Application description

STANDARDdue
KNX RTH push-button RGB 2-gang

4472-B

10.KNX4472B-E.1512/151215

Feller
by Schneider Electric

1 General

This document explains the individual parameters of the STANDARDdue KNX RTH push-button RGB 2-gang, and serves as configuration aid.

STANDARDdue RTH push-button RGB 2-gang
Application: RTH push-button 2-gang V1.1

The STANDARDdue KNX RTH push-button RGB is an input device that is used in KNX systems as a room thermostat for regulating the temperature in closed rooms such as apartments, offices etc. and for controlling fans (→ *chapter 3.9*). The two push-buttons can be used to operate the room thermostat or to switch different consumers On and Off, to dim lights, operate blinds, store and recall scenes and/or to start sequences. If any actuators are being controlled, both single-button and two-button operation is possible (→ *chapter 3.2.1*). In this case the room thermostat must be controlled externally, e.g. using a 7" home panel or the HomeServer.

The KNX RTH push-button RGB features RGB LEDs, which can display 6 different basic colours as well as 4 freely definable user colours. For the user colours, the values red, green and blue can be set in the ETS or sent via the KNX bus with a 3 byte object.

1.1 Technical data

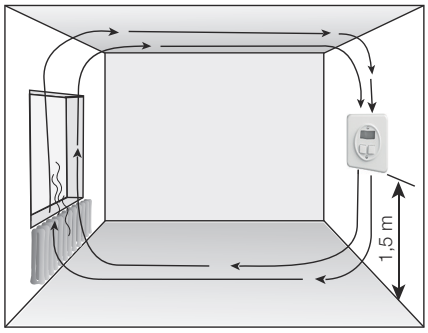
Ambient conditions:	
- Type of protection (IEC 60529)	IP20, dry installation
- Ambient temperature	operation: –5 °C up to +45 °C storage: –25 °C up to +70 °C
KNX supply	
- Voltage	21–30 V DC SELV
- Connection	KNX bus connecting terminal
Power consumption	
- Basic power requirement	max. 170 mW
- additionally per lighting push-button	max. 155 mW
- additionally for LCD backlighting	max. 170 mW
Service life	at least 10^5 switching operations
Installation depth	31 mm

Note:
For further information on the installation, please refer to the installation instructions.

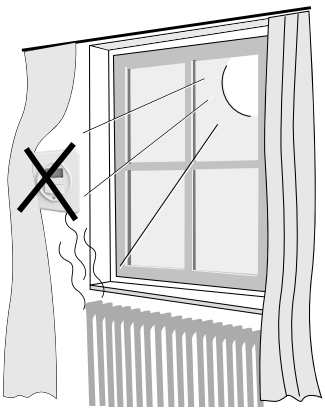
1.2 Notes on choice of installation site

The following should be noted when choosing an installation site so that the integrated room thermostat can work efficiently:

- The KNX RTH push-button should be installed at a distance of approximately 150 cm from the floor and 50 cm from the edge of the door.
- The KNX RTH push-button should be installed on a wall opposite the radiator.
- The heater and the KNX RTH push-button should not be isolated from each other by angled room architecture.



- It is not advisable to install the KNX RTH push-button close to the radiator or behind curtains.
- This also applies to installation on an external wall – low external temperatures influence the temperature control.
- The heat that is radiated by electrical consumers and direct sunlight can adversely affect control performance.



1.3 Typographical conventions

The following typographical conventions are used in this application description:

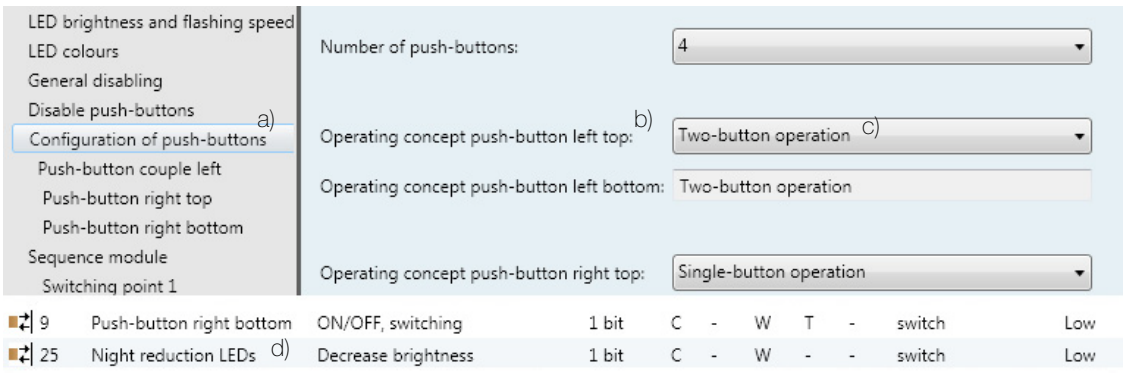
a) Names of parameter pages are enclosed in double quotation marks " ".
e.g. parameter page "Configuration of push-buttons"

b) Parameter names are shown in **bold** letters.
e.g. parameter **Operating concept push-button x** determines the operating concept of the push-button.

c) Parameter values are shown in *italics*, while the standard values defined in the ETS are shown in ***bold-italics***

e.g. **Operating concept push-button x** *Two-button operation*
Single-button operation

d) Objects are shown in angle brackets < > Object name and function are separated using a dash –, while the object number (if indicated) is placed before the bracket.
e.g. object 25 <Night reduction LEDs – Decrease brightness> is visible in the ETS.



2 The application "RTH push-button 2-gang V1.1"

2.1 Overview

Number of communication objects: 89
max. number of group addresses: 254
max. number of allocations: 255

For planning as well as for commissioning and the diagnostics of a KNX system, a programming software is required: KNX Tool Software ETS version 3 or later. It is used to select and/or create the application programme and its parameters as well as loading them into the device.

The product database required for the KNX RTH push-button RGB is available at **www.feller.ch** The KNX label guarantees that the products of different manufacturers are able to communicate with each other and that the commands are interpreted in the same way by devices of different manufacturers (command compatibility).

The ETS for KNX push-button includes the following parameter pages (with explanations):

2.2 Communication objects

Communication flags:

Flag	Name	Meaning
R	Read	Object status can be viewed (ETS / display etc.)
W	Write	Object can receive
T	Transmit	Object can send
U	Update	Objekt can accept answer to own read requests

2.2.1 Object table push-button

The following objects are visible depending on the parameterisation.

No.	Object name	Function	Type	DPT	Flags			
					R	W	T	U
12 12, 15	Push-button couple Push-button x	ON/OFF, switching	1 bit	1.001		x	x	
	1 bit object for sending and receiving switching telegrams (ON, OFF). The object is visible if the following parameter setting is selected: "Push-button x" – **Push-button function** = *Switching*							
					R	W	T	U
12 12, 15	Push-button couple Push-button x	ON/OFF, dimming	1 bit	1.001		x	x	
	1 bit object for sending and receiving switching telegrams (ON, OFF). The object is visible if the following parameter setting is selected: "Push-button x" – **Push-button function** = *Dimming*							
					R	W	T	U
12 12, 15	Push-button couple Push-button x	UP/DOWN, blind	1 bit	1.008		x	x	
	1 bit object for sending and receiving telegrams with which blinds can be moved up- or downwards. The object is visible if the following parameter setting is selected: "Push-button x" – **Push-button function** = *Blind*							
					R	W	T	U
12 12, 15	Push-button couple Push-button x	Recall, scene Recall/save, scene	8 bit	18.001			x	
	8 bit object for recalling or saving one of a maximum of 64 scenes in the actuator. The object is visible if the following parameter setting is selected: "Scene module" – **Scene function** = *Decentralised scene saving (in actuator)* "Push-button x" – **Push-button function** = *Scene*							
					R	W	T	U
12 12, 15	Push-button couple Push-button x	Recall scene x	1 bit	1.010		x	x	
	1 bit object for starting a local scene. The object is visible if the following parameter setting is selected: "Scene module" – **Scene function** = *Local scene saving (in push-button)* "Push-button x" – **Push-button function** = *Scene* Further information on the scene function → *chapter 3.5*							
					R	W	T	U
12 12, 15	Push-button couple Push-button x	Send, value	8 bit	5.001		x	x	
	8 bit object for sending and receiving values 0–255. The object is visible if the following parameter setting is selected: "Push-button x" – **Push-button function** = *Value*							

No.	Object name	Function	Type	DPT	Flags			
					R	W	T	U
12 12, 15	Push-button couple Push-button x	Forced position	2 bit	2.001		x	x	
	2 Bit for sending and receiving forced telegrams. Polarity → *chapter 2.3.3* The object is visible if the following parameter setting is selected: "Push-button x" – **Push-button function** = *Forced position*							
					R	W	T	U
13 13, 16	Push-button couple Push-button x	Brighter/darker, dimming	4 bit	3.007			x	
	4 bit object for sending relative dimming telegrams. The object is visible if the following parameter setting is selected: "Push-button x" – **Push-button function** = *Dimming*							
					R	W	T	U
13 13, 16	Push-button couple Push-button x	Step/stop, blind	1 bit	1.009		x	x	
	1 bit object for sending and receiving telegrams with which blinds can be stopped or slats can be readjusted. The object is visible if the following parameter setting is selected: "Push-button x" – **Push-button function** = *Blind*							
					R	W	T	U
13, 16	Push-button x (longer press)	ON/OFF, switching	1 bit	1.001		x	x	
	1 bit object for sending and receiving switching telegrams (ON, OFF). The object is visible if the following parameter setting is selected: "Push-button x" – **Push-button function** = *Switching / Value / Forced position* "Push-button x" – **Longer press push-button x** = *Active* "Push-button x" – **Longer press function** = *Switching*							
					R	W	T	U
13, 16	Push-button x (longer press)	Value, dimming	8 bit	5.001			x	
	8 bit object for sending dimming values. The object is visible if the following parameter setting is selected: "Push-button x" – **Push-button function** = *Switching / Value / Forced position* "Push-button x" – **Longer press push-button x** = *Active* "Push-button x" – **Longer press function** = *Dimming value in %*							
					R	W	T	U
13, 16	Push-button x (longer press)	UP/DOWN, blind	1 bit	1.008			x	
	1 bit object for sending telegrams with which blinds can be moved up- or downwards. The object is visible if the following parameter setting is selected: "Push-button x" – **Push-button function** = *Switching / Value / Forced position* "Push-button x" – **Longer press push-button x** = *Active* "Push-button x" – **Longer press function** = *Blind UP / DOWN*							
					R	W	T	U
13, 16	Push-button x (longer press)	Send, value	8 bit	5.001			x	
	8 bit object for sending values 0–255. The object is visible if the following parameter setting is selected: "Push-button x" – **Push-button function** = *Switching / Value / Forced position* "Push-button x" – **Longer press push-button x** = *Active* "Push-button x" – **Longer press function** = *Value*							

No.	Object name	Function	Type	DPT	Flags			
					R	W	T	U
13, 16	Push-button x (longer press)	Recall, scene	8 bit	18.001			x	
	8 bit object for recalling or saving one of a maximum of 64 scenes in the actuator. The object is visible if the following parameter setting is selected: "Push-button x" – **Push-button function** = *Switching / Value / Forced position* "Push-button x" – **Longer press push-button x** = *Active* "Push-button x" – **Longer press function** = *Scene*							
					R	W	T	U
16	Push-button couple, double-click	UP/DOWN, move shading	1 bit	1.008			x	
	1 bit object for sending telegrams with which the shading can be moved up- or downwards by means of blind actuators. The object is visible if the following parameter setting is selected: "Configuration of push-buttons" – **Operating concept push-button x** = *Two-button operation* "Push-button x" – **Push-button function** = *Blind* "Push-button x" – **Advanced functions blind** = *Move shading (double-click: long/short)*							
					R	W	T	U
14 14, 17	Push-button couple, Signal LED Push-button x, signal LED	Show on LED	1 bit	1.001		x		x
	1 bit object used for activating the push-button LED. Polarity: 1 = LED illuminated ; 0 = LED extinguished The object is visible if the following parameter setting is selected: "Push-button x" – **LED function** = *Status signal LED object (external signal)*							
					R	W	T	U
14 14, 17	Push-button couple, Signal LED Push-button x, signal LED	Show on RGB LED	3 byte	232.600		x		x
	3 byte object for receiving RGB telegrams which may affect the colour of the push-button LED. The object is visible if the following parameter setting is selected: "Push-button x" – **LED-function** = *RGB signal LED object (external signal)*							
					R	W	T	U
14 14, 17	Push-button couple, Signal LED Push-button x, signal LED	Override/show on LED	1 bit	1.001		x		x
	1 bit object used for overriding the push-button LED function. Polarity can be parameterised. The object is visible if the following parameter setting is selected: "Push-button x" – **LED function** = *Orientation light (always switched on) / Push-button status (internal signal) / Press: ON / Release: OFF (feedback)* "Push-button x" – **LED function overridable with object signal LED** = *Yes*							
					R	W	T	U
24	All involved push-buttons	Disable push-buttons	1 bit	1.001		x		
	1 bit object for enabling or disabling the push-button functions. Polarity can be parameterised. The object is visible if the following parameter setting is selected: "General disabling" – **Disable push-buttons function** unequal *Not active*							

No.	Object name	Function	Type	DPT	Flags			
					R	W	T	U
25	Night reduction LEDs & display	Decrease brightness	1 bit	1.001		x		
	1 bit object for activating or deactivating the night reduction (modified brightness of all active LEDs as well as backlighting of the LC display). Polarity can be parameterised. The object is visible if the following parameter setting is selected: "LED brightness and flashing speed" – **Night reduction LEDs function** unequal *Not active*							

2.2.2 Object table scene module

Notes:

- The objects are only visible during the parameter setting "Scene module" – **Scene function** = *Local scene saving (in push-button)*
- The number of visible objects varies between 10 (objects 31–40) and 15 (objects 31–45). This depends on the parameter setting "Scene module" – **Number of scene values per scene**.

No.	Object name	Function	Type	DPT	Flags			
					R	W	T	U
31–45	Scene value x	ON/OFF, UP/DOWN	1 bit	1.001		x	x	x
	1 bit object for sending and receiving switching telegrams (ON, OFF) or telegrams with which blinds can be moved up- or downwards. The object is visible if the following parameter setting is selected: "Data type scene value" – **Data type scene value x** = *1 bit (switching ON/OFF, blind UP/DOWN)*							
					R	W	T	U
31–45	Scene value x	Send, value	8 bit	5.001		x	x	x
	8 bit object for sending and receiving dimming values or telegrams with which blinds can be moved into the respective position. The object is visible if the following parameter setting is selected: "Data type scene value" – **Data type scene value x** = *8 bit (dimming value, blind value)*							

2.2.3 Object table sequence module

Notes:

- The objects are only visible with the parameter setting "Sequence module" – **Sequence module** = *Active*
- The number of visible objects varies between 0 and 10 (first object: 79). This depends on the parameter setting "Sequence module" – **Switching point x** = *Active*.

No.	Object name	Function	Type	DPT	Flags			
					R	W	T	U
87	Sequence module	Recall sequence	1 bit	1.010		x	x	
	1 bit object for starting or stopping the sequence module. Polarity: 1 = start; 0 = stop.							
					R	W	T	U
88	Sequence module	Status	1 bit	1.010			x	
	1 bit object for reading out the status of the sequence module. Polarity: 1 = sequence has been started and will be processed; 0 = sequence completed							
					R	W	T	U
79–86	Sequence switching point x	ON/OFF, switching	1 bit	1.001			x	
	1 bit object for sending switching telegrams (ON, OFF). The object is visible if the following parameter setting is selected: "Sequence module" – **Switching point x** = *Active* "Switching point x" – **Switching point x function** = *Switching*							
					R	W	T	U
79–86	Sequence switching point x	Value, dimming	8 bit	5.001			x	
	8 bit object for sending dimming values. The object is visible if the following parameter setting is selected: "Sequence module" – **Switching point x** = *Active* "Switching point x" – **Switching point x function** = *Dimming value in %*							
					R	W	T	U
79–86	Sequence switching point x	UP/DOWN, blind	1 bit	1.008			x	
	1 bit object for sending telegrams with which blinds can be moved up- or downwards. The object is visible if the following parameter setting is selected: "Sequence module" – **Switching point x** = *Active* "Switching point x" – **Switching point x function** = *Blind UP/DOWN*							
					R	W	T	U
79–86	Sequence switching point x	Send, value	8 bit	5.001			x	
	8 bit object for sending values 0–255. The object is visible if the following parameter setting is selected: "Sequence module" – **Switching point x** = *Active* "Switching point x" – **Switching point x function** = *Value*							
					R	W	T	U
79–86	Sequence switching point x	Recall, scene	8 bit	18.001			x	
	8 bit object for recalling one of a maximum of 64 scenes in the actuator. The object is visible if the following parameter setting is selected: "Sequence module" – **Switching point x** = *Active* "Switching point x" – **Switching point x function** = *Scene*							

2.2.4 Object table room thermostat

No.	Object name	Function	Type	DPT	Flags			
					R	W	T	U
0–9	Window monitoring	Input 1–10	1 bit	1.001		x	x	x
	1 bit object for coupling window contacts: Polarity: 1 = window opened; 0 = window closed. The object is visible if the following parameter setting is selected: "Window monitoring" – **Window monitoring** = *Active* "Window monitoring" – **Number of windows to be monitored** = *1–10*							
					R	W	T	U
10	Window monitoring	Output	1 bit	1.001			x	
	1 bit object for sending the window monitoring depending on the window contact objects 0–9 (OR-links). Polarity: 1 = at least 1 window opened; 0 = all windows closed The object is visible if the following parameter setting is selected: "Window monitoring" – **Window monitoring** = *Active*							
					R	W	T	U
47	Controller operating mode	All operating modes	8 bit	20.102		x		
	8 bit object for switching over the operating mode of the room thermostat in accordance with the KNX specification Values: 01 = comfort operation; 02 = standby operation; 03 = night operation; 04 = frost/heat protection. The object is visible if the following parameter setting is selected: "Operating modes / Status" – **Switch operating mode via** = *8 bit object*							
					R	W	T	U
47	Controller operating mode	Comfort	1 bit	1.001		x		
	1 bit object for switching over into the comfort operating mode. The object is visible if the following parameter setting is selected: "Operating modes / Status" – **Switch operating mode via** = *Individual objects (1 bit)*							
					R	W	T	U
48	Controller operating mode	Night	1 bit	1.001		x		
	1 bit object for switching over into the night operating mode. The object is visible if the following parameter setting is selected: "Operating modes / Status" – **Switch operating mode via** = *Individual objects (1 bit)*							
					R	W	T	U
49	Controller operating mode	Frost/heat protection	1 bit	1.001		x		
	1 bit object for switching over into the frost/heat protection operating mode. The object is visible if the following parameter setting is selected: "Operating modes / Status" – **Switch operating mode via** = *Individual objects (1 bit)*							
					R	W	T	U
50	Controller operating mode	Holidays	1 bit	1.001		x		
	1 bit object for switching over into the holiday operating mode. The object is visible if the following parameter setting is selected: "Operating modes / Status" – **Switch operating mode via** = *Individual objects (1 bit)*							
					R	W	T	U
51	Controller operating mode	Dewpoint	1 bit	1.001		x		
	1 bit object for switching over into the dewpoint operating mode.							

No.	Object name	Function	Type	DPT	Flags			
					R	W	T	U
52	Correcting variable	Heating	1 bit	1.001			x	
	1 bit object for sending the switching correcting variable for the heating function. The object is visible if the following parameter setting is selected: "Heating/cooling system" – **Activation of the heating/cooling function** = *Heating / Heating and cooling* "Heating/cooling system" – **Type of heating function** = *Switching PI control / Switching 2-point control*							
					R	W	T	U
52	Correcting variable	Heating	8 bit	5.001			x	
	8 bit object for sending the continuous correcting variable for the heating function. The object is visible if the following parameter setting is selected: "Heating/cooling system" – **Activation of the heating/cooling function** = *Heating / Heating and cooling* "Heating/cooling system" – **Type of heating function** = *Continuous PI control / Continuous 2-point control*							
					R	W	T	U
52	Correcting variable	Cooling	1 bit	1.001			x	
	1 bit object for sending the switching correcting variable for the cooling function. The object is visible if the following parameter setting is selected: "Heating/cooling system" – **Activation of the heating/cooling function** = *Cooling* "Heating/cooling system" – **Type of cooling function** = *Switching PI control / Switching 2-point control*							
					R	W	T	U
52	Correcting variable	Cooling	8 bit	5.001			x	
	8 bit object for sending the continuous correcting variable for the cooling function. The object is visible if the following parameter setting is selected: "Heating/cooling system" – **Activation of the heating/cooling function** = *Cooling* "Heating/cooling system" – **Type of cooling function** = *Continuous PI control / Continuous 2-point control*							
					R	W	T	U
52	Correcting variable	Basic heating	1 bit	1.001			x	
	1 bit object for sending the switching correcting variable for the basic heating. The object is visible if the following parameter setting is selected: "Heating/cooling system" – **Activation of the heating/cooling function** = *2-stage heating* "Heating/cooling system" – **Type of basic level** = *Switching PI control / Switching 2-point control*							
					R	W	T	U
52	Correcting variable	Basic heating	8 bit	5.001			x	
	8 bit object for sending the continuous correcting variable for the basic heating. The object is visible if the following parameter setting is selected: "Heating/cooling system" – **Activation of the heating/cooling function** = *2-stage heating* "Heating/cooling system" – **Type of basic level** = *Continuous PI control / Continuous 2-point control*							
					R	W	T	U
52	Correcting variable	Basic cooling	1 bit	1.001			x	
	1 bit object for sending the switching correcting variable for the basic cooling. The object is visible if the following parameter setting is selected: "Heating/cooling system" – **Activation of the heating/cooling function** = *2-stage cooling* "Heating/cooling system" – **Type of basic level** = *Switching PI control / Switching 2-point control*							

No.	Object name	Function	Type	DPT	Flags			
					R	W	T	U
52	Correcting variable	Basic cooling	8 bit	5.001			x	
	8 bit object for sending the continuous correcting variable for the basic cooling. The object is visible if the following parameter setting is selected: "Heating/cooling system" – **Activation of the heating/cooling function** = *2-stage cooling* "Heating/cooling system" – **Type of basic level** = *Continuous PI control / Continuous 2-point control*							

					R	W	T	U
53	Correcting variable	Cooling	1 bit	1.001			x	
	1 bit object for sending the switching correcting variable for the cooling function in the mixed operating mode. The object is visible if the following parameter setting is selected: "Heating / cooling system" – **Activation of the heating / cooling function** = *Heating and cooling* "Heating/cooling system" – **Type of cooling function** = *Switching PI control / Switching 2-point control*							

					R	W	T	U
53	Correcting variable	Cooling	8 bit	5.001			x	
	8 bit object for sending the continuous correcting variable for the cooling function in the mixed operating mode. The object is visible if the following parameter setting is selected: "Heating / cooling system" – **Activation of the heating / cooling function** = *Heating and cooling* "Heating/cooling system" – **Type of cooling function** = *Continuous PI control / Continuous 2-point control*							

					R	W	T	U
53	Correcting variable	Additional heating	1 bit	1.001			x	
	1 bit object for sending the switching correcting variable for the additional heating. The object is visible if the following parameter setting is selected: "Heating/cooling system" – **Activation of the heating/cooling function** = *2-stage heating* "Heating/cooling system" – **Correcting variable of the additional level** = *Switching*							

					R	W	T	U
53	Correcting variable	Additional heating	8 bit	5.001			x	
	8 bit object for sending the continuous correcting variable for the additional heating. The object is visible if the following parameter setting is selected: "Heating/cooling system" – **Activation of the heating/cooling function** = *2-stage heating* "Heating/cooling system"– **Correcting variable of the additional level** = *Continuous*							

					R	W	T	U
53	Correcting variable	Additional cooling	1 bit	1.001			x	
	1 bit object for sending the switching correcting variable for the additional cooling. The object is visible if the following parameter setting is selected: "Heating/cooling system" – **Activation of the heating/cooling function** = *2-stage cooling* "Heating/cooling system" – **Correcting variable of the additional level** = *Switching*							

					R	W	T	U
53	Correcting variable	Additional cooling	8 bit	5.001			x	
	8 bit object for sending the continuous correcting variable for the additional cooling. The object is visible if the following parameter setting is selected: "Heating/cooling system" – **Activation of the heating/cooling function** = *2-stage cooling* "Heating/cooling system"– **Correcting variable of the additional level** = *Continuous*							

No.	Object name	Function	Type	DPT	Flags			
					R	W	T	U
54	Room temperature base set point value	Specification	2 byte	9.001		x		
	2 byte object for receiving an external specification of the base set point value (≡ heating comfort set point value). The room thermostat rounds up the temperature values received via the object to 0.1°C.							
					R	W	T	U
55	Room temperature set point value	set	2 byte	9.001			x	
	2 byte object for sending the currently set set point value.							
					R	W	T	U
56	Room temperature actual value	Control value	2 byte	9.001			x	
	2 byte object for sending the actual temperature measured by the temperature sensor. The object is visible if the following parameter setting is selected: "Room temperature measurement" – **Use external temperature sensor** = *No*							
					R	W	T	U
56	Room temperature actual value	External sensor	2 byte	9.001		x	x	x
	2 byte object for receiving and passing the actual temperature measured by the external sensor. The object is visible if the following parameter setting is selected: "Room temperature measurement" – **Use external temperature sensor** = *Yes*							
					R	W	T	U
57	Heating/cooling	Switch operating mode	1 bit	1.100		x	x	x
	1 bit object for switching between heating and cooling in the mixed operating mode. The object is visible if the following parameter setting is selected: "Heating / cooling system" – **Activation of the heating / cooling function** = *Heating and cooling* "Functionality" – **Switchover between heating and cooling** = *With the "Heating/cooling" object*							
					R	W	T	U
58	Disable additional level	Operating mode	1 bit	1.003		x		
	1 bit object for disabling or enabling the correcting value output for the additional level. Polarity: 1 = disabled; 0 = enabled The object is visible if the following parameter setting is selected: "Heating / cooling system" – **Activation of the heating / cooling function** = *2-stage heating / 2-stage cooling*							
					R	W	T	U
59	Room thermostat status	Feedback signal	8 bit				x	
	8 bit object for reporting the current operating mode of the room thermostats. Design: | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Bit 0: Comfort; Bit 1: Standby; Bit 2: Night operation; Bit 3: Frost/heat protection; Bit 4: Dewpoint alarm; Bit 5: Heating / cooling; Bit 6: Controller inactive; Bit 7: Frost alarm							
					R	W	T	U
60	Room thermostat status	Feedback signal	2 byte	22.101			x	
	2 byte object for reporting the current operating status of the room thermostat. Design: | Bit 15 | Bit 14 | Bit 13 | Bit 12 | Bit 11 | Bit 10 | Bit 9 | Bit 8 | | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Bit 0: Error; Bit 1: 0 ; Bit 2: 0 ; Bit 3: 0 ; Bit 4: Heating additional level; Bit 5: 0 ; Bit 6: 0 ; Bit 7: Heating inactive; Bit 8: Heating / cooling; Bit 9: 0 ; Bit 10: Cooling additional level; Bit 11: Cooling inactive; Bit 12: Dewpoint alarm; Bit 13: Frost alarm; Bit 14: Temperature alarm; Bit 15: 0							

2.2.5 Object table fan (fan coil)

Note:The objects are only visible during the parameter setting
"Fan (fan coil)" – **Changing of fan operating mode at the device** = *Enabled*

No.	Object name	Function	Type	DPT	Flags			
					R	W	T	U
72	Fan operating mode	Switch "manual/auto"	1 bit	1.003			x	
	1 bit object for switching the FanCoil to the manual fan control. The object is visible if the following parameter setting is selected: "Fan (Fan Coil)" – **"Switch manual/auto" object type** = *1 bit*							
					R	W	T	U
72	Fan operating mode	Switch "manual/auto"	8 bit	5.010			x	
	8 bit object for switching the FanCoil to the manual fan control. The object is visible if the following parameter setting is selected: "Fan (Fan Coil)" – **"Switch manual/auto" object type** = *8 bit 0..100% / 8 bit 0..255*							
					R	W	T	U
73	Fan operating mode	"Level" fan operating mode	1 bit	1.003			x	
	1 bit object for specifying a fan level on a FanCoil by means of switching direction commands. The object must have the same data format as the analogue object of the FanCoil actuator. Polarity: 1 = Switch fan level up, 0 = Switch fan level down The object is visible if the following parameter setting is selected: "Fan (Fan Coil)" – **"Fan levels" object type** = *1 bit*							
					R	W	T	U
73	Fan operating mode	"Level" fan operating mode	8 bit	5.010			x	
	8 bit object for specifying a fan level at a FanCoil by means of value telegrams. The object must have the same data format as the analogue object of the FanCoil actuator. The object is visible if the following parameter setting is selected: "Fan (Fan Coil)" – **"Fan levels" object type** = *8 bit 0..100% / 8 bit 0..255*							
					R	W	T	U
74	Fan operating mode	Frost/heat protection	1 bit	1.003			x	
	1 bit object for switching the FanCoil into the frost/heat protection. The object is visible if the following parameter setting is selected: "Fan (Fan Coil)" – **"Frost/heat protection" object type** = *1 bit*							
					R	W	T	U
74	Fan operating mode	Frost/heat protection	8 bit	5.010			x	
	8 bit object for switching the FanCoil into the frost/heat protection. The object is visible if the following parameter setting is selected: "Fan (Fan Coil)" – **"Frost/heat protection" object type** = *8 bit 0..100% / 8 bit 0..255*							

2.2.6 Object table display

No.	Object name	Function	Type	DPT	Flags			
					R	W	T	U
46	Display	ON/OFF, backlighting	1 bit	1.001		x		
	1 bit object for activating and deactivating the backlighting of the LC display. Polarity can be parameterised. The object is visible if the following parameter setting is selected: "Configuration display" – **Display lighting** = *According to object value (1=ON, 0=OFF) / (1=OFF, 0=ON)*							
					R	W	T	U
46	Display	RGB backlighting	3 byte	232.600		x		
	3 byte object for receiving RGB telegrams, which can influence the backlighting colour. The object is visible if the following parameter setting is selected: "Configuration display" – **Display lighting** = *RGB signal LED object (external signal)*							
					R	W	T	U
61	Display information	External temperature	2 byte	9.001		x	x	x
	2 byte object for receiving the external temperature for the value display.							
					R	W	T	U
62	Display information	DateTime	3 byte	19.001		x	x	x
	8 byte object for receiving the date and time for the value display. The object is visible if the following parameter setting is selected: "Configuration of display" – **Receive time and date via** = *1 common 8 byte object Time & Date*							
					R	W	T	U
62	Display information	Time	3 byte	10.001		x	x	x
	3 byte object for receiving the current time (e.g. from a DCF-77 master clock) for the value display. The object is visible if the following parameter setting is selected: "Configuration of display" – **Receive time and date via** = *2 separate 3 byte objects Time / Date*							
					R	W	T	U
63	Display information	Date	3 byte	11.001		x	x	x
	3 byte object for receiving the date for the value display. The object is visible if the following parameter setting is selected: "Configuration of display" – **Receive time and date via** = *2 separate 3 byte objects Time / Date*							
					R	W	T	U
64	Display information	Fan status automatic	1 bit	1.002		x	x	x
	1 bit object for receiving an active manual fan control for the value display.							
					R	W	T	U
65	Display information	Fan level	8 bit	5.010		x	x	x
	8 bit object for receiving the current fan level for the value display.							
					R	W	T	U
66	Display information	Wind speed	2 byte	9.005		x	x	x
	2 byte object for receiving the wind speed for the value display.							
					R	W	T	U
67	Display information	Air humidity	8 bit	5.001		x	x	x
	8 bit object for receiving the air humidity for the value display.							
					R	W	T	U
68	Display information	Air quality	2 byte	9.008		x	x	x
	2 byte object for receiving the air quality for the value display.							

No.	Object name	Function	Type	DPT	Flags			
					R	W	T	U
69	Display information	Free value	2 byte			x	x	x
	2 byte object for receiving a free value for the value display.							
					R	W	T	U
70	Display information	Change	1 bit	1.016		x		
	1 bit object for switching over the LC display. The object is visible if the following parameter setting is selected: "Configuration display" – **Change between displays via object** = *Yes*							
					R	W	T	U
71	Display	Language of display	8 bit	18.001		x		
	8 bit object for changing the language of the user interface. The value (1–64) is parameterisable (→ *chapter 2.6.2*). The object is visible if the following parameter setting is selected: "Configuration of display" – **Change language via object** = *8 bit object*							
					R	W	T	U
71	Display	Language of display	2 byte	234.001		x		
	2 byte object for changing the language of the user interface in accordance with ISO 639-1. Interpretable values: 6465h for German / 6672h for French / 6974h for Italian / 656Eh for English The object is visible if the following parameter setting is selected: "Configuration of display" – **Change language via object** = *2 byte object*							
					R	W	T	U
75	Display	Switch operating mode	1 bit	1.001		x		
	1 bit object for changing the operating mode display. Polarity can be parameterised.							

2.3 Parameters push-button 2-gang

Parameter page "Quick reference guide"

At first, the desired parameters must be set in the ETS application. These parameter settings will be saved when downloading the ETS application data to the KNX RTH push-button RGB. Before starting the ETS download, the KNX push-button must be programmed with a unique physical address by the ETS.

Since certain dependencies exist between the parameters in the ETS, care must be taken in order to ensure that the parameters are set according to the following procedure:

1. Parameter page "Scene module"
2. Parameter page "Configuration of push-buttons"
3. The configuration of the remaining parameters can be carried out in any order.

For the room thermostat:

1. Parameter page "Heating / cooling system"
2. The configuration of the remaining parameters can be carried out in any order.

Caution! Make sure to follow the correct order of parameterisation. If this order is not observed, any settings already made during the configuration will be lost.

Note: Always carry out the parameterisation in a top-down order.

2.3.1 Parameter page "Configuration of push-buttons"

The functionality of the two push-buttons (**Number of push-buttons** = *2*) can be defined on the parameter page "Configuration of push-buttons".

Parameter **Push-buttons control** defines whether both push-buttons are used to operate the room thermostat or for controlling any actuators.

Push-buttons control	*Room thermostat - / +* *Room thermostat + / -* *Any actuators*
Room thermostat - / +	The room thermostat is operated using the push-buttons. The left-hand push-button is used as the minus button (–) and the right-hand push-button as the plus button (+).
Room thermostat + / -	The room thermostat is operated using the push-buttons. The left-hand push-button is used as the plus button (+) and the right-hand push-button as the minus button (–).
Any actuators	Any actuators are controlled using the push-buttons. The room thermostat cannot be operated by the end user (LC display is for display only). The parameters **Operating concept push-button left** and the **Operating concept push-button right** are visible.

Parameter **Operating concept push-button left** defines the operating concept of the push-buttons (→ *chapter 3.2.1*).

Operating concept push-button left	*Two-button operation* *Single-button operation*
Two-button operation	The two push-buttons complement each other; they control the same actuator. One push-button issues the inverted command of the other. e.g. left: Light ON – right: Light OFF The parameter page "Push-button couple" is visible.
Single-button operation	The two push-buttons are independent of each other; each controls a different actuator, e.g. left: Light ON/OFF – right: Blind UP/DOWN The parameter pages "Push-button left" and "Push-button right" are visible.

2.3.2 Parameter page "Push-button x" (Room thermostat)

Note: A specific parameter page "Push-button left" and "Push-button right" is available for the parameterisation of the push-buttons. However, since both parameter pages are identically structured, they will be explained altogether at this point.

Parameter **LED function** specifies when the LED of the corresponding push-buttons should light up.

LED function	*Not active (always switched off)* *Orientation light (always switched on)* *Status signal LED object (external signal)* *RGB signal LED object (external signal)* *Press: ON / Release: OFF (feedback)*
Not active (always switched off)	The LED is always switched off.
Orientation light (always switched on)	The LED is always switched on. The parameters **LED colour** and **LED function overridable with object signal LED** are visible.
Status signal LED object (external signal)	If an ON telegram is sent to the object <Push-button x, signal LED – Show on LED>, the LED will light up. If an OFF telegram is sent to the object <Push-button x, signal LED – Show on LED>, the LED will go out. The parameters **LED display mode** and **LED colour** are visible. The parameter **LED function overridable with object signal LED** is permanently set to *No*.
RGB signal LED object (external signal)	If an RGB telegram is sent to the object <Push-button x, signal LED – Show on RGB LED>, the LED will light up in the corresponding colour. The RGB telegram must contain the proportions of the colour values for red, green and blue. In order to turn off the LED, the value 0 must be sent to all 3 colour values. The parameter **LED display mode** is visible. The parameter **LED function overridable with object signal LED** is permanently set to *No*.
Press: ON / Release: OFF (feedback)	If the push-button is pressed, the LED will light up, if it is released, the LED will go out. The parameters **LED colour** and **LED function overridable with object signal LED** are visible.

Parameter **LED display mode** specifies the lighting mode of the LED.

LED display mode	LED function = *Status signal LED object (external signal)* ***Status normal*** *Status inverted* *Status normal flashing* *Status inverted flashing* *Status normal soft-flashing* *Status inverted soft-flashing* LED function = *RGB signal LED object (external signal)* ***Status normal*** *Status normal flashing* *Status normal soft-flashing*
Status normal	The LED lights up if an ON telegram is present at the corresponding object.
Status inverted	The LED lights up if an OFF telegram is present at the corresponding object.
Status normal flashing	The LED lights up if an ON telegram is present at the corresponding object.
Status inverted flashing	The LED starts flashing if an OFF telegram is present at the corresponding object.
Status normal soft-flashing	The LED starts soft-flashing (→ *chapter 3.3*) if an ON telegram is present at the corresponding object.

Status inverted soft-flashing	The LED starts soft-flashing (→ *chapter 3.3*) if an OFF telegram is present at the corresponding object.

Parameter **LED colour** specifies in which colour the LED lights up.

LED colour	*Red* / *Green* / ***Blue*** / *White* / *Yellow* / *Vviolet* *User colour 1–4*
Red / *Green* / *Blue* / *White* / *Yellow* / *Violet*	The LED lights up in the selected colour.
User colour 1–4	The LED lights up in the colour mixed on the parameter page "LED colours" (→ *chapter 2.3.5*).

Parameter **LED function overridable with object signal LED** specifies if the LED can be overridden.

LED function overridable with object signal LED	***No*** *Yes*
No	The LED cannot be overridden.
Yes	The LED is overridden as soon as the corresponding telegram is present at the object <Push-button x, signal LED – Override/show on LED>. The parameters **Signal LED display mode** and **Signal LED colour** are visible.

Parameter **Signal LED display mode** specifies how the LED is to be overridden.

Signal LED display mode	*Status normal* *Status inverted* ***Status normal flashing*** *Status inverted flashing* *Status normal soft-flashing* *Status inverted soft-flashing*
Status normal	The LED lights up and overrides the normal LED function if an ON telegram is present at the object <Push-button x, signal LED – Override/show on LED>.
Status inverted	The LED lights up and overrides the normal LED function if an OFF telegram is present at the object <Push-button x, signal LED – Override/show on LED>.
Status normal flashing	The LED starts flashing and overrides the normal LED function if an ON telegram is present at the object <Push-button x, signal LED – Override/show on LED>
Status inverted flashing	The LED starts flashing and overrides the normal LED function if an OFF telegram is present at the object <Push-button x, signal LED – Override/show on LED>
Status normal soft-flashing	The LED starts soft-flashing (→ *chapter 3.3*) and overrides the normal LED function if an ON telegram is present at the object <Push-button x, signal LED – Override/show on LED>.
Status inverted soft-flashing	The LED starts soft-flashing (→ *chapter 3.3*) and overrides the normal LED function if an OFF telegram is present at the object <Push-button x, signal LED – Override/show on LED>.

Parameter **Signal LED colour** specifies with which colour the LED is overridden.

Signal LED colour	*Red* / ***Green*** / *Blue* / *White* / *Yellow* / *Violet* *User colour 1–4*
Red / *Green* / *Blue* / *White* / *Yellow* / *Violet*	The LED lights up in the selected colour.
User colour 1–4	The LED lights up in the colour mixed on the parameter page "LED colours" (→ *chapter 2.3.5*).

2.3.3 Parameter page "Push-button x" (Any actuators)

Note: A specific parameter page (parameter page "Push-button couple" or parameter pages "Push-button left" and "Push-button right") is available for the parameterisation of the push-buttons. However, since all parameter pages are identically structured, they will be explained altogether at this point.

Parameter

Push-button function specifies the basic function of a push-button.

Push-button function

Parameter page "Push-button couple"
Switching
Dimming
Blind
Forced position

Parameter page "Push-button left" / "Push-button right"
Switching
Dimming
Blind
Scene
Value
Forced position
Sequence module

Switching

The push-button has the basic function of switching.
With the parameter **Switching function**, the behaviour for a brief press is specified.

The object <Push-button x – ON/OFF, switching> is visible.

An alternative function can be defined for a longer press (parameter **Longer press**).

Dimming

The push-button has the basic function of dimming.
With the parameter **Dimming function**, the behaviour is specified for when the push-button is pressed.

The object <Push-button x – ON/OFF, dimming> is visible.
The object <Push-button x – Brighter/darker, dimming> is visible.

Blind

The push-button has the basic function blind.
With the parameter **Blind function**, the behaviour is specified for when the push-button is pressed.

The object <Push-button x – UP/DOWN, blind> is visible.
The object <Push-button x – Step/stop, blind> is visible.

Scene

The push-button has the basic function scene.
With the parameter **Scene function**, the scene to be called up is specified.

The object <Push-button x – …, scene> is visible.

Value

The push-button has the basic function value.
The parameter **Value function** is set to the value *Send 8 bit value*.

The parameter **8 bit value** defines the value (*0..255*) that is sent to the bus upon a brief press.

The object <Push-button x – Send, value> is visible.

An alternative function can be defined for a longer press (parameter **Longer press**).

Forced position

The push-button has the basic function forced position.
With the parameter **Forced position function**, the behaviour for a brief press is specified.

The object <Push-button x – Forced position> is visible.

An alternative function can be defined for a longer press (parameter **Longer press**).

Sequence module

The push-button has the basic function sequence module.
With the parameter **Sequence module**, the behaviour is specified for when the push-button is pressed.

Parameter **Switching function** specifies which command is triggered if a push-button with the basic function of switching is briefly pressed.

Switching function	Operating concept push-button x = *Two-button operation* ***Press: ON*** *Press: OFF* Operating concept push-button x = *Single-button operation* ***Press: INV*** *Press: ON* *Press: OFF* *Press: ON / Release: OFF* *Press: OFF / Release: ON*
Press: INV	If the push-button is pressed, the state of the object <Push-button x – ON/OFF, switching> is inverted and transferred.
Press: ON	If the push-button is pressed, an ON telegram is sent to the object <Push-button x – ON/OFF, switching>.
Press: OFF	If the push-button is pressed, an OFF telegram is sent to the object <Push-button x – ON/OFF, switching>.
Press: ON / Release: OFF	If the push-button is pressed, an ON telegram is sent to the object <Push-button x – ON/OFF, switching>. An OFF telegram is sent once it is released again.
Press: OFF / Release: ON	If the push-button is pressed, an OFF telegram is sent to the object <Push-button x – ON/OFF, switching>. An ON telegram is sent once it is released again.

Parameter **Dimming function** specifies which commands are triggered by the push-button with the basic function of dimming.

Dimming function	Operating concept push-button x = *Two-button operation* ***ON/brighter (short/long)*** *OFF/darker (short/long)* *INV/brighter (short/long)* *INV/darker (short/long)* Operating concept push-button x = *Single-button operation* ***Single-button op. (short/long: INV/dimming INV)*** *ON/brighter (short/long)* *OFF/darker (short/long)*
Single-button op. (short/long: INV/dimming INV)	If the push-button is pressed shortly, the state of the object <Push-button x – ON/OFF, dimming> is inverted and transferred. If the push-button is pressed and held, the dimming brightness is increased or decreased (brighter/darker) (object <Push-button x – Brighter/darker, dimming>). The corresponding dimming direction is determined by inverting the direction of the object value.
ON/brighter (short/long)	If the push-button is pressed shortly, an ON telegram is sent to the object <Push-button x – ON/OFF, dimming>. If the push-button is pressed and held, the dimming brightness is increased (brighter) (object <Push-button x – Brighter/darker, dimming>). If the push-button is released again, the dimming process is stopped.
OFF/darker (short/long)	If the push-button is pressed shortly, an OFF telegram is sent to the object <Push-button x – ON/OFF, dimming>. If the push-button is pressed and held, the dimming brightness is decreased (darker) (object <Push-button x – Brighter/darker, dimming>). If the push-button is released again, the dimming process is stopped.
INV/brighter (short/long)	If the push-button is pressed shortly, the state of the object <Push-button x – ON/OFF, dimming> is inverted and transferred. If the push-button is pressed and held, the dimming brightness is increased (brighter) (object <Push-button x – Brighter/darker, dimming>). If the push-button is released again, the dimming process is stopped.

INV/darker (short/long)

If the push-button is pressed shortly, the state of the object <Push-button x – ON/OFF, dimming> is inverted and transferred.
If the push-button is pressed and held, the dimming brightness is decreased (darker) (object <Push-button x – Brighter/darker, dimming>).
If the push-button is released again, the dimming process is stopped.

Parameter **Blind function** specifies which commands are triggered by the push-button with the basic function blind.

Blind function

Operating concept push-button x = *Two-button operation*
UP (short: step/stop, long: move)
DOWN (short: step/stop, long: move)
UP (short: move, long: move/stop)
DOWN (short: move, long: move/stop)

Operating concept push-button x = *Single-button operation*
UP & DOWN (short: move, long: move/stop)
UP (short: move, long: move/stop)
DOWN (short: move, long: move/stop)
UP (short: step/stop, long: move)
DOWN (short: step/stop, long: move)

UP & DOWN (short: move, long: move/stop)

If the push-button is pressed shortly, the blind will move (object <Push-button x – UP/DOWN, blind>). If the push-button is briefly pressed again, the blind is stopped.
If the push-button is pressed and held, the blind will move (object <Push-button x – UP/DOWN, blind>). If the push-button is released, the blind will be stopped (object <Push-button x – Step/stop, blind>).
The corresponding move direction is determined by inverting the direction of the last actuation of the push-button.

UP (short: move, long: move/stop)

If the push-button is pressed shortly, the blind will move upwards (object <Push-button x – UP/DOWN, blind>). If the push-button is briefly pressed again, the blind is stopped.
If the push-button is pressed and held, the blind will move upwards (object <Push-button x – UP/DOWN, blind>). If the push-button is released, the blind will be stopped (object <Push-button x – Step/stop, blind>).

DOWN (short: move, long: move/stop)

If the push-button is pressed shortly, the blind will move downwards (object <Push-button x – UP/DOWN, blind>). If the push-button is briefly pressed again, the blind is stopped.
If the push-button is pressed and held, the blind will move downwards (object <Push-button x – UP/DOWN, blind>). If the push-button is released, the blind will be stopped (object <Push-button x – Step/stop, blind>).

UP (short: step/stop, long: move)

If the push-button is pressed shortly, the angle of the slats is adjusted upwards (object <Push-button x – Step/stop, Blind>) or the blind is stopped if it was moving before.
If the push-button is pressed and held, the blind will move upwards (object <Push-button x – UP/DOWN, Blind>)

DOWN (short: step/stop, long: move)

If the push-button is pressed shortly, the angle of the slats is adjusted downwards (object <Push-button x – Step/stop, Blind>) or the blind is stopped if it was moving before.
If the corresponding push-button is pressed and held, the Blind will move downwards (object <Push-button x – UP/DOWN, Blind>).

Parameter **Advanced functions blind** is only visible if **Operating concept push-button x** = *Two-button operation* is set. It can only be changed if the **Blind function left** is configured to *UP (short: step/stop, long: move)* or *DOWN (short: step/stop, long: move)*.

Advanced functions blind	***Not active*** *Move shading (double-click: long/short)*
Not active	The advanced function is not activated.
Move shading (double-click: long/short)	If the push-button is pressed and held (>0.5 sec) at first and then pressed again within one second, an UP/DOWN telegram is sent via the object <Push-button x, double-click – UP/DOWN, move shading>.

Parameter **Scene function** specifies which commands are triggered by the push-button with the basic function scene. (explanations regarding scenes → *chapter 3.5*)

Scene function	**Scene function** = *Decentralised scene saving (in actuator)* ***Recall scene*** *Recall / save scene* **Scene function** = *Local scene saving (in push-button)* ***Recall / save scene 1*** *Recall / save scene 2* … *Recall / save scene 8*
Recall scene	A short press of the push-button results in a simple scene recall (object <Push-button x – Recall, scene>). Pressing the push-button longer does not have any function. **Note:** Not all of the actuators do support the maximum number of 64 scenes.
Recall / save scene	A short press of the push-button results in a simple scene recall (object <Push-button x –Recall/save, scene>). If the push-button is pressed and held, a storage telegram is sent to the bus and the involved actuators save the current value. After 3 seconds, the LED starts flashing and after another 3 seconds the LED will light up permanently, indicating that the scene has been saved. **Note:** Not all of the actuators do support the maximum number of 64 scenes.
Recall / save scene x	If the push-button is pressed shortly, the scene values saved on the push-button will be sent to all assigned actuators. When using the saving function on the parameter page "Scene module" (→ *chapter 2.5.1*), the scenes can also be saved.

Parameter **Scene number** determines the respective scene number in the actuator.

Scene number	*1..64*

Parameter **Forced position function** specifies the behaviour for a brief press.

Forced position function			*Forced switch on (11)* *Forced switch off (10)* *Cancel forced position (00)*
Forced switch on (11)	1	1	If the push-button is only pressed shortly, a forced telegram with bit 0 = 1 and bit 1 = 1 is sent to the object <Push-button x – Forced position>.
Forced switch off (10)	1	0	If the push-button is only pressed shortly, a forced telegram with bit 0 = 0 and bit 1 = 1 is sent to the object <Push-button x – Forced position>.
Cancel forced position (00)	0	0	If the push-button is pressed shortly, the forced position is disabled and sent to bit 0 = 0 and bit 1 = 0. The control system will then be released for normal operation again.
		bit 0	forced state, polarity: 1 = ON/DOWN ; 0 = OFF/UP
	bit 1		forced position, polarity: 1 = active ; 0 = inactive

Parameter **Sequence module function** specifies the behaviour for a press.

Sequence module function	*Start* *Start (short) / Stop (longer press)*
Start	If the push-button is pressed, the sequence defined on the parameter page "Sequence module" (→ *chapter 2.4.1*) will start. Since the sequence cannot be stopped by a press of a push-button, care must be taken in order to ensure that the parameter **Restart sequence module after last switching point?** is set to *No*.
Start (short) / Stop (longer press)	If the push-button is pressed shortly, the sequence defined on the parameter page "Sequence module" (→ *chapter 2.4.1*) will start. Pressing the push-button longer will stop the sequence.

Parameter **LED function** specifies when the LED of the corresponding push-buttons should light up.

LED function	*Not active (always switched off)* *Orientation light (always switched on)* *Push-button status (internal signal)* *Status signal LED object (external signal)* *RGB signal LED object (external signal)* *Press: ON / Release: OFF (feedback)*
Not active (always switched off)	The LED is always switched off.
Orientation light (always switched on)	The LED is always switched on. The parameters **LED colour** and **LED function overridable with object signal LED** are visible.
Push-button status (internal signal)	The LED is internally linked to the first push-button object (group address) of <Push-button x – ON/OFF, …>. The parameters **LED display mode**, **LED colour** and the **LED function overridable with object signal LED** are visible. **Note:** This function is only useful in combination with the push-button function switching or dimming.
Status signal LED object (external signal)	If an ON telegram is sent to the object <Push-button x, signal LED – Show on LED>, the LED will light up. If an OFF telegram is sent to the object <Push-button x, signal LED – Show on LED>, the LED will go out. The parameters **LED display mode** and **LED colour** are visible. The parameter **LED function overridable with object signal LED** is permanently set to *No*.

RGB signal LED object (external signal)	If an RGB telegram is sent to the object <Push-button x, signal LED – Show on RGB LED>, the LED will light up in the corresponding colour. The RGB telegram must contain the proportions of the colour values for red, green and blue. In order to turn off the LED, the value 0 must be sent to all 3 colour values. The parameter **LED display mode** is visible. The parameter **LED function overridable with object signal LED** is permanently set to *No*.
Press: ON / Release: OFF (feedback)	If the push-button is pressed, the LED will light up, if it is released, the LED will go out. The parameters **LED colour** and **LED function overridable with object signal LED** are visible.

Parameter **LED display mode** specifies the lighting mode of the LED.

LED display mode	**LED function** = *Push-button status (internal signal)* **LED function** = *Status signal LED object (external signal)* ***Status normal*** *Status inverted* *Status normal flashing* *Status inverted flashing* *Status normal soft-flashing* *Status inverted soft-flashing* **LED function** = *RGB signal LED object (external signal)* ***Status normal*** *Status normal flashing* *Status normal soft-flashing*
Status normal	The LED lights up if an ON telegram is present at the corresponding object.
Status inverted	The LED lights up if an OFF telegram is present at the corresponding object.
Status normal flashing	The LED lights up if an ON telegram is present at the corresponding object.
Status inverted flashing	The LED starts flashing if an OFF telegram is present at the corresponding object.
Status normal soft-flashing	The LED starts soft-flashing (→ *chapter 3.3*) if an ON telegram is present at the corresponding object.
Status inverted soft-flashing	The LED starts soft-flashing (→ *chapter 3.3*) if an OFF telegram is present at the corresponding object.

Parameter **LED colour** specifies in which colour the LED lights up.

LED colour	*Red* / *Green* / ***Blue*** / *White* / *Yellow* / *Vviolet* *User colour 1–4*
Red / *Green* / *Blue* / *White* / *Yellow* / *Violet*	The LED lights up in the selected colour.
User colour 1–4	The LED lights up in the colour mixed on the parameter page "LED colours" (→ *chapter 2.3.5*).

Parameter **LED function overridable with object signal LED** specifies if the LED can be overridden.

LED function overridable with object signal LED	***No*** *Yes*
No	The LED cannot be overridden.
Yes	The LED is overridden as soon as the corresponding telegram is present at the object <Push-button x, signal LED – Override/show on LED>. The parameters **Signal LED display mode** and **Signal LED colour** are visible.

Parameter **Signal LED display mode** specifies how the LED is to be overridden.

Signal LED display mode	*Status normal* *Status inverted* ***Status normal flashing*** *Status inverted flashing* *Status normal soft-flashing* *Status inverted soft-flashing*
Status normal	The LED lights up and overrides the normal LED function if an ON telegram is present at the object <Push-button x, signal LED – Override/show on LED>.
Status inverted	The LED lights up and overrides the normal LED function if an OFF telegram is present at the object <Push-button x, signal LED – Override/show on LED>.
Status normal flashing	The LED starts flashing and overrides the normal LED function if an ON telegram is present at the object <Push-button x, signal LED – Override/show on LED>
Status inverted flashing	The LED starts flashing and overrides the normal LED function if an OFF telegram is present at the object <Push-button x, signal LED – Override/show on LED>
Status normal soft-flashing	The LED starts soft-flashing (→ *chapter 3.3*) and overrides the normal LED function if an ON telegram is present at the object <Push-button x, signal LED – Override/show on LED>.
Status inverted soft-flashing	The LED starts soft-flashing (→ *chapter 3.3*) and overrides the normal LED function if an OFF telegram is present at the object <Push-button x, signal LED – Override/show on LED>.

Parameter **Signal LED colour** specifies with which colour the LED is overridden.

Signal LED colour	*Red* / ***Green*** / *Blue* / *White* / *Yellow* / *Violet* *User colour 1–4*
Red / *Green* / *Blue* / *White* / *Yellow* / *Violet*	The LED lights up in the selected colour.
User colour 1–4	The LED lights up in the colour mixed on the parameter page "LED colours" (→ *chapter 2.3.5*).

Parameter **Longer press left push-button** and/or **Longer press right push-button** specifies if an additional function is carried out by pressing the push-button longer.

Longer press left push-button **Longer press right push-button**	***Not active*** *Active*
Not active	Pressing the push-button longer does not have any effect.
Active	By pressing the push-button longer, an additional command is sent to an additional object. The object <Push-button x (longer press)> is visible. The parameters **Time for longer press** and **Longer press function** are visible.

Parameter **Time for longer press** specifies for how long the push-button is to be pressed in order to trigger the command defined with the parameter **Longer press function**.

Time for longer press	***0.5 sec.*** */ 1 sec. / 2 sec. .. 10 sec.*

Parameter **Longer press function** specifies the additional command to be triggered by pressing the push-button longer.

Longer press function	***Switching*** *Dimming value in %* *Blind UP / DOWN* *Value* *Scene*
Switching	By pressing the push-button longer, the telegram specified with the parameter **Switching function** is sent to the object <Push-button x (longer press) – ON/OFF, switching>.
Dimming value in %	By pressing the push-button longer, the telegram specified with the parameter **Dimming value function** is sent to the object <Push-button x (longer press) – Value, dimming>.
Blind UP / DOWN	By pressing the push-button longer, the telegram specified with the parameter **Blind function** is sent to the object <Push-button x (longer press) – UP/DOWN, Blind>.
Value	By pressing the push-button longer, the value specified with the parameter **8 bit value** (*0..255*) is sent to the object <Push-button x (longer press) – Send, value>.
Scene	By pressing the push-button longer, the scene saved in the actuator with the parameter **Scene number** (*1..64*) is recalled. (object <Push-button x (longer press) – Recall, scene>)

2.3.4 Parameter page "LED brightness and flashing speed"

Parameter **Design colour** adjusts the LED brightness depending on the color of the STANDARDdue cover set.

Design colour	***white (darker LED)*** *black (brighter LED)*

Parameter **LED brightness during normal operation** specifies how bright the LED will light up during normal operation. The luminosity is indicated as a percentage of the maximum possible luminosity of the LED.

LED brightness during normal operation	*0..100* (***100***)

Parameter **Night reduction LEDs function** specifies if the LEDs and the LC display (→ *chapter 2.6.1*) are to shine with a decreased brightness/backlighting during night-time.

Night reduction LEDs function	***Not active*** *ON = night reduction active / OFF = inactive* *ON = inactive / OFF = night reduction active*
Not active	The LEDs and the LC display always shine with the specified brightness value for normal operation. **Note:** *Not active* is not to be interpreted as «.. are not lit». Only the object 25 <Night reduction LEDs & display – Decrease brightness> is not displayed thus making the night reduction function unavailable.
ON = night reduction active / OFF = inactive	If an ON telegram is sent to object 25 <Night reduction LEDs & display – Decrease brightness>, the LEDs and the LC display only shine with the degree of brightness specified for night reduction. If an OFF telegram is sent to object 25 <Night reduction LEDs & display – Decrease brightness>, the LEDs and the LC display will return to the illumination value for normal operation. The object 25 <Night reduction LEDs & display – Decrease brightness> is visible. The parameter **LED brightness during night reduction** is visible.
ON = inactive / OFF = night reduction active	If an OFF telegram is sent to object 25 <Night reduction LEDs & display – Decrease brightness>, the LEDs and the LC display only shine with the degree of brightness specified for night reduction. If an ON telegram is sent to object 25 <Night reduction LEDs & display – Decrease brightness>, the LEDs and the LC display will return to the illumination value for normal operation. The object 25 <Night reduction LEDs & display – Decrease brightness> is visible. The parameter **LED brightness during night reduction** is visible.

Parameter **LED brightness during night reduction** specifies how bright the LEDs will be lit during night operation, which is activated via the object 25 <Night reduction LEDs & display – Decrease brightness>. The luminosity is indicated as a percentage of the maximum possible luminosity of the LED.

LED brightness during night reduction	*0..100* (***50***)

Parameter **Flashing speed LEDs** specifies at what cadence the LEDs will be flashing.

Flashing speed LEDs	*very fast (0.5 sec.)* ***fast (1 sec.)*** *slow (2 sec.)* *very slow (4 sec.)*

2.3.5 Parameter page "LED colours"

On the parameter page "LED colours", two user-specific colours (LED user colour 1–4) can be «mixed» in an additive manner in order to match them to the environment.

Parameter

Red, **Green** and **Blue** determine the numeric portion of the colours red, green and blue in the user colour. Further information on additive colour mixing → *chapter 3.6*.

Red
Green
Blue *0..255*

Since for hardware reasons light control is different for push-button lighting and the LC display, the «same» colour must be defined differently to achieve the «same» colour effect. For the predefined colours (*Red*, *Green*, *Blue*, *White*, *Yellow*, *Violet*) this adjustment is stored in the ETS, meaning that the «right» RGB values are used depending on the application (LED or display).

	RGB value push-button			RGB value display		
Colour	Red:	Green:	Blue:	Red:	Green:	Blue:
Red	102	000	000	102	000	000
Green	000	098	008	025	140	030
Blue	000	000	255	002	020	255
White	105	128	110	255	205	255
Yellow	128	110	000	175	100	010
Violet	089	000	255	089	010	180

For individually mixed colours (*User colour 1–4*) an individual RGB value must be defined for each application, e.g. **User colour 1** for the LED colour assigned to the parameter page "Button x" and **User colour 2** for the colour of the lighting assigned to the parameter page "Configuration of Display".

Matching STANDARDdue the following settings for a white LED color are recommended.
This must be defined manually as *User colour x*.

	RGB value push-button			RGB value display		
STANDARDdue	Red:	Green:	Blue:	Red:	Green:	Blue:
black (..60)	090	128	140	255	205	255
white (..61)	124	131	074	124	090	074

The following definitions generate the same colour effect:

	RGB value push-button			RGB value display		
Colour	Red:	Green:	Blue:	Red:	Green:	Blue:
Orange	204	038	000	244	038	000
Pink	191	000	077	255	045	002
Warm white	124	131	074	144	090	074
Cold white	090	128	140	255	205	255

Parameter

Use colour correction allows you to compensate colour differences of LEDs between two different push-buttons.

Use colour correction *No*
Yes

No The colour correction is not used.

Yes The colour correction is used for all LEDs.
The parameters **Red**, **Green** and **Blue** are visible in percent (-100..30) for the correction.

Note: With these parameters, no colours may be set. They should only be used for any correction of colour deviations which may be required in the case of a deviating aging of the LEDs or with minor colour differences of LEDS of different batches.

2.3.6 Parameter page "General disabling"

With the object 24 <All involved push-buttons – Disable push-buttons>, all or individual push-buttons (parameter page "Disable push-buttons" → *chapter 2.3.7*) can be disabled. If a push-button is disabled, it is no longer able to send a signal until the push-button is enabled again. A disabled push-button can be signalled by means of LEDs.

Parameter **Disable push-buttons function** determines the polarity of the disabling object 24 <All involved buttons – Disable buttons>.

Disable push-buttons function	*Not active* *ON = disable / OFF = operation* *ON = operation / OFF = disable*
Not active	The push-buttons cannot be disabled.
ON = disable / OFF = operation	If an ON telegram is sent to the object 24, the push-buttons will be disabled depending on the configuration on the parameter page "Disable push-buttons". If an OFF telegram is sent to object 24, these push-buttons will be enabled again. The object 24 <All involved push-buttons – Disable push-buttons> is visible.
ON = operation / OFF = disable	If an OFF telegram is sent to the object 24, the push-buttons will be enabled depending on the configuration on the parameter page "Disable push-buttons". If an ON telegram is sent to object 24, these push-buttons will be enabled again. The object 24 <All involved push-buttons – Disable push-buttons> is visible.

Parameter **Behaviour for disabling event** specifies if and which telegrams are sent before the push-buttons are disabled.

Behaviour for disabling event	*Maintain state and disable* *ON/DOWN, then disable* *OFF/OFF, then disable*
Maintain state and disable	Only the push-button is disabled. The state of the actuator is not changed.
ON/DOWN, then disable	If the push-button is disabled, an ON telegram is sent to the corresponding group address (1 / DPT 1.001) and the push-button is disabled.
OFF/OFF, then disable	If the push-button is disabled, an OFF telegram is sent to the corresponding group address (0 / DPT 1.001) and the push-button is disabled.

Note: During disabling, the telegram is always sent via the group address of the push-button object with the lowest object number. The telegram is only sent via 1 bit objects. If the object has another data type, no telegram will be sent.

Parameter **LED display mode, if disabled** specifies if and how the LEDs will react if the push-button has been disabled via the object 24 <All involved push-buttons – Disable push-buttons>.

LED display mode, if disabled	*Not active (is not overridden)* *ON (switched on if disabled)* *OFF (switched off if disabled)* ***Sequence (3x flashing / 3x pause if disabled)*** *Flashing (flashing if disabled)* *Soft-flashing (soft-flashing if disabled)*
Not active (is not overridden)	If the push-button is disabled, this does not have an effect on the state of the LED. If available and parameterised, the LED will show its «normal» function.
ON (switched on if disabled)	If the push-button is disabled, the LED will be lit permanently.
OFF (switched off if disabled)	If the push-button is disabled, the LED will be switched off.
Sequence (3x flashing / 3x pause if disabled)	If the push-button is disabled, the LED starts flashing in a specific disable flashing sequence.
Flashing (flashing if disabled)	If the push-button is disabled, the LED will flash continuously.
Soft-flashing (soft-flashing if disabled)	If the push-button is disabled, the LED will soft-flash continuously (→ *chapter 3.3*).

The flashing speed is determined by the general parameter **Flashing speed LEDs** on the parameter page "LED brightness and flashing speed" (→ *chapter 2.3.4*).

Parameter **LED colour** specifies in which colour the LED lights up.

LED colour	***Red*** */ Green / Blue / White / Yellow / Violet* *User colour 1–4*
Red / Green / Blue / White / Yellow / Violet	The LED lights up in the selected colour.
User colour 1–4	The LED lights up in the colour mixed on the parameter page "LED colours" (→ *chapter 2.3.5*).

2.3.7 Parameter page "Disable push-buttons"

On the parameter page "Disable push-buttons", individual push-buttons can be excluded from the disabling function on the parameter page "General disabling" (→ *chapter 2.3.6*).

Note: The following parameter is available for each of the individual push-buttons. To simplify matters, the parameterisation is described using only one parameter as example.

Parameter **Push-button x** determines whether or not the push-button can be disabled via object 24 <All involved push-buttons – Disable push-buttons>.

Push-button x	*Yes* *No*

2.4 Parameters sequence module

2.4.1 Parameter page "Sequence module"

Parameter **Sequence module** enables the definition of a sequence (→ *chapter 3.4*) with up to 8 switching points which can be parameterised.

Sequence module	*Not active* *Active*
Not active	No sequence has been defined. All follow-up parameters are hidden.
Active	The sequence can be defined with up to 8 parameterisable switching points. The object 87 <Sequence module – Recall sequence> is visible. Object 88 <Sequence module – Status> is visible.

Parameter **Switching point x** specifies if the switching point is passed through in the sequence.

Switching point x	*Not active* *Active*
Not active	The switching point is not active.
Active	The switching point is active and will be «passed through». The command to be executed is specified with the parameter **Switching point x function** on the parameter page "Switching point x" (→ *chapter 2.4.2*). The object <Switching point x – ON/OFF, switching> is visible.

Parameter **Restart sequence module after last switching point?** specifies whether the sequence is restarted from the beginning after the last switching point has been processed.

Restart sequence module after last switching point?	*No* *Yes*
No	The sequence can be stopped by pressing the push-button longer when an OFF telegram is received at the object 87 <Sequence module – Recall sequence>. If it is not stopped manually, this will happen after the last switching point.
Yes	The sequence restarts from the beginning. It is only stopped by the press of a push-button (if the push-button has been parameterised accordingly) or if an OFF telegram is received at the object 87 <Sequence module – Recall sequence>.

2.4.2 Parameter page "Switching point x"

Parameter **Time interval to starting point** and **Time interval to previous active switching point** indicate the time interval to the starting point or to the previous switching point in seconds.

Time interval to starting point	
Time interval to previous active switching point	*0..3600* (***0***)

Parameter **Switching point x function** specifies the function to be executed at the corresponding switching point.

Switching point x function	***Switching*** *Dimming value in %* *Blind UP / DOWN* *Value* *Scene*
Switching	The telegram specified with the parameter **Switching function** is sent to the object <Switching point x – ON/OFF, switching>.
Dimming value in %	The value specified with the parameter **Dimming value function** is sent to the object <Switching point x – Value, dimming>.
Blind UP / DOWN	The telegram specified with the parameter **Blind function** is sent to the object <Switching point x – UP/DOWN, Blind>.
Value	The value specified with the parameter **8 bit value** (*0..255*) is sent to the object <Switching point x – Send, value>.
Scene	The scene saved in the actuator with the parameter **Scene number** (*1..64*) is recalled (object <Switching point x – Recall, scene>)

2.5 Parameters scene module

2.5.1 Parameter page "Scene module"

On the parameter page "Scene module", the number of group addresses and the functioning of the scene saving is specified when using local scene saving.

Parameter **Scene function** specifies the type of scenes (→ *chapter 3.5*).

Scene function	***Decentralised scene saving (in actuator)*** *Local scene saving (in push-button)*
Decentralised scene saving (in actuator)	The scene values are remotely saved in the actuators (8-bit scene).
Local scene saving (in push-button)	The scene values are locally saved in the KNX push-button (conventional scene).

Parameter **Number of scene values per scene** specifies the maximum number of scene values per scene. The value applies to all scenes.

Number of scene values per scene	***max. 10 values/objects per scene*** *max. 15 values/objects per scene*
max. 10 values/objects per scene	Per scene, a maximum of 10 different scene values can be recalled and saved.
max. 15 values/objects per scene	Per scene, a maximum of 15 different scene values can be recalled and saved.

Parameter **Scene mode for the user during the operation** specifies if and how scenes can be saved by the user.

Scene mode for the user during the operation	*Only recall scene* ***Recall scene and save all*** *Recall scene and save selectively*
Only recall scene	The scene can be recalled by the push-button but it cannot be saved. The saving of scenes is only carried out via ETS.
Recall scene and save all	The scene can be recalled and saved by the push-button. If the push-button is pressed and held, the current state of all group addresses assigned to the scene is queried and saved. After approx. 3 seconds, the LED starts flashing quickly, after another 4 seconds it will light up permanently, indicating that the scene has been saved. If the push-button is pressed for a very long time (approx. 12 seconds), the scene will be deleted.
Recall scene and save selectively	The scene can be recalled and saved by the push-button. Only changed values will be taken into account in the new scene. Group addresses, that were not changed during the scene saving procedure, will not be saved. If the push-button is pressed and held, the LED will start flashing after approx. 3 seconds. If the push-button is released now, the desired loads can be set within a time of 4 minutes. If the push-button is pressed and held again, the LED will light up permanently after approx. 3 seconds indicating that the scene has been saved. If the push-button is pressed shortly during the programming, the programming mode will be exited without saving. If the push-button is pressed for a very long time (approx. 12 seconds), the scene will be deleted.

Note: The actuator value will not be saved in the scene if **Presetting scene value x** = *Disabled* (parameter page "Scene x [value 1…10/1…15]" → *chapter 2.5.3*).

Parameter **Transmission delay between scene telegrams** specifies the duration of the pauses between the individual telegrams of a scene when the scene is recalled.

Transmission delay between scene telegrams	*25 ms* / ***50 ms*** / *75 ms* / *100 ms*

Note: The more quickly the telegrams follow in sequence, the higher the bus load.

Parameter **Recall scene via object** is permanently set to *1 = recall scene*. Via the corresponding objects in the ETS, scenes can be recalled using additional switches by sending an ON telegram to the corresponding object number.

2.5.2 Parameter page "Data type scene value 1…10/1..15"

Note: For the parameterisation of the data types of the scene values per scene, a designated parameter page is available. However, since both parameter pages are progressively structured, they will both be explained together at this point.

Parameter **Data type scene value x** specifies the data type (DPT) of the individual save points.

Data type scene value x	*1 bit (switching ON/OFF, blind UP/DOWN)* *8 bit (dimming value, blind value)*
1 bit (switching ON/OFF, blind UP/DOWN)	When the scene is triggered, a 1 bit telegram is sent to the corresponding group address. This way, the state of the object <Scene value x – ON/OFF, UP/DOWN> is switched according to the state saved in the scene.
8 bit (dimming value, blind value)	When the scene is triggered, a 8 bit telegram is sent to the corresponding group address. This way, the value of the object <Scene value x – Send, value> is switched according to the value saved in the scene.

2.5.3 Parameter page "Scene x [value 1…10/1…15]"

Note: For the parameterisation of the presetting of the scene values per scene, a designated parameter page is available. However, since all parameter pages are identically structured, they will be explained altogether at this point.

Parameter **Presetting scene value x** specifies the scene value to be sent. During operation, new values can be saved via the push-button.

Presetting scene value x	**Data type scene value x** = *1 bit (switching ON/OFF, blind UP/DOWN)* ***Disabled*** *Switching ON, blind DOWN* *Switching OFF, blind UP* **Data type scene value x** = *8 bit (dimming value, blind value)* ***Disabled*** *0 % / 5 % / 10 % / 15 % .. 100 %*
Disabled	The scene value x is not involved in the scene x. Therefore, the corresponding group address remains unchanged upon recall of scene.
Switching ON, blind DOWN	When the scene is triggered, a 1 bit telegram with the value (1) is sent to the corresponding group address (DPT 1.001/1.008). This causes the light to be switched on or the blind to be closed.
Switching OFF, blind UP	When the scene is triggered, a 1 bit telegram with the value (0) is sent to the corresponding group address (DPT 1.001/1.008). This causes the light to be switched off or the blind to move up.
0 % / 5 % / 10 % / 15 % .. 100 %	When the scene is triggered, a 8 bit telegram with the preset value is sent to the corresponding group address (DPT 5.001). This causes the light to be adjusted to the desired brightness or the blind to move to the corresponding position.

2.6 Display parameters

2.6.1 Parameter page "Configuration of display"

What and in which form items appear on the LC display is defined in the parameter page "Configuration of display" (→ *chapter 3.2.2*).

* The parameters marked with a star are only visible if the push-buttons for operating the room thermostat are used (parameter **Push-buttons control** = *Room thermostat* → *chapter 2.3.1*).

Parameter **Display operating mode** defines the display on the LC display.

Display operating mode	***Lighted background / black font*** *Black background / lighted font*
Lighted background / black font	The display consists of black font on a lighted background. This setting is particularly suitable if the room thermostat is installed in a bright environment. Thanks to the transreflective LC display, the font is clearly legible in a bright environment, even if it is not lighted.
Black background / lighted font	The display is set to a lighted font on a black background. With this setting the font is almost illegible when it is not lighted - the display looks like it is switched off.

Parameter **Switch operating mode via object** defines the switching command for switching over the operating mode display via the bus.

Switch operating mode via object	***ON = black font / OFF = lighted font*** *ON = lighted font / OFF = black font*
ON = black font / OFF = lighted font	If an ON telegram is sent to object 75 <Display – Switch operating mode>, the display is set to a black font on a lighted background. If an OFF telegram is sent, a lighted font is displayed on a black background.
ON = lighted font / OFF = black font	If an ON telegram is sent to object 75 <Display – Switch operating mode>, the display is set to a lighted font on a black background. If an OFF telegram is sent, a black font is displayed on a lighted background.

* Parameter **Change operating mode after pressing push-button** defines whether the operating mode of the LC display is to be changed for a certain time after a push-button press. By changing the operating mode it can be indicated to the user that the RTH push-button is in setting mode and the user can make settings (if required).

Change operating mode after pressing push-button	***not active*** *Lighted background / black font* *Black background / lighted font*
not active	The operating mode is not changed.
Lighted background / black font	The displays are set to black font on a lighted background. The parameter **Duration of change operating mode after pressing push-button** is visible
Black background / lighted font	The display is set to a lighted font on a black background. The parameter **Duration of change operating mode after pressing push-button** is visible

* Parameter **Duration of change operating mode after pressing push-button** defines in seconds how long the operating mode change remains valid after pressing a push-button. The time duration begins again each time a push-button is pressed.

Duration of change operating mode after pressing push-button	*1..3600* (***30***)

Parameter **Display lighting** defines whether and how the LCD backlighting is switched on.

Display lighting	*Always ON* *Always OFF* *According to object value (1=ON, 0=OFF)* *According to object value (1=OFF, 0=ON)* *RGB signal LED object (external signal)* ***Temporarily ON after pressing push-button***
Always ON	The backlighting is switched on permanently.
Always OFF	The backlighting is always switched off and is not temporarily switched on, even when a push-button is pressed.
According to object value (1=ON, 0=OFF)	If an ON telegram is sent to object 46 <Display – ON/OFF, backlighting>, the backlighting is switched on. If an OFF telegram is transmitted, it switches off. If the backlighting is switched off, it is switched on for the set time and then off again when a push-button is pressed. The Object 46 <Display – ON/OFF, backlighting> is visible.
According to object value (1=OFF, 0=ON)	If an OFF telegram is sent to object 46 <Display – ON/OFF, backlighting>, the backlighting is switched on. If an ON telegram is transmitted, it switches off. If the backlighting is switched off, it is switched on for the set time and then off again when a push-button is pressed. The Object 46 <Display – ON/OFF, backlighting> is visible.
RGB signal LED object (external signal)	If an RGB telegram is sent to object 46 <Display – RGB backlighting>, the backlighting lights up in the relevant colour. The RGB telegram must contain the proportions of the colour values for red, green and blue. To switch off the backlighting, the value 0 must be sent for all 3 colour values. Object 46 <Display – RGB backlighting> is visible. The parameter **Colour lighting** is hidden.
** Temporarily ON after pressing push-button*	The backlighting is switched on for the set time after a push-button is pressed. The parameter **Duration of lighting after pressing push-button** is visible.

* Parameter **Duration of lighting after pressing push-button** defines in seconds how long the backlighting remains on after pressing a push-button. The time duration begins again each time a push-button is pressed.

Duration of lighting after pressing push-button	*1..3600* (***30***)

The lighting duration is independent of the value for the parameter **Duration of change operating mode after pressing push-button**. The fact that setting mode has been exited can be indicated to the user by means of a small difference between the two values.

Parameter **Brightness during normal operation** defines the brightness of the backlighting. The brightness is specified as a percentage of the maximum possible brightness.

Brightness during normal operation	*0..100* (***100***)

Parameter **Brightness during night reduction** defines the brightness of the backlighting during night operation, activated via object 25 <Night reduction LEDs & display – decrease brightness>. The brightness is specified as a percentage of the maximum possible brightness.

The night reduction can only be activated if a value that is not equal to *not active* is entered on parameter page "LED brightness and flashing speed" (→ *chapter 2.3.4*) for the parameter **Night reduction LEDs function**.

Brightness during night reduction	*0..100* (***50***)

Parameter **Colour lighting** defines the colour with which the LCD display is lit.

Colour lighting	***Red*** / *Green* / *Blue* / *White* / *Yellow* / *Vviolet* *User colour 1–4*
Red / *Green* / *Blue* / *White* / *Yellow* / *Violet*	The lighting comes on in the selected colour.
User colour 1–4	The LED lights up in the colour mixed on the parameter page "LED colours" (→ *chapter 2.3.5*).

Parameter **Contrast** defines the contrast of the liquid crystal display.

Contrast	*–3* / *–2* / *–1* / ***0*** / *+1* / *+2* / *+3*

Parameter **Language** defines the language of the user interface (GUI).

Language	***German*** *French* *Italian* *English*

Parameter **Change language via object** defines whether and with which data format the language is changed via the bus.

Change language via object	*No* ***2 byte object*** *8 bit object*
No	The language cannot be changed.
2 byte object	The language is changed using the standard 2 byte object in accordance with ISO 639-1. The following values must be transmitted: - 6465h for German de - 6672h for French fr - 6974h for Italian it - 656Eh for English en Object 71 <Display – Language of display> is visible.
8 bit object	The language is changed with an 8 bit object. The value to be sent (1–64) is defined on the parameter page "Language scenes". The parameter page "Language scenes" is visible. Object 71 <Display – Language of display> is visible.

Parameter **Display x** (1–5) in the 'Display information' section defines which information is displayed.

Display x	*Not active* *Actual temperature* *Set point temperature* *External temperature* *Controller operating mode* *Time* *Date* *Fan levels* *Wind speed* *Relative air humidity* *CO2* *Free value* *Only symbols (empty value display)*
Not active	The relevant **Display x** is not used, i.e. it is skipped when the change takes place.
Actual temperature	The room temperature measured by the temperature sensor (object 56 <Room temperature actual value – control value>) is displayed. Headline: *Actual temp.*
Set point temperature [1)]	The set point temperature setting (object 56 <Room temperature set point value – set>) is displayed. Headline: *Set point temp.*
External temperature	The external temperature measured by a KNX weather station (e.g. Feller 4720 MS) and signalled via object 61 <Display information – external temperature> is displayed. Headline: *External temp.*
Controller operating mode [1)]	The current controller operating mode (object 47 <Controller operating mode.) is displayed as text. Headline: *Operating mode* The operating modes that can be selected by the user can be configured in the section entitled 'Operating modes selectable at device'.
Time [1)]	The time that is signalled via object 62 <Display information – Time> or set by the end user is displayed in format hh.mm. Headline: *Time*
Date [1)]	The date that is signalled via object 63 <Display information – Date> or set by the end user is displayed in format dd.mm.yy. Headline: *Date*
Fan levels [1)]	The fan level returned via object 65 <Display information – Fan level> is displayed. Headline: *Fan level*
Wind speed	The wind speed measured by a KNX weather station (e.g. Feller 4720 MS) and signalled via object 66 <Display information – Wind speed> is displayed in m/s. Headline: *Wind speed*
Relative air humidity	The relative air humidity signalled via object 67 <Display information – Air humidity> (ratio of actual amount of water vapour in the air compared with the amount it can hold) is displayed as a percentage. Headline: *Air humid.*
CO2	The concentration of CO_2 in the air (room air quality) signalled via object 68 <Display information – Air quality> is displayed in ppm (parts per million 10^{-6}). Headline: *CO2*
Free value	The value signalled via object 69 <Display information – Free value> is displayed. This can be defined for all four languages on parameter page "Free value" (→ *chapter 2.6.3*).
Only symbols (empty value display)	No value is displayed, i.e. the value display is empty.

1) By configuring the push-buttons accordingly (parameter **Push-buttons control** = *Room thermostat*) the values can be modified by the user (see *Bedienungsanleitung STANDARDdue elegance KNX-RTH-Taster RGB mit Raumthermostat*).

Parameter **Heating/cooling symbol is active** defines whether the room thermostat function is displayed in the top right-hand corner of the display.

Heating/cooling symbol is active	*Do not display* *Display if operating mode is active* ***If heating or cooling required***
Do not display	The function is not displayed.
Display if operating mode is active	The symbol ♨ (heat emission) or ❄ (cooling) is displayed if the relevant function is active.
If heating or cooling required	The symbol ♨ (heat emission) or ❄ (cooling) is only displayed if the relevant function has been activated and the controller is demanding heating or cooling power.

Parameter **Controller operating mode symbol** defines whether the symbol for the active controller operating mode appears in the top left-hand corner of the display.

Controller operating mode symbol	*Do not display* ***Display***

Parameter **Fan level and speed symbol** defines whether the returned fan level and the symbol ✣ appear in the bottom right-hand corner of the display.

Fan level and speed symbol	*Do not display* ***Display***

Parameter **Justification top** defines the justification of the headline (text at top in the centre).

Justification top	*Left* ***In the centre*** *Right*

Parameter **Justification bottom** defines the alignment of the text display (text at bottom in the centre).

Justification bottom	*Left* ***In the centre*** *Right*

Parameter **Display "Auto"** defines the polarity for displaying automatic fan mode.

Display "Auto"	***If fan status automatic = "0"*** *If fan status automatic = "1"*
If fan status automatic = "0"	Automatic mode A✣ is displayed if an OFF telegram is sent to object 64 <Display information – Fan status automatic>.
If fan status automatic = "1"	Automatic mode A✣ is displayed if an ON telegram is sent on object 64 <Display information – Fan status automatic>.

Parameter **Decimal places shown in display for actual and external temperature** defines the format for displaying the actual temperature and the external temperature.

Decimal places shown in display for actual and external temperature	***0 decimal places (1 °C step)*** *1 decimal place (0.5 °C step)* *1 decimal place (0.1 °C step)*

Parameter **Decimal places shown in display for set point temperature** defines the format for displaying the set point temperature. The set point temperature can be altered by the user (with appropriate push-button configuration).

Decimal places shown in display for set point temperature	*0 decimal places (1 °C step)* ***1 decimal place (0.5 °C step)*** *1 decimal place (0.1 °C step)*

Parameter **Set point temperature display** defines how the set point temperature is displayed.

Set point temperature display	*relative* ***absolute***
relative	The set point temperature that has been retrospectively modified by the user is displayed relative to the value defined on the parameter page "Set point values" with the parameter **Basic set point (comfort temperature)**. If the value has not been changed, 0 °C is displayed
absolute	The set point temperature is displayed absolute in °C.

Parameter **Automated change between displays** defines whether the value display is changed automatically.

Automated change between displays	*Yes* ***No***
Yes	The value display changes alternates between the individual items of information. The parameter **Change every x sec.** is displayed.
No	Value display does not change automatically.

Parameter **Change every x sec.** defines in seconds how long a value appears in the display until it switches atuomatically to the next value.

Change every x sec.	*1..3600* (***3***)

Parameter **Change between displays via object** defines whether the value display can also be changed via the bus.

Change between displays via object	*Yes* ***No***
Yes	The value display is switched via object 70 <Display information – Change>. Object 70 <Display information – Change> is visible.
No	The change only takes place if both push-buttons (parameter **Push-buttons control** = *Room thermostat*) are pressed simultaneously or automatically.

Parameter **Receive time and date via** defines the data format in which the time and date are received via the bus.

Receive time and date via	***2 separate 3 byte objects Time / Date*** *1 common 8 byte object Time & Date*
2 separate 3 byte objects Time / Date	The time and date are received by two separate 3 byte objects (DTP 10.001 and 11.001). Objects 62 <Display information – Time> and 63 <Display information – Date> are visible.
1 common 8 byte object Time & Date	The date and time are received by one 8 byte object (DTP 19.001). Object 62 <Display information – DateTime> is visible.

Parameter **Comfort operation** defines if it is possible to switch to comfort operation (🏠) at the device when the controller operating mode is displayed (parameter **Display**).

Comfort operation	*Yes* *No*

Parameter **Standby operation** defines if it is possible to switch to standby operation (⌂) at the device when the controller operating mode is displayed (parameter **Display**).

Standby operation	*Yes* *No*

Parameter **Night operation** defines if it is possible to switch to night operation (☾) at the device when the controller operating mode is displayed (parameter **Display**).

Night operation	*Yes* ***No***

Parameter **Comfort extension** defines if it is possible to active comfort extension (🚶) at the device when the controller operating mode is displayed (parameter **Display**).

Comfort extension	*Yes* *No*

Parameter **Duration of the comfort extension** defines the duration of the comfort extension. This value can be adapted by the user (with appropriate push-button configuration).

Duration of the comfort extension	*0.5 hours* / ***1.0 hour*** / *1.5 hours* / *2.0 hours* *3.0 hours* / *4.0 hours*

Parameter **Frost/heat protection** defines if it is possible to activate the frost/heat protection (❄) at the device when the controller operating mode is displayed (parameter **Display**).

Frost/heat protection	*Yes* *No*

2.6.2 Parameter page "Language scenes"

If the display language is changed using an 8 bit object (parameter **Change language via object** = *8 bit object*), the values to be received can be defined on the parameter page "Language scenes".

Parameter **German** defines the value to be received for the german user interface.

German	*1..64* (***1***)

Parameter **French** defines the value to be received for the french user interface.

French	*1..64* (***2***)

Parameter **Italian** defines the value to be received for the italian user interface.

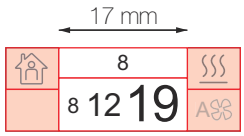
Italian	*1..64* (***3***)

Parameter **English** defines the value to be received for the english user interface.

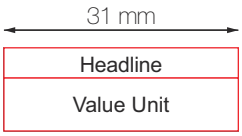
English	*1..64* (***4***)

2.6.3 Parameter page "Free value"

The display for **Display** = *Free value* is defined for all four languages on the parameter page "Free value".



Display symbols on the left and the right side = *Yes*



Display symbols on the left and the right side = *No*

The font used in the display is ***Arial Unicode***. Depending on whether the symbols are displayed or not (parameter **Display symbols on left and right margin**), the display has a width of 17 mm (*Yes*) or 31 mm (*No*).

Parameter **Headline (DE) / (FR) / (IT) / (EN)** defines the headline in the respective language.

Headline (DE) **Headline (FR)** **Headline (IT)** **Headline (EN)**	The display font size is ***8 pt***. The number of characters that can be displayed depends on the respective text (proportional font, can be tried out with a text editor) and is a maximum of 19 characters.

Parameter **Format** defines the data type of object 69 <Display information – Free value>.

Format	*1 bit (DPT 1.xxx)* *8 bit unsigned value (DPT 5.001, percentage)* *8 bit unsigned value (DPT 5.xxx)* *8 bit signed value (DPT 6.xxx)* *2 byte unsigned value (DPT 7.xxx)* *2 byte signed value (DPT 8.xxx)* ***2 byte float value without fraction (DPT 9.xxx)*** *2 byte float value with fraction (DPT 9.xxx)* *4 byte unsigned value (DPT 12.xxx)* *4 byte signed value (DPT 13.xxx)* *4 byte float value without fraction (DPT 14.xxx)* *4 byte float value with fraction (DPT 14.xxx)*

Parameter **Value adjustment** defines whether the transferred value is to be converted.

Value adjustment	***None*** *divided* *multiplied*
None	No value adjustment is made.
divided	The received value is divided by the number contained in the parameter **Value**. The parameter **Value** is displayed.
multiplied	The received value is multiplied by the number contained in the parameter **Value**. The parameter **Value** is displayed.

Parameter **Value** defines the divisor or multiplier for the received value.

Value	*0..10000000* (***1000***)

Parameter **Minimum value** and **Maximum value** define the minimum and maximum value to be displayed.

Minimum value **Maximum value**	*-9999999..99999999*

Parameter **Unit** defines the unit (text) to be displayed with the value.

Unit	max. 5 characters

Parameter **Font size of value and unit** defines the font size for the output.

Font size of value and unit	*small* ***average*** *big*
small	The display font size is ***8 pt***.
average	The display font size is ***12 pt***.
big	The display font size is ***19 pt***.

Parameter **Display of symbols at left and right margin** defines whether the symbols defined on the parameter page "Configuration of display" (→ *chapter 2.6.1*) are displayed.

Display of symbols at left and right margin	*Yes* *No*

2.7 Parameters room thermostat

2.7.1 Parameter page "Heating/cooling system"

On the parameter page "Heating/cooling system", the function of the room thermostat (→ *chapter 3.7.1*) as well as the control algorithm used (→ *chapter 3.8*) are determined.

Parameter **Activation of the heating/cooling function** determines the type of system to be controlled.

Activation of the heating/ cooling function	*Heating* *Cooling* *Heating and cooling* *2-stage heating* *2-stage cooling*
Heating	The room thermostat controls a heating system. If the current actual value is below the current set point value, this difference is balanced out by emitting a calculated correcting variable with the object 52 <Correcting variable – Heating>.
Cooling	The room thermostat controls a cooling system. If the current actual value exceeds the current set point value, this difference is balanced out by emitting a calculated correcting variable with the object 52 <Correcting variable – Cooling>.
Heating and cooling	The room thermostat controls a heating and a cooling system. For each function, an own control algorithm can be specified. The calculated correcting variables are issued with the objects 52 <Correcting variable – Heating> and 53 <Correcting variable – Cooling>. With the parameter **Switchover between heating and cooling** (parameter page "Functionality" → *chapter 2.7.4*) it is determined how it is switched between heating and cooling.
2-stage heating	The room thermostat controls a heating system with basic and additional levels. With the parameter **Interval between basic level and additional level** (parameter page "Set point values" → *chapter 2.7.2*) it is determined up to which temperature the additional level remains active. For the basic and the additional level, separate correcting variables are calculated and transmitted to the bus with the objects 52 <Correcting variable – Basic heating> and 53 <Correcting variable – Additional heating>. The parameters **Correcting variable of the additional level** and **Hysteresis of the additional level** are visible.
2-stage cooling	The room thermostat controls a cooling system with a basic and an additional level. With the parameter **Interval between basic level and additional level** (parameter page "Set point values" → *chapter 2.7.2*) it is determined up to which temperature the additional level remains active. For the basic and the additional level, separate correcting variables are calculated and transmitted to bus with the objects 52 <Correcting variable – Basic cooling> and 53 <Correcting variable – Additional cooling>. The parameters **Correcting variable of the additional level** and **Hysteresis of the additional level** are visible.

Parameter **Type of heating function / cooling function / basic level** determines the control algorithm (→ *chapter 3.8*) of the heating or cooling system to be controlled.

Type of heating function **Type of cooling function** **Type of basic level**	*Continuous PI control* ***Switching PI control*** *Continuous 2-point control* *Switching 2-point control*
Continuous PI control	The control variable calculated by the room thermostat (0–100 %) is sent via an 1 byte value object directly via the bus to the system, which in turn implements it directly in a certain degree of openness. The parameter **Adjustment of the PI control to the heating system / cooling system** is visible.
Switching PI control	The correcting variable calculated by the room thermostat (0–100 %) is converted into an equivalent pulse width modulated (PWM) correcting variable. Within an adjustable cycle time (3–30 minutes) the actuator is opened via an 1 bit switching object (1) for the calculated duration in percent and then closed again (0). The parameter **Adjustment of the PI control to the heating system / cooling system** is visible.
Continuous 2-point control	The actuators are activated (100 %) or deactivated (0 %) via an 8 bit object. These parameter setting is only sensible in specific cases, e.g. for controlling a constant valve with 2-point correcting variables. The parameter **Hysteresis of the 2-point controller heating / cooling** is visible.
Switching 2-point control	The actuators are activated (1) or deactivated (0) via an 1 bit object. The parameter **Hysteresis of the 2-point controller heating / cooling** is visible.

Parameter **Adjustment of the PI control to the heating system** determines predefined values for different heating systems, for the control parameters **Proportional range for heating** and **Reset time for heating** (→ *chapter 3.8.2*).

Adjustment of the PI control to the heating system	***Warm water heating (5 K / 150 min)*** *Underfloor heating (5 K / 240 min)* *Electrical heating (4 K / 100 min)* *Fan coil unit (4 K / 90 min)* *SplitUnit (4 K / 90 min)* *Via control parameters*
Via control parameters	Provided that sufficient expert knowledge is given, the adjustment may be implemented via the control parameters **Proportional range for heating** and **Reset time for heating**.

Parameter **Adjustment of the PI control to the cooling system** determines predefined values for different cooling systems, for the control parameters **Proportional range for cooling** and **Reset time for cooling** (→ *chapter 3.8.2*).

Adjustment of the PI control to the cooling system	***Cooling ceiling (5 K / 240 min)*** *Fan coil unit (4 K / 90 min)* *SplitUnit (4 K / 90 min)* *Via control parameters*
Via control parameters	Provided that sufficient expert knowledge is given, the adjustment may be implemented via the control parameters **Proportional range for cooling** and **Reset time for cooling**.

Parameter **Proportional range for heating** / **cooling** in steps of 0.1 K. A small proportional range leads to great overshoots in the case of set point value changes (potentially also continuous oscillations) and a rapid regulation to the set point value, while a large proportional range leads to no (or only minor) overshoots but to a slow regulation.

Proportional range for heating
Proportional range for cooling *10..200* (***40***)

Parameter **Reset time for heating** / **cooling** in minutes. A short reset time leads to a rapid regulation of control deviations (ambient conditions) with the risk of continuous oscillation, while a longer reset time leads to a slow regulation of control deviations.

Reset time for heating
Reset time for cooling *0..240* (***120***)

0 inactive; only the P algorithm is applied (→ *chapter 3.8*).

Note: Changing the control parameter by small values leads to a significantly modified control behaviour.

Parameter **Hysteresis of the 2-point controller heating** / **cooling** determines the temperature range (in steps of 0.1 K) around the set point value to be undercut or exceeded in order to trigger a switching of the 2-point controller. A small hysteresis leads to lower temperature fluctuations but to more frequent switching and thus to a greater bus load. With a greater hysteresis, it is switched less frequently; however, this may lead to uncomfortable temperature fluctuations.

Hysteresis of the 2-point controller heating
Hysteresis of the 2-point controller cooling *0..255* (***2***)

Parameter **Correcting variable of the additional level** determines the type of correcting variable of the 2-point control for the 2-stage control operation. Additional levels may only be controlled via the 2-point control.

Correcting variable of the additional level *Switching*
Continuous

Switching The actuators are activated (1) or deactivated (0) via an 1 bit object.

Continuous The actuators are activated (100%) or deactivated (0%) via an 8 bit object.

Parameter **Hysteresis of the additional level** in steps of 0.1 K for activating the additional level. With heating systems, the additional level is activated if the actual value is larger than the set point value minus the **Interval between basic level and additional level** plus the **Hysteresis of the additional level** and reactivated again if the actual value is lower than the set point value minus the **Interval between basic level and additional level** minus the **Hysteresis of the additional level**. The same applies to cooling systems accordingly.

Hysteresis of the additional level *0..255* (***2***)

2.7.2 Parameter page "Set point values"

On the parameter page "Set point values", the respective set point values (→ *chapter 3.7.3*) are determined for every operating mode.

Parameter **Base set point value (comfort temperature)** determines the room temperature when the room is used.

Base set point value (comfort temperature)	*16 °C .. 31 °C* (***21 °C***)

Note: Overheated rooms are unhealthy: The room temperature should not exceed 20–21°C.
Rule of thumb: A 1°C increase in room temperature consumes approx. 6% more energy.

Parameter **Heating reduction during standby operation** determines the value (based on the base set point value) by which the temperature is to be lowered if the room is temporarily out of use.

Heating reduction during standby operation	*0 K .. 8 K* (***2 K***)

Parameter **Heating reduction during night operation** determines the value (based on the base set point value) by which the temperature is to be reduced during night operation.

Heating reduction during night operation	*0 K .. 8 K* (***4 K***)

Parameter **Set point value frost protection** determines the set point temperature for frost protection.

Set point value frost protection	*4 °C .. 10 °C* (***7 °C***)

Parameter **Increase of cooling during standby operation** determines the value (based on the base set point value) by which the temperature is to be increased if the room is temporarily out of use.

Increase of cooling during standby operation	*0 K .. 8 K* (***2 K***)

Parameter **Increase of cooling during night operation** determines the value (based on the base set point value) by which the temperature is to be increased during night operation or at the weekend.

Increase of cooling during night operation	*0 K .. 8 K* (***4 K***)

Parameter **Set point value heat protection** determines the set point temperature for heat protection.

Set point value heat protection	*4 °C .. 10 °C* (***7 °C***)

Parameter **Dead zone between heating and cooling** determines the temperature zone for mixed operation (*Heating and cooling*), during which neither the heating nor the cooling is active. The comfort temperature for heating corresponds to the **Base set point value (comfort temperature)**, while the comfort temperature for cooling can be derived from the **Base set point value (comfort temperature)** plus the **Dead zone between heating and cooling**.

Dead zone between heating and cooling	*1 K .. 8 K* (***2 K***)

Parameter **Interval between basic level and additional level** determines the temperature difference towards the basic level until which the additional level is to be included in the control, for the 2-stage control operation.

Interval between basic level and additional level	*1 K .. 3 K* (***3 K***)

2.7.3 Parameter page "Operating modes / status"

Parameter **Switch operating mode via** determines whether or not the switching of the operating modes is to be implemented via 1 bit individual objects or 8 bit value object.

Switch operating mode via	*Individual objects (1 bit)* ***8 bit object***
Individual objects (1 bit)	The switching of the operating modes is implemented via the bus by means of the 1 bit switching objects 47 <Controller operating mode – Comfort>, 48 <Controller operating mode – Night>, 49 <Controller operating mode – Frost/heat protection> and 50 <Controller operating mode – Holidays>.
8 bit object	The switching of the operating modes is implemented via the bus in accordance with the KNX specification by means of the 8 bit value object 47 <Controller operating mode – All operating modes>.

For communicating with other systems (e.g. visualisation software etc.), the KNX compatible objects 59 and 60 <Room thermostat status – Feedback signal> are available.

2.7.4 Parameter page "Functionality"

Depending on the function (→ *chapter 2.7.1*), the following parameters are visible on the parameter page "Functionality":

Activation of the heating/cooling function =	*Heating* *2-stage heating*	*Cooling* *2-stage cooling*	*Heating and cooling*
Allocation of the correcting variables to the objects "Heating" and "Cooling"			x
Switchover between heating and cooling			x
Heating/cooling function			x
Operating mode after reset	x	x	x
Activate valve protection	x		x
Valve protection On time	x		x
Cycle of the valve protection	x		x

Parameter **Allocation of the correcting variables to the objects "Heating" and "Cooling"** determines whether or not the correcting variables for heating and cooling are sent via a common object in the mixed operating mode. The parameter may only be changed if the same control type (continuous or switching) is used for both functions; otherwise it is fixed at *Isolated*.

Allocation of the correcting variables to the objects "Heating" and "Cooling"	***Isolated*** *Together on "Heating" object"*
Isolated	Separate objects are available for the correcting variables of the heating system (object 52 <Correcting variable – Heating>) and the cooling system (object 53 <Correcting variable – Cooling>)
Together on "Heating" object"	If the heating and the cooling system are a combined system, the correcting variables can be issued with the same object 52 <Correcting variable – Heating>. The switching between heating and cooling is always implemented via object 57 <Heating/cooling – Switch operating mode>.

For instance, a combined correcting variable object may also be required if it is both heated as well as cooled via a one-tube system (combined heating and cooling system). For this purpose, initially, the temperature of the medium in the one-tube system is to be changed via the system control. Subsequently, the operating mode is set via the object 57 <Heating/cooling – Switch operating mode> (often it is cooled with cold water in the one-tube system during summer, while it is heated by means of hot water in winter).

Parameter **Switchover between heating and cooling** determines how it is switched between heating and cooling in the mixed operating mode.

Switchover between heating and cooling	*Automatic* ***With the "Heating/cooling" object***
Automatic	The switching is automatically implemented depending on the parameterised set point values, the dead zone and the current actual value.
With the "Heating/cooling" object	The switching is exclusively implemented via the object 57 <Heating/cooling – Switch operating mode>. The parameter **Heating/cooling function** is visible.

Parameter **Heating/cooling function** determines the switching command for the object 57 <Heating/cooling – Switch operating mode>.

Heating/cooling function	***OFF = cooling / ON = heating*** *OFF = heating / ON = cooling*

Parameter **Operating mode after reset** determines which operating mode is to be activated after a bus voltage return or a programming process via the ETS. Thus, the respective set point values apply.

Operating mode after reset	***Standby operation*** *Comfort operation* *Night operation* *Frost/heat protection* *Operating mode same as before reset*

Parameter **Activate valve protection** determines whether or not the valve protection is activated. The valve protection prevents that the valves are trapped at the radiator in the case of a longer deactivation of the heating (e.g. in summer) due to deposits in the heating water.

Activate valve protection	***Yes*** *No*
Yes	The valves are opened after an adjustable cycle (**Cycle of the valve protection**) for an adjustable period of time (**Valve protection On time**) (correcting variable 1 resp. 100% unless inverted) and are then closed again (correcting variable 0 resp. 0% unless inverted). Generally, the valve protection is only started for inactive correcting variable objects, i.e. only for those objects which have not demanded any heating energy within the specified cycle. The parameters **Valve protection On time** and **Cycle of the valve protection** are visible.
No	The valve protection is deactivated.

Parameter **Valve protection On time** determines the period of time during which the correcting variable is sent for ON in minutes.

Valve protection On time	*1..10* (***3***)

Parameter **Cycle of the valve protection** determines how often the correcting variable for ON is sent.

Cycle of the valve protection	*once per day* *once per week* ***once per month***

2.7.5 Parameter page "Room temperature measurement"

On the parameter page "Room temperature measurement", the actual values can be compared.

Parameter **Use external temperature sensor** determines whether or not an external sensor is used for the room temperature measurement.

Use external temperature sensor	*Yes* ***No***
Yes	The temperature is measured via an externally connected temperature sensor. Its temperature measurement values can be read via the 2 byte input object 56 <Room temperature actual value – External sensor>. All follow-up parameters are hidden.
No	The temperature is measured locally with the temperature sensor integrated in the room thermostat.

Parameter **Adjustment of the room thermostat to the ambient** determines the type of installation of the room thermostat.

Adjustment of the room thermostat to the ambient	***Flush-mounted*** *Via installation location parameters*
Via installation location parameters	The influences of installations are manually balanced out with the parameters **Time constant** and **Dynamic Offset**, provided that sufficient expert knowledge is given.

Note: In order to be able to determine the room temperature with the internal temperature sensor, the self-heating of the device must be taken into consideration. The influence of the self-heating on the temperature depends on the type of installation. For this reason, it is important that this parameter is set correctly.

Parameter **Time constant** determines the time constant in seconds.

Time constant	*1..7000* (***750***)

Parameter **Dynamic offset** determines the offset in steps of 0.01 K.

Dynamic offset	*10..1000* (***123***)

Parameter **Change of the room temperature for automatic sending** determines the temperature value by which the actual value must change in order to be automatically sent via object 56 <Room temperature actual value – Control value> to the bus.

Change of the room temperature for automatic sending	*Inactive* *0.1 K* / *0.2 K* / ***0.5 K*** / *1.0 K* / *1.5 K* / *2.0 K*
Inactive	The actual value is not sent automatically.

Parameter **Adjustment direction of the room temperature measurement** determines whether the value defined with the parameter **Adjustment value of the room temperature measurement** is added to the menu value or deducted from the measured value.

Adjustment direction of the room temperature measurement	***Increase measured value*** *Reduce measured value*
Increase measured value	The measured value is to be increased if the value measured by the temperature sensor is below the actual room temperature. Actual value = measured value + **Adjustment value of the room temperature measurement**
Reduce measured value	The measured value is to be reduced if the value measured by the temperature sensor exceeds the actual room temperature. Actual value = measured value – **Adjustment value of the room temperature measurement**

Parameter **Adjustment value of the room temperature measurement** determines the value by which the measured value is corrected.

Adjustment value of the room temperature measurement	***0.0 K*** */ 0.5 K / 1.0 K / 1.5 K .. 5.0 K*

Note: The room temperature measurement is in a steady state after an operating time of approx. 45 minutes as of the last restart and/or ETS download. It is thus important that the adjustment value is determined after having been operated for 45 minutes at the earliest.

Parameter **Cycle time for the automatic sending of the room temperature** determines the time interval for the output of the determined actual value via object 56 <Room temperature actual value – Control value>. The output is independent from the change in the actual value.

Cycle time for the automatic sending of the room temperature	***Inactive*** */ 2 min / 10 min / 40 min*
Inactive	The time interval is deactivated. The actual value is not sent cyclically.

2.7.6 Parameter page "Output correcting variable"

Depending on the selection of the control algorithm (→ *chapter 2.7.1*), the following parameters are visible on the parameter page "Output correcting variable":

	PI control		2-point control	
	continuous	switching	continuous	switching
Output of the correcting variable	x	x	x	x
Change for automatic sending	x			
Cycle time of the switching correcting variable		x		
Cycle time for automatic sending	x	x	x	
Filter correcting variable output	x	x	x	x
Minimum correcting variable	x			
Maximum correcting variable	x			
Correcting variable Off			x	
Correcting variable On			x	

Parameter **Output of the correcting variable heating / cooling / basic level / additional level** (HCBA) determines whether the correcting variable telegrams are output in a normal or inverted manner.

Output of the correcting variable HCBA	***Normal*** *Inverted*
Normal	1 (switching) and/or 100% (continuous) corresponds to the maximum heating and/or cooling performance. The greater the correcting variable, the greater the heating and/or cooling performance.
Inverted	0 corresponds to the maximum heating and/or cooling performance. The lower the correcting variable, the lower the heating and/or cooling performance.

Parameter **Change for automatic sending** determines the value by which the correcting variable must change for the 1 byte object 52/53 <Correcting variable – ...> to be sent to the bus for the continuous PI control.

Change for automatic sending	*0..100* (***1***)
0	The function is inactive, the object 52/53 <Correcting variable – ...> is sent after the period of time defined with the parameter **Cycle time for automatic sending** respectively.

Parameter **Cycle time of the switching correcting variable** determines the time interval for the pulse width modulated correcting variables (PWM) for the switching PI control. A short cycle time is used for fast heating systems (e.g. electronic heating); here, the switching frequency and the bus load increases. In the case of a long cycle time, temperature fluctuations in the room occur; it is used for slow heating systems (e.g. underfloor heating/warm water heating).

Cycle time of the switching correcting variable	*3 min* / *5 min* / *10 min* / ***15 min*** / *20 min* / *30 min*

Parameter **Cycle time for automatic sending** determines the time interval for the cyclic sending of the correcting variable via the objects 52/53 <Correcting variable – …>. The sending is implemented independently from a change in the correcting variable.

Cycle time for automatic sending	*Inactive* ***2 min*** / *10 min* / *40 min*
Inactive	The time interval is deactivated. The correcting variable is not sent cyclically.

Parameter **Filter correcting variable output** determines whether or not the output of correcting variable telegrams is restricted to 1 telegram per minute.

Filter correcting variable output	***Do not filter*** *Only 1 telegram per minute*
Do not filter	There are not limitations regarding the number of correcting variables sent per minute.
Only 1 telegram per minute	Maximally 1 telegram per minute is sent to the address of the objects 52/53 <Correcting variable – …>.

Parameter **Minimum correcting variable heating** / **cooling** / **basic level** / **additional level** (HCBA) determines the correcting variable for the continuous PI control to be issued if no heating or cooling performance is demanded. It is used for balancing a valve offset and is to be set to the value at which the valve just remains closed.

Minimum correcting variable HCBA	***0 %*** / *5 %* / *10 %* / *15 %* / *20 %* / *25 %* / *30 %*

Parameter **Maximum correcting variable heating** / **cooling** / **basic level** / **additional level** (HCBA) determines the correcting variable for the continuous PI control to be issued if the full heating or cooling performance is demanded. This parameter corresponds to the value at which the valve is completely opened.

Maximum correcting variable HCBA	*70 %* / *75 %* / *80 %* / *85 %* / *90 %* / *95 %* / ***100 %***

Parameter **Correcting variable Off heating** / **cooling** / **basic level** / **additional level** (HCBA) determines the value to be sent as off-command with the 8 bit object <Correcting variable – …> for the continuous 2-point control.

Correcting variable Off HCBA	***0 %*** / *5 %* / *10 %* / *15 %* / *20 %* / *25 %* / *30 %*

Parameter **Correcting variable On heating** / **cooling** / **basic stage** / **additional level** (HCBA) determines with the continuous 2-point control which value is sent as On-command with the 8 bit object <Correcting variable – …>.

Correcting variable On HCBA	*70 %* / *75 %* / *80 %* / *85 %* / *90 %* / *95 %* / ***100 %***

2.7.7 Parameter page "Manual set point setting"

On the parameter page "Manual set point setting", it can be determined if and in which limits the set point values can be adjusted at the device.

Parameter **Set point values can be adjusted during running time** determines whether or not the set point values can be adjusted during the running time.

Set point values can be adjusted during running time	*Yes* *No*
Yes	The set point values can be adjusted within the parameterised limits during the running time.
No	The set point values cannot be adjusted during the running time at the device. All follow-up parameters are hidden.

Parameter **Maximum increase of the set point value in heating mode** / **cooling mode** determines the maximum upward set point value adjustment.

Maximum increase of the set point value in heating mode **Maximum increase of the set point value in cooling mode**	*0 K* / *1 K* / *2 K* / ***3 K*** / *4 K* / *5 K*

Parameter **Maximum reduction of the set point value in heating mode** / **cooling mode** determines the maximum downward set point value adjustment.

Maximum reduction of the set point value in heating mode **Maximum reduction of the set point value in cooling mode**	*0 K* / *1 K* / *2 K* / ***3 K*** / *4 K* / *5 K*

Parameter **Behaviour when receiving a base set point value** determines the behaviour when receiving the base set point value via the object 54 <Room temperature base set point value – Specification>.

Behaviour when receiving a base set point value	*Reset manual set point value specification* *Manual set point value specification unchanged*
Reset manual set point value specification	The manually set set point value adjustment is reset to 0.
Manual set point value specification unchanged	The manually set set point value adjustment is maintained.

2.7.8 Parameter page "Window monitoring"

With the active window monitoring, one input object <Window monitoring – Input x> exists for each monitored window (1–10). The value of the output object 10 <Window monitoring – Output> is determined from disjunction (OR) of the values of the input objects (1=window open / 0=window closed), i.e. it takes the value 1 if the first input object receives the value 1 while it takes the value 0 when all input objects have the value 0 again.

Typically, the output object is linked to the frost protection object so that the room thermostat immediately changes to frost protection. Thus, radiators under the respective window can be temporarily deactivated, e.g. during ventilation, which in turn leads to savings in terms of energy and heating costs.

Since this is not very reasonable in the case of a brief intensive airing of the room (many heating systems, in particular underfloor heating, are extremely inert, or valves are unnecessarily driven during a short opening of the window which in turn leads to unnecessary wear), a period of time may be specified additionally (parameter **Delay until frost protection**) which delays the sending of a 1 of the output object. If the output object returns to the value 0 (all windows are closed), this is sent immediately.

Parameter **Window monitoring** determines whether or not the window contacts are monitored.

Window monitoring	*Not active* *Active*
Not active	The window monitoring is switched off.
Active	The window monitoring is active. The parameter **Number of windows to be monitored** and **Delay until frost protection** are visible.

Parameter **Number of windows to be monitored** determines the number of window contacts to be monitored.

Number of windows to be monitored	*1..10* (***1***)

Parameter **Delay until frost protection** determines the period of time until object 10 <Window monitoring – Output> sends a 1, in minutes.

Delay until frost protection	*0..255* (***15***)

2.8 Parameters fan (fan coil)

2.8.1 Parameter page "Fan (fan coil)"

Parameter **Changing of fan operating mode at the device** determines whether or not the fan level of the fan coil can be changed with the room thermostat (→ *chapter 3.9*).

Changing of fan operating mode at the device	*Enabled* ***Disabled***
Enabled	The user has the possibility to control the fan levels of the fan coil via the room thermostat.
Disabled	No fan coil is controlled. All follow-up parameters are hidden.

Parameter **Number of fan levels** determines the number of fan levels which can be selected.

Number of fan levels	*1..9* (***6***)

Parameter **Automatic switchover to automatic mode** determines whether or not the room thermostat switches the fan back into the automatic operating mode.

Automatic switchover to automatic mode	*Enabled* ***Disabled***
Enabled	After each activation of the manual fan control, the room thermostat switches the fan back into the automatic mode after a certain period of time. The parameter **Time** is visible.
Disabled	The room thermostat does not automatically switch the fan back into the automatic operating mode.

Parameter **Time** determines the number of minutes after expiration of which the manual fan control is deactivated.

Time	*1..3600* (***60***)

Parameter **"Manual Off" fan operating mode at the device** determines whether or not the fan can be deactivated manually.

"Manual Off" fan operating mode at the device	*Enabled* ***Disabled***
Enabled	The user has the possibility to manually deactivate the fan. The parameter page "Level 0 (Man.Off) fan operating mode" is visible.
Disabled	The user cannot deactivate the fan manually.

Parameter **"Switch manual/auto" object type** determines the data format for object 72 <Fan operating mode – Switch "manual/auto">.

"Switch manual/auto" object type	***1 bit*** *8 bit 0..100%* *8 bit 0..255*
1 bit	The data format is set to 1 bit.
8 bit 0..100%	The data format is set to 8 bit value specification in percent.
8 bit 0..255	The data format is set to 8 bit value specification 0..255.

Parameter **"Fan levels" object type** determines the data format for object 73 <Fan operating mode – "Level" fan operating mode> fan operating mode.

"Fan levels" object type	*1 bit* *8 bit 0..100%* ***8 bit 0..255***
1 bit	The data format is set to 1 bit.
8 bit 0..100%	The data format is set to 8 bit value specification in percent. The fan level is stipulated in a percentage value of the maximum fan performance.
8 bit 0..255	The data format is set to 8 bit value specification 0..255. The fan level is selected directly as value.

Parameter **"Frost/heat protection" object type** determines the data format for object 74 <Fan operating mode – Frost/heat protection>

"Frost/heat protection" object type	***1 bit*** *8 bit 0..100%* *8 bit 0..255*
1 bit	The data format is set to 1 bit.
8 bit 0..100%	The data format is set to 8 bit value specification in percent.
8 bit 0..255	The data format is set to 8 bit value specification 0..255.

Parameter **Waiting time for fan coil response** determines the period of time during which the system waits for the response of the Fan coil actuator, in seconds. If the user has selected a fan level at the room thermostat, it is sent to the bus. Subsequently, the operation for the user is disabled until the actuator reports the set fan level or until the set period of time has expired. If no response is received within the set period of time, the fan is reset to its previous state. Ensure that the response time of the Fan coil actuator (depending on the bus load in the building) is shorter than the period of time set here.

Waiting time for fan coil response	*5..255* (***20***)

2.8.2 Parameter page "Automatic fan operating mode"

On the parameter page "Automatic fan operating mode", the telegrams are determined which are sent to the fan coil actuator when the manual fan control is deactivated (switching to automatic mode).

Parameter **On "Switch manual/auto" object** determines which telegram is sent to object 72 <Fan operating mode – Switch "manual/auto"> in order to activate the automatic fan control.

On "Switch manual/auto" object	*Do not send telegram* ***Send telegram***
Do not send telegram	No telegram is sent to the object.
Send telegram	The telegram set under the **Value** parameter is sent to the bus.

Parameter **On "Fan levels" object** determines whether or not the fan level is to be sent to object 73 <Fan operating mode – "Level" fan operating mode>.

On "Fan levels" object	***Do not send telegram*** *Send telegram*
Do not send telegram	No telegram is sent to the object.
Send telegram	The telegram set under the **Value** parameter is sent to the bus.

Parameter **On "Frost/heat protection" object** determines whether or not the frost/heat protection is to be activated via object 74 <Fan operating mode – Frost/heat protection>.

On "Frost/heat protection" object	*Do not send telegram* ***Send telegram***
Do not send telegram	No telegram is sent to the object.
Send telegram	The telegram set under the **Value** parameter is sent to the bus.

Parameter **Value** determines the value to be sent with the 1 bit telegram.

Value	*Send ON* *Send OFF*

Parameter **Send value in %** determines the value to be sent with the 8 bit telegram.

Send value in %	*0..100*

Parameter **Send value 0..255** determines the value to be sent with the 8 bit telegram.

Send value 0..255	*0..255*

2.8.3 Parameter page "Level x fan operating mode"

On the parameter page "Level x fan operating mode", the telegrams are determined which are sent to the fan coil actuator when the corresponding fan level x is selected at the room thermostat.

Parameter **On "Switch manual/auto" object** determines whether or not the manual fan control is to be simultaneously activated via the object 72 <Fan operating mode – Switch "manual/auto">.

On "Switch manual/auto" object	***Do not send telegram*** *Send telegram*
Do not send telegram	No telegram is sent to the object.
Send telegram	The telegram set under the **Value** parameter is sent to the bus.

Parameter **On "Fan levels" object** determines the fan level which is to be sent via object 73 <Fan operating mode – "Level" fan operating mode>

On "Fan levels" object	***Do not send telegram*** *Send telegram*
Do not send telegram	No telegram is sent to the object.
Send telegram	The telegram set under the **Value** parameter is sent to the bus.

Parameter **On "Frost/heat protection" object** determines whether or not the frost/heat protection is to be activated via object 74 <Fan operating mode – Frost/heat protection>.

On "Frost/heat protection" object	*Do not send telegram* ***Send telegram***
Do not send telegram	No telegram is sent to the object.
Send telegram	The telegram set under the **Value** parameter is sent to the bus.

Parameter **Value** determines the value to be sent with the 1 bit telegram.

Value	*Send ON* *Send OFF*

Parameter **Send value in %** determines the value to be sent with the 8 bit telegram.

Send value in %	*0..100*

Parameter **Send value 0..255** determines the value to be sent with the 8 bit telegram.

Send value 0..255	*0..255*

2.8.4 Parameter page "Level 0 (Man.Off) fan operating mode"

On the parameter page "Level 0 (Man.Off) fan operating mode", the telegrams are stipulated with which the fan (and, normally, also the valves) are switched off manually.

Parameter **On "Switch manual/auto" object** determines whether or not the manual fan control is to be simultaneously activated via the object 72 <Fan operating mode – Switch "manual/auto">.

On "Switch manual/auto" object	*Do not send telegram* *Send telegram*
Do not send telegram	No telegram is sent to the object.
Send telegram	The telegram set under the **Value** parameter is sent to the bus.

Parameter **On "Fan levels" object** determines the fan level which deactivates the fan.

On "Fan levels" object	*Do not send telegram* *Send telegram*
Do not send telegram	No telegram is sent to the object.
Send telegram	The telegram set under the **Value** parameter is sent to the bus.

Parameter **On "Frost/heat protection" object** determines whether or not the frost/heat protection is to be activated via object 74 <Fan operating mode – Frost/heat protection>.

On "Frost/heat protection" object	*Do not send telegram* ***Send telegram***
Do not send telegram	No telegram is sent to the object.
Send telegram	The telegram set under the **Value** parameter is sent to the bus.

Parameter **Value** determines the value to be sent with the 1 bit telegram.

Value	*Send ON* *Send OFF*

Parameter **Send value in %** determines the value to be sent with the 8 bit telegram.

Send value in %	*0..100*

Parameter **Send value 0..255** determines the value to be sent with the 8 bit telegram.

Send value 0..255	*0..255*

3 Functional description

3.1 Behaviour after ETS download or bus voltage return

After the application has been downloaded to the device by means of the ETS, the device will restart. After a few seconds, the device will be ready for operation.

If all of the LEDs on the push-button are flashing red, this means that the download could not be carried out properly or that the ETS application is not compatible with the hardware.

Procedure:

1. Shortly disconnect the device from the KNX bus voltage
2. Check the application compatibility
3. Check the physical address
4. Download the application again

Attention:

> KNX devices with the additional designation **RGB** can only be programmed using the corresponding application with the additional designation RGB.

> Older applications (without the additional designation RGB) cannot be loaded to the present hardware with the additional designation **RGB**. Feller shall not assume any liability or consequential costs for projecting errors.

After an interruption of the bus voltage, the device will start automatically after the voltage has returned. The settings made during parameterisation will remain unchanged.

Note: Depending on the settings on the parameter pages "General disabling" and "Disable push-buttons", it may occur that telegrams are sent to the bus after the restart.

3.2 RTH push-button

3.2.1 Operating concept

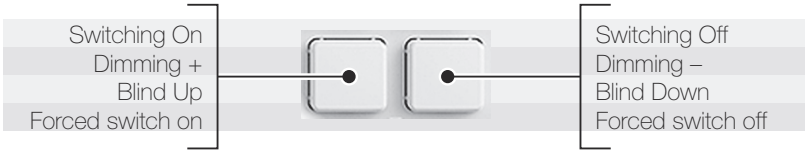
Thanks to a flexible operating concept, the KNX RTH push-button RGB can be used in different ways. For example, the two push-buttons can be used to operate the room thermostat **or** for controlling any actuators. The functions that the push-buttons have are defined at the beginning of parameterisation on the parameter page "Configuration of push-buttons" (→ *chapter 2.3.1*).

Room thermostat

The room thermostat settings can be changed. The arrangement of the + (Plus) and – (Minus) buttons are user selectable. No consumers are controlled.

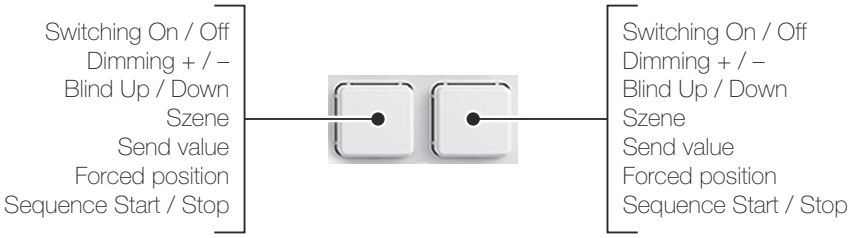
Actuator: Two-button operation

The same consumer is always actuated, but the function depends on whether the left push-button or the right push-button is pressed. The room thermostat cannot be operated, the LC display is for display only.



Actuators: Single-button operation

The two push-buttons are independent of each other, each controls a different consumer. The room thermostat cannot be operated, the LC display is for display only.

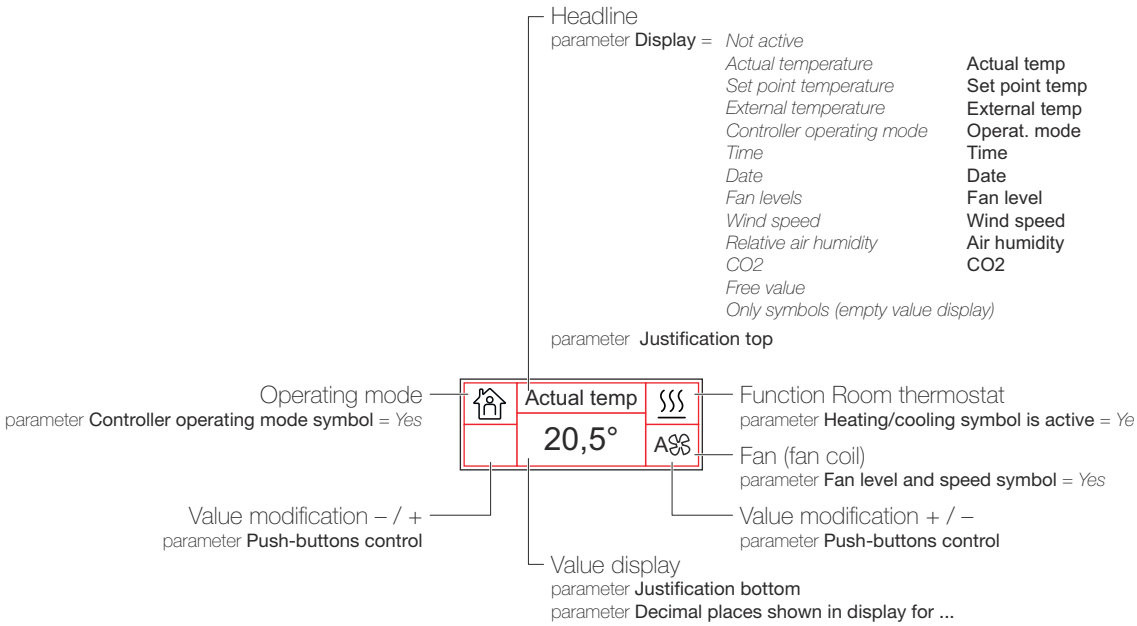


3.2.2 LC display (liquid crystal display)

The KNX RTH push-button RGB has a transreflective LC display that provides optimum legibility, depending on the light conditions. This means that the displays can be read in direct sunlight and, in conjunction with the LED backlighting, even in darkness.

Display

The LC display is divided into 6 areas. It can be adapted to the needs of the user on the parameter page "Configuration of display" (→ *chapter 2.6.1*).



Operating mode

The display uses black text on a lighted background or a lighted font on a black background (parameter **Display operating mode**). It is possible to switch operating modes via the bus using object 75 <Display – Switch operating mode>.

parameter **Display operating mode** = *Lighted background / black font*

parameter **Display operating mode** = *Black background / lighted font*

Lighting

The lighting of the display can optionally be switched always on or off or actuated using separate communication objects; the colour can be determined in the ETS. For the KNX RTH push-button RGB, user colours can be individually mixed on the parameter page "LED colours" (→ *chapter 2.3.5*). The colour can optionally be changed via the bus.

Operation

When configured accordingly, room thermostat functions are controlled using the two push-buttons (parameter **Function of buttons** = *Room thermostat*). Press the RTH push-button once to switch to setting mode. Press both push-buttons simultaneously to toggle between the parameterised displays.

Notes:

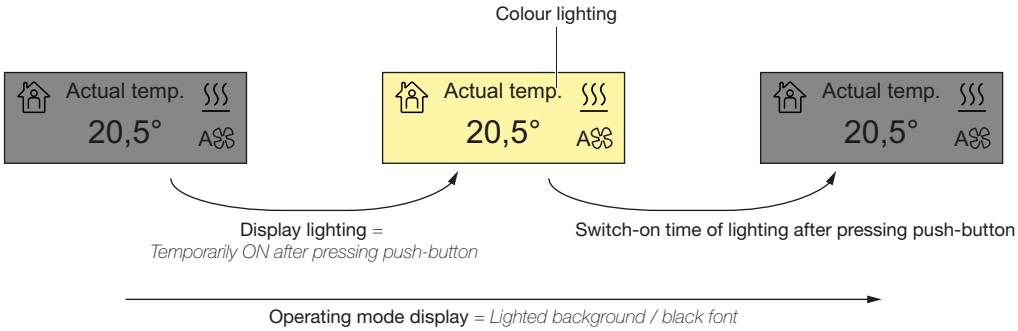
> Apart from the set point temperature, settings can only be modified if they are displayed.

> The brightness of the lighting can only be modified by the user when the actual temperature is displayed.

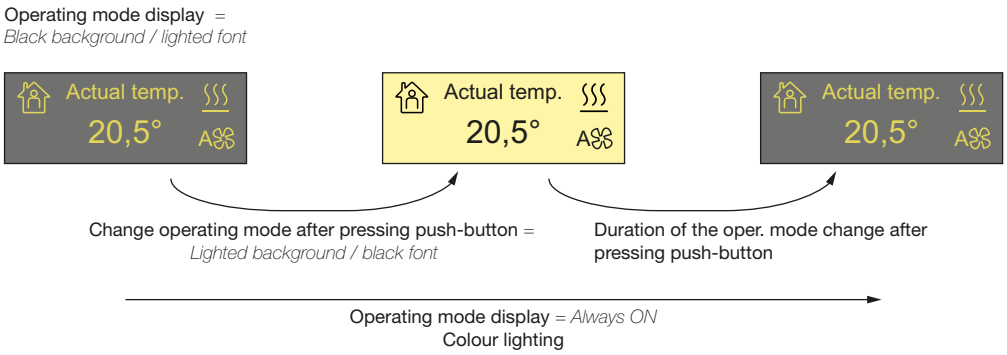
Changes after pressing a push-button

To visually indicate to the user that the device has switched to the setting mode

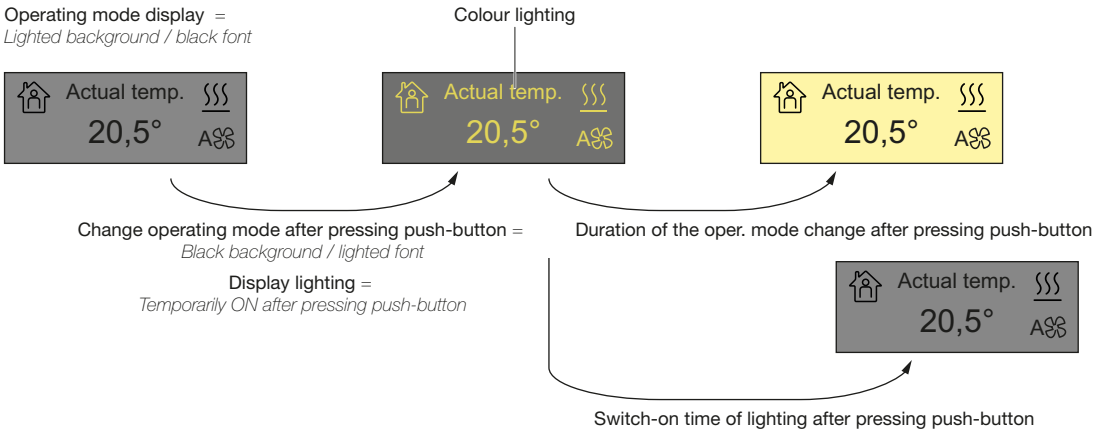
- the lighting can be switched on for a certain amount of time



- the operating mode can be changed over for a certain time

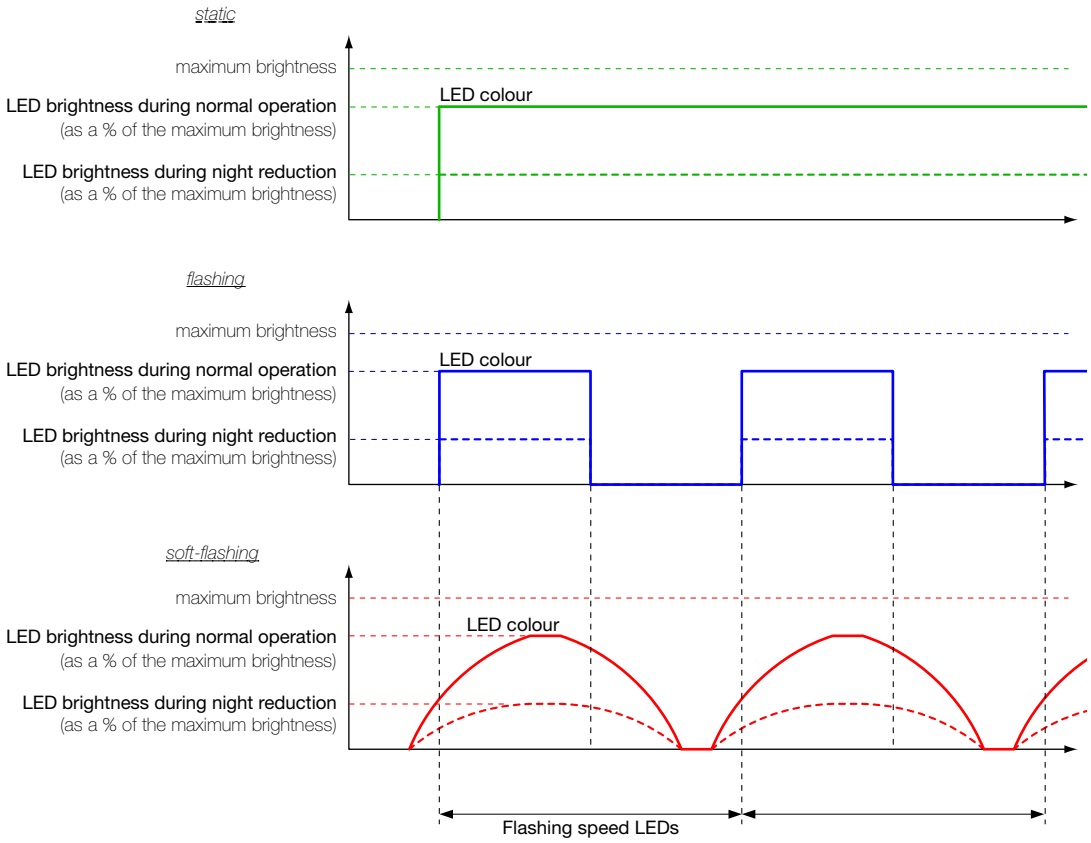


- after a push-button is pressed



3.3 LEDs

The KNX push-buttons RGB are equipped with LEDs, which each LED can be individually configured. They can optionally be activated or deactivated (orientation light), serve as a status display or be used for feedback purposes (LED function). A control via separate communication objects is also possible. The LEDs are able to display a static, flashing or soft-flashing status (display mode). The soft-flashing status can be used as subtle source of information since the LEDs appear more vivid due to the increasing and decreasing dimming brightness.



An individual colour can be set for each LED in the ETS. Optionally, the function of the LED can be overridden via the bus thus enabling a change in the colour and the display mode of individual LEDs depending on priority. For the KNX push-button RGB, two user colours can be individually mixed on the parameter page "LED colours". This enables an optimal adjustment of the LEDs to both the colours of the Feller design parts as well as to the environment (see also *chapter 3.6*).

The brightness during normal operation and the flashing speed of all LEDs is globally defined on the parameter page "LED brightness and flashing speed". This ensures a unified visual appearance and a synchronised flashing of the LEDs [1]. The brightness can optionally be adjusted during operation via a 1 bit communication object. This adjustment can be used to reduce the brightness during night-time, for example. If you wish to adjust the brightness via the object, the parameter **Night reduction LEDs function** needs to be set. In this case, the object 25 <Night reduction LEDs & display – Decrease brightness> will be visible in the ETS.

[1] The increasing and decreasing dimming brightness of the soft-flashing LED starts at approx. 10% of the flashing speed prior to switch on/off of the flashing LED. When reaching the upper or lower peak, this state is maintained for approx. 10% of the flashing speed.

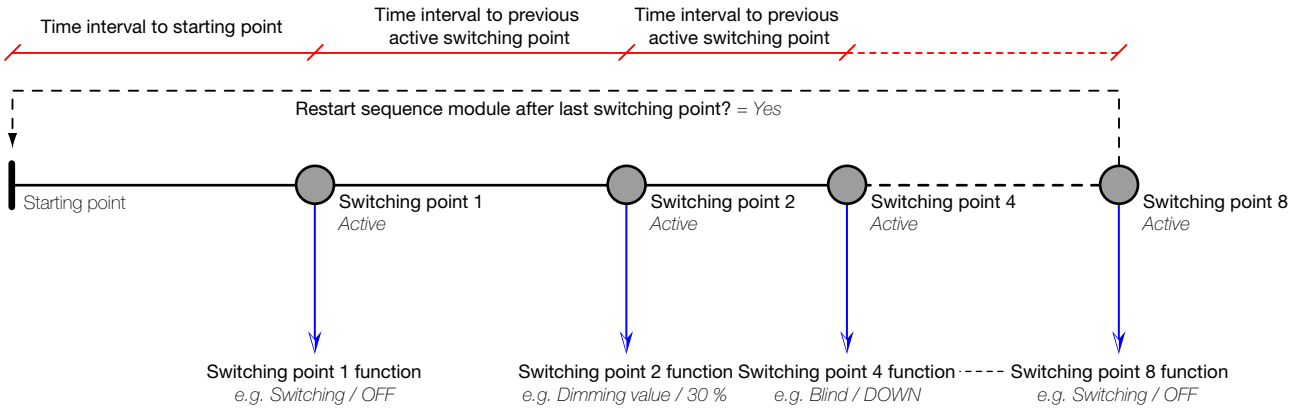
3.4 Sequence module

A possible application for the sequence module in residential buildings is, for example, a time-delayed «Central OFF». Compared to the pure scene solution, this provides the advantage of «organically shutting down» the house and helps to avoid current spikes. At first, the light in the hallway is reduced to 30%, then, the blinds are closed and the lights are turned off on all floors on a time-delayed basis, before the lights in the hallway are also turned off.

Since the time interval to the previous switching point can be up to an hour long, the restart can also be activated by means of a simple presence simulation.

In functional buildings, the sequence module can also be used for presentations, for example. At first, the beamer is turned on, then, the blinds are closed after 30 seconds, and after another 15 seconds, the lighting is dimmed down.

A sequence of up to 8 parametrisable switching points can be defined on the parameter page "Sequence module". There is an output object for each switching point. Each switching point is triggered with a time-delay following the starting point or previous switching point.



The sequence is started by pressing a push-button, provided that this push-button has been parameterised accordingly (→ *chapter 2.3.3*), or by writing ON into the object 87 <Sequence module – Recall sequence>.

The sequence is stopped by pressing a push-button for a longer time, provided that this push-button has been parameterised accordingly (→ *chapter 2.3.3*), or by writing OFF into the object 87 <Sequence module – Recall sequence>.

While the sequence is processed, the object 88 <Sequence module – Status> is set to ON. At the end, it is set back to OFF.

If the sequence is started again by a press of a push-button or by writing ON into the object 87 <Sequence module – Recall sequence> while it is being processed, the sequence will restart from the beginning (retrigger).

3.5 Scene module

With a scene, a group of actuators can be set to a desired state simultaneously by a press of a push-button. This way, the desired ambience can be achieved by pressing a push-button (e.g. meal, leaving the house, blinds down, lighting off, set heating to standby operation etc.). This scene functionality often provides advantages in functional buildings as well. A museum or a gallery could, for example, showcase the exhibition objects in the right light by a press of a push-button.

There are two concepts for the KNX push-button RGB with regard to triggering or saving scenes:

Decentralised scene saving in the actuator (8-bit scene)

The scene values are remotely saved in the scene storage of the actuator. At the press of a push-button, a preset scene number (1..64) is sent to the bus via a separate communication object. This way, the scene is called up in the actuator or – when using the saving function – also saved. The KNX push-button RGB and the actuators communicate with each other via an 8-bit telegram.

For the 8-bit scene, only one telegram is sent in order to control all corresponding actuators simultaneously.

For every push-button, it can be set whether a scene is only to be recalled or if it is to be recalled and saved using the parameter **Scene function** (→ *chapter 2.3.3*). When saving the scene, care must be taken in order to ensure that all involved devices are in the right state. A scene cannot be deleted by the user.

Local scene saving in the push-button (conventional scene)

The scene values are locally saved in the KNX push-button RGB. At the press of the push-button, the corresponding scene value is sent to all involved actuators via the bus. A snapshot of the default values and/or actuator states can be saved as scene value. The scenes are permanently stored and remain available even after a voltage interruption.

Up to 15 group addresses can be assigned to the scene function. A maximum of 8 different scenes is possible. The same actuators and/or group addresses participate in each scene.

For the conventional scene, up to 15 telegrams are serially sent to the bus (delay time between the sending of the individual telegrams can be set using the parameter **Transmission delay between scene telegrams**). This causes a «high» bus load and may result in visible delays when scenes are called up. (When using the 8-bit scene, this mechanism does not occur.)

The parameter **Scene mode for the user during the operation** can be used to set whether scenes can only be recalled or if they can be recalled and saved (all or selective) (→ *chapter 2.5.1*).

The link of the KNX push-button RGB with the actuators is established via the scene objects. They must be linked to the same ETS group address that is used to link the local push-button and display objects to the actuator.

In order to properly configure the KNX push-button RGB, please also note the following points:

- Enter the correct object type (1 bit for switching, 8 bit for dimming brightness or blind position) in the settings on the parameter page "Data type scene value 1…10/1…15" (→ *chapter 2.5.2*).
- In the settings on the parameter page "Scene x [value 1…10/1…15]" (→ *chapter 2.5.3*), define the parameters **Presetting scene value 1** to **Presetting scene value 10/15**.
  Note: These parameters are only valid until a new scene is saved. If the device is programmed with the ETS again afterwards, all scenes are reset to the values saved in the ETS (presetting).
- The transfer (Ü) and/or read (L) flag must be set for the actuator for 1-byte scene groups. Both flags, however, may only be set for one actuator per scene group if several actuators are connected to a scene group.
- In the parameter settings **Scene mode for the user during the operation** = *Recall scene and save all* on the parameter page "Scene module" (→ *chapter 2.5.1*), the read flag (L) must be set for the 1-byte object of the actuator and the current brightness/position of the actuator must be legible.
- In the parameter settings **Scene mode for the user during the operation** = *Recall scene and save selectively* on the parameter page "Scene module" (→ *chapter 2.5.1*), the transfer (Ü) flag must be set for the 1-byte object of the actuator and the current brightness/position of the actuator must be legible.

Notes:

> Depending on the programming via the ETS, a scene may also be called up by other push-buttons (so called extensions) by means of an ON telegram.

> The «Program scene» function can be disabled via the ETS parameter settings so that a scene may only be recalled (parameter **Scene mode for the user during the operation** = *Only recall scene*). The scene can then not be programmed by the end user.

> Not all of the actuators are scene-capable. Please note the relevant information provided in the product specifications of the manufacturers.

3.6 RGB colour theory

Source: Colour theory and colour design (www.ipsi.fraunhofer.de/~crueger/farbe/)

Man perceives light on a certain wavelength ranging from 380 nm (nanometer) to 750 nm as colours. There are three different types of colour-sensitive photoreceptors located in the retina of the human eye, also referred to as cones. They are sensitive for three different wavelength ranges of light, namely long-wave, medium-wave and short-wave light. The cones collect the rays of their wavelength that incidents in the human eye, and direct them to the brain, where the real colour perception evolves. We see long-wave light as red, medium-wave light as green and short-wave light as blue.

Primary colours

Combinations of 2 or 3 different wavelengths in equal proportions and full intensity result in overall 8 extreme colour perceptions, also referred to as primary colours.
The 8 primary colours are red, green, blue, cyan, magenta, yellow, white and black.
Black and white are the achromatic primary colours, the 6 others are chromatic primary colours.

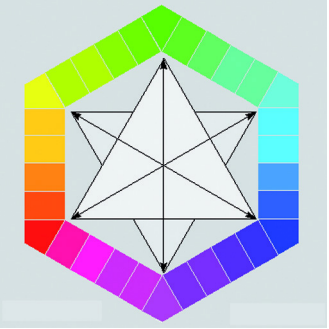
The additive colour mixing (RGB)

The RGB colour range is used for self-luminous (colour-displaying) systems that are subject to the principle of additive colour mixing, also referred to as light mixing. According to the three cone types of the human retina, it is based on the three primary colours red, green and blue. Brighter colour shades can be created by mixing. Yellow is created by mixing red and green, mixing green and blue results in cyan and blue mixed with red in magenta. If all three colours come together in full intensity and in equal proportions, they will create the colour white.

The LEDs of the KNX push-buttons RGB as well as colour television and the colour display of a computer are working based on this principle. In graphics software, it is known as the RGB model.

Colour hexagon

The colour hexagon consists of a triangle comprising the elementary colours red, green and blue and a triangle comprising the primary colours magenta, yellow and cyan.

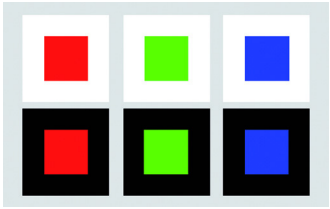


The colours are arranged in such a way that their mixed colour shades are located between the three elementary colours. Therefore, yellow is located between red and green, cyan between green and blue and magenta between blue and red. This way, two colours are facing each other that will complement each other and create the colour white when using the additive colour mixing. Such colour pairs are referred to as complementary colours.

The 6 primary colours are positioned in the corners of the hexagon, the mixed colour shades created from two neighbouring primary colours are located on the legs in between. The colour hexagon can be divided into two halves: one half contains cold colour shades while the other one contains warm ones. The warm colour shades range from green, yellow and red to magenta. The cold colour shades range from magenta, blue and cyan to green. Green and magenta are placed on the intersection points between warm and cold and are considered neutral.

Simultaneous contrast

When informing yourself about colour design, you will soon discover that colours change their character depending on their environment. These are the effects of the simultaneous contract.



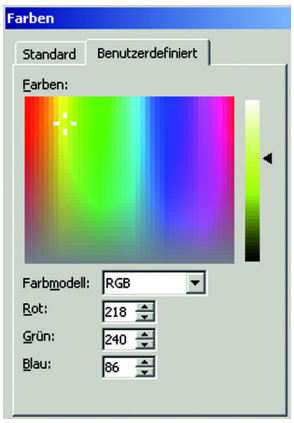
Example:
One and the same colour appears brighter in front of a dark background and darker in front of a bright background. A bright background will put a colour into the foreground and a dark background will decrease its effect. Achromatic environments let chromatic colours shine more brightly, which particularly applies to black.

This effect also occurs if the LED colour is combined with an Feller cover set.

The effect of the simultaneous contrast is caused by the fact that the human eye is not made to reproduce colours as true to the physical values that they are based on as possible but is instead aiming at pointing out differences. This means that changing a colour shade in a colourful design – by adding a new colour shade or removing a colour shade – can fundamentally change the character of the design.

LED colours of the KNX push-buttons RGB

A user colour is defined by the numeric portion (0 … 255) of the colours red, green and blue. The colour value 255 represents the full colour shade of a primary colour, while the colour value 0 indicates that no portions of this primary colour are included.



Auxiliary means such as colour mixers that are used in almost every computer programme can be used to define colours.

Numerous colour tables including colour patterns and their corresponding codes are available on the internet as well, e.g. www.ipsi.fraunhofer.de/~crueger/farbe/farb-must.html or www.farb-tabelle.de/de/farbtabelle.htm.

Please note that the colours mixed on your screen can only serve as general guide and **that the perception on site significantly depends on the combination of background – Feller design parts – lighting etc.**.

3.7 Room thermostat

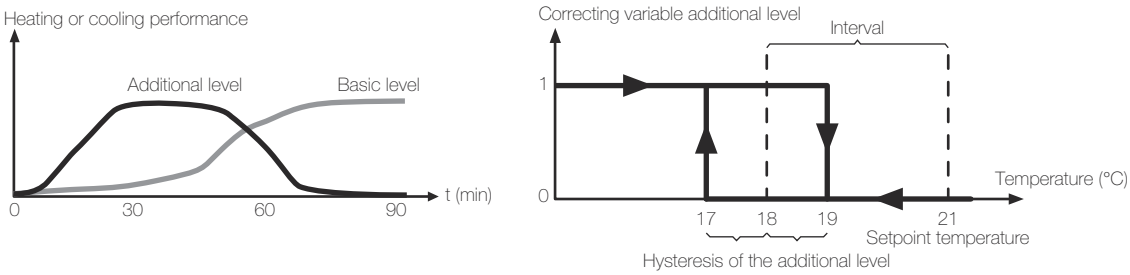
The room thermostat (RTH) of the KNX RTH push-button RGB may be used for single-room temperature control. Depending on the function, the operating mode, the current set point value and the room temperature, the correcting variables for heating or cooling control and for fan control (FanCoil) are sent to the KNX bus. They are analysed by the controlled KNX actuators or directly by means of bus-compatible actuators and converted into physical parameters for room climate control.

The room thermostat is a stand-alone functional device of the KNX RTH push-button RGB and is equipped with its own parameter and object area in the ETS.

3.7.1 Function

The room thermostat may be used for controlling heating systems (*Heating* function) or cooling systems (*Cooling* function). Mixed operation is also possible (*Heating and cooling* function); here the room thermostat may switch in an automatic or controlled manner, via the object 57 <Heating/cooling – Switch operating mode>.

In order to shorten the heating phase with inert heating systems (such as floor heating), often a second, less inert heating system is used, which achieves a faster heating effect during the long starting period of the mains system (basic level) (*2-stage heating* function). This is the same for cooling systems (*2-stage cooling* function).



The additional level controlled via the 2-point control (→ *chapter 3.8.3*) remains activated until it reaches the specified interval to the set point temperature (parameter **Interval between basic level and additional level** → *chapter 2.7.2*) plus hysteresis (parameter **Hysteresis of the additional level**). Then, the additional level is deactivated and only the basic level remains switched on.
The additional level (e.g. heating) is only switched on again when the actual value is lower than the set point value (e.g. 21 °C) minus interval (e.g. 2 °C) minus hysteresis of the additional level (e.g. 1 °C).

3.7.2 Operating modes

The room thermostat knows 5 operating modes all of which have been allocated their own set point value for heating and cooling. The status' are shown at the push-button display by means of symbols.

- Comfort operation
  It is used for controlling the room temperature when the room is in use.
  The comfort operation is activated when (e.g. a pirios presence detector) signalises a presence via the <Controller operating mode> object or by means of actuating the operating mode key at the device.
- Standby operation
  It is used for a minor decreasing/increasing of the room temperature when heating or cooling if the room is preliminary out of use. A short heating-up or cooling-down phase is the result of a minor reduction or increase in the room temperature.
- Night operation
  It is used for a greated reduction or increase in the room temperature during the night or at the weekend. If the night operation is terminated, any possibly active comfort extension is terminated, as well.
- Frost/heat protection
  It is used for deactivating the heating or cooling until a critical temperature is reached (freezing or over-heating of the room). If the frost/heat protection is used, the previous condition is reinstated.
- Dewpoint operation
  It is used for the unconditional deactivation of the heating or cooling, e.g. in the case of condensation at the cooling system. The dewpoint operation is activated via object 51 <Controller operating mode – Dew-point>.
  All symbols for the operating mode are deactivated. If object 51 <Controller operating mode – Dewpoint> is deleted, the previous condition is reinstated.

- Comfort extension
  The additional operating mode comfort extension has an identical effect than the comfort operation. However, after a certain preset period of time (parameter **Duration of the comfort extension**) it is automatically left again. It is used for the preliminary suppression of the night operation, e.g. if the room is to be used for a longer period of time at night.
  The comfort extension is activated if the operating mode key is activated during night operation and, on the parameter page "Configuration of display" in the section **Operating modes selectable at the device** the parameter **Night operation** is set to *No*.
  The comfort extension is terminated when the parameterised duration has expired, the night operation is activated or when the night operation is left by actuating the operating mode key at the device.

  Note: If the comfort extension is left early (if the comfort extension time has not yet expired), the timer is reset.

Switching between operating modes

It can be switched in different ways between these operating modes:

- by actuating the operating mode key at the device, if on the parameter page "Configuration of display" in the section **Operating modes selectable at the device** the respective operating type is enabled.
- via 1 bit individual objects 47–51 <controller operating mode – ...>
  (when **Switch operating mode via** = *individual objects (1 bit)*)

Comfort	Night	Frost/heat protection	Holidays	Dewpoint	resulting operating mode
1	x	0	0	0	Comfort operation
0	0	0	0	0	Standby operation
0	1	0	0	0	Night operation
x	x	1	0	0	Frost/heat protection
x	x	x	1	0	Frost/heat protection
x	x	x	x	1	Dewpoint operation

- via 8 bit value object 47 <Controller operating mode - All operating modes> and
  1 bit individual object 51 <Controller operating mode - Dewpoint>
  (when **Switch operating mode via** = *8 bit object*)

object value <Controller operating mode – All operating modes>	– Dewpoint>	resulting operating mode
01	0	Comfort operation
02	0	Standby operation
03	0	Night operation
04	0	Frost/heat protection
x	1	Dewpoint operation

x = any value

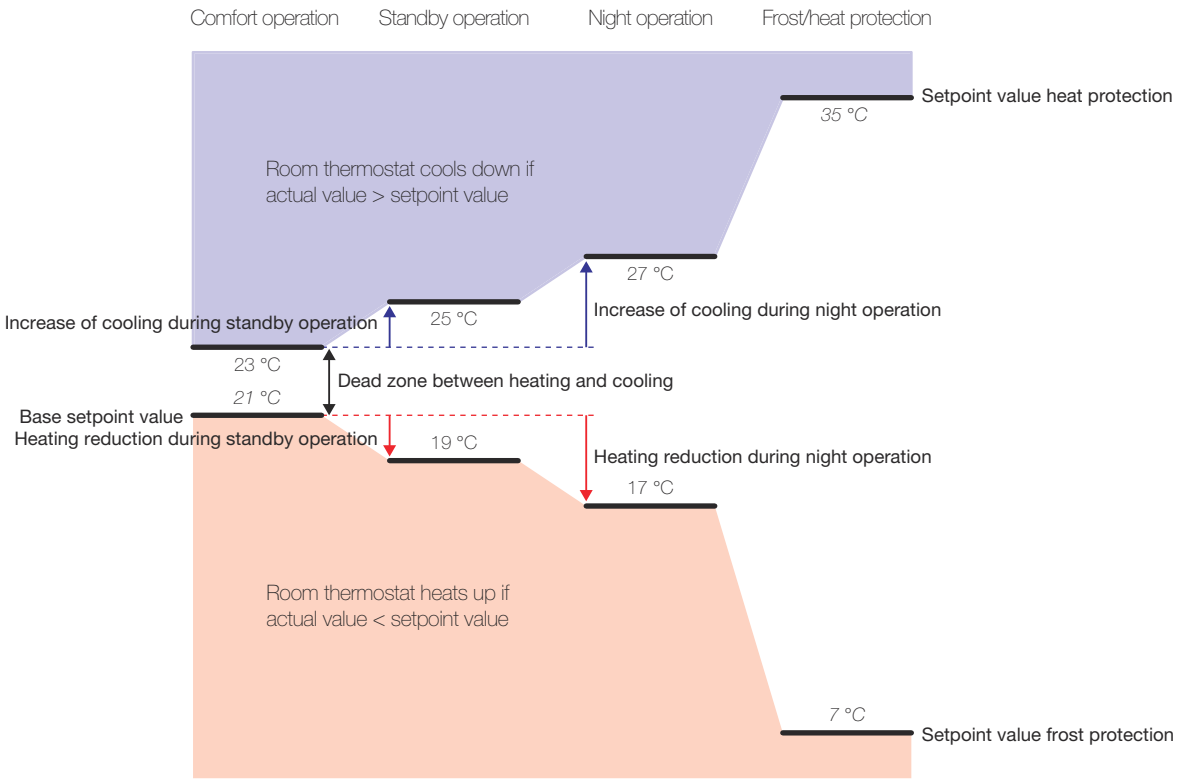
3.7.3 Set point values, set point value adjustment and dead zone

For every operating mode, a set point value is determined at the parameter page "Setpoing values". When changing the operating mode, the respective set point value is used for the further room temperature control. The set point values of all operating modes (excluding frost/heat protection) may be adjusted manually via the keys of the room thermostat (set point value adjustment) within the limits to be set (parameter page "Manual set point setting").
For mixed operation (*Heating and cooling* function), a continuous changing of the room thermostat between heating and cooling is presented by means of the parameterised dead zone.

Calculation of the set point values

Operating mode	Heating set point value =	Cooling set point value =
Comfort operation and Comfort extension	Base set point value + Set point value adjustment	Base set point value + Dead zone between heating and cooling*) + Set point value adjustment
Standby operation	Base set point value – Heating reduction during standby operation + Set point value adjustment	Base set point value + Increase of cooling during standby operation + Dead zone between heating and cooling*) + Set point value adjustment
Night operation	Base set point value – Heating reduction during night operation + Set point value adjustment	Base set point value + Increase of cooling during night operation + Dead zone between heating and cooling*) + Set point value adjustment
Frost/heat protection	Set point value frost protection	Set point value heat protection

*) parameterised for mixed operation (*Heating and cooling*) only, otherwise = 0



3.7.4 Room temperature measurement

The room thermostat cyclically measures the room temperature (actual value) and compares it with the prescribed set point value of the active operating mode. From the difference between actual value and set point value, the correcting variable is calculated by means of the set control algorithm (→ *chapter 3.8*).

In order to guarantee a control of room temperature which is always flawless and effective, it is of utmost importance that an accurate actual value is determined. The room thermostat has an integrated room temperature sensor via which the temperature can be measured. Alternatively (e.g. with unsuitable installation locations of the room thermostat), an external temperature connected via bus telegrams may be used for determining the actual value.

When selecting the installation location of the room thermostat, the following aspects should be taken into consideration:

- any installation in combinations, in particular if flush-mounted dimmers are installed, is to be avoided
- do not mount in the proximity of large electrical consumers (avoid heat impact)
- do not install in the proximity of radiators or cooling systems
- keep the room thermostat out of direct sunlight
- an installation at the internal side of an exterior wall may have a negative influence on the temperature measurement
- the room thermostat should be installed within a distance of at least 30 cm away from doors, windows or ventilation systems and should be installed at minimum height of 1.5 m above the floor

Comparison of measuring values after sufficient delay

In some cases, in the course of the room temperature measurement, a comparison of individual temperature values may be required. A comparison is required, for instance, if the temperature measured by the temperature sensor is permanently lower or higher than the actual temperature around the room thermostats. To determine the temperature deviations, the actual room temperature should be determined by means of a reference measurement with a **gauged** temperature measurement device.

With the **Adjustment direction of the room temperature measurement** and **Adjustment value of the room temperature measurement** parameters, the temperature comparison can be parameterised in the area of 0–5 K. The comparison is statically set once only and is the same for all operating modes of the room thermostat.

For the room temperature control, the room thermostat always uses the compared value for calculating the correcting variables. The compared value may be sent out to the bus via the 2 byte object 56 <Room temperature actual value – Control value>.

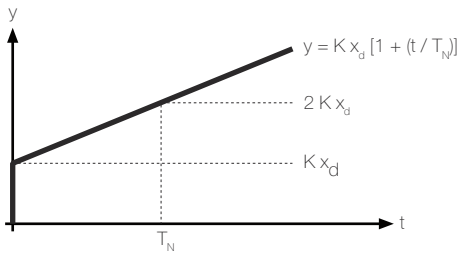
3.8 Control algorithms

In order to enable a comfortable temperature control in a living room or office, a special control algorithm is required, which controls the installed heating or cooling systems. Taking into consideration the specified set point values as well as the actual room temperature, the room thermostat determines correcting variables which control the heating or cooling system. The control system (control loop) consists of a room thermostat, the actuator or the switching actuator (when using electrothermic drives), the actual heating or cooling element (e.g. radiator or cooling ceiling) and the room. This comprises the controlled system.

The room thermostat measures the temperature (actual value) and compares it to the specified set point value. From the difference between actual value and set point value, the correcting variable is calculated by means of the set control algorithm. By means of the correcting variable, valves or fans for heating or cooling systems are controlled by means of which heating or cooling energy in the heat or cold exchangers is rendered to the room. When regularly resetting the correcting variable, the room thermostat is able to compensate for the deviations between the actual and the set point values in the control loop, which are caused by external influences.

3.8.1 PI control

A PI control is an algorithm consisting of a **P**roportional and an **I**ntegral component.



PI control algorithm: Correcting variable $y = K\,x_d\,[1 + (t / T_N)]$

$x_d = x_{set\ point} - x_{actual}$: Control difference

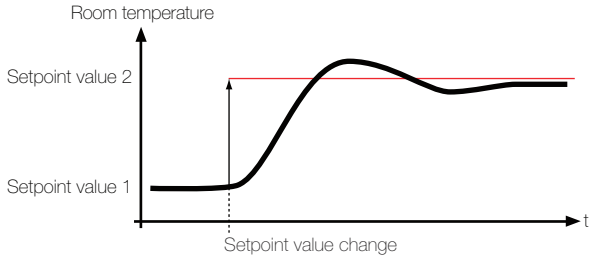
P: **Proportional range** which can be parameterised

K = 1 / P: Reinforcement factor

T_N: **Reset time** which can be parameterised

By deactivating the reset time (= 0) →
P control algorithm: Correcting variable $y = K\,x_d$

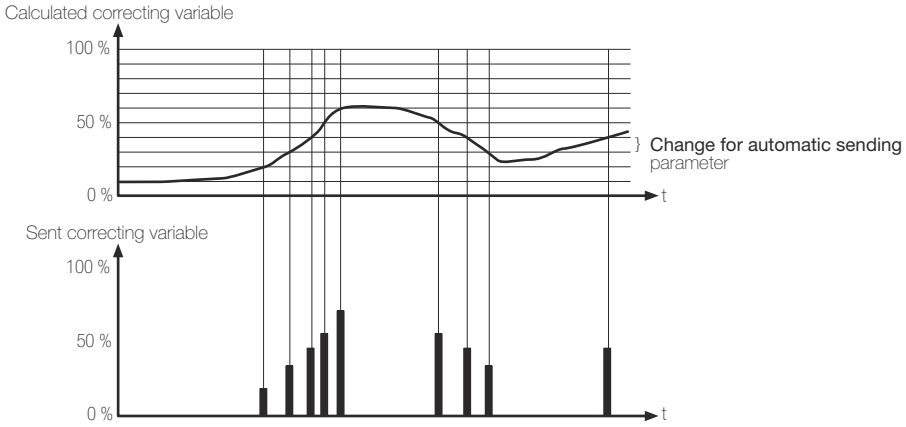
By combining these control characteristics, a correction of the room temperature which is as fast and as accurate as possible can be achieved without any or with only minor control deviations.



dynamic behaviour of the PI algorithm (e.g. during heating)

Constant PI control

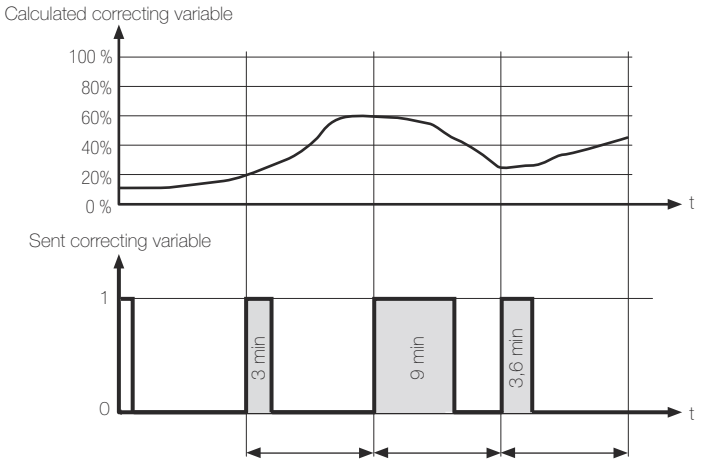
For the steady PI control, the room thermostat cyclically calculates a new constant correcting variable (0–100 %) and sends it to the bus via a 8 bit value object if the calculated correcting variable value has changed by a specified value (**Change for automatic sending** parameter).



Additionally, the current correcting variable may be sent cyclically to the bus. By doing so, it can be ensured that telegrams are received with a cyclical safety monitoring of the correcting variable in the actuator or in the controlled switching actuator within the monitoring time. The time interval which is determined by the **Cycle time for automatic sending** parameter should correspond to the monitoring time in the actuator (preferably, the cycle time in the room thermostat is to be parameterised lower).

Switching PI control

With the switching PI control, which is also known as PWM control, the correcting variable (0–100 %) calculated by the room thermostat is converted into an equivalent pulse width modulated (PMW) correcting variable signal and emitted to the bus via a 1 bit switching object on expiration of the cycle time. For instance, if the room thermostat calculates a correcting variable of 20 %, with a **Cycle time of the switching correcting variable** amounting to *15 min*, then a logical **1** is sent for 3 minutes (20 % of 15 minutes) and, subsequently, a **0** is sent for 12 minutes. Upon expiration of the cycle time, the current correcting variable is reconverted into a new PWM.
It is also by means of this control algorithm that the room temperature is kept at a constant level. Averaged over time, this results in the same behaviour of the control system as with a constant controller.



In most cases, the pulse width modulated correcting variables are used for controlling electrothermal drives. Here, the room thermostat sends the switching correcting variable telegrams to a switching actuator with semiconductor switching elements to which the drives are connected (e.g. heating or room actuator). By setting the cycle time, it is possible to adjust the control to the drives used. The cycle time determines the switching frequency of the pulse width modulated signal and allows for the adjustment to the adjustment cycle times of the used actuators (travel time, which the drive requires for adjusting the valve of the completely closed position up to the completely open position). In addition to the adjustment cycle time, the dead time (period of time during which the actuators do not show any reaction when being activated or deactivated) is to be taken into consideration as well. If several drives are used with different adjustment cycle times, the greater of the times is to be taken into consideration. As a rule, the specifications of the drives' manufacturers are to be taken into consideration.

3.8.2 Adjustment of the PI control

In order to allow for the PI control algorithm to control all common heating or cooling systems efficiently and to ensure that the room temperature regulation functions as quickly as possible without control deviations, an adjustment of the control parameters is required. With a PI control, specific factors, which have a significant influence on control behaviours, may be set for this purpose. For this reason, for the most common heating and cooling systems, the room thermostat may be set to the predefined «experience values» (parameter **Adjustment of the PI control to the heating system / cooling system**). For the heating or cooling mode, the following heating or cooling types can be set:

Heating / cooling system	Proportional area (preset)	Reset time (preset)	recommended PI control	recommended **Cycle time of the switching correcting variable**
Warm water heating	5 K	150 min	constant / switching	– *15 min*
Underfloor heating	5 K	240 min	switching	*15 min / 20 min*
Electrical heating	4 K	100 min	switching	*10 min / 15 min*
FanCoil	4 K	90 min	constant	–
SplitUnit	4 K	90 min	switching	*10 min / 15 min*
Cooling ceiling	5 K	240 min	switching	*15 min / 20 min*

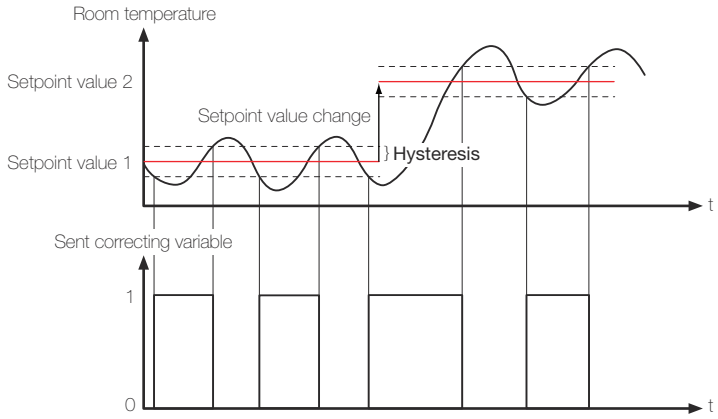
If, by means of selecting a corresponding heating or cooling system, no satisfying control result can be achieved with the specified values, the adjustment may be optimised via control parameters provided that sufficient expert knowledge is given.

3.8.3 2-point control

The 2-point control is the most simple type of control. Here, no correcting variable is calculated. The controller is activated when the room temperature has undercut a certain temperature and it is deactivated as soon as a certein value has been exceeded. When exceeding the hysteresis, the heating is deactivated while, when undercutting the hysteresis, it is activated.

Example: Set point value 20 °C, hysteresis 1 K => heating is activated at 19 °C and deactivated at 21 °C.

The benefit of the extremely simple control is opposed by the disadvantage of a constantly fluctuating room temperature. The temperature overshoots because an actuator needs some time until it is closed completely. Furthermore, even when being switched off, the radiator passes stored heat onto the room.
When activating the heating, the system reacts in a similarly delayed manner. Inert heating and/or cooling systems may not be controlled via a 2-point control since, here, extreme overshoots and thus a significant comfort loss occur.



3.8.4 Application examples

Warm water radiator heating with motorised actuators

Characteristics	Parameter	Setting
Heating only	Activation of the heating/cooling function	*Heating*
	Type of heating function	*Continuous PI control*
	Adjustment of the PI control to the heating system	*Warm water heating (5 K / 150 min)*

Floor heating

Characteristics	Parameter	Setting
Heating only	Activation of the heating/cooling function	*Heating*
	Type of heating function	*Switching PI control*
	Adjustment of the PI control to the heating system	*Underfloor heating (5 K / 240 min)*

Cooling ceiling with motorised actuators

Characteristics	Parameter	Setting
Cooling only	Activation of the heating/cooling function	*Cooling*
	Type of heating function	*Continuous PI control*
	Adjustment of the PI control to the heating system	*Via control parameters*
	Proportional range for cooling	approx. 5 K (depending on application)
	Reset time for cooling	approx. 240 min (depending on application)

Switching electrical radiator heating

Characteristics	Parameter	Setting
Heating only	**Activation of the heating/cooling function**	*Heating*
	Type of heating function	*Switching PI control*
	Adjustment of the PI control to the heating system	*Electrical heating (4 K / 100 min)*

Air conditioning by means of the 2-tube FanCoil system / air conditioning system with heat pump and reversing valve

Characteristics	Parameter	Setting
Alternatively heating or cooling (manual switching)	**Activation of the heating/cooling function**	*Cooling and heating*
	Type of heating function	e.g. *Switching 2-point control*
	Hysteresis of the 2-point controller heating	ca. 1 K
	Type of cooling function	e.g. *Switching 2-point control*
	Hysteresis of the 2-point controller cooling	ca. 1 K
only one actuator is switched	**Allocation of the correcting variables to the objects "Heating" and "Cooling"**	*Together on "Heating" object*

Note for heat pump: Object 57 <Heating/cooling – Switch operating mode> must be connected with the status of the reversing valve.

Air-conditioning with 4-tube (2 cycle) FanCoil system (e.g. with switching actuators)

Characteristics	Parameter	Setting
Alternatively heating or cooling with automatic switching	**Activation of the heating/cooling function**	*Cooling and heating*
	Type of heating function	e.g. *Switching PI control*
	Adjustment of the PI control to the heating system	*Fan coil unit (4 K / 90 min)*
	Type of cooling function	e.g. *Switching PI control*
	Adjustment of the PI control to the heating system	*Fan coil unit (4 K / 90 min)*
two actuators are switched	**Allocation of the correcting variables to the objects "Heating" and "Cooling"**	*Isolated*
e.g. automated change between heating and cooling	**Switchover between heating and cooling**	*Automatic*

Temperature limitation by means of shading arrangement

Characteristics	Parameter	Setting
Cooling only	**Activation of the heating/cooling function**	*Cooling*
	Type of cooling function	*Switching 2-point control*
	Hysteresis of the 2-point controller cooling	great (e.g. 5 K)

3.9 Fan (fan coil)

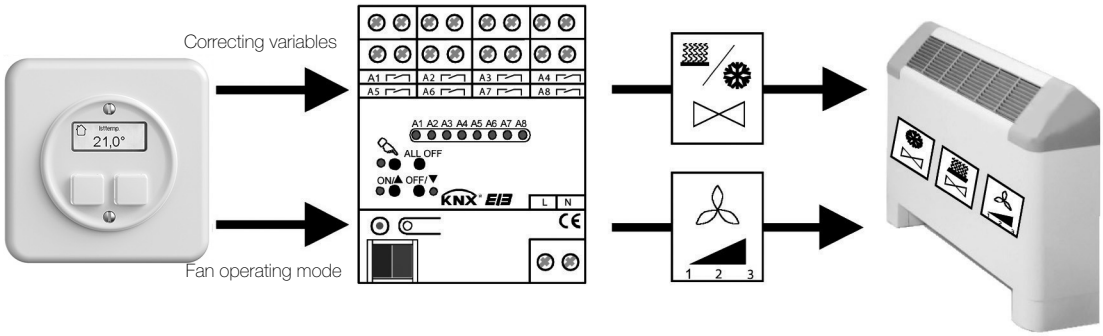
The term «FanCoil» has its origins in the English-speaking world and is a combination of the terms fan and coil. These terms immediately stand for the functioning principle of a FanCoil: A fan blows the sucked-in air through a heat exchanger, which mostly consists of a coil- or fan-type heating or cooling register. For this reason, the sucked-in air is conditioned, i.e. heated or cooled. In German, the term «Gebläsekonvektor» is used.

FanCoils are used for the room temperature control and are to be allocated to the group of air/water air conditioning systems. Such devices are either operated according to the recycled air principle or predominantly in greater air conditioning systems in the fresh air or mixed air operation. FanCoils are available in different construction types, which can be found frequently: Devices for wall, ceiling or duct mounting, free-standing or horizontally or vertically integrated in the covers or intermediate ceilings.

Basically, the FanCoil functions like a conventional radiator. However, the air circulation is supported by a fan unit. Thus, the heating and cooling performance can be significantly increased so that these devices may also be used for heating larger rooms. It is possible to heat rooms up to comfortable air temperatures within a short period of time.
The devices which are normally equipped with filters have multi-stage fans, the speed of which can be modified by means of fan stage inlets which in turn leads to a change in the fan performance. In practice, fans with up to 6 fan stages exist. Often, the fans are designed as tangential fans (cross-flow fan).

Manual fan control

The room thermostat enables the manual fan control in the FanCoil, independent from the correcting variable specification. Thus, it is possible to air condition rooms in any manually specified fan stage according to the respective requirements. It can be determined for the individual actuators whether the manual operation may be implemented with or without additional heating/cooling.



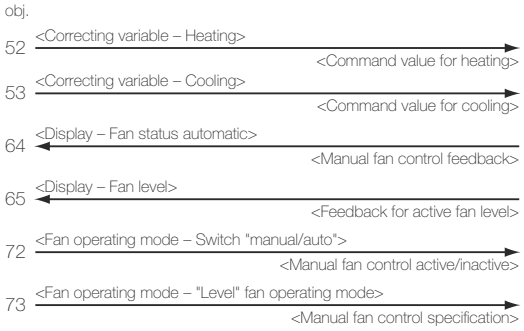
3.9.1 Feller FanCoil actuator 36363-1.REG

By means of its relay outlets, the Feller KNX FanCoil actuator 36363-1.REG controls the electric fan levels and valve inlets of one or two FanCoils. Depending on the device version, FanCoils are integrated in 2-tube systems (heating only, cooling only or heating and cooling via a common tubing system) or, alternatively, in 4-tube systems (heating and cooling via separate tubes). The FanCoil actuator supports both tubing principles. Additionally, the FanCoil actuator also enables a manual fan control by means of which pure ventilation functions without heating or cooling operations or an individual room ventilation with active heating or cooling is practical.

The manual fan control of the actuator is activated as soon as a telegram is received by object 73 <Fan operating mode – "Level" fan operating mode> via object 19 <Manual fan control specification>. The telegram is immediately evaluated as control specification so that the fan is switched into a fan level as specified. As long as the manual fan control is active, the fan may be controlled via object 19 <Manual fan control specification>.

For deactivating the manual fan control, the 1 bit object 18 <Manual fan control active/inactive> must be written-on with an OFF telegram. An ON telegram on this object will not show any reaction. When deactivating the manual fan control, the normal operating mode (automated operating mode) is activated again, insofar as no function with high priority (e.g. locking function) is active. In the normal operating mode, the FanCoil actuator controls the outlets in accordance with the latest correcting variable and operating mode received.

KNX RTH push-button RGB 2-gang
4472-B



Feller FanCoil actuator
36363-1.REG

The Feller FanCoil actuator differentiates between two functioning principles of the manual fan control, which may be configured with the **Manual fan control only with active heating/cooling** parameter in the ETS alternatively toward each other. Thus, the manual fan control can be activated entirely independently from the correcting variables. In such case, the manual ventilation without active heating or cooling is possible even with closed valves, as pure ventilation function. Furthermore, a manual fan control may only be executed when the heating or cooling valve is open, i.e. the heating or cooling mode is active.

For the correct interaction between the room thermostat and the Feller FanCoil actuator, the following parameters should be set correctly:

KNX RTH push-button RGB

Parameter page "Configuration display"	
Display "Auto"	*if fan status automatic = "0"*
Parameter page "Fan (fan coil)"	
Changing of fan operating mode at the device	*Enabled*
Number of fan levels	same as for the FanCoil actuator **Number of fan levels**
"Switch manual/auto" object type	*1 bit*
"Fan levels" object type	*8 bit 0..255*
Waiting time for fan coil response	*5* (to be adjusted to the bus load in the building)
Parameter page "Automatic fan operating mode"	
On "Switch manual/auto" object	*Send telegram*
Value	*Send OFF*
On "Fan Levels" object	*Do not send telegram*
On "Frost/heat protection" object	*Do not send telegram*
Parameter page "Level x fan operating mode"	
On "Switch manual/auto" object	*Do not send telegram*
On "Fan stages" object	*Send telegram*
Value	Fan level *x*
On "Frost/heat protection" object	*Do not send telegram*

Feller FanCoil actuator 36363-1.REG

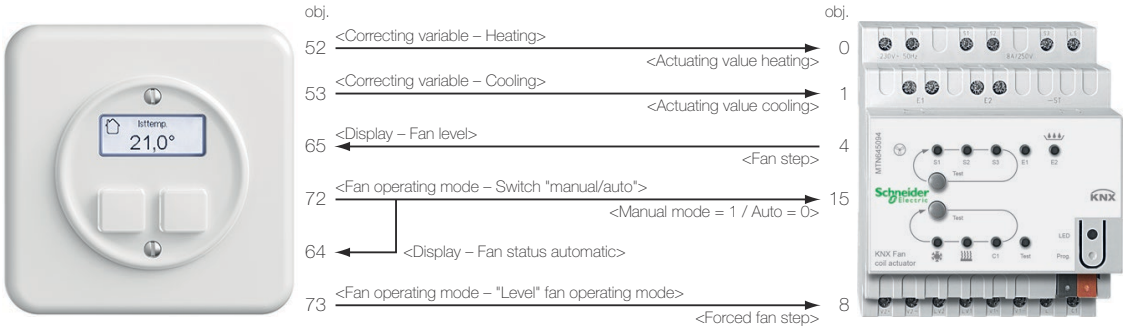
Parameter page "manual fan control"	
Manual fan control	*enabled*
Activation of manual fan control	*via object "Man. fan lev. specification"*
Fan level change-over in case of manual specification via	Value object (8 bit)
Parameter page "Kx fan feedback"	
Feedback for the active fan level	*yes, active signalling object*
Type of feedback	*Fan levels via value*

3.9.2 Schneider Electric FanCoil actuator MTN645094

The Schneider Electric FanCoil actuator MTN645094 is suited for 2-tube and 4-tube systems. It controls up to 3 fan levels as well as 2- or 3-point heating and/or cooling valves each. Via an additional relay, an electrical additional stage may be controlled.

For the manual fan control, the forced position is activated with the Schneider Electric FanCoil actuator. This is implemented when object 15 <Manual mode = 1 / Auto = 0> receives the value ON by object 72 <Fan operating mode – Switch "manual/auto">. The desired fan stage is set via object 8 <Forced fan step>; the data type is 8 bit percent. The forced position of the fan does not influence the valve control. The forced position is left if object 15 <Manual mode = 1 / Auto = 0> receives the value OFF from object 72 <Fan operating mode – Switch "manual/auto">.

Since the actuator does not comprise any response object for the "manual/auto" operating mode, object 64 <Display – Fan status automatic> of the room thermostat is to be connected with object 72 <Fan operating mode – Switch "manual/auto"> of the room thermostat.



KNX RTH push-button RGB 2-gang 4472-B

Schneider Electric FanCoil actuator MTN645094

For the correct interaction between the room thermostat and the Schneider Electric FanCoil actuator, the following parameters should be set correctly:

KNX RTH push-button RGB

Parameter page "Configuration display"

Display "Auto"	*if fan status automatic = "0"*

Parameter page "Fan (fan coil)"

Changing of fan operating mode at the device	*Enabled*
Number of fan levels	same as for the FanCoil actuator **Number of fan steps**
"Switch manual/auto" object type	*1 bit*
"Fan levels" object type	*8 bit 0..100%*
Waiting time for fan coil response	*5* (to be adjusted to the bus load in the building)

Parameter page "Automatic fan operating mode"

On "Switch manual/auto" object	*Send telegram*
Value	*Send OFF*
On "Fan levels" object	*Do not send telegram*
On "Frost/heat protection" object	*Do not send telegram*

Parameter page "Level x fan operating mode"

On "Switch manual/auto" object	*Send telegram*
value	*Send ON*
On "Fan levels" object	*Send telegram*
Value	same as with FanCoil actuator; recommended *25 % / 55 % / 85 %*
On "Frost/heat protection" object	*Do not send telegram*

Schneider Electric FanCoil actuator MTN645094

The standard values may be taken over.

Numerics

A

B

C

D

T

U

V

W

FELLER AG | Postfach | CH-8810 Horgen
Telefon +41 44 728 77 77 | Telefax +41 44 728 72 99

FELLER SA | Caudray 6 | CH-1020 Renens
Téléphone +41 21 653 24 45 | Téléfax +41 21 653 24 51

Service Line | Telefon +41 44 728 74 74 | info@feller.ch | www.feller.ch

10.KNX4472B-E.1512/151215