTsunblindSTOP
UCPT Index: 74, SNVT_setting
Function: By means of this configuration parameter you can adjust which SNVT_setting value shall be sent when the blind/shutter is stopped. (Preset value: SET_STOP 0.0 0.0)

UCPTsunblindIDLE
UCPT Index: 75, SNVT_setting
Function: By means of this configuration paramenter you can adjust which SNVT_setting value shall be sent for the stand-by mode of the blind/shutter. *UCPTsunblindIDLE* is sent 500ms after the stop of the blind/shutter, if *UCPTsunblindIDLE* is unequal *UCPTsunblindSTOP*. (Preset value: SET_NUL 0.0 0.0)

6326-xx Triton Triple Room Temperature Controller

A rocker is fix allocated to a switch object.

Function: Switch light / buttons: Byte[0] = 0:

Configuration of the rocker for lighting ON / OFF by UCPTdiConfig							
Allocation of the rocker function block		Function left button		Function right button		Function LED	
Byte[0]	Description	Byte[1]	Description	Byte[2]	Description	Byte[3]	Description
0	Switching / Buttons	1	Light only ON (Switch_A_n)	1	Light only ON (Switch_A_n)	1	Left rocker red Right rocker green
		2	Light only OFF(Switch_A_n)	2	Light only OFF (Switch_A_n)	2	Left rocker green Right rocker red
		3	Command Automatic (= 0.0 – 1) (Switch_A_n)	3	Command Automatic (= 0.0 – 1) (Switch_A_n)	3	Left rocker red Right rocker red
						4	Left rocker green Right rocker red

Examples:

A: Rocker 1 shall be configured as a switching function. The left button turns the light on and the right button turns the light off. The LED shall be red and green.

UCPTdiConfig[0].*Byte[0] = 0* ➔ switch
UCPTdiConfig[0].*Byte[1] = 2* ➔ left ON
UCPTdiConfig[0].*Byte[2] = 4* ➔ right OFF
UCPTdiConfig[0].*Byte[3] = 1* ➔LED

6327-xx Triton Quintiple Room Temperature Controller

A rocker is fix allocated to a switch object. Each object has two output variables (xxx_A; xxx_B). Thus, a individual switch object can be allocated to each button. This is useful for a toggle function. (see chapter3)

Function: Switch light / buttons: Byte[0] = 0:

Configuration of the rocker for lighting ON / OFF by UCPTdiConfig							
Allocation of the rocker function block		Function left button		Function right button		Function LED	
Byte[0]	Description	Byte[1]	Description	Byte[2]	Description	Byte[3]	Description
0	Switching / Buttons	1	Light toggle ON / OFF (Switch_A_n)	1	Light toggle ON / OFF (Switch_B_n)	1	Left rocker red Right rocker green
		2	Light only ON (Switch_A_n)	2	Light only ON (Switch_A_n)	2	Left rocker green Right rocker red
		3	Light only ON (Switch_B_n)	3	Light only ON (Switch_B_n)	3	Left rocker red Right rocker red
		4	Light only OFF(Switch_A_n)	4	Light only OFF (Switch_A_n)	4	Left rocker green Right rocker red
		5	Light only OFF (Switch_B_n)	5	Light only OFF (Switch_B_n)		
		6	Command Automatic (= 0.0 –1) (Switch_A_n)	6	Command Automatic (= 0.0 –1) (Switch_A_n)		
		7	Command Automatic (= 0.0 –1) (Switch_B_n)	7	CommandI Automatic (= 0.0 –1) (Switch_B_n)		

Examples:

A: Rocker 1 shall be configured as a switching function. The left button turns the light on and the right button turns the light off. The LED shall be red and green.

UCPTdiConfig[0].*Byte[0] = 0* ➔ switch
UCPTdiConfig[0].*Byte[1] = 2* ➔ left ON
UCPTdiConfig[0].*Byte[2] = 4* ➔ right OFF
UCPTdiConfig[0].*Byte[3] = 1* ➔ LED

B: Rocker 2 shall be configured as a switching function. The left and right button toggle the light on and off. The LED shall be green.

UCPTdiConfig[1].*Byte[0] = 0* ➔ switch
UCPTdiConfig[1].*Byte[1] = 1* ➔ toggle left
UCPTdiConfig[1].*Byte[2] = 1* ➔ toggle right
UCPTdiConfig[1].*Byte[3] = 4* ➔ LED

C: Rocker 3 shall be configured as a switching function. The left button toggles the light on and off. The right button puts the lighting in the automatic mode. The LED shall be green in the toggle mode and red in the automatic mode.

UCPTdiConfig[2].*Byte[0] = 0* ➔ switch
UCPTdiConfig[2].*Byte[1] = 1* ➔ toggle left
UCPTdiConfig[2].*Byte[2] = 6* ➔ toggle automatic
UCPTdiConfig[2].*Byte[3] = 2* ➔ LED

Function: Dim Light: Byte[0] = 1:

6326-xx Triton Triple Room Temperature Controller

Configuration of rockers for lighting with dimming by UCPTdiConfig							
Allocation rocker function block		Function left button		Function right button		Function LED	
Byte[0]	Description	Byte[1]	Description	Byte[2]	Description	Byte[3]	Description
1	Lighting by dimming	1	Light brighter by dimming(Switch_A_n)	1	Light brighter by dimming (Switch_A_n)	1	Ligth on red Ligth off green
			Turn-on value = Max-value		Turn-on value = Max-value		
		2	Light brighter by dimming (Switch_A_n)	2	Light brighter by dimming (Switch_A_n)	2	Ligth on green Ligth off red
			Turn-on value = last On-value		Turn-on value = last On-value		
		3	Light darker by dimming (Switch_A_n)	3	Light darker by dimming only (Switch_A_n)	3	Left rocker red Right rocker red
						4	Left rocker green Right rocker green

Example:

A: Rocker 1 shall be configured as a dimming function. Short button actuations result in a toggling of the switch status. The left button dims the light brighter and the right button darker. The LED shall be red and green.

UCPTdiConfig[0].*Byte[0] = 1* ➔ Dim
UCPTdiConfig[0].*Byte[1] = 1* ➔ Left Brighter
UCPTdiConfig[0].*Byte[2] = 3* ➔ Right Darker
UCPTdiConfig[0].*Byte[3] = 1* ➔ LED

6327-xx Triton Quintuple Room Temperature Controller

Configuration of the rocker for lighting with dimming by UCPTdiConfig							
Allocation of rocker function block		Function left button		Function right button		Function LED	
Byte[0]	Description	Byte[1]	Description	Byte[2]	Description	Byte[3]	Description
1	Lighting with Dimming	1	Toggle light by dimming (Switch_A_n) Turn-on value = Max-value	1	Toggle light by dimming (Switch_B_n) Turn-on = Max-value	1	Left rocker red Right rocker green
		2	Toggle light by dimming (Switch_A_n) Turn-on value = last ON value	2	Toggle light by dimming (Switch_B_n) Turn-on value = last ON value	2	Left rocker green Right rocker red
		3	Light brighter only by dimming (Switch_A_n) Turn-on value = Max-value	3	Light brighter only by dimming (Switch_A_n) Turn-on value = Max-value	3	Left rocker red Right rocker red
		4	Light only brighter by dimming (Switch_B_n) Turn-on value = Max-value	4	Light only brighter by dimming (Switch_B_n) Turn-on value = Max-value	4	Left rocker green Right rocker green
		5	Light only brighter by dimming (Switch_A_n) Turn-on value = last ON value	5	Light only brighter by dimming (Switch_A_n) Turn-on value = last ON value		
		6	Light only brighter by dimming (Switch_B_n) Turn-on value= last ON value	6	Light only brighter by dimming (Switch_B_n) Turn-on value = last ON value		
		7	Light only darker by dimming(Switch_A_n)	7	Light only darker by dimming (Switch_A_n)		
		8	Light only darker by dimming (Switch_B_n)	8	Light only darker by dimming (Switch_B_n)		
		9	Command Automatic (= 0.0 – 1) (Switch_A_n)	9	Command Automatic (= 0.0 – 1) (Switch_A_n)		
		A	Command Automatic (= 0.0 – 1) (Switch_B_n)	A	Command Automatic (= 0.0 – 1) (Switch_B_n)		

Examples:

A: Rocker 1 shall be configured as a dimming function. Thus, the left button turns the light on/brighter and the right button turns the light off/darker. The LED shall be red and green.

UCPTdiConfig[0].*Byte[0] = 1* ➔ Dim
UCPTdiConfig[0].*Byte[1] = 3* ➔ Left: On/Brighter
UCPTdiConfig[0].*Byte[2] = 7* ➔ Right: Off/Darker
UCPTdiConfig[0].*Byte[3] = 1* ➔ LED

B: Rocker 2 shall be configured as a switching function. Thus, the left button toggles the light on by *nvoSwitch_A_n* and the right button toogles the light off by *nvoSwitch_B_n.* The LED shall be green.

UCPTdiConfig[1].*Byte[0] = 1* ➔ Dim
UCPTdiConfig[1].*Byte[1] = 1* ➔ Left: Toggle by dimming
UCPTdiConfig[1].*Byte[2] = 1* ➔ Right: Toggle by dimming
UCPTdiConfig[1].*Byte[3] = 4* ➔ LED

Function Blinds: Byte[0] = 2:

Configuration of button for blind with UCPTdiConfig							
Allocation of rocker function block		Function left button		Function right button		Function LED	
Byte[0]	Desription	Byte[1]	Desription	Byte[2]	Description	Byte[3]	Description
2	Blind	1	Blind Up (Setting_A_n)	1	Blind Up (Setting_A_n)	1	Left rocker red Right rocker green
		2	Blind Down(Setting_A_n)	2	Blind Down (Setting_A_n)	2	Left rocker green Right rocker red
		3	Blind Up (Setting_B_n)	3	Blind Up (Setting_B_n)	3	Left rocker red Right rocker red
		4	Blind Down(Setting_B_n)	4	Blind Down (Setting_B_n)	4	Left rocker green Right rocker green

Example: Rocker 1 shall be configured as a blind function. Thus, the left button switchs the blind up and the right button switchs the blind down. The LED shall be red.

UCPTdiConfig[0].*Byte[0] = 2* ➔ Blind
UCPTdiConfig[0].*Byte[1] = 1* ➔ Left: Up
UCPTdiConfig[0].*Byte[2] = 2* ➔ Right: Down
UCPTdiConfig[0].*Byte[3] = 3* ➔ LED

Function: Shutter: Byte[0] = 3:

Configuration of rocker for shutter with UCPTdiConfig							
Allocation rocker function block		Function left button		Function right button		Function LED	
Byte[0]	Description	Byte[1]	Description	Byte[2]	Description	Byte[3]	Description
3	Shutter	1	Shutter Up (Setting_A_n)	1	Shutter Up (Setting_A_n)	1	Left rocker red Right rocker green
		2	Shutter Down (Setting_A_n)	2	Shuter Down (Setting_A_n)	2	Left rocker green Right rocker red
		3	Shutter Up (Setting_B_n)	3	Shutter Up (Setting_B_n)	3	Left rocker red Right rocker red
		4	Shutter Down (Setting_B_n)	4	Shuter Down (Setting_B_n)	4	Left rocker red Right rocker red

Example: Rocker 3 shall be configured as a shutter function. Thus, the left button switchs the shutter up and the right button switchts the shutter down. The LED shall be red.

UCPTdiConfig[2].*Byte[0] = 3* ➔ Shutter
UCPTdiConfig[2].*Byte[1] = 1* ➔ Left Up
UCPTdiConfig[2].*Byte[2] = 2* ➔ Right Down
UCPTdiConfig[2].*Byte[3] = 3* ➔ LED

Function Scene Control: Byte[0] = 4:

Configuration of button for scene control with UCPTdiConfig							
Allocation of rocker function block		Function left button		Function right button		Function	
Byte[0]	Description	Byte[1]	Description	Byte[2]	Description	Byte[3]	Description
4	Scene	01-FE	Scene number	01-FE	Scene number	1	Left rocker red Right rocker green
						2	Left rocker green Right rocker red
						3	Left rocker red Right rocker red
						4	Left rocker green Right rocker green

Example: Rocker 2 shall be configured as a scene call. Thus, the left button calls scene 5 and the right button calls scene 12. The LED shall be red.

UCPTdiConfig[1].*Byte[0] = 4* ➔ Scene
UCPTdiConfig[1].*Byte[1] = 5 dez* ➔ Call scene 5
UCPTdiConfig[1].*Byte[2] = 12 dez* ➔ Call scene 12
UCPTdiConfig[1].*Byte[3] = 3* ➔ LED

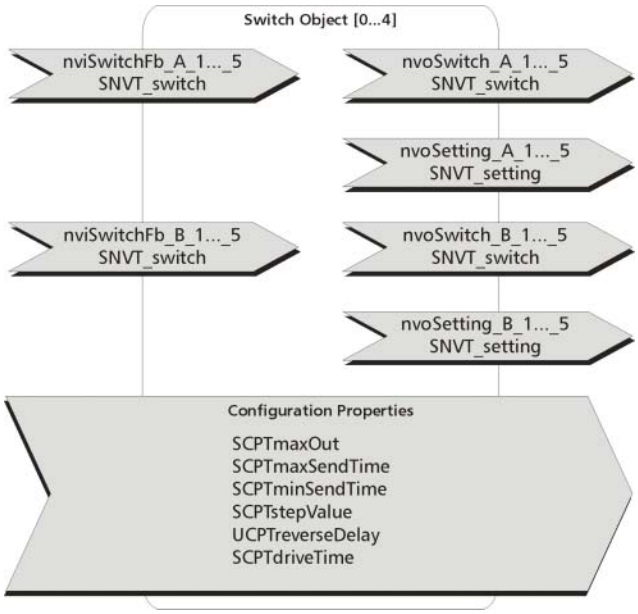
Function Fan Stage: Byte[0] = 5:
The fan stage adjustment is continuously changed in 5 % steps. An automatic operation is not possible.

Configuration of rocker for continuous fan stage control by UCPTdiConfig							
Allocation rocker function block		Function left button		Function right button		Function LED	
Byte[0]	Description	Byte[1]	Description	Byte[2]	Description	Byte[3]	Description
5	Fan stage	1	Fan stage -5 % (nvoFanSpeed)	1	Fan stage +5 % (nvoFanSpeed)	1	Left rocker red Right rocker green
						2	Left rocker green Right rocker red
						3	Left rocker red Right rocker red
						4	Left rocker green Right rocker green

Example: Rocker 2 shall be configured for a fan stage adjustment. The LED shall be red.

UCPTdiConfig**[1]**.*Byte[0] = 5* ➔ fan stage adjustment
UCPTdiConfig**[1]**.*Byte[1] = 1* ➔ left + 5 % steps
UCPTdiConfig**[1]**.*Byte[2] = 1* ➔ right - 5 % steps
UCPTdiConfig**[1]**.*Byte[3] = 3* ➔ LED

4.2 Switch Object



Five identical switch objects for light and blind control. The switch objects are directly allocated to the rockers:

Switch Object[0] = rocker 1
Switch Object[1] = rocker 2
Switch Object[2] = rocker 3
Switch Object[3] = rocker 4
Switch Object[4] = rocker 5

The allocation of the rockers to the individual functions is made as described above by means of the configuration properties UCPTdiConfig[0]...[4] in the Node Object. Possible functions are light control ON/OFF, light control ON/OFF by dimming or blinds UP/DOWN by automatic run. Each switch object has 2 variable groups which are distinguished by the appendix *xxx_A* and *xxx_B.* Thus, a network variable can be allocated to each button.

4.2.1 Input Variables Switch Object:

nviSwitchFb_A_1...5, nviSwitchFb_B_1...5
SNVT Type: SNVT_switch, Index 95
Function: Input variable for the current status of the light groups controlled by nvoSwitch_A_1...5 respectively nvoSwitchFb_B_1...5.

4.2.2 Output Variables Switch Object:

nvoSwitch_A_1...5, nvoSetting_A_1...5, nvoSwitch_B_1...5, nvoSetting_B_1...5
SNVT Type: SNVT_switch, Index 95; SNVT_setting, Index 117
Function:Output variable for control of light groups and of light and blind controllers. The output values are depending on the function setting. A switch object can be allocated to each rocker. Each switch object has two different variable groups, which are distinguished by the appendix *_A* respectively *_B* . Thus, a network variable can be allocated to each button.
BY
UCPTdiConfig[x].Byte[0] a function is allocated to the rockers.
UCPTdiConfig[x].Byte[1] a function is allocated to the left button.
UCPTdiConfig[x].Byte[2] a function is allocated to the right button.
UCPTdiConfig[x].Byte[3] a function is allocated to the LED

Switching/ Buttons

UCPTdiConfig[x].Byte[0] = 00_{hex}

Button pressed/ not pressed

UCPTdiConfig[x].Byte[1...2] = 01_{hex}

Button pressed	nvoSwitch_A_x.value	= SCPTmaxOut
	nvoSwitch_A_x.state	= 1
	nvoSetting_A_x.function	= SET_ON;
	nvoSetting_A_x.setting	= SCPTmaxOut;
Button not pressed	nvoSwitch_A_x.value	= 0
	nvoSwitch_A_x.state	= 0
	nvoSetting_A_x.function	= SET_OFF;
	nvoSetting_A_x.setting	= 0;

UCPTdiConfig[x].Byte[1...2] = 02_{hex}

Button pressed	nvoSwitch_B_x.value	= SCPTmaxOut
	nvoSwitch_B_x.state	= 1
	nvoSetting_B_x.function	= SET_ON;
	nvoSetting_B_x.setting	= SCPTmaxOut;
Button not pressed	nvoSwitch_B_x.value	= 0
	nvoSwitch_B_x.state	= 0
	nvoSetting_B_x.function	= SET_OFF;
	nvoSetting_B_x.setting	= 0;

Toggle of Lighting

UCPTdiConfig[x].Byte[1] = 03_{hex}

Each button actuation results in a toggling of the lighting, i.e. between ON and OFF

Lighting ON	nvoSwitch_A_x.value	= SCPTmaxOut
	nvoSwitch_A_x.state	= 1
	nvoSetting_A_x.function	= SET_ON;
	nvoSetting_A_x.setting	= SCPTmaxOut;
Lighting OFF	nvoSwitch_A_x.value	= 0
	nvoSwitch_A_x.state	= 0
	nvoSetting_A_x.function	= SET_OFF;
	nvoSetting_A_x.setting	= 0;

UCPTdiConfig[x].Byte[2] = 03_{hex}

Each button actuation results in a toggling of the lighting, i.e. between ON and OFF.

Lighting ON	nvoSwitch_B_x.value	= SCPTmaxOut
	nvoSwitch_B_x.state	= 1
	nvoSetting_B_x.function	= SET_ON;
	nvoSetting_B_x.setting	= SCPTmaxOut;
Lighting OFF	nvoSwitch_B_x.value	= 0
	nvoSwitch_B_x.state	= 0
	nvoSetting_B_x.function	= SET_OFF;
	nvoSetting_B_x.setting	= 0;

Lighting ON

UCPTdiConfig[x].Byte[1...2] = 04_{hex}

Each button actuation leads to a switching-on of the lighting.

Lighting ON	nvoSwitch_A_x.value	= SCPTmaxOut
	nvoSwitch_A_x.state	= 1
	nvoSetting_A_x.function	= SET_ON;
	nvoSetting_A_x.setting	= SCPTmaxOut;

UCPTdiConfig[x].Byte[1...2] = 05_{hex}

Each button actuation results in a switching-on of the lighting.

Lighting ON	nvoSwitch_B_x.value	= SCPTmaxOut
	nvoSwitch_B_x.state	= 1
	nvoSetting_B_x.function	= SET_ON;
	nvoSetting_B_x.setting	= SCPTmaxOut;

Lighting OFF

UCPTdiConfig[x].Byte[1...2] = 06_{hex}

Each button actuation results in a switching-off of the lighting.

Lighting OFF	nvoSwitch_A_x.value	= 0
	nvoSwitch_A_x.state	= 0
	nvoSetting_A_x.function	= SET_OFF;
	nvoSetting_A_x.setting	= 0;

UCPTdiConfig[x].Byte[1...2] = 07_{hex}

Each button actuation results in a switching-off of the lighting.

Lighting OFF	nvoSwitch_B_x.value	= 0
	nvoSwitch_B_x.state	= 0
	nvoSetting_B_x.function	= SET_OFF;
	nvoSetting_B_x.setting	= 0;

Automatic

UCPTdiConfig[x].Byte[1..2] = 08_{hex}

An actuation of an “automatic button“ switchs the variables nvoSwitch_A_x to the value 0,0 -1. Thus, for example a light controller can be reset from an external override to the automatic mode.

UCPTdiConfig[x].Byte[1...2] = 08_{hex}

The actuation of an “automatic button“ switchs the variables nvoSwitch_B_x to the value 0,0 -1. Thus, for example a light controller can be reset from an external override to the automatic mode.

Dimming

6327-xx Triton Qunituple Room Temperature Controller

UCPTdiConfig[x].Byte[0] = 01_{hex}

Toggle lighting by dimming, turn-on value = max. value

Short button acutations (< 1 s) result in a toggling of the current lighting status, whereas the .value –turn-on value always is SCPTmaxOut. By longer button actuations (> 1 s) the dimming function is activated, i.e. based on the current lighting status, the .value-value of the switch variables is raised or lowered in precent steps of UCPTstepValue as long as the button is pressed.

UCPTdiConfig[x].Byte[1] = 01_{hex}

Lighting to maximum value	nvoSwitch_A_x.value	= SCPTmaxOut
	nvoSwitch_A_x.state	= 1
Lighting on 50%	nvoSwitch_A_x.value	= 50,0
	nvoSwitch_A_x.state	= 1
Lighting OFF	nvoSwitch_A_x.value	= 0
	nvoSwitch_A_x.state	= 0

The variables nvoSetting_A_x are not changed in that setting.

UCPTdiConfig[x].Byte[2] = 01_{hex},

Lighting to maximum value	nvoSwitch_B_x.value	= SCPTmaxOut
	nvoSwitch_B_x.state	= 1
Lighting on 50%	nvoSwitch_B_x.value	= 50,0
	nvoSwitch_B_x.state	= 1
Lighting OFF	nvoSwitch_B_x.value	= 0
	nvoSwitch_B_x.state	= 0

The variables nvoSetting_B_x are not changed in that setting.

Toggle lighting by dimming, turn-on value = last on-value

UCPTdiConfig[x].Byte[1] = 02_{hex}

Function as with 01hex, but with the difference, that not the value SCPTmaxOut but the last turn-on value is taken over. The smallest turn-on value is limited to 20%.

Lighting ON by dimming brighter, turn-on value = max. value

If the lighting is switched-off, a button actuation results in an immediate switching-on of the lighting. By longer button actuations (> 1 s) the function "dim brighter" is activated, i.e. based on the current light status the .value – value of the switch variable is reduced in precent steps of UCPTstepValue as long as the maximum value SCPTmaxOut is reached. The sending interval in the mode dimming is adjusted by SCPTminSendTime and is preadjusted to approx. 300ms.

UCPTdiConfig[x].Byte[1...2] = 03_{hex}

Switch-on of lighting	nvoSwitch_A_x.value	= SCPTmaxOut
	nvoSwitch_A_x.state	= 1
	nvoSetting_A_x.function	= SET_ON;
	nvoSetting_A_x.setting	= SCPTmaxOut;
Dim lighting brighter	nvoSwitch_A_x.value	= last value + UCPTstepValue
	nvoSwitch_A_x.state	= 1
	nvoSetting_A_x.function	= SET_UP;
	nvoSetting_A_x.setting	= UCPTstepValue;

UCPTdiConfig[x].Byte[1...2] = 04_{hex}

Switch-on of lighting	nvoSwitch_B_x.value	= SCPTmaxOut
	nvoSwitch_B_x.state	= 1
	nvoSetting_B_x.function	= SET_ON;
	nvoSetting_B_x.setting	= SCPTmaxOut;
Dim lighting brighter	nvoSwitch_B_x.value	= last value + UCPTstepValue
	nvoSwitch_B_x.state	= 1
	nvoSetting_B_x.function	= SET_UP;
	nvoSetting_B_x.setting	= UCPTstepValue;

Lighting ON by dimming brighter, turn-on value = last on-value

UCPTdiConfig[x].Byte[1...2] = 05_{hex}, 06_{hex},

Function as with 03hex, but with the difference, that not the value SCPTmaxOut is taken over when switching-on the light, but the last turn-on value. The smallest turn-on value is limited to 20%.

Lighting OFF by dimming darker

If the lighting is turned-on, a short button actuation (< 1 s) leads to an immediate switching-off of the lighting. By longer button actuations (> 1 s) the function "dim darker" is activated, i.e. based on the current lighting status the .value –value of the switch variables is reduced in percent steps of UCPTstepValue as long as the value 0 is reached. The sending interval in the mode dimming is adjusted by SCPTminSendTime and amounts to approx. 300ms preset.

UCPTdiConfig[x].Byte[1...2] = 07_{hex}

Switch-off lighting	nvoSwitch_A_x.value	= 0
	nvoSwitch_A_x.state	= 0
	nvoSeting_A_x.function	= SET_OFF;
	nvoSetting_A_x.setting	= 0;
Dim lighting darker	nvoSwitch_A_x.value	= last value - UCPTstepValue
	nvoSwitch_A_x.state	= 1
	nvoSetting_A_x.function	= SET_DOWN;
	nvoSetting_A_x.setting	= UCPTstepValue;

UCPTdiConfig[x].Byte[1,2] = 08_{hex}

Switching-off of lighting	nvoSwitch_B_x.value	= 0
	nvoSwitch_B_x.state	= 0
	nvoSeting_B_x.function	= SET_OFF;
	nvoSetting_B_x.setting	= 0;
Dim lighting darker	nvoSwitch_B_x.value	= last value - UCPTstepValue
	nvoSwitch_B_x.state	= 1
	nvoSetting_B_x.function	= SET_DOWN;
	nvoSetting_B_x.setting	= UCPTstepValue;

Automatic

UCPTdiConfig[x].Byte[1..2] = 09_{hex}

The actuation of an "automatic button" switchs the variable nvoSwitch_A_x to the value 0,0 -1. Thus, a light controller can be reset from an external override to the automatic mode.

UCPTdiConfig[x].Byte[1...2] = $0A_{hex}$

The actuation of an "automatic button" switchs the variable nvoSwitch_B_x to the value 0,0 -1. Thus, a light controller can be reset from external override to the automatic mode.

6326-xx Triton Triple Room Temperature Controller

UCPTdiConfig[x].Byte[0] = 01_{hex}

Lighting brighter by dimming, turn-on value = max. value

Short button actuations (< 1 s) lead to a toggling of the current lighting status, whereas the.value – turn-on value is always SCPTmaxOut. By longer button actuations (> 1 s) the function „dim brighter" is activated, i.e. based on the current lighting status the .value – value of the switch variables is increased in percent steps of UCPTstepValue as long as the maximum value SCPTmaxOut is reached. The sending interval in the dimming mode is set by SCPTminSendTime and amounts to approx. 300ms preset.

UCPTdiConfig[x].Byte[1] = 01_{hex}

Switching-on of lighting	nvoSwitch_A_x.value	= SCPTmaxOut
	nvoSwitch_A_x.state	= 1
	nvoSetting_A_x.function	= SET_ON;
	nvoSetting_A_x.setting	= SCPTmaxOut;
Lighting OFF	nvoSwitch_A_x.value	= 0
	nvoSwitch_A_x.state	= 0
	nvoSeting_A_x.function	= SET_OFF;
	nvoSetting_A_x.setting	= 0;
Dim lighting brighter	nvoSwitch_A_x.value	= last value + UCPTstepValue
	nvoSwitch_A_x.state	= 1
	nvoSetting_A_x.function	= SET_UP;
	nvoSetting_A_x.setting	= UCPTstepValue;

Lighting ON by dimming brighter, turn-on value = last ON-value

UCPTdiConfig[x].Byte[1,2] = 02_{hex}

Function as with 01hex with the only difference that not the value SCPTmaxOut is used when switching-on the lighting, but the last turn-on value. The smallest turn-on value is limited to 20%.

Lighting darker by dimming

Short button actuations (< 1 s) lead to a toggling of the current lighting status, whereas the.value – turn-on value always is SCPTmaxOut. By longer button actuations (> 1 s) the function "dim darker" is activated, i.e. based on the current lighting status, the .value – value of the switch variables is reduced in percent steps of UCPTstepValue as long as the value 0 is reached. The sending interval in the dimming mode is set by SCPTminSendTime and amounts to approx. 300ms preset.

UCPTdiConfig[x].Byte[1,2] = 03_{hex}

Switching-on of lighting	nvoSwitch_A_x.value	= SCPTmaxOut
	nvoSwitch_A_x.state	= 1
	nvoSetting_A_x.function	= SET_ON;
	nvoSetting_A_x.setting	= SCPTmaxOut;
Switching-off of lighting	nvoSwitch_A_x.value	= 0
	nvoSwitch_A_x.state	= 0
	nvoSeting_A_x.function	= SET_OFF;
	nvoSetting_A_x.setting	= 0;
Lighting darker by dimming	nvoSwitch_A_x.value	= last value - UCPTstepValue
	nvoSwitch_A_x.state	= 1
	nvoSetting_A_x.function	= SET_DOWN;
	nvoSetting_A_x.setting	= UCPTstepValue;

Blind

UCPTdiConfig[x].Byte[0] = 02_{hex}

Blind UP

In the configuration mode “blind UP“ only the nvoSetting variables are changed and sent. Short button actuations (< 2 s) are used for a fine adjustment of the lamellas. A long button acutation (>2 s) starts the automatic run and drives the blind contiuously in the direction open for the time SCPTdriveTime. The automatic run can be stopped by a renewed button actuation.

UCPTdiConfig[x].Byte[1…2] = 01_{hex}

Open blind	nvoSetting_A_x.function	= UCPTsunblindUP;
Stop blind	nvoSetting_A_x.function	= UCPTsunblindSTOP;

UCPTdiConfig[x].Byte[1…2] = 03_{hex}

Open blind	nvoSetting_B_x.function	= UCPTsunblindUP;
Stop blind	nvoSetting_B_x.function	= UCPTsunblindSTOP;

With a delay of 500ms the command *UCPTsunblindIDLE* for idle mode is sent after the command *UCPTsunblindSTOP* if *UCPTsunblindIDLE* is unequal to *UCPTsunblindSTOP.*

Blind DOWN

In the configuration mode “blind DOWN“ only the nvoSetting variables are changed and sent. Short button actuations (< 2 s) are for the fine adjustment of the lamellas. A long button actuation (> 2 s) starts the automatic run and drives the blind for the time SCPTdriveTime continuously into the direction close. The automatic run can be stopped by a renewed button actuation.

UCPTdiConfig[x].Byte[1...2] = 02_{hex}

Close blind	nvoSetting_A_x.function	= UCPTsunblindDOWN;
Stop blind	nvoSetting_A_x.function	= UCPTsunblindSTOP;

UCPTdiConfig[x].Byte[1...2] = 04_{hex}

Close blind	nvoSetting_B_x.function	= UCPTsunblindDOWN;
Stop blind	nvoSetting_B_x.function	= UCPTsunblindSTOP;

With a delay of 500ms the command *UCPTsunblindIDLE* for idle mode is sent after the command *UCPTsunblindSTOP* if *UCPTsunblindIDLE* is unequal to *UCPTsunblindSTOP.*

Shutter

UCPTdiConfig[x].Byte[0] = 03_{hex}

Shutter UP

In the configuration mode “Shutter UP“ only the nvoSetting variables are changed and sent. Short button actuation (< 2 s) starts the automatic run and drives the shutter continuously in the direction open for the time SCPTdriveTime. The automatic run can be stopped by a renewed button actuation. By a long button actuation (> 2 s) the position of the shutter can be individually adjusted.

UCPTdiConfig[x].Byte[1...2] = 01_{hex}

Open shutter	nvoSetting_A_x.function	= UCPTsunblindUP;
Stop shutter	nvoSetting_A_x.function	= UCPTsunblindSTOP;

UCPTdiConfig[x].Byte[1...2] = 03_{hex}

Open shutter	nvoSetting_B_x.function	= UCPTsunblindUP;
Stop shutter	nvoSetting_B_x.function	= UCPTsunblindSTOP;

With a delay of 500ms the command *UCPTsunblindIDLE* for idle mode is sent after the command *UCPTsunblindSTOP* if *UCPTsunblindIDLE* is unequal to *UCPTsunblindSTOP.*

Shutter DOWN

In the configuration mode “shutter DOWN“ only the nvoSetting variables are changed and sent. Short button actuation (< 2 s) starts the automatic run and drives the shutter continuously into the direction close for the time SCPTdriveTime. The automatic run can be stoped by a renewed button actuation. By a long button actuation (> 2 s) the position of the shutter can be adjusted individually.

UCPTdiConfig[x].Byte[1...2] = 02_{hex}

Close shutter	nvoSetting_A_x.function	= UCPTsunblindDOWN;
Stop shutter	nvoSetting_A_x.function	= UCPTsunblindSTOP;

UCPTdiConfig[x].Byte[1...2] = 04_{hex}

Close shutter	nvoSetting_B_x.function	= UCPTsunblindDOWN;
Stop shutter	nvoSetting_B_x.function	= UCPTsunblindSTOP;

With a delay of 500ms the command *UCPTsunblindIDLE* for idle mode is sent after the command *UCPTsunblindSTOP* if *UCPTsunblindIDLE* is unequal to *UCPTsunblindSTOP.*

4.2.3 Configuration Parameter Switch Object:

SCPTmaxOut
SCPT Index: 93, SNVT_lev_cont
Function: This configuration property determines the maximum output value of the variable nvoSwitch.value. (Preset value: 100.0)

SCPTmaxSendTime
SCPT Index: 49, SNVT_time_sec
Function: Heartbeat function. This configuration property stipulates the interval time after which the output variable is sent. By input values = 0, the heartbeat function is deactivated. (Preset value: 0,0 s)

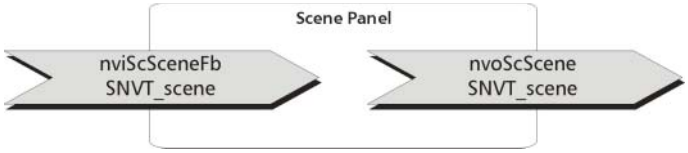
SCPTminSendTime
SCPT Index: 52, SNVT_time_sec
Function: This configuration property stipulates the sending interval of the output variable in the mode dimming. By input values = 0, the function is deactivated. (Preset value: 0,3 s)

SCPTstepValue
SCPT Index: 92, SNVT_lev_cont
Function: This configuration property defines the step size of the variable nvoSwitch.value in the mode dimming. (Preset value: 5.0)

UCPTreverseDelay
UCPT Index: 14, SNVT_count
Function: This configuration property defines the toggling delay with a rotation reversing of the blind motors. Thus, a change command from e.g. nvoSetting = SET_UP to nvoSetting = SET_DOWN is output delayed. (Preset value: 500 ms)

SCPTdriveTime
UCPT Index: 45, SNVT_time_sec
Function: This configuration property defines the maximum turn-on time of the blind motors in the automatic run. (Preset value: 100,0 s)

4.3 Scene Panel



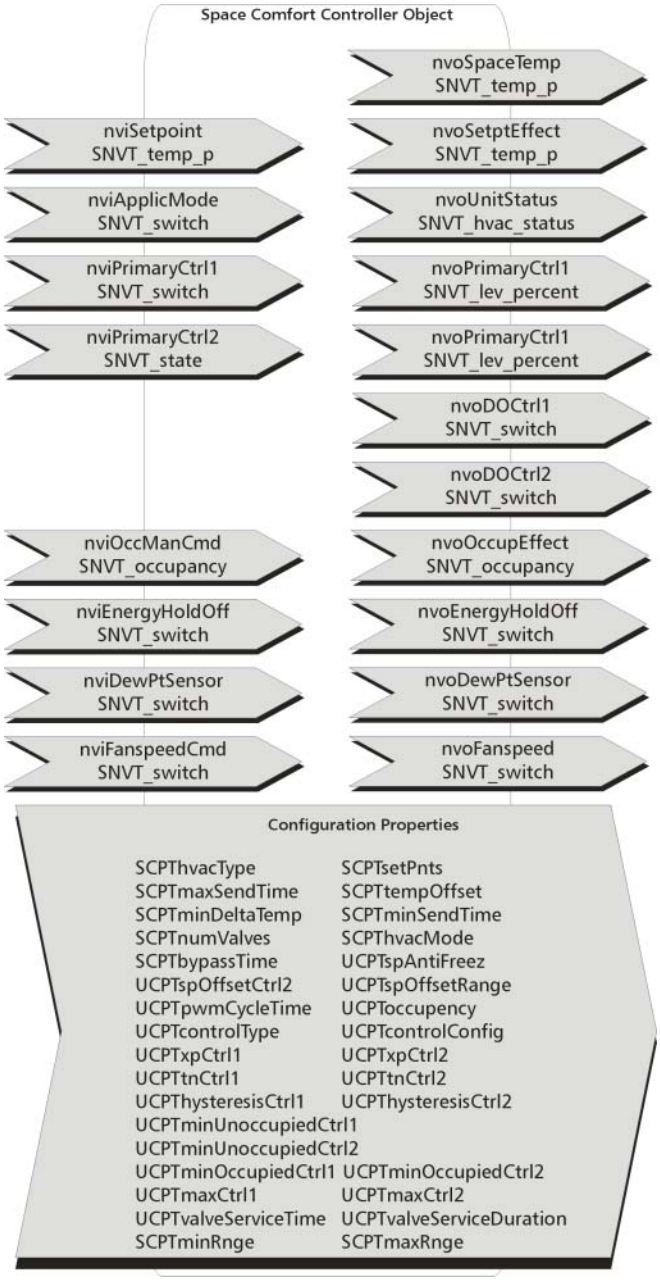
4.3.1 Input Variable Scene Panel

nviScSceneFb
SNVT Type: SNVT_scene, Index 115
Function: Input variables with the current lighting scene in the room.

4.3.2 Output Variable Scene Panel:

nvoScScene
SNVT Type: SNVT_scene, Index 115
Function: Output variable for driving a scene controller. The output variables are depending on the function adjustments. By UCPTdiConfig[x].Byte[0] = 3 buttons are allocated to the Scene-Object.
UCPTdiConfig[x].Byte[1...2] = 01_{hex} - FE_{hex} allocates a scene number to these buttons, which is called by short button actuations with SC_RECALL. With longer button actuations (> 2s) the scene is relearned with SC_LEARN. It is also possible to allocate several buttons to the scene panel.

4.4 Space Comfort Controller Object



Temperature
Temperature detection is made by an inside temperature sensor. An external temperature measuring is not possible. For an afterwards calibration of the inside sensor the configuration property *SCPTtempOffset* is available.

Set Point
The effective set point *nvoSetptEffect* is calculated depending on the room occupancy (*nvoOccupEffect*), of the set point defaults via *SCPTsetPnts* respetively *nviSetpoint* and the offset value (manual adjustment).

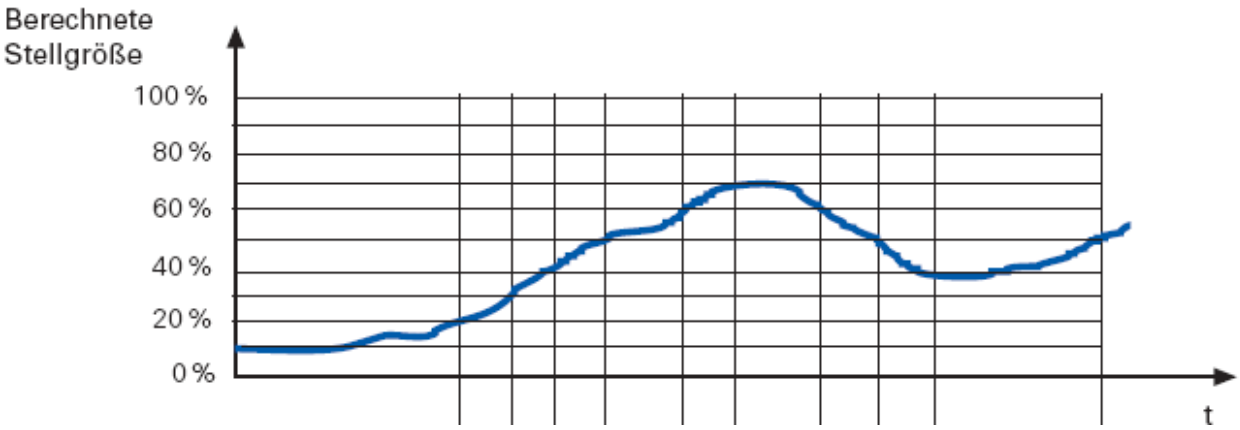
Control
For the temperature control the control algorithm uses the effective set point described above. The neutral (energy-free) zone around the basic set point adapts itself automatically to the current room occupancy and can be parameterized via *SCPTsetPnts*.

The controller can be configured for heating and cooling or bilevel heating by *UCPTcontrolType.* The control variables of the PI-controller for heating and cooling are output by the variables *SNVT_lev_percent* for continuous actuators and the variables *SNVT_switch* for thermionic two-point actuators (PWM-control and two-point control). The control parameter can be indiviudally adapted to the conditions of a room. The monitoring of window contacts and dew point detectors is made by the input variables *nviEnergyHoldOff* and *nviDewPtSensor*.

Change of Configuration Properties
After parameters have been changed, the changes must be activated. The new settings are taken over into the operating unit by adjusting nviProgDevice.bit0 from 0 to 1 respectively after a voltage reset.

Continuous PI-Control

The continuous controller outputs the control variable to the bus. The room temperature is kept constant by the control algorithm. The control variable sent has an effect on the actuator, which is mounted on an angle valve. It measures the quantity of heat going through the radiator (0 to 100%) according to the control variable. The higher the deviation of the set point from the temperature, the bigger the control variable gets.



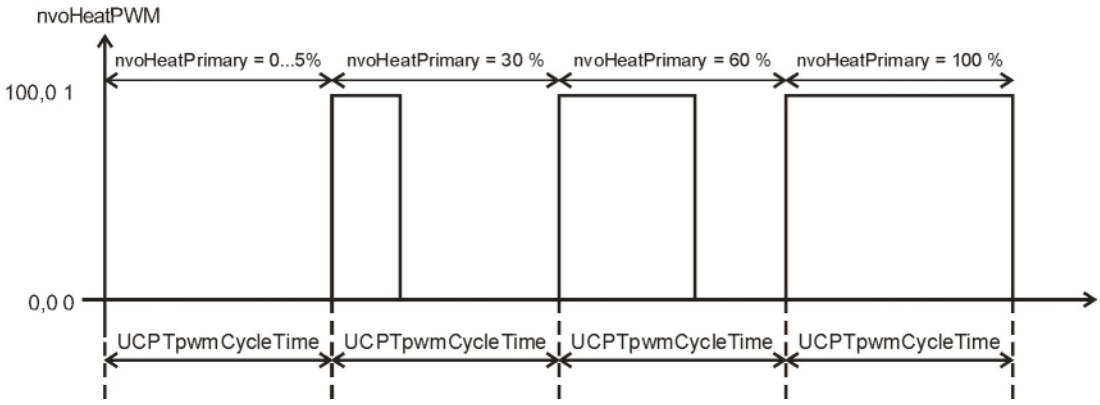
Switching PI-Control (PWM-Control)

With the PWM control (Pulse-Width-Modulation) the values (0..100%) calculated by the control algorithm are converted into a PWM. The same is based on a constant cycle period. If the controller calculates a control variable of 20%, a logical “1” is sent for 3 minutes (20% of 15 minutes) and afterwards a “0” for 12 minutest is sent for a cycle period of the switching control variable of 15 minutes. After the expiration of the cycle time, the current control variable is converted into a new PWM, again.

The room temperature is kept constant by the control algorithm. Averaging the complete period, you have the same behaviour of the control system as with a continuous controller.

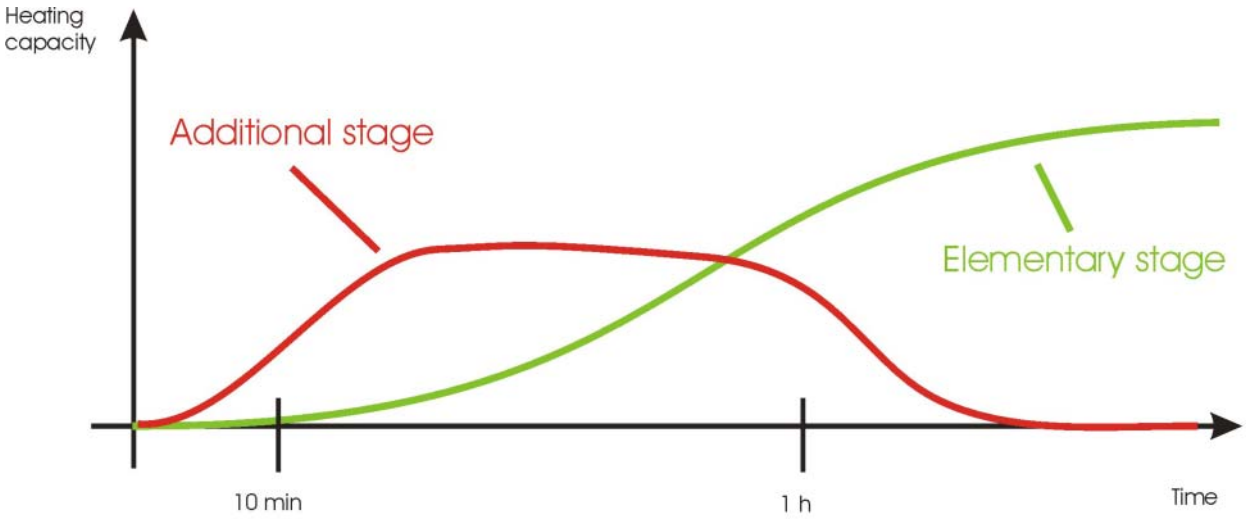
With the PWM- and the 2-point control the room temperature controller controls a switching actuator. The switching actuator opens and closes the actuator.

Example PWM-Control for Controller 1 (Ctrl1):



Bilevel Heating

The bilevel heating is very often used in connection with an underfloor heating. The underfloor heating is a very inertial system. The heating-up of a room takes a long time, thus (up to some hours). In order to shorten the heating-up phase, a fast heating system (e.g. warm water – convection heating) is used, additionally. As soon as the set temperature is considerably increased, the additional stage (fast heating system) is switched-on together with the elementary stage (e.g. underfloor heating). The room is mainly heated up by the additional stage, as the same is faster and thus the thermal output is available much faster. If the room is heated up to a special temperature (e.g. set temperature –1K, parameterizable), the additional stage is switched-off. In the meantime, the basic elementary stage can supply the required thermal output to the room and takes over the control solely. The elementary stage is parametrized as usual (e.g. with a underfloor heating: PWM with 30 minutes cycle period). For the booster heating system a 2-point control is absolutely sufficient, as the same is not used for the control but only for the heating-up phase.



Thermionic behaviour, as soon as the elementary and additional stage are switched-on at the same time, e.g. in the morning when the heating is activated.

4.4.1 Input Variables Space Comfort Controller Object

nviApplicMode

SNVT Type: SNVT_hvac_mode, Index 108

Function: Input variable for selection of the controller operating mode.

HVAC_AUTO	==>	automatic toggling between heating and cooling
HVAC_HEAT	==>	only heating
HVAC_COOL	==>	only cooling
HVAC_OFF	==>	control variables are set to 0%

The initialization status after reset is determined by the configuration property *SCPThvacMode*.

!! When making a change of HVAC_AUTO to HVAC_HEAT respectively HVAC_COOL as well as !!of HVAC_HEAT respectively HVAC_COOL to HVAC_AUTO a reset is made in the operation !!unit.

nviPrimaryCtrl1

SNVT Type: SNVT_lev_percent, Index 81

Function: Control variable for the network variable nvoPrimaryCtrl1 and *nvoDOCtrl1.*

nviPrimaryCtrl1 = 0x7FFF (163,835 %) ==> inside controller cooling ON (initialization value after reset)

nviPrimaryCtrl1 = 0 ... 100 % ==> inside controller OFF

==> nviPrimaryCtrl1 determines the output variables

!! The external override has top priority, thus also a parallel control of heating and !! cooling vales is possible.

nviPrimaryCtrl2
SNVT Type: SNVT_lev_percent, Index 81
Function: Control variable for the network variable nvoPrimaryCtrl2 und *nvoDOCtrl2.*
nvoPrimaryCtrl2 = 0x7FFF (163,835 %) ==> insinde controller heating ON (initialization value after reset)
nvoPrimaryCtrl2 = 0 ... 100 % ==> inside controller OFF
==>nvoPrimaryCtrl2 determines the output variables
!! The external override has top priority, thus also a parallel control of heating and !! cooling vales is possible.

nviOccManCmd
SNVT Type: SNVT_occupancy, Index 109
Function: Input variable for default fo the room occupancy. The current room occupancy determines the variables of the control parameter "effective set point" and "neutral zone" and thus the set points for heating and cooling (see table 1). Initialization value adjustable via *UCPToccupancy*.

nviOccManCmd: Default via GLT by: OC_OCCUPIED, OC_STANDBY, OC_UNOCCUPIED

nviOccManCmd	Wippe	>>>	nvoOccupEffect	NvoSetptEffect
OC_OCCUPIED	****	>>>	OCCUPIED	SCPTsetpoint + manual set point offset **or** nviSetpoint + manual set point offset
****	OC_OCCUPIED	>>>		
OC_STANDBY	****	>>>	STANDBY	SCPTsetpoint + manual set point offset **or** nviSetpoint + manual set point offset
OC_OCCUPIED	OC_STANDBY	>>>		
OC_UNOCCUPIED	****	>>>	UNOCCUPIED	SCPTsetpoint + manual set point offset **or** nviSetpoint + manual set point offset
OC_UNOCCUPIED	OC_STANDBY	>>>		

Table 1: Set point, room occupancy

nviSetpoint
SNVT Type: SNVT_temp_p, Index 105
Function: Input variable for default of basic set point temperature.
It is not obligatory necesary to bind this network variable to a superior node. If no update for nviSetpoint is made, the initialization value 0x7FFF (=327,67°C) is maintained. The values of the configuration properties *SCPTsetPnts* are used for the calculation of the effective set point (basic set point + offset). If *nviSetpoint* receives an update with a valid set point, the effective set point is calculated with the value of the input variable.

nviEnergyHoldOff

SNVT Type: SNVT_switch, Index 95
Function: Input variable of e.g. window or door contact for activation of the energy-saving function. By nviEnergyHoldOff = 100.0 1 the function is activated and the control variables heating/cooling are reset to their minimum values. With an active energy-saving function the antifreeze function is switched-on (seeUCPTspAntiFreez). After deactivation of the energy-hold-off function the temperature control is restarted.

nviDewPtSensor

SNVT Type: SNVT_switch, Index 95
Function: Input variable for evaluation of a dew point detector in the operating mode cooling. By nviDewPtSensor = 100.0 1 the control variable cooling is reset to its minimum value. After deactivation of this function, the temperature control is started restarted.

nviFanSpeedCmd

SNVT Typ: SNVT_switch, Index 95
Funktion: Input variable for fanspeed. The valid value range is the same as nvoFanSpeed.

4.4.2 Output Variables Space Comfort Controller Object

nvoSpaceTemp

SNVT Type: SNVT_temp_p, Index 105
Function: Output variable for measured temperature value. Measuring range 0-50°C, resolution 1/100°C. Data output is made depending on *SCPTmaxSendTime* and 1,5s- 4s after reset.

nvoUnitStatus

SNVT Type: SNVT_hvac_status, Index 112
Function: Output variable for operating status and the control variables heating/cooling of the controller.

.mode = HVAC_AUTO	==> automatic toggling between heating and cooling		
HVAC_HEAT	==>	only heating	
HVAC_COOL	==>	only cooling	
HVAC_OFF	==>	control off	
.heat_output_primary	0...100 %	==>	control variable heating
.cool_output_primary	0...100 %	==>	control variable cooling

nvoPrimaryCtrl1

SNVT Type: SNVT_lev_percent, Index 81
Function: Output variable with the control variable of controller 1 (heating) for control of a continuous actuator. Data output is made depending on *SCPTmaxSendTime* and 1,5s- 4s after reset.

nvoDOCtrl1

SNVT Type: SNVT_switch, Index 95
Function: Output variable with the control variable of controller 1 (heating) for pulse width modulated control or two-point control of a thermionic two-point actuator. Data output is made immediately with a switch command or otherwise depending on *SCPTmaxSendTime* and 1,5s- 4s after reset.

nvoPrimaryCtrl2

SNVT Type: SNVT_lev_percent, Index 81
Function: Output variable with the control variable of controller 2 (bilevel heating and cooling) for control of a contiuous actuator. Data output is made depending on *SCPTmaxSendTime* and 1,5s- 4s after reset.

nvoDOCtrl2

SNVT Type: SNVT_switch, Index 95
Function: Ouptut variable with the control variable of controller 2 (bilevel heating and cooling) for pulse width modulated control or for two-point control of a thermic two-point actuator. Data output is made immediately with a switch command or otherwise depending on *SCPTmaxSendTime* and 1,5s- 4s after reset.

nvoSetptEffect
SNVT Type: SNVT_temp_p, Index 105
Function: The output variable send the set point used by the control algorithm. Output is depending on the operating mode of the controller:

nviApplicMode = HVAC_AUTO	==>	nvoSetptEffect = basic set point = (set point heating + set point cooling)/2
nviApplicMode = HVAC_HEAT	==>	nvoSetptEffect = set point heating
nviApplicMode = HVAC_Cool	==>	nvoSetptEffect = set point cooling

The effective set point is calculated depending on *nviSetpoint*, *nviOccManCmd*, *SCPTsetPnts* and the manual adjustment (see table 1). Data output is made depending on *SCPTmaxSendTime*, with value changes and 1,5s- 4s after reset.

nvoOccupEffect
SNVT Type: SNVT_occupancy, Index 109
Function: Output variable for the effective room occupancy (see table 1). Data output is made depending on *SCPTmaxSendTime*, with value change and 1,5s- 4s after reset.

nvoEnergyHoldOff
SNVT Type: SNVT_switch, Index 95
Function: Output variable for status indication of the energy function.

nvoEnergyHoldOff = 0.0 0	==>	window contcat inactive
nvoEnergyHoldOff = 100.0 1	==>	window contact active

Data output is made depending on *SCPTmaxSendTime*, with value changes and 1,5s- 4s after reset.

nvoDewPtSensor
SNVT Type: SNVT_switch, Index 95
Function: Output variable for status indication of the condensation detector.

nvoDewPtSensor = 0.0 0	==>	condensation detector inactive
nvoDewPtSensor = 100.0 1	==>	condensation detector active

Data output is made depending on *SCPTmaxSendTime*, with value changes and 1,5s- 4s after reset.

nvoFanSpeed
SNVT Typ: SNVT_switch, Index 95
Funktion: Output variable for status of the fanspeed. The readout in the display shows the current fan speed adjustment. The adjustment is continuously made with 5 % steps.

4.4.2 Configuration Properties Space Comfort Controller Object -

SCPThvacType – Controller type
Index: 169, SNVT_hvac_type
Function: Configuration parameter for indentifying the controller type.
Preset value: SCPThvacType = HVT_GENERIC

SCPThvacMode - Initialization
Index: 74, SNVT_hvac_mode
Function: The configuration property determines the initialisation status of the input variable *nviApplicMode* and thus also the start configuration of the temperature controller.
(Preset value: HVAC_AUTO)

SCPTmaxSendTime - Heartbeat
Index: 49, SNVT_time_sec
Function: The configuration property defines the sending time of the output variable. By input values =0, data output is deactivated. (Preset value: 30 s).

SCPTtempOffset – Temperature Offset
Index: 272, SNVT_temp_diff_p
Function: Offset for the temperature value. By means of this parameter a software calibration is possible. (Preset value: 0.0 °C)
Value range: –12,8 K .... 12,7 K

SCPTminDeltaTemp

Index: 64, SNVT_temp_p
Function: If the temperature changes by the adjusted value „SCPTminDeltaTemp“, the new temperature value is transmitted. The function is depending on the adjustment of the parameter „SCPTminSendTime“. (Range >= 0 °C; Preset value: 0,30 °C)

SCPTminSendTime

Index: 52, SNVT_time_sec
Function: Stipulates the smallest update interval of the output variables nvoSpaceTemp. An update is made after expiration of „SCPTminSendTime“, if the temperature value of the output variables has changed by more than „SCPTminDeltaTemp“. By input values = 0 the function is deactivated. (Preset value: 5,0 sec)

SCPTnumValves – Heating mode
Index: 59, SNVT_count
Function: The configuration property is used for selecting 2-pipe or 4-pipe systems. With a 2-pipe system (1 valve), the output variables with the control variables heating and cooling are receiving the same values.

SCPTnumValves = 1: ==> 2-pipe system
Mode heating: nvoPrimaryCtrl1 = nvoPrimaryCtrl2 = control variable heating
Mode cooling: nvoPrimaryCtrl1 = nvoPrimaryCtrl2 = control variable cooling
SCPTnumValves = 2: ==> 4-pipe system (standard value)
Mode heating: nvoPrimaryCtrl1 = control variable heating
Mode cooling: nvoPrimaryCtrl2 = control variable cooling

SCPTminRnge

Index: 23, SNVT_switch
Function: Lower limiting of fanspeed.
Value range: 0 – 30 %

SCPTmaxRnge

Index: 20, SNVT_switch
Function: Upper limiting of fanspeed.
Value range: 70 – 100 %

UCPTpwmCycleTime – PWM-cycle time
Index: 35, SNVT_time_min
Function: The configuration property determines the cycle time for a pulse width modulated control of the actuators by *nvoDOCtrl1* and *nvoDOCtrl2*. (Preset value: 15 min)

UCPTcontrolType – Controller type
Index: 51, typedef struct {unsigned short Byte[4]} UNVT_str_hex4
Function: By UCPTcontrolType.Byte[0] controller type 1 and by UCPTcontrolType.Byte[1] controller type 2 can be preset. Controller 1 (Ctrl1) can only heat whereas controller 2 (Ctrl2) 2-steps can heat or cool. Both controllers can control continuously or by a two-point behaviour. With a continuous control the pulse-width modulated control is output via *nvoDOCtrl1* respectively *nvoDOCtrl2* (table2).

Configuration of the controller for air-conditioning control UCPTcontrolType			
Allocation Button – Function block		Function	
Byte[0]	desription	Byte[1]	description
0	no controller	0	no controller
1	continuous heating	1	continuous cooling
2	2-point heating	2	2-point cooling
		3	2. heating stage continuous heating
		4	2. heating stage 2-point heating

Table 2: Selection of Controller

UCPTcontrolConfig – Controller settings
Index: 61, SNVT_state
Function: By UCPTcontrolConfig controller settings can be made.
UCPTcontrolConfig.

bit0
0 ==> reset of manual set point adjustment upon receipt of a basic set point (nviSetpoint)

1* ==> manual set point adjustment remains unchanged upon receipt of a basic set point (nviSetpoint)

bit1
0* ==> party function not with antifreeze
1 ==> party function also with antifreeze

bit2
0* ==> normal heating output
1 ==> inverting heating output

bit3
0* ==> normal cooling output
1 ==> inverting cooling output

UCPTvalveServiceTime – Maintenance interval
Index: 68, SNVT_time_hour
Function: The configuration parameter determines the maintenance interval for the valve. After expiration of the maintenance interval, the valve is opened completely and closed again. Thus, the valve is protected and prevented against lime deposit.

UCPTvalveServiceDuration – Duration maintenance interval
Index: 69, SNVT_time_min
Function: The configuration property determines the periodic time, i.e. how long the valve shall be opened as a protection against lime deposit. This time shall at least guarantee a complete opening of the valve.

4.4.3 Configuration Properties Space Comfort Controller Object – Set Point

SCPTsetPnts – Set points
Index: 60, SNVT_temp_setpt
Function: Configuration property for default of the set points for heating and cooling depending on the room occupancy. By nviSetpoint the values can be overwritten.
Preset values:

.occupied_heat	21 °C
.standby_heat	19 °C
.unoccupied_heat	16 °C
.occupied_cool	23 °C
.standby_cool	25 °C
.unoccupied_cool	28 °C

Value range:

.occupied_heat	16 ... 31 °C
.standby_heat	occupied_heat - 1...8 K
.unoccupied_heat	occupied_heat - 1...8 K
.occupied_cool	occupied_heat + 1...8 K
.standby_cool	occupied_cool + 1...8 K
.unoccupied_cool	occupied_cool + 1...8 K

Attention: The decimal place is not considered.

SCPTbypassTime – Party extension
Index: 34, SNVT_time_min
Function: Configuration property for default of a party extension. In the night operation it is possible to toggle the adjusted time in the comfort operation via the operating panel.
Value range: 0 min; 30 min; 60 min; 90 min; 120 min; 150 min; 180 min; 240 min

UCPTspAntiFreez - Antifreeze
Index: 18, SNVT_temp_p
Function: Set point for heating as a antifreeze function with on opened window contact, i.e. with active energy saving function. (Preset value: 10 °C)
Value range: 5 ... 10 °C

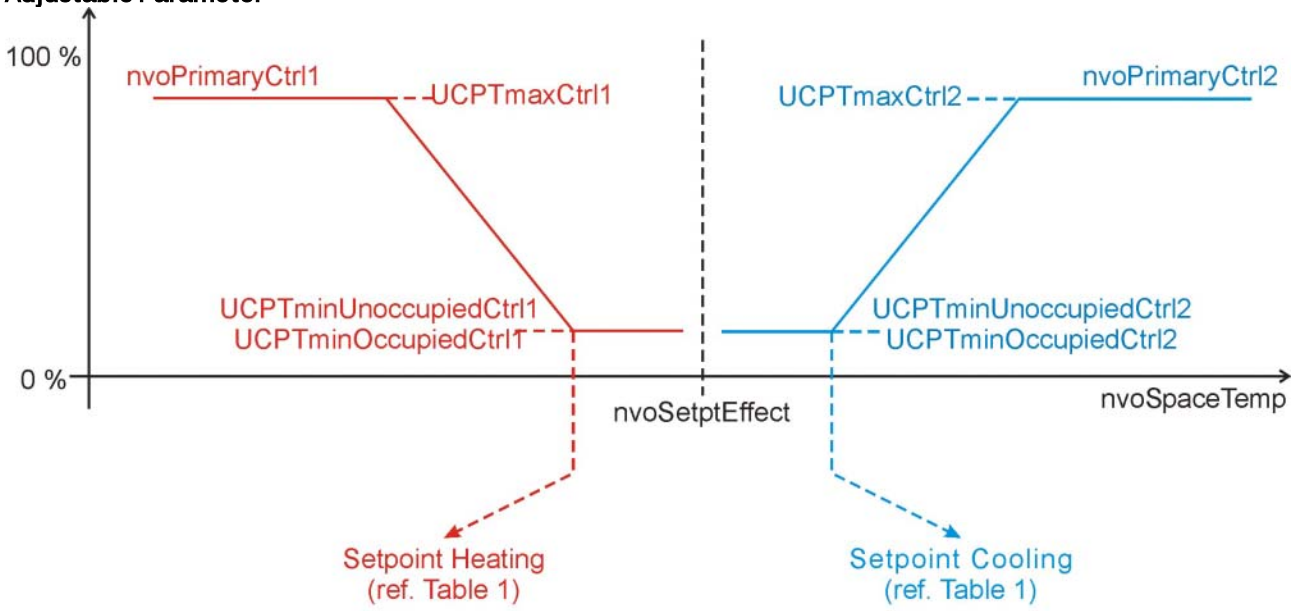
UCPToccupancy – Room occupancy after reset
Index: 60, SNVT_occupancy
Function: Configuration property for default of the room occupany after a reset.
(Preset value:OC_OCCUPIED)

UCPTspOffsetRng – Manual set point adjustment
Index: 12, SNVT_temp_p
Function: Configuration parameter for the value range of the adjustable set point correction, i.e. the default set point can be changed by the user by the value +/- UCPTspOffsetRng . (Preset value: 3K)
Value range: +/- 1K; +/- 3K; +/- 5K

UCPTspOffsetCtrl2 – Set point second heating stage
Index: 58, SNVT_temp_p
Function: Configuration property for 2-point heating. Indicates the stage range of the basic stage to the additional stage. (Preset value: 2,0 K)
Value range: 1...8 K

4.4.4 Configuration Properties Space Comfort Controller Object – Continuous Controller

Adjustable Parameter



Picture 4-2: Parameter

UCPTxpCtrl1 – Proportional range controller 1
Index: 54, SNVT_temp_p
Function: Parameter for adjustment of the proportional range. With UCPTxpCtrl1 Xp = 0 the controller 1 (for heating) is deactivated. (Preset value: 4 K)
Value range: 0...10 K

UCPTtnCtrl1 – Reset time controller 1
Index: 55, SNVT_time_min
Function: Parameter for adjustment of reset time of I-proportion.
By input values = 0 the I-proportion is deactivated. (Preset value: 100 min)
Value range: 0 min ... 240 min

UCPTminUnoccupiedCtrl1 – Control variable limit
Index: 62, SNVT_lev_percent
Function: Control variable limit downwards into the operation mode UNOCCUPIED.
(Preset value: 0 %)

UCPTminOccupiedCtrl1- Control variable limit
Index: 64, SNVT_lev_percent
Function: Control variable limit downwards into the operation mode OCCUPIED and STANDBY.
(Peset value: 0 %)

UCPTmaxCtrl1- Control variable limit
Index: 66, SNVT_lev_percent
Function: Control variable limit upwards.(Preset value: 100 %)

UCPTxpCtrl2– Proportional range controller 1
Index: 56, SNVT_temp_p
Function: Parameter for adjustment of the proportional range. By UCPTxpCtrl2 = 0 controller 2 is deactivated. (Preset value: 4 K)
Value range: 0...10 K

UCPTtnCtrl2– Reset time controller 2
Index: 57, SNVT_time_min
Function: Parameter for adjustment of reset time of the I-proportion.
By input values = 0 the I-proportion is deactivated. (Preset value: 100 min)
Value range: 0 min ... 240 min

UCPTminUnoccupiedCtrl2 – Control variable limit
Index: 63, SNVT_lev_percent
Function: Control variable limit downwards into the operating mode UNOCCUPIED.
(Preset value: 0 %)

UCPTminOccupiedCtrl2- Control variable limit
Index: 65, SNVT_lev_percent
Function: Control variable limit downwards into the operation mode OCCUPIED and STANDBY.
(Preset value: 0 %)

UCPTmaxCtrl2- Control variable limit
Index: 67, SNVT_lev_percent
Function: Control variable limit upwards. (Preset value: 100 %)

4.4.5 Configuration Properties Space Comfort Controller Object – Two-Point Controller

UCPThysteresisCtrl1 - Hysterese
Index: 52, SNVT_temp_p
Function: Parameter for adjustment of hysteresis loop. (Preset value: 1 K)
Value range: 0...10 K

UCPThysteresisCtrl2- Hysteresis
Index: 53, SNVT_temp_p
Function: Parameter for adjustment of hysteresis loop. (Preset value: 1 K)
Value range: 0...10 K